Instituto Superior de Gestão e Administração de Santarém

Mestrado em Engenharia de Tecnologias e Sistemas Web

Dissertação

Análise de Sentimento Textual para Correspondência de Perfis Vocacionais

Ana Paula Silva

Orientador: Professor Doutor Ricardo Vardasca

Santarém

Ano 2023

Instituto Superior de Gestão e Administração de Santarém

Mestrado em Engenharia de Tecnologias e Sistemas Web

Dissertação

Análise de Sentimento Textual para Correspondência de Perfis Vocacionais

Ana Paula Silva

Dissertação submetida para satisfação parcial dos requisitos do grau de Mestre em Engenharia de Tecnologias e Sistemas Web sob a orientação do Prof. Doutor Ricardo Vardasca

Santarém

Ano 2023

RESUMO

Ao terminar o 3ºciclo, os alunos estão ansiosos para saber qual o percurso educativo que será o melhor para o seu perfil de modo a corresponder ao seu objetivo de vida e, sendo eu professora, esta etapa motivou-me para a minha investigação. Identificou-se uma lacuna que foi a quantidade reduzida de soluções tecnológicas que permitam dar o contributo para a resolução deste problema, especialmente com o uso de algoritmos de *Machine Learning* e *Deep Learning*. Sendo assim, pretendeu-se analisar o uso de alguns destes algoritmos na problemática acima referida, no contexto de análise de sentimentos textual, verificando assim a possibilidade de, através de texto, conseguir resultados significativos para tomada de decisões nesta área. O objetivo desta dissertação é a procura e validação de soluções de Inteligência Artificial (IA) para dar resposta à falta de soluções tecnológicas que permitam a correspondência de perfis vocacionais. A metodologia utilizada e os respetivos instrumentos foram os seguintes: revisão da literatura, construção de um questionário no Google Forms, Diagrama de fluxo, Pré-processamento dos dados, análise de sentimento com a linguagem python, construção de léxicos, classificação com vários algoritmos, construção de modelos de IA, análise estatística, validação dos modelos, análise dos resultados obtidos e construção de uma aplicação web. Os participantes do questionário online foram 267 alunos do 9º ano de escolaridade de três Agrupamentos de Escolas (Agrupamento de Escolas de Coruche, Agrupamento de Escolas Eduardo Gageiro e Agrupamento de Escolas de Marinha Grande Poente) e a recolha dos dados foi efetuada durante o ano letivo 2022/2023. Podemos observar que com base nos resultados apresentados, o classificador Random Forest obteve a maior exatidão (0,54), indicando um desempenho geralmente melhor em comparação com os outros classificadores. Apesar dos meus resultados não terem sido tão elevados quanto alguns dos resultados relatados na revisão de literatura, a comparação é desafiadora devido às diferenças nos dados, algoritmos, objetivos, pré-processamento e outros fatores. A baixa precisão e exatidão podem ser atribuídas à quantidade e qualidade limitadas dos dados no meu dataframe. Em relação às minhas questões de pesquisa e hipótese, destaco que, apesar dos resultados da literatura terem sido promissores, o meu estudo concentrou-se num grupo demográfico específico e numa área de pesquisa única. Reconheço as limitações do meu estudo, incluindo a falta de adesão de escolas, a qualidade das respostas dos alunos e desafios no pré-processamento de dados. Estas limitações são identificadas como oportunidades de melhoria no futuro. Apesar da minha hipótese inicial não ter sido totalmente confirmada, este trabalho abre portas para pesquisas futuras com o objetivo de aprimorar modelos e expandir a sua aplicação a nível nacional. Em resumo, esta pesquisa oferece uma visão esclarecedora e promissora da interseção entre a Inteligência Artificial e a orientação vocacional, apesar dos desafios encontrados.

Palavras-Chave: Análise de Sentimento Textual, Inteligência Artificial, Machine Learning

ABSTRACT

At the end of the 3rd cycle, students are eager to know which educational path will be the best for their profile in order to meet their life goal and, being a teacher, this stage motivated me for my research. A gap was identified that was the reduced amount of technological solutions that allow to contribute to the resolution of this problem, especially with the use of Machine Learning and Deep Learning algorithms. Thus, it was intended to analyze the use of some of these algorithms in the aforementioned problem, in the context of textual sentiment analysis, thus verifying the possibility of achieving significant results for decision-making in this area through text. The objective of this dissertation is the search and validation of Artificial Intelligence (AI) solutions to respond to the lack of technological solutions that allow the matching of vocational profiles. The methodology used and the respective instruments were as follows: literature review, construction of a questionnaire in Google Forms, Flow diagram, Pre-processing of the data, sentiment analysis with the python language, construction of lexicons, classification with various algorithms, construction of AI models, statistical analysis, validation of the models, analysis of the results obtained and construction of a web application. The participants of the online questionnaire were 267 students of the 9th grade of three Schools (Agrupamento de Escolas de Coruche, Agrupamento de Escolas Eduardo Gageiro and Agrupamento de Escolas de Marinha Grande Poente) and the data collection was carried out during the academic year 2022/2023. We can observe that based on the results presented, the Random Forest classifier obtained the highest accuracy (0.54), indicating a generally better performance compared to the other classifiers. Although my results were not as high as some of the results reported in the literature review, the comparison is challeging due to differences in data, algorithms, objectives, preprocessing, and other factors. The low precision and accuracy can be attributed to the limited quantity and quality of the data in my dataframe. Regarding my research questions and hypothesis, I highlight that, although the results of the literature were promising, my study focused on a specific demographic group and a single research area. I recognize the limitations of my study, including the lack of adherence from schools, the quality of student responses, and challenges in pre-processing data. These limitations are identified as opportunities for improvement in the future. Although my initial hypothesis has not been fully confirmed, this work opens doors for future research with the aim of improving models and expanding their application at the national level. In summary, this research offers an enlightening and promising insight into the intersection between Artificial Intelligence and vocational guidance, despite the challenges encountered.

Keywords: Artificial Intelligence, Machine Learning, Textual Sentiment Analysis

Agradecimentos

Agradeço a todas as pessoas que me acompanharam neste percurso e que me deram força e resiliência ao longo desta etapa tão importante.

À minha mãe, que sempre me incentivou a nunca desistir, ao meu pai que me educou com bons valores e deu-me o exemplo de motivação para estudar sempre e ao meu padrasto que sempre me apoiou nas minhas decisões.

Ao meu namorado, que sempre me apoiou e encorajou a fazer melhor.

Aos meus colegas de turma que viveram comigo esta caminhada.

Ao Professor Doutor Ricardo Vardasca, que me orientou de forma excelente, sempre com paciência, competência, saber e determinação.

Aos docentes deste Mestrado, tanto do ISLA de Santarém como de Gaia, bem como os seus Diretores.

Aos Diretores de Escolas do Agrupamento de Escolas Eduardo Gageiro, o professor Carlos Candeias, do Agrupamento de Escolas de Coruche, a professora Isabel Cordeiro e do Agrupamento de Escolas de Marinha Grande Poente, o professor Cesário Silva, que aceitaram que a sua escola participasse no meu estudo.

Aos professores das escolas que aplicaram o meu questionário, em especial ao professor Humberto Barreiras, do AE de Coruche, bem como aos alunos que responderam.

A todos os que contribuíram para me tornar mais forte, exigente e experiente, o meu sincero obrigado.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

Lista de Abreviaturas e Acrónimos

AE	Agrupamento de Escolas
AdaBoost	Adaptative Boosting
CA	Computação Afetiva
AUC	Área sob a curva ROC
CNN	Rede Neuronal Convolutional
DL	Deep Learning
DLNN	Rede Neuronal de Aprendizagem Profunda
DTREE	Decision Tree
GAN	Generative Adversarial Network
IA	Inteligência Artificial
K-NN	K-Nearest Neighbors
LSTM	Long short-term memory
ML	Machine Learning
MLP	Multi-layer Perceptron
NB	Naive Bayes
NLTK	Biblioteca Natural Language Toolkit
PLN	Processamento de Linguagem Natural
RMSE	Root-mean-square error
RN	Redes Neuronais
RNN	Rede Neuronal Recursiva
RNTN	Rede de tensor neuronal recursiva
ROC Curve	Receiver operating characteristic curve
SVM	Support Vector Machines
TEIP	Territórios Educativos de Intervenção Prioritária
TI	Tecnologias Informáticas

VADER	Valence Aware Dictionary and Sentiment Reasoner
R2 Score	Coeficiente de Determinação
TP	True Positive
FP	False Positive
FN	False Negative
TN	True Negative

1. INTRODUÇÃO

Desde sempre um dos grandes desafios do adolescente passa por descobrir a sua vocação e decidir a melhor escolha dentro dos seus gostos, competências e habilidades. As transições entre ciclos de escolaridade são etapas de alguma incerteza e instabilidade, tendo vindo a receber, por isso, atenção de variados campos de investigação e das políticas educativas.

Tem-se visto, ao longo do tempo, uma crescente preocupação em detetar emoções, com o desenvolvimento da Inteligência Artificial (IA), *Machine Learning* (ML) e *Deep Learning* (DL), como se pode observar no desenvolvimento de algoritmos que recebem tweets, por exemplo, e que conseguem detetar emoções e a sua polaridade.

Irá ser desenvolvida uma investigação com algoritmos que de alguma forma possam ser utilizados no âmbito da Dissertação, mesmo que o seu objeto de estudo seja diferente (um blog, redes sociais, Profissionais de Tecnologias Informáticas (TI) e críticas de filmes), dado que há poucos estudos publicados nesta área. Serão aprofundados alguns métodos de IA, apresentados os resultados e por fim será feita uma discussão e proposto o trabalho futuro a desenvolver.

Nesta dissertação a finalidade está no estudo e conceção e validação de modelos sobre a análise de emoções em texto, bem como criação de correspondência de perfis vocacionais. Este trabalho desenvolveu-se no âmbito académico do Instituto Superior de Gestão e Administração (ISLA) de Santarém e tem como objetivos: a deteção e a procura de emoções num determinado texto que permitam validar perfis vocacionais, elaborar um modelo através de algoritmos de inteligência artificial que possam dar resposta ao problema, treinando algoritmos de ML e DL.

Identificou-se uma lacuna que foi a quantidade reduzida de soluções tecnológicas que permitam dar o contributo para a resolução deste problema, especialmente com o uso de algoritmos de ML e DL.

Assim, pretende-se dar um contributo, através de pesquisa de soluções de IA, nomeadamente as que são possíveis de aplicar a análise de sentimentos em texto, de modo a ajudar os estudantes a definir, a escolher ou percecionar quais são as melhores alternativas face às suas características para prosseguimento de estudos. Coloca-se a hipótese de o uso de modelos de IA fornecerem uma exatidão (accuracy) de elevada percentagem nos casos estudados na revisão da literatura e serem possíveis de ser usadas neste campo de

investigação e aplicação. Pretende-se assim avaliar o contributo que as técnicas associadas à IA podem dar na resolução deste problema.

1.2 Problema, questões de investigação e hipótese

Problema: Identificou-se uma lacuna que foi a quantidade reduzida de soluções tecnológicas que permitam dar o contributo para a resolução deste problema, especialmente com o uso de algoritmos de ML e DL.

Questões de investigação:

Como um modelo de Inteligência Artificial aplicado a análise sentimental textual pode ajudar os estudantes a escolher a sua decisão de percurso vocacional futuro?

Quais os melhores algoritmos para solucionar o problema com melhor eficácia, precisão e exatidão?

Pode a solução final ser aplicada a nível nacional e ser uma referência nesta problemática?

Hipótese: Os usos de modelos de IA fornecem uma exatidão de elevada percentagem nos casos estudados na revisão da literatura e são possíveis de ser usadas neste campo de investigação e aplicação.

1.3 Motivação

A tomada de decisão dos alunos do 3º ciclo é repleta de incertezas e devido a haver poucas tecnologias com base na Inteligência Artificial que contribuam para esta área, a minha investigação é nesse sentido, nomeadamente no uso de algoritmos de ML e de DL. A minha motivação surge essencialmente da minha prática profissional, como professora de TIC (Tecnologias da Informação e Comunicação) do Ensino Básico e Secundário.

1.4 Contexto

Uma parte integrante da tomada de decisão humana é a emoção, e esta componente poderia potencialmente ser dada aos computadores, (Agrawal, Ameeta; An, Aijun, 2012). A análise de textos (*Text Analytics*) é uma das áreas de ML e tem aplicações diretas na análise de sentimentos e modelo estatístico de tópicos. A Análise de Sentimentos visa identificar o sentimento que os utilizadores apresentam a respeito de alguma entidade de interesse (um

produto específico, uma empresa, uma pessoa, etc.) baseado no conteúdo disponível na Web (Moreira, João, 2020).

Com os recentes avanços na aprendizagem profunda, conhecida como *Deep Learning*, a capacidade dos algoritmos de analisar textos melhorou consideravelmente. No caso de algoritmos de Processamento de Linguagem Natural (PLN) ou utilizados na computação afetiva, necessitam de palavras para poderem ser avaliados. O PLN é uma área de pesquisa que surgiu nos anos 50 e é uma das áreas da IA, cuja preocupação é estabelecer a ligação entre os computadores e a linguagem natural das pessoas. Esta interação homem-computador permite aplicações como, resumos automáticos de textos, análise de sentimentos, a extração de tópicos, etc. Na classificação/análise de sentimentos mede-se a polaridade da opinião, se é positiva ou negativa, através de algoritmos de ML, por exemplo. Estes algoritmos analisam palavras-chave que caracterizam uma palavra como positiva ou negativa – como por exemplo "alegria" - positivo, "raiva" - negativo.

A Computação Afetiva (CA) (*Affective Computing*), destacada por Picard (1997) aborda modelos para o reconhecimento computacional de emoções humanas e faz uma ligação entre Computação, Engenharia e emoções. Duas das técnicas de aprendizagem profunda mais populares para a análise de sentimentos são as Redes Neuronais Convolutionais (CNNs) e a *Long short-term memory* (LSTM) (Faria, E., 2018). As CNNs apenas extraem as características locais, enquanto os LSTMs são um tipo de rede que tem uma memória que se lembra de dados de input anteriores e toma decisões com base nesse conhecimento, ou seja, os LSTM são mais apropriados para texto, visto que cada palavra tem significado baseado nas palavras circundantes.

1.5 Finalidade e objetivos

Nesta dissertação a finalidade está no estudo, conceção e validação de modelos de IA sobre a análise de emoções em texto, bem como criação de correspondência de perfis vocacionais.

Objetivos Gerais:

- Colmatar a lacuna existente a nível tecnológico de uma ferramenta que ajude os alunos a tomar decisões relativas ao seu percurso escolar;
- Detetar emoções num determinado texto que permita validar perfis vocacionais;

- Pesquisar sobre o Estado da Arte acerca do tema;
- Aplicar a análise sentimental de textos;
- Criar um modelo validado de análise de sentimentos textuais.

Objetivos Específicos:

- Elaborar um modelo através de algoritmos de inteligência artificial que possa dar resposta ao problema, treinando algoritmos de ML e DL.
- Proporcionar aos alunos a análise vocacional e correspondência de perfil mais adequado a cada um através de aplicação de um questionário a alunos de 9º ano de escolaridade e análise de sentimento textual feita a partir das respostas do mesmo.

1.6 Contributos

O desenvolvimento deste modelo irá focar-se na melhoria da habilidade de tomada de decisão dos alunos de 9º ano de escolaridade, em relação à sua vocação e percurso escolar futuro. O modelo desenvolvido no âmbito desta dissertação possibilitará obter respostas quanto à vocação dos alunos de modo a permitir a tomada de decisões mais conscientes acerca do seu futuro escolar.

1.7 Estrutura do documento

O presente documento é composto por seis capítulos que compõem o corpo principal da dissertação, seguida de uma secção para Referências Bibliográficas e outra para Anexos.

O primeiro capítulo tem o objetivo de introduzir o problema principal enquanto remete para o contexto propício, objetivos e contributos.

No segundo capítulo é reunida a informação bibliográfica relacionada com o tema desta dissertação para fundamentar a melhor forma de desenvolvimento e que passos seguir na fase de desenvolvimento.

O terceiro capítulo tem o objetivo de descrever o método aplicado e explicar as ferramentas utilizadas para as decisões tomadas.

O quarto capítulo tem o objetivo de apresentar os resultados obtidos.

O quinto capítulo remete para a discussão dos resultados obtidos com esta investigação e avalia a hipótese levantada no primeiro capítulo de modo a verificar o sucesso do modelo.

O sexto e último capítulo apresenta as conclusões obtidas no desenvolvimento do modelo apresentado nesta dissertação.

Após este capítulo existem duas secções, as Referências Bibliográficas utilizadas na elaboração desta dissertação e uma secção de anexos que complementam a informação apresentada na Dissertação.

2. REVISÃO DA LITERATURA

A revisão da literatura é uma etapa crucial na pesquisa científica, pois permite ao pesquisador conhecer o estado atual do conhecimento sobre o assunto de estudo. Pesquisou-se temas relacionados com a área em estudo, nomeadamente, algoritmos de IA aplicados ao problema. A revisão da literatura foi realizada com recursos aos repositórios IEEE Xplore, Google Academic, ResearchGate, Springer e Google, pesquisando palavras como *análise de sentimentos, machine learning, análise vocacional, tomada de decisão, redes neuronais em análise de sentimentos, aprendizagem profunda, text mining, Deep learning, Redes Neuronais,* entre outras.

Os objetivos da revisão da literatura são os seguintes:

1. Identificar os principais algoritmos de IA utilizados em análise de sentimento textual.
2. Avaliar as vantagens e desvantagens dos diferentes algoritmos de IA.
3. Compreender as tendências e evoluções recentes na área de algoritmos de IA.
4. Verificar se há falhas de conhecimento na área de algoritmos de IA e identificar áreas para pesquisa futura.
5. Analisar o desempenho dos algoritmos de IA em comparação com outras técnicas e métodos.
6. Investigar os desafios e limitações enfrentados na utilização de algoritmos de IA e como eles podem ser superados.

A pesquisa revela que as emoções constituem motores potentes, previsíveis, por vezes prejudiciais e também benéficos para a tomada de decisões (Lerner, Jennifer S., 2004).

Para processar a linguagem natural, pode-se processar os textos em diversos níveis, tais como léxico, morfológico, semântico, etc, permitindo que as máquinas compreendam como os humanos falam (Liddy, E.D., 2001).

2.1 Aprendizagem Automática

No processo de *Data Mining*, muitas vezes não é viável a identificação de padrões estatisticamente. Os algoritmos de aprendizagem automática consistem na construção de um modelo matemático baseado num conjunto de dados, conhecido por training data. Este modelo matemático vai sendo moldado pelo algoritmo, consoante os dados que se obteve do

modelo, de modo que este modelo consiga fazer previsões e/ou decisões sobre dados futuros, o mais corretamente possível (Cardoso, José, 2020). A análise de sentimentos é um método utilizado para extrair a polaridade subjetiva bem como os algoritmos de ML e DL usados para classificação de sentimentos.

2.2 Linguagem Python

A linguagem de programação Python vem com uma extensa biblioteca padrão, incluindo componentes para programação gráfica, processamento numérico e conectividade web (Eckman, P., 1994). Esta linguagem é bastante utilizada em ML, já que vem com um pacote Scikit-learn, um pacote de ML simples de usar que facilita esta utilização. Segundo a pesquisa realizada é adequada a utilização da Biblioteca Natural Language Toolkit (NLTK), visto que é uma ajuda na compreensão da aplicação de algoritmos de análise de texto (Pyhton), e do PLN. A Análise de Sentimentos, usando a biblioteca NLTK pode ser usada para identificar o sentimento, a opinião ou a crença de uma afirmação, de muito negativa, a neutra, a muito positiva. (Bird, S., Klein, E., & Loper, E. 2009)

O Python vem com uma extensa biblioteca padrão, incluindo componentes para programação gráfica, processamento numérico e conectividade web (Eckman, P., 1994). O Python é bastante utilizado em ML, já que vem com um pacote Scikit-learn, um pacote de ML simples de usar que facilita esta utilização.[1] Na etapa da análise lexical, ocorre a conversão de uma cadeia de caracteres numa cadeia de palavras. Na fase seguinte, artigos, preposições e conjunções são candidatos naturais à lista de stopwords, ou seja, a serem eliminados. Então, pode ser executada a normalização lexical porque frequentemente o utilizador especifica uma palavra na consulta (Eckman, P., 1994). Na figura 1 pode-se observar um exemplo de uma lista de stopwords.

1. https://harve.com.br/blog/programacao-python-blog/python-para-que-serve-top-5-utilidades/

De	A	O	Que	E	Do
um	Para	Com	não	Os	No
ao	Das	À	as	Dos	Como
já	Nos	Também	Só	Pelo	Pela
Eu	Ela	Entre	Sem	Mesmos	Aos
depois	Nas	Me	esse	Você	Essa
eles	Suas	Meu	Às	minha	Elas
numa	Lhe	Deles	Essas	Esses	Tivesse
pelos	Dele	Tu	Te	Vocês	Vos
pelas	Minhas	Teus	Tuas	Nosso	Nossa
este	Dela	Teu	tua	Delas	Esta

Figura 1-Lista de stopwords

Fonte: Adaptado de (Minhaneli, Gustavo, 2018)

2.3 Data Mining (Mineração de dados)

A mineração de dados é um método muito conhecido para extrair padrões e conhecimentos classificados, escondidos num grande número de conjuntos de dados. A fim de extrair o conhecimento, existem várias técnicas para categorizar.

2.4 Métodos de IA

2.4.1. Soluções com Machine Learning

Machine Learning consiste na execução de algoritmos que criam de modo automático modelos de representação de conhecimento com base num conjunto de dados. Tem por base o treino das máquinas para que o algoritmo possa aprender e melhorar o seu desempenho.

2.4.1.1 Naive Bayes

O algoritmo Naive Bayes (NB) é utilizado como base na classificação de textos por ser rápido e fácil de usar. Não é um único algoritmo, mas sim, uma família de algoritmos onde todos partilham um princípio comum, ou seja, cada par de funcionalidades classificadas é independente uma da outra.[2] O teorema de Bayes é afirmado matematicamente com a seguinte equação, onde A e B são eventos e $P(B) \neq 0$.

[2]https://www.geeksforgeeks.org/naive-bayes-classifiers/

$$P(A/B) = \frac{P(B/A)xP(A)}{P(B)}$$

Figura 2-Teorema de Bayes,

Fonte: Minhaneli, Gustavo, (2018)

Por exemplo, para usar o NB o autor Gustavo Minhaneli mostrou que para que um texto extraído do Twitter possa ser considerado positivo, por exemplo, a fórmula ficaria conforme apresentado na Figura 3 (Minhaneli, Gustavo, 2018).

$$P(positive|tweet) = \frac{P(tweet|positive)\ P(positive)}{P(tweet)}$$

Figura 3-Teorema de Bayes aplicado a tweets

Fonte: (Minhaneli, Gustavo, 2018)

Esta análise permitiu encontrar os seguintes resultados: Ao avaliar as 357 frases, observou-se que foram classificadas, 195 como alegria, 34 como desgosto, 16 como medo, 39 como raiva, 13 como surpresa e 28 como tristeza, obtendo uma precisão de 78,27%, totalizando 194 classificadas corretamente e 32 erroneamente (Minhaneli, Gustavo, 2018).

Um estudo realizado por Kularbphettong e Tongsiri visou desenvolver um sistema de apoio à decisão para a seleção de um curso superior dos estudantes utilizando dois métodos de ML, J48 (Árvore de Decisão) e NB. Os seus resultados mostraram que o NB teve o melhor desempenho, com 92,13%, 93% e 91% de exatidão **(**Kularbphettong, K., & Tongsiri, C., 2014).

2.4.1.2 Support Vector Machines

Os Support Vector Machines (SVM) são uma das técnicas de aprendizagem automática amplamente utilizadas para a deteção da polaridade a partir de texto. Os SVMs mostram ser altamente eficientes na categorização tradicional do texto, geralmente superando o algoritmo NB **(**Pang, B., Lee, L., & Vaithyanathan, S., 2002).

No caso de duas categorias, o procedimento de treino representado pelo vetor w, que não só separa os vetores de documento numa classe, mas para o qual a separação, ou margem, é a maior quanto possível. Para cj $\in$ {1,-1}, correspondente a positivo, negativo, onde αj é obtido por uma dupla otimização do problema, a solução pode ser escrita como:

$$\vec{w} := \sum_j \alpha_j c_j \vec{d}_j, \quad \alpha_j \geq 0,$$

Figura 4-Solução com SVM

Fonte: (Pang, B., Lee, L., & Vaithyanathan, S., 2002)

O vetor dj em que αj é maior que zero é o Support Vector e contribui para o vetor W. A classificação do teste é determinada pelo lado em que caírem os vetores **(**Pang, B., Lee, L., & Vaithyanathan, S., 2002). Num estudo, em que se usou o NB e o SVM, o resultado de precisão do Naive Bayes foi de 65,6%, tendo em conta o dataset analisado retirado dos posts de um blog, enquanto a precisão máxima alcançada do SVM foi de 73,89% (Aman, S., & Szpakowicz, S., 2007). Esse estudo detetou, dentro das frases de emoção, sete possíveis categorias de emoção às quais podem ser atribuídas a uma frase. A tabela 1 mostra o valor do kappa para cada uma destas categorias de emoções para cada par anotador (Aman, S., & Szpakowicz, S., 2007).

Tabela 1-Categorias de emoções com valor de k

Category	a↔b	a↔c	a↔d	average
happiness	0.76	0.84	0.71	0.77
sadness	0.68	0.79	0.56	0.68
anger	0.62	0.76	0.59	0.66
disgust	0.64	0.62	0.74	0.67
surprise	0.61	0.72	0.48	0.60
fear	0.78	0.80	0.78	0.79
mixed emotion	0.24	0.61	0.44	0.43

	a↔b	a↔c	a↔d	average
Kappa	0.73	0.84	0.71	0.76

Fonte: Aman, S., & Szpakowicz, S., (2007)

Os indicadores de emoções usados foram palavras e conjuntos de palavras selecionados pelos anotadores.

Aman, S., & Szpakowicz, S. (2007) no seu estudo ainda estudaram sinais de pontuação na classificação de sentimentos, como o ponto de exclamação. Neste estudo foram usados os algoritmos NB e o SVM, sendo que o SVM atingiu uma precisão máxima de 73.89% e o NB uma precisão máxima de 72,08%. Neste caso surge uma limitação que pode ser a capacidade de atingir uma maior precisão nos resultados, visto existir a necessidade de classificar a análise a nível semântico e de contexto. Num outro estudo que analisou 839

palavras contidas num dvd, obteve um resultado de 684 com sentimento positivo e 155 casos com sentimento negativo (Elangovan, D., & Subedha, V., 2020).

Tabela 2-Comparação de métodos utilizados com o dataset do dvd

Sl. No	Dataset	Classifier	Sensitivity	Specificity	Accuracy	F-score	Kappa
1	DVD	FF-MLP	98.52	91.19	97.14	98.24	90.59
2		ACO-K	98.23	89.93	96.66	97.94	89.03
3		ACO	97.92	86.50	95.70	97.35	86.03
4		PSO	96.57	79.69	92.61	95.23	78.76
5		CSK	96.27	72.85	90.10	93.48	73.05
6		SVM	96.93	74.88	91.18	94.20	75.85
7		NN	93.93	71.61	87.84	91.82	68.15

Fonte: Elangovan, D., & Subedha, V., (2020)

Na tabela 2 pode-se comprovar que em termos de exatidão o algoritmo que obteve maior percentagem foi o FF-MLP, e o kappa com valor de 90.59, ficando o SVM em sexto lugar, com 91,18% de exatidão e valor de kappa 75.85, contudo todos tiveram um bom resultado acima dos 85% de exatidão. É de salientar que o FF-MLP foi um modelo proposto pelos autores do estudo, sendo um modelo de DL, enquanto que o SVM é um algoritmo de ML.

2.4.1.3 K-Nearest Neighbors (K-NN)

O K-Nearest Neighbors (K-NN) É um algoritmo simples e de fácil implementação, de aprendizagem supervisionada, que pode ser utilizado para resolver problemas de classificação e regressão. Este algoritmo armazena todos os casos disponíveis e classifica novos casos baseados numa medida de semelhança, ou seja, classifica um novo caso pela maioria de votos dos seus vizinhos.

Num estudo realizado para ajudar licenciados em TI a escolherem a sua carreira, foram recolhidos dados de 2255 empregados no sector de TI na Arábia Saudita. Os dados foram então treinados e feitos testes supervisionados com algoritmos de ML para identificar a carreira mais apropriada entre 3 carreiras (Programador, Analista e Engenheiro). O algoritmo XGBoost forneceu a exatidão mais elevada de 70.47%. Para obter as informações foi realizado um questionário em que os participantes responderam sobre as suas softskills

e skills de programação, aplicado entre fevereiro e outubro de 2018. Na resposta sobre profissão, obteve-se 565 valores/profissões (Al- Dossari, H., Nughaymish, F. A., Al-Qahtani, Z., Alkahlifah, M., & Alqahtani, A., 2020). Através da bilioteca de python scikitlearn, implementou-se os algoritmos seguintes: K-Nearest Neighbors (K-NN), Decision Tree (DT), Bagging meta-estimator, Gradient Boosting e o XGBoost. Estes algoritmos foram treinados com um dataset de 1707 empregados e foi obtida a seguinte percentagem de exatidão para cada um deles (Al- Dossari, H., Nughaymish, F. A., Al-Qahtani, Z., Alkahlifah, M., & Alqahtani, A., 2020).

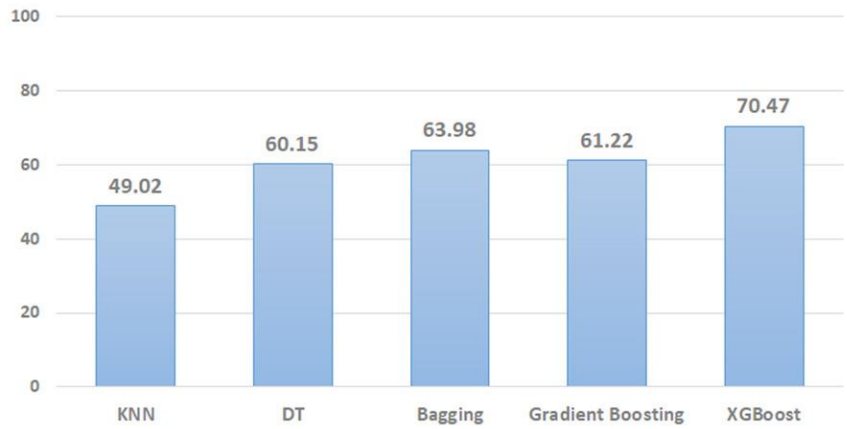


Figura 5- Exatidão dos 5 algoritmos

Fonte: (Al- Dossari, H., Nughaymish, F. A., Al-Qahtani, Z., Alkahlifah, M., & Alqahtani, A., 2020)

Os treinos permitiram obter o resultado de que o XGBoost foi o que obteve exatidão mais elevada de 70,47%. A percentagem baixa de exatidão obtida, poderá ser justificada pela pouca quantidade de dados no dataset. Neste estudo o K-NN obteve a exatidão mais baixa dos 5 algoritmos (Al- Dossari, H., Nughaymish, F. A., Al-Qahtani, Z., Alkahlifah, M., & Alqahtani, A., 2020).

2.4.1.4. Árvores de Decisão

Um importante ramo da mineração de dados, a classificação, pode ser resolvido por vários modelos de aprendizagem indutiva. As árvores de decisão, devido às suas vantagens, são uma das técnicas mais populares (Pong-Inwong, Chakrit; Kaewmak, Konpusit, 2016). A Árvore de decisão é uma técnica de ML utilizada para criar modelos de classificação que

funciona criando ramificações com base em diferentes critérios, a partir dos quais é possível chegar a uma decisão ou previsão.

Na análise de sentimento, uma árvore de decisão pode ser usada para classificar textos em diferentes categorias de sentimento, como positivo, negativo ou neutro. É preciso treinar o modelo com um conjunto de dados rotulados, ou seja, textos já classificados em diferentes categorias de sentimento. O modelo então aprende a identificar padrões nos dados e a criar ramificações com base nesses padrões e depois de treinado, pode ser usado para classificar novos textos com base nas ramificações criadas. Existem vários estudos que mostram o desempenho das árvores de decisão na análise de sentimento. Em geral, elas têm um desempenho bastante bom, embora possam ser menos precisas do que outros algoritmos de ML em alguns casos.

2.4.2 Soluções com Deep Learning

2.4.2.1. Redes Neuronais

Redes Neuronais (RN) são um método prático para se aprender aproximações de funções apartir de exemplos. Estes algoritmos podem mudar a sua própria função ou valores aplicados à entrada de dados após uma execução. São caracterizadas pela utilização de neurónios, ou seja, unidades de processamento de informação. Pode-se dizer que as RNs são capazes de reconhecer e classificar padrões e generalizar o conhecimento obtido.

2.4.2.1.1. Rede Neuronal Convolucional

As recentes arquiteturas da Rede Neuronal Profunda têm vindo a realizar um desempenho significativo em tarefas PLN devido à quantidade crescente de informação. A Rede Neuronal Convolutional (CNN) é uma das arquiteturas de rede neuronal mais populares na classificação de imagem devido à sua composição e invariabilidade local, como também mostrou uma performance significativa na identificação de sentimentos em textos

(Haque, M. R., Lima, S. A., & Mishu, S. Z., 2019). Num estudo sobre comentários de turistas chineses, com um dataset de 2018 com 1,000,000 comentários retirados de uma base de dados da Wiki Dump, foram aplicados os algoritmos LSTM e CNN e foram obtidos os seguintes resultados com o CNN:

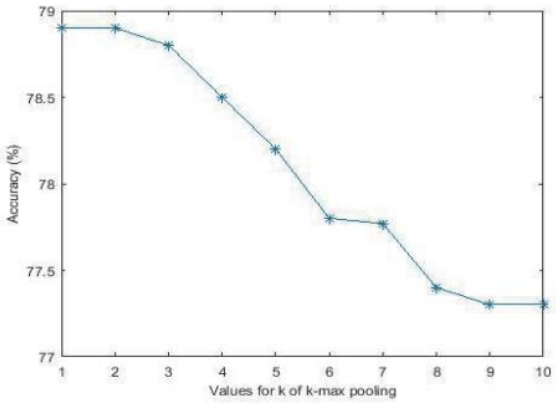


Figura 6-Exatidão com diferentes resultados k para o k-max pooling

Fonte: Gao, J., Yao, R., Lai, H., & Chang, T. C., (2019)

Exploraram o k de 1 a 10 quando o número de mapas estava definido para 100 e o tamanho da região única estava definido para 13, e foi obtida a precisão (Gao, J., Yao, R., Lai, H., & Chang, T. C., 2019). As camadas de pooling são normalmente utilizadas imediatamente a seguir às camadas convolucionais e são responsáveis por encontrar a informação mais significativa obtida nas camadas convolucionais (Faria, E., 2018).

2.4.2.1.2. Long Short-Term memory (LSTM)

A arquitetura Long Short-Term Memory (LSTM) consiste num conjunto de subredes recorrentemente ligadas, conhecidas como blocos de memória. Estes blocos podem ser considerados como uma versão diferenciável dos chips de memória num computador digital (Graves, A., 2012). No estudo anteriormente mencionado sobre análise de sentimentos em comentários de turistas chineses, obteve-se os seguintes resultados:

A dimensionalidade da palavra incorporada com o intervalo de 160 a 300 tem sido comumente usada. Neste estudo considerou-se variações de 160 a 240. Para cada dimensão, usou-se um tamanho diferente de janela de contexto de 20 a 100 para produção de 25 configurações diferentes. A partir da tabela 1, a definição do parâmetro d=200 e k=80 contribuiu com a melhor precisão de reconhecimento, sendo *d* as dimensões das palavras e *k* o tamanho das janelas de contexto (Gao, J., Yao, R., Lai, H., & Chang, T. C., 2019).

Tabela 3-Precisões do sistema com diferentes dimensões

d\k	20	40	60	80	100
160	79.52	79.92	80.87	81.52	81.43
180	80.18	81.23	81.63	82.13	82.07
200	80.27	81.34	81.92	**82.85**	82.71
220	80.31	81.39	81.87	82.72	82.68
240	80.24	81.27	81.71	82.55	82.53

Fonte: Gao, J., Yao, R., Lai, H., & Chang, T. C., (2019)

2.4.2.1.3. Multi-Layer Perceptron (MLP)

O Multi-layer Perceptron (MLP) é uma rede neuronal artificial que gera um conjunto de saídas a partir de um conjunto de entradas. Um MLP é caracterizado por várias camadas de nós de entrada ligados como um grafo direcionado entre as camadas de entrada e de saída. O MLP usa a técnica de aprendizagem supervisionada backpropagation para treinar a rede[3]. Este algoritmo pode ser dividido em duas fases (forward e backward).

Baseado num dataset educacional retirado do repositório Kalboard 360 dataset e feito o data mining através da ferramenta Weka, foi realizada uma investigação usando alguns algoritmos como o SVM, DTREE, K-Star, Bayes Net e MLP. O objetivo era obter uma previsão da performance de estudantes (Sultana, J., Sultana, N., Yadav, K., & AlFayez, F., 2018).

Tabela 4-Comparação de Métodos dataset educacional

Methods	Accuracy	RMSE	TP	FP	ROC
SMO	78.75	0.35	0.78	0.12	0.86
MLP	78.33	0.34	0.78	0.12	0.89
Random Forest	76.66	0.33	0.76	0.13	0.89
Simple Logistics	5	0.34	0.76	0.13	0.88
Decision Tree	75.83	0.36	0.75	0.13	0.83
K-Star	73.75	0.38	0.73	0.14	0..87
Multi-Class Classifier	70.62	0.37	0.70	0.17	0.85
Bays Net	70.20	0.37	0.70	0.16	0.84

Fonte: Sultana, J., Sultana, N., Yadav, K., & AlFayez, F., (2018)

[3] https://www.techopedia.com/definition/20879/multilayer-perceptron-mlp

Observa-se que o método SVM e MLP teve um bom desempenho na previsão do desempenho dos alunos com uma exatidão de 78,75 % e 78,33 % e a Bayes Net teve uma exatidão de 70,20% (Sultana, J., Sultana, N., Yadav, K., & AlFayez, F., 2018). Os resultados indicam que os métodos de aprendizagem SVM e MLP indicaram um bom desempenho em geral em comparação com os outros classificadores em termos de exatidão de classificação, RMSE, sensibilidade e especificidade e curva ROC.

2.4.2.1.4. Rede de Tensor Neuronal Recursiva

As redes de tensores neuronais recursivas consideram frases de entrada de qualquer comprimento. Representam uma frase através de vetores de palavras e uma árvore de parse e, em seguida, computam vetores para nós mais altos na árvore usando a mesma função de composição baseada em tensor (Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., 2013). Este modelo RNTN foi proposto e implementado por Shocher, (2013), que utiliza a função tensor para todos os nós e utiliza as seguintes equações:

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix}\right)$$

$$p_2 = f\left(\begin{bmatrix} a \\ p_1 \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} a \\ p_1 \end{bmatrix} + W \begin{bmatrix} a \\ p_1 \end{bmatrix}\right)$$

Figura 7- Computação p1 e p2

Fonte: Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., (2013)

Na figura 7, temos p1 como o nó formado por dois vetores, b e c, e p2 como sendo a composição de a e p1. O autor considera que este modelo tem vantagens sobre o modelo Redes Neuronais Recursivas (RNN), dado que quando V é zero, o tensor pode relacionar vetores de entrada. A Universidade de Stanford disponibilizou a Stanford Sentiment Treebank que consiste em 11,855 frases extraídas de críticas de filmes, que surgiu de uma publicação de Pang e Lee (2005) (Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., 2013).

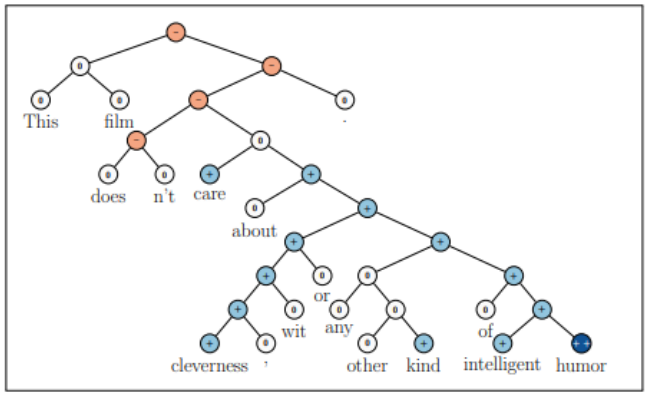


Figura 8- Exemplo de Rede Tensor Neuronal Recursiva

Fonte: Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., (2013)

O topo da árvore de sentimentos possui a polaridade associada da frase inteira, pelo que se verifica que a polaridade é negativa, sendo o modelo bottom-up. A Rede de Tensor Neuronal Recursiva (RNTN) prevê com exatidão 5 classes de sentimento, de muito negativas para muito positivas (– –, 0, +, +, +, +), em cada nó de uma árvore de parse e capturando a negação e o seu âmbito nesta frase.

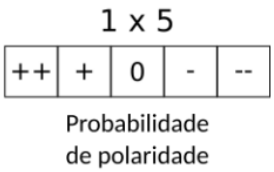


Figura 9- Representação da probabilidade de polaridade

Fonte: Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., (2013)

Cada fatia do tensor será multiplicada pela concatenação dos vetores-composição dos filhos do nó e pela matriz, gerando um valor numérico. A RNTN é composta por duas camadas, enquanto a análise de sentimentos é feita pelo classificador softmax aplicado aos vetores-composição da árvore de sentimentos, ou seja, o softmax escolhe a polaridade indicada.

Tabela 5-Exatidão para 5 classes e previsões binárias

Model	Fine-grained		Positive/Negative	
	All	Root	All	Root
NB	67.2	41.0	82.6	81.8
SVM	64.3	40.7	84.6	79.4
BiNB	71.0	41.9	82.7	83.1
VecAvg	73.3	32.7	85.1	80.1
RNN	79.0	43.2	86.1	82.4
MV-RNN	78.7	44.4	86.8	82.9
RNTN	**80.7**	**45.7**	**87.6**	**85.4**

Fonte: Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., (2013)

O desempenho ótimo foi alcançado para todos os modelos com tamanhos de vetores entre 25 e 35 dimensões e tamanhos de batch de 20 a 30. O desempenho decresceu nos tamanhos de vetores e batch maiores ou menores (Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C., 2013). Observou-se que o RNTN obteve 80.7% de exatidão na previsão de sentimentos fine-grained e assim foi mais preciso que os outros algoritmos que foram submetidos à mesma análise, como o NB, o RNN e o SVM.

2.4.2.1.5. *AugmentGAN*

O AugmentGAN, um simples, mas eficaz modelo de aumento de texto Generative Adversarial Network (GAN), que garante a coerência sintática nas amostras recém-geradas. Dada uma entrada com um rótulo, o AugmentGan pretende gerar uma sequência semântica semelhante que segue a estrutura sintática da amostra original.

Tabela 6- Exemplo de cenário

Input text	Label	Output text
I liked the movie.	Positive	*I loved the song.* *Movie was great!* *Its a beautiful day.*

Fonte: Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty, (2021)

O AugmentGAN baseia-se na GAN, composta por dois submódulos - um gerador e um discriminador. Dada a entrada rotulada, o gerador pretende produzir uma saída que pertença ao mesmo rótulo que a entrada (Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty;

2021). Pandey implementou este modelo para cinco tarefas de classificação: classificação de sentimento, reconhecimento de emoções, detetor de sarcasmo, classificação de intenções, classificações de spam, em doze datasets e três línguas.

O Gerador de Aumento utilizado é um modelo baseado no LSTM, que, para cada palavra, gera uma semelhante, que transmite a semântica original. O discriminador utilizado foi baseado no CNN que faz previsões sobre a originalidade dos dados de entrada e classifica se o texto é original ou não, ou seja, se é inventado pelo gerador ou é uma amostra do set de treino. Nesta investigação, foi implementado um módulo PoS-GAN que dá uma recompensa R*p* ao gerador para uma estrutura sequencial correta, caso contrário penaliza o gerador. AugmentGAN é modelado após SeqGAN, ou seja, o PoS-GAN funciona como um filtro para refinar a qualidade das amostras geradas. Usando o PoS-GAN, encorajou-se o gerador de amostras a gerar frases mais estruturadas (Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty; 2021)

Tabela 7-Resultados Estatísticos com vários sentimentos e datasets

	IMDB		SST		Alexa		Hindi-Movie				Bengali-Tweets	
Set	Pos	Neg	Pos	Neg	Pos	Neg	Pos	Neg	Neu	Con	Pos	Neg
Original	12.5K	12.5K	3896	3896	2320	200	700	424	500	160	2800	2800
Generated	10K	10K	5000	5000	1000	1500	200	500	500	800	4000	4000
Total	22.5K	22.5K	8896	8896	3320	1700	900	924	1000	960	6800	6800

(a)

	Sarcasm	
Set	Sar	N-Sar
Original	23.9K	20.2K
Generated	30K	25K
Total	53.9K	45.2K

(b)

	Ubuntu					Chat		Web							
Set	MU	SP	SC	SR	No	DT	FC	CP	DA	DV	ED	FS	FA	SA	No
Original	10	10	13	17	3	43	57	2	7	1	2	6	7	3	2
Generated	50	50	50	50	70	100	90	200	200	200	200	200	200	200	200
Total	60	60	63	67	73	143	147	202	207	201	202	206	207	203	202

(c)

	Mail Spam		SMS Spam	
Set	Spm	N-S	Spm	N-S
Original	1368	4362	453	2889
Generated	4000	1000	3000	1000
Total	5368	5362	3453	3889

(d)

	Emotion												
Set	EM	SD	EN	NE	WR	SR	LV	FN	HT	HP	BR	RL	AN
Original	659	4828	522	6340	7433	1613	2068	1088	1187	2986	157	1021	98
Generated	300	500	2000	300	200	3000	2000	2000	2000	2000	3000	3000	3000
Total	959	5328	2522	6640	7633	4613	4068	3088	3187	4986	3157	4021	3098

(e)

Fonte: Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty, (2021)

Neste trabalho foram utilizados os datasets:

A) Classificação de sentimentos: 1) IMDB: 50k Críticas de filmes 50k 2) Stanford Sentiment Treebank (SST): 7792 frases 3) Kaggle Amazon-Alexa1 com 3150 reviews, 2320 positivas e 200 negativas 4) Críticas de filmes Hindi: 200 positivas, 424 negativas, 500 neutros, e 160 conflituosos. 5) Bengali Sentiment Dataset:2800 positivas e 700 negativas.

B) Emotion Recognition: 1) Kaggle Emotion Detection: 13 emoções com 30k data points de treino. Legenda: Emoções vazias (EM), tristeza (SD), entusiasmo (EN), neutras (NE), preocupação (WR), surpresa (SR), amor (LV), divertimento (FN), ódio (HT), felicidade (HP), aborrecimento (BR), alívio (RL), e raiva (AN).

C) Detector de Sarcasmo: 1) Kaggle News Headline: 55 328 expressões de sarcasmo e não-sarcasmo no total. No set de treino não-sarcástico 23 976 pontos e 20 286 de sarcasmo.

D) Classificação de Spam: 1) Kaggle Mail: 5730 entidades, 1368 de spam e 4362 não spam. 2) Kaggle SMS: 4179 entidades (treino: 3342 e teste: 837). No set de treino, o spam foi de 453 e não-spam 2889.

E) Classificação de Intenção 1) AskUbuntu: Consiste em 5 intenções: Update (MU), Setup Impressora (SP), Desligar Computador (SC), Recomendação de software (SR), e nenhuma (No). O dataset contém 162 amostras (treino: 53 e teste: 109). 2) Chatbot: Duas intenções diferentes— Tempo de partida (DT) e Encontrar conexões (FCs), com o total de 206 questões (treino: 100 e teste: 106). 3) Aplicação Web: 89 amostras (treino: 30 e teste: 59) e 8 intenções—Mudar Password (CP), Apagar conta (DA), Download de video (DV), Exportar data (ED), Filtrar spam (FS), Encontrar alternativas (FAs), Sincronizar contas (SAs), e nenhuma (No) (Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty; 2021). O gerador computou a probabilidade de distribuição sobre o vocabulário disponível, na fase inicial do treino (∼ 2 epochs), com o vocabulário (dataset) completo no sistema. Nos epochs subsequentes, treinou-se na especificidade da legenda, para datasets específicos. O processo foi repetido para *n* iterações, sendo que *n* representa o número desejado de amostras a ser geradas para um dado dataset (Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty; 2021). Em cada epoch, se a perda de validação for a melhor até ao momento, o processo é guardar os parâmetros para posteriormente usar esse modelo no set de teste. Foram usados os seguintes modelos:

a) Naive Bayes: com máximo valor posterior sobre o vetor Tf-Idf.

b) CNN: usou-se 300-D GloVe pré-treinados e 100 filtros de convolução de tamanho 3, 4, and 5 g.

c) RNN: modelo com 300-D GloVe pré-treinados e dimensão escondida h = 256.

d) Bi-LSTM: duas LSTM de tamanho 256.

e) FastText: Usou-se o bag-of-2gram com o Glove de 300 dimensões.

f) Bi-GRU: usou-se o BERT com a dimensão de 256 (Suraj Pandey; Md. Shad Akhtar;Tanmoy Chakraborty; 2021).

As novas amostras geradas pelo AugmentGAN tiveram performances superiores em 6 modelos e classificação. Nas tarefas de análise de sentimentos, o AugmentGAN atingiu as melhores previsões com 92.8%, 91.3%, e 94.5% para IMDB, SST, e Alexa datasets, respetivamente. As limitações encontradas estão ligadas à necessidade de grande quantidade de treino.

2.5 Reflexão Crítica da Revisão da Literatura

A análise de polaridades de frases é uma tarefa complicada do ponto de vista computacional. Os algoritmos de ML tendem a encontrar a sensação de positividade através de análise de combinações de palavras.

Ao contrário da análise de sentimentos, não existem muitos estudos sobre a análise sentimental em texto, no contexto dos sistemas de apoio à tomada de decisão vocacional escolar. O trabalho de Socher et al (2013) trouxe uma enorme contribuição para o campo da análise de sentimentos, desenvolvido na Universidade de Stanford, através do Treebank de sentimentos e de um modelo de rede neuronal recursivo associado a um tensor, o RNTN. Este algoritmo de DP alcançou uma precisão de 80.7% na previsão de sentimentos em todos os nós, superando todos os outros algoritmos analisados. Para muitas tarefas de PLN, o aumento de texto oferece um excelente suporte na aprendizagem supervisionada, como visto no caso do modelo AugmentGAN que gera texto fabricado. Muitas das limitações encontradas nos algoritmos mencionados, estão relacionadas com três características básicas de um algoritmo de ML: (i) um menor custo computacional possível; (ii) a utilização de menos padrões de exemplo; (iii) e um menor envolvimento humano na construção do modelo antes do treino.[6] Na área do Deep Learning, apesar de terem definições de problemas simples, estas tarefas sempre foram desafiantes para um sistema computacional. Uma das principais razões é a falta de dados de treino suficientes que um sistema de aprendizagem profunda exige.

Conclui-se que esta área é importante, tendo já sido desenvolvida muita investigação na área da análise sentimental de textos, mas há falta de investigação e desenvolvimento de tecnologia ao nível de aplicações que usem a IA na área da tomada de decisão, e especificamente na área da escolha vocacional.

O desenvolvimento de investigações com datasets na língua portuguesa iria com certeza ser de um contributo enorme na investigação e abrir caminhos para outras investigações, além de ajudar os investigadores a terem bases sólidas para apresentar casos de teste na nossa língua. As Redes Neuronais de Aprendizagem Profunda (DLNN) permitem uma maior capacidade de aprendizagem e seria importante a existência de maior investigação nesta área aplicada à análise de sentimentos em texto. A investigação está em aberto na área da análise de sentimentos aplicada a data sets de estudantes de 9º ano com a utilização de algoritmos de ML e/ou DL.

2.6 Tecnologias a serem utilizadas no modelo:

- Linguagem Python;
- Bibliotecas de pré-processamento de texto;
- Bibliotecas de ML como a scikit-learn e a TensorFlow;
- Ferramentas de análise de sentimento;
- Base de dados com os dados obtidos nas respostas do questionário;
- Algoritmos de ML/DL como a regressão linear, árvores de decisão, Naïve Bayes, K-NN, MLP, entre outros;
- Ferramentas de visualização de dados;
- Aplicação Web demonstrativa da aplicabilidade do modelo.

3 MÉTODO

3.1 Metodologias

A metodologia a ser utilizada para a produção da Dissertação envolve as componentes seguintes:

a. A revisão de literatura;

b. Construção de questionários, aplicação e análise dos mesmos a alunos de 9º ano;

c. Construção de Léxicos;

d. Pré-processamento de dados;

e. Treino de algoritmos de ML e DL;

f. Construção de modelo de IA aplicável à problemática;

g. Validação e análise dos resultados obtidos, com conjunto de teste.

h. Construção de aplicação Web demonstrativa da aplicabilidade do modelo.

No diagrama abaixo está representada a metodologia a ser seguida para a obtenção de resultados.

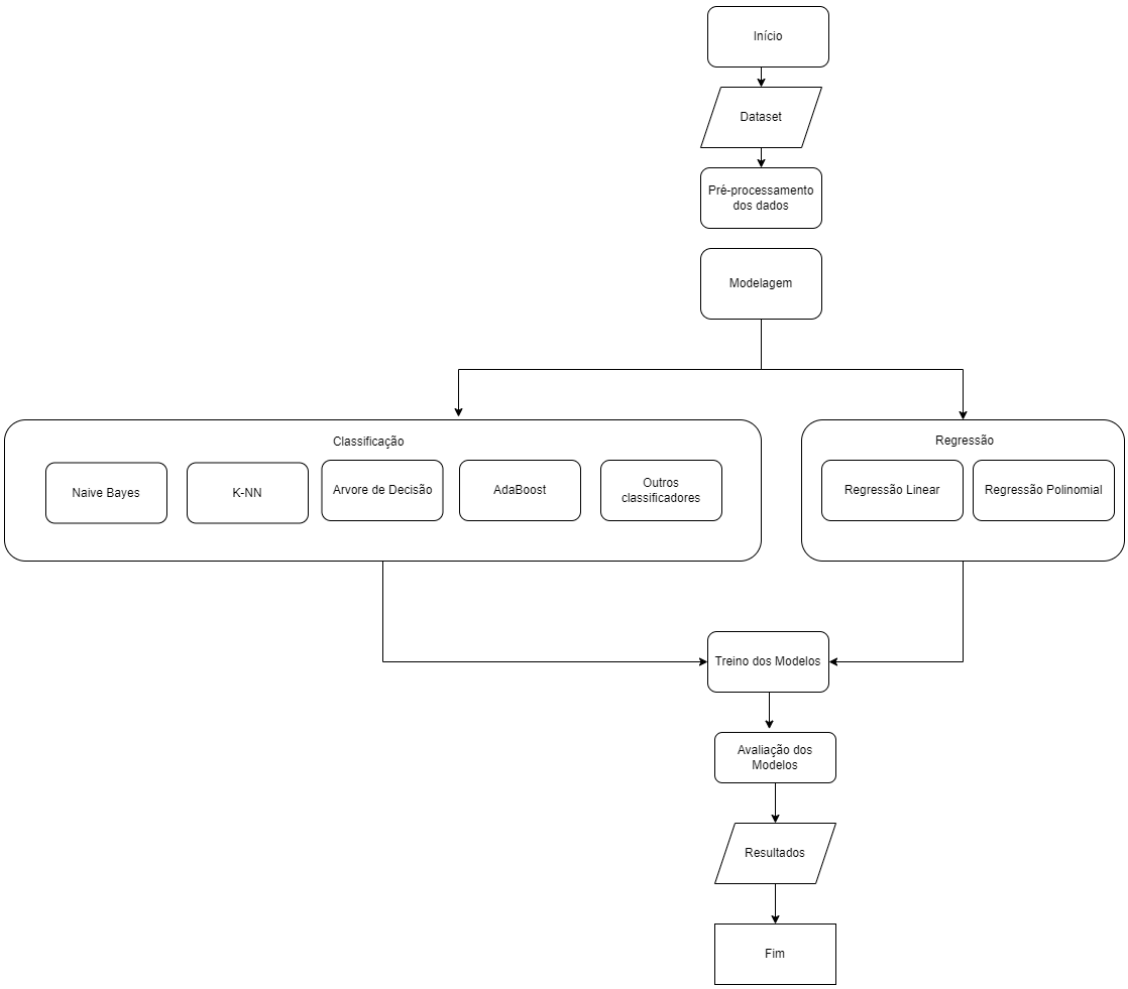


Figura 10-Diagrama de Fluxo para obtenção de resultados, autoria própria

O questionário foi criado no *google forms* com 20 perguntas, sendo 18 abertas, e 2 fechadas, sendo estas, na sua maioria, de resposta longa, e pode ser acedido na hiperligação: https://forms.gle/hDGPDZryePw8mrVY6 . Antes do envio do questionário para as Escolas teve de passar pelo processo de pedido de aprovação ao DGE/MIME (Direção Geral da Educação/ Monitorização de Inquéritos em Meio Escolar), que demorou cerca de um mês a ser aprovado, obtendo o número de registo: 1283900001. Após aprovação do Inquérito, foram realizados contatos a cerca de 60 Agrupamentos de Escolas (AE) a nível nacional, no sentido de aplicação do meu questionário aos alunos do 9º ano, tendo obtido respostas de três agrupamentos, nomeadamente, o AE Eduardo Gageiro, Sacavém, o AE Marinha Grande Poente e o AE de Coruche, estando estes três AE integrados num TEIP (Territórios Educativos de Intervenção Prioritária). De acordo com o "Programa TEIP 2017-2021:

Territórios Educativos de Intervenção Prioritária", o objetivo do programa é "a promoção do sucesso educativo e da inclusão escolar, através da implementação de medidas e estratégias que contribuam para a melhoria da qualidade do ensino e das aprendizagens, a valorização das escolas e a promoção da equidade e da coesão social" (Ministério da Educação, 2017). (ver Anexo E).

3.2 Caracterização das Escolas

3.2.1 Agrupamento de Escolas Eduardo Gageiro

O AE Eduardo Gageiro (AEEG) situa-se na zona oriental do Concelho de Loures, distrito de Lisboa, marcada pela localização do rio Trancão e serve na sua maioria, habitantes da União de Freguesias de Sacavém e Prior Velho. As escolas do AEEG inserem-se numa área urbanizada, sendo um dormitório da Área Metropolitana de Lisboa, com bairros recentes destinados à classe média, zonas de habitação social e algumas áreas degradadas das antigas vilas industriais.[4] Segundo o Plano Plurianual de Melhorias do AEEG 2018-2021 (PPM), a população, é oriunda, na sua maioria, de países africanos de língua portuguesa (PALOP), e foi realojada na Urbanização Terraços da Ponte, sendo as suas gerações descendentes ainda representantes dessa realidade. O Agrupamento é constituído por sete escolas, sendo a Escola Secundária de Sacavém, com 3º ciclo e Ensino Secundário, a escola sede.

Figura 11- Escola Secundária de Sacavém

[4] Retirado de: https://www.cm-loures.pt/Ligacao.aspx?DisplayId=88

O AEEG integra um TEIP, tem 9 turmas de 9º ano no ano letivo de 2022/2023 e, além dos cursos regulares de ensino secundário, tem como oferta educativa no presente ano letivo, os seguintes cursos profissionais na Escola Secundária de Sacavém:

- Técnico Comercial;
- Técnico de Design de Comunicação Gráfica;
- Técnico de Gestão e Programação e Sistemas Informáticos;
- Técnico de Instalações Elétricas;
- Técnico de Turismo.

3.2.2. Agrupamento de Escolas Marinha Grande Poente

O AE Marinha Grande Poente (AEMGP) está localizado na cidade da Marinha Grande, que se situa na zona do Pinhal Litoral, a cerca de 12 quilómetros a oeste da capital do distrito, Leiria. Segundo o Plano Plurianual de Melhorias do AEMGP 2018-2021 (PPM), a comunidade escolar deste AE é cada vez mais multicultural, havendo um número crescente e significativo de alunos, originários de vinte e seis países estrangeiros: indianos, paquistaneses, cidadãos do Leste, brasileiros e cidadãos do Médio Oriente. Fazem parte deste AE dez estabelecimentos de ensino, sendo a sede a Escola Secundária Eng.º Acácio Calazans Duarte (3ºciclo e ensino secundário).

Figura 12- Escola Secundária Eng Acácio Calazans Duarte, fonte:
http://mrg.pt/uploads/portfolio/imagens/big_portfolio_1456489931_2013.jpg

O AEMGP integra um TEIP, e no ano letivo de 2022/2023, além dos cursos regulares de ensino secundário, tem como oferta educativa no presente ano letivo, os seguintes cursos profissionais na Escola Secundária Eng.º Acácio Calazans Duarte (ESEACD):

- Técnico Assistente Dentário;
- Técnico Auxiliar de Farmácia;
- Técnico Cozinha/Pastelaria;
- Técnico de Ação Educativa;
- Técnico de Auxiliar Protésico Dentário;
- Técnico de Design Industrial;
- Técnico de Massagem de Estética e Bem-Estar;
- Técnico de Multimédia;
- Técnico de Planeamento Industrial;
- Técnico de Programação e Maquinação CNC;
- Técnico Recursos Florestais e Ambientais.

3.2.3 Agrupamento de Escolas de Coruche

Segundo o Plano Plurianual de Melhorias do Agrupamento de Escolas de Coruche 2018-2022 (PPM), o agrupamento está localizado no concelho de Coruche, distrito de Santarém, sendo cerca de 80% da população do concelho residente em pequenos aglomerados com menos de 2 mil habitantes, tendo a vila de Coruche cerca de 3 mil e agrega todas as escolas do concelho de Coruche: Escola Secundária com 3º CEB de Coruche, escola sede do Agrupamento, Escola EB 2,3 Dr. Armando Lizardo, EBIJI do Couço e restantes Escolas Básicas e Jardins de Infância do concelho, o que significa um Agrupamento com uma alta dispersão geográfica, com 16 estabelecimentos de ensino.

Figura 13-Escola Secundária de Coruche, fonte: https://maisribatejo.pt/wp-content/uploads/2023/02/escola-secundaria-coruche.jpg

O Agrupamento de Escolas de Coruche integra um TEIP, e no ano letivo de 2022/2023, tem como oferta educativa para o ensino secundário os quatro cursos regulares científico-humanísticos, não tendo este ano letivo oferta de cursos profissionais.

3.3.Áreas de Ensino

As quatro grandes áreas do ensino regular do Secundário são:

Cursos cientifico-humanísticos:

- Ciências e Tecnologias.
- Ciências Socioeconómicas.
- Línguas e Humanidades.
- Artes Visuais.

Cursos Profissionais:

Tabela 8- Tabela de Áreas de Educação e Formação

Áreas sectoriais (Conselhos sectoriais para a Qualificação)	Áreas de Educação e Formação	Exemplos de cursos nível 4:
Agroalimentar	541 - Indústrias Alimentares	Técnico de Controlo de qualidade Alimentar, Técnico de Produção Agropecuária; Técnico de Máquinas Florestais, entre outros
	621 - Produção Agrícola e Animal	
	622 - Floricultura e Jardinagem	
	623 - Silvicultura e Caça	
Artesanato e Ourivesaria	215 – Artesanato	Técnico de Vidro Artístico; Técnico de Pintura Decorativa, entre outros
Comércio e Marketing	341 – Comércio	Técnico comercial; Técnico de vendas; Técnico de organização de eventos, entre outros
	342 - Marketing e Publicidade	
	544 - Indústrias Extrativas	

Construção civil e Urbanismo	582 - Construção Civil e Engenharia Civil	Técnico de Medições e Orçamentos; técnico de topografia;
Cultura, Património e Produção de Conteúdos	212 - Artes do Espetáculo	Técnico de Produção e Tecnologias da Música; Técnico de Multimédia; Intérprete/ator/atriz; Assistente de Arqueólogo, entre outros
	213 - Audiovisuais e Produção dos Media	
	225 - História e Arqueologia	
	322 - Biblioteconomia, Arquivo e Documentação	
Defesa e Segurança	861 - Proteção de Pessoas e Bens	Técnico de proteção civil; Bombeiro, entre outros
Economia do Mar	624 – Pescas	Técnico de aquicultura, entre outros
Energia e Ambiente	522 - Eletricidade e Energia	Técnico Instalador de Sistemas Eólicos; Técnico de Redes Elétricas; Técnico de Eletrotecnia, entre outros
	850 - Proteção do Ambiente - Programas Transversais	
Indústrias Químicas, Cerâmica, Vidro e Outras	524 - Tecnologia dos Processos Químicos	Técnico de Análise Laboratorial; Técnico de Química Industrial, entre outros
	543 - Materiais (Indústrias da Madeira, Cortiça, Papel, Plástico, Vidro e Outro	
Informática, Eletrónica e Telecomunicações	481 - Ciências Informáticas	Técnico de Eletrónica e Telecomunicações; Técnico de Mecatrónica; Programador de Informática; Técnico de Informática-Sistemas, entre outros
	523 - Eletrónica e Automação	
Madeiras, Mobiliário e Cortiça	543 - Materiais (Indústrias da Madeira, Cortiça, Papel, Plástico, Vidro e Outro	Técnico de Desenho de Mobiliário e Construções em Madeira; Técnico de Cerâmica, entre outros
Metalurgia e Metalomecânica	521 - Metalurgia e Metalomecânica	Técnico de CAD/CAM; Técnico de Soldadura, entre outros
Moda	542 - Indústrias do Têxtil, Vestuário, Calçado e Couro	Técnico de Design de Moda; Técnico de Tecelagem, entre outros
Saúde e Serviços à Comunidade	724 - Ciências Dentárias	Técnico Auxiliar de Saúde; Técnico de Ação Educativa; Animador
	725 - Tecnologias de Diagnóstico e Terapêutica	

	727 - Ciências Farmacêuticas	Sociocultural; Técnico de Geriatria, entre outros
	729 - Saúde - Programas não Classificados Noutra Área de Formação	
	761 - Serviços de Apoio a Crianças e Jovens	
	762 - Trabalho Social e Orientação	
	861 - Proteção de Pessoas e Bens	
Serviços às Empresas	343 - Finanças, Banca e Seguros	Técnico Administrativo; Técnico de Apoio à Gestão; Técnico de Segurança no Trabalho, entre outros
	344 - Contabilidade e Fiscalidade	
	345 - Gestão e Administração	
	346 - Secretariado e Trabalho Administrativo	
	347 - Enquadramento na Organização/Empresa	
	380 – Direito	
	862 - Segurança e Higiene no Trabalho	
Serviços Pessoais	814 - Serviços Domésticos	Esteticista; Cabeleireiro, entre outros
	815 - Cuidados de Beleza	
Transportes e Logística	525 - Construção e Reparação de Veículos a Motor	Técnico de Produção Automóvel; Técnico de Gestão de Transportes, entre outros
	840 - Serviços de Transporte	
Turismo e Lazer	812 - Turismo e Lazer	Técnico de Restaurante/Bar; Técnico de Informação e Animação Turística; Técnico de Desporto, Cozinheiro/a, entre outros
	813 – Desporto	

Fonte: adaptado de https://catalogo.anqep.gov.pt/conselhosSectoriais

3.4 Arquitetura da solução:

Possível arquitetura para a solução:

1. Recolha de dados: A primeira etapa é recolher as respostas dos questionários respondidos pelos alunos e armazená-los numa base de dados (Excel).
2. Pré-processamento de dados: Posteriormente, os dados precisam ser pré-processados para remover as informações irrelevantes e transformar o texto num formato que possa ser processado por algoritmos de ML e DL. Assim, é possível extrair as palavras relevantes dos questionários, remover as stopwords, fazer a tokenização, ou seja, dividir as frases em palavras e convertê-las em vetores para que possam ser alimentadas no modelo de IA e aplicar técnicas de normalização de texto.
3. Análise de sentimentos: Após o pré-processamento dos dados, o passo a seguir é a análise de sentimentos, através de algoritmos de ML e DL como a regressão logística, árvores de decisão, Naïve Bayes, SVM e Redes Neuronais. A análise de sentimentos poderá ser simples, como uma classificação binária de positivo/negativo, ou complexa, como a análise de polaridade. Estes algoritmos são treinados com um conjunto de dados de referência, para que possam classificar automaticamente o sentimento dos textos.
4. Visualização dos resultados: Depois de analisar os sentimentos, será possível visualizar os resultados por meio de gráficos e tabelas, que irão mostrar a distribuição do sentimento em relação às perguntas do questionário, que permitirá identificar os padrões e nas respostas dos alunos.
5. Feedback para as escolas: Por fim, a solução poderá fornecer feedback para escolas, por exemplo, apresentando sugestões de áreas que possam ser mais adequadas às vocações dos alunos com base nas respostas dadas no questionário e apresentação de resultados gerais. Como o questionário elaborado é anónimo, não permitirá obter-se resultados individuais, contudo o modelo poderá ser usado numa outra investigação a nível individual.

3.5 Levantamento e análise de Requisitos

O tipo de questionário a aplicar é de perguntas abertas (ver Anexo E) com perguntas que pretendem extrair informação relacionada com áreas vocacionais, cujos dados serão armazenados num ficheiro em Excel. Foi elaborada uma declaração de consentimento

informado para os encarregados de educação dos alunos poderem assinar e concordar com o uso dos dados na investigação. O questionário foi submetido para análise da Direção-Geral de Estatísticas da Educação e Ciência no site http://mime.dgeec.mec.pt/, visto ser necessário a sua autorização para aplicação em meio escolar. Será feita a análise de sentimento positivo/negativo e de polaridade e feita a métrica que será usada para avaliar a eficácia da solução. Poderá verificar-se a existência de dados ambíguos ou incompletos, o que causará limitações ao estudo.

Requisitos de privacidade e segurança

As informações serão apenas utilizadas neste estudo e poderão ser apagadas e alteradas consoante a vontade dos Encarregados de Educação dos alunos, aspeto que está referido no consentimento informado.

3.6 Fundamentação para elaboração das perguntas do questionário

Foi importante definir quais as variáveis de interesse abordadas no questionário. Escolheu-se o tipo de perguntas abertas relevantes que possam transmitir sentimentos para obter palavras que possam ser analisadas com análise de sentimento textual. É importante a definição de um conjunto de categorias de análise que possam ser usadas para classificar as respostas dos alunos, como por exemplo, "positivo", "negativo" e "neutro" e, em seguida, usar essas categorias para classificar as respostas dos alunos. A escolha das perguntas abertas foi fundamentada em diversas fontes, tais como literatura científica, especialistas em orientação vocacional e consulta a professores e orientador da Dissertação. Foram ainda revistas as competências necessárias para as profissões mais comuns ou desejadas pelos alunos.

3.7 Construção de Léxico

Um léxico é uma lista de palavras usadas num contexto específico. Considera-se alguns exemplos de palavras que podem ser incluídos num léxico para determinar área em perguntas específicas de área vocacional: "não gosto nada", “não gosto muito”, “desnecessário”, "desinteressante", "complicada", "Ainda não sei", “gosto”, “gostaria”, “importante”, “bom”, “muito bom”, “adorava”, “boa opção”, “favorita”, e algumas palavras que poderão ser

incluídas em perguntas genéricas, para determinar a área, ou seja palavras representantes de áreas vocacionais: Por exemplo para a área de Ciências e Tecnologias: lexico_ciencias_tecnologias = ["tecnologia", "computador", "veterinária", "matemática", "programar", "computadores", "informática", "médico", "cirurgia", "medicina", "saúde", "biologia", "ciências", "engenharia", "arquitetura"] e para a área de Artes e Visuais: lexico_artes_visuais_1 = ["artes", "desenho", "cores", "texturas", "desenhar", "moldes", "pintura", "arquitetura", "criatividade", "escultura", "geometria", "fotografia", "design", "multimédia"]. O trabalho de construir os léxicos para este problema envolve muita pesquisa e tempo, pelo que é um trabalho moroso, que envolve também a criação de um léxico para cada disciplina, para obter melhores resultados na pergunta acerca da disciplina favorita.

3.8 Treino de Algoritmos

Bag of words

No processamento de linguagem natural, a representação "bag of words" é uma forma de representar texto como dados numéricos que envolve a criação de um vocabulário de todas as palavras que ocorrem num determinado conjunto de documentos de texto e, em seguida, codificar esses documentos como vetores numéricos, com cada elemento do vetor indicando a presença ou ausência de uma palavra particular no vocabulário. O bag of words não leva em consideração a ordem em que as palavras aparecem no documento. Em vez disso, apenas olha para a presença ou ausência de cada palavra, tratando o documento como um saco de palavras que pode ser contado e comparada com outros documentos e assim permite que os dados de texto sejam processados e analisados usando técnicas numéricas, como algoritmos de ML, mas não captura o significado ou contexto do texto no documento.

3.9 Processo de análise de texto com ML/DL:

1. Recolher o conjunto de dados obtidos nos questionários;
2. Pré-processamento dos dados: inclui tarefas como remover a pontuação, remover palavras sem significado (como "o" e "a"), e converter todas as palavras para minúsculas.

3. Criar modelos de ML/DL: Existem vários algoritmos que podem ser usados para análise de texto, como o *Naïve Bayes (NB),* a *Árvore de Decisão,* o *K-Nearest Neighbors* (K-NN*),* Redes Neuronais, Regressão Linear, entre outros.
4. Treinar os modelos: Depois de criar os modelos, é necessário treiná-los utilizando o conjunto de dados pré-processado.
5. Testar o modelo: Depois de treinar o modelo, é importante testá-lo para ver como ele se comporta com novos dados, fornecendo o modelo com um conjunto de dados de teste e comparando as previsões com as categorias conhecidas.
6. Utilizar o modelo: Depois de treinar e testar o modelo, começa-se a utilizá-lo para realizar análises de texto, que poderá incluir novas previsões com outros dados.

4. RESULTADOS

No presente capítulo segue-se a apresentação dos resultados obtidos a partir da análise dos questionários aplicados aos alunos de 9º ano. Nesta investigação, tendo em conta o público-alvo, algumas respostam das perguntas dos questionários, revelaram-se ambíguas, incompletas e difíceis de analisar através de ML, contudo, penso relevante a apresentação destes resultados.

4.1 Caracterização do dataset

Esta investigação usou um Dataset composto por respostas a 20 perguntas realizadas através de questionário a alunos de 9º ano de três AE do país, nomeadamente, o AE Eduardo Gageiro, o AE de Marinha Grande Poente e o AE de Coruche. Foram realizados contactos com cerca de 60 Agrupamentos de Escolas (AE) a nível nacional, no entanto, obtive respostas de 267 alunos dos 3 agrupamentos mencionados.

N	O
8. Qual é a tua disciplina preferida?	9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve.
Nenhuma	eu não quero seguir essas áreas mas acho que é interessante
Ciências Naturais	eu acho essa área um bocadinho chata
História	O que eu acho é que são matérias difíceis e muito complicadas
Geografia	Não gosto muito porque e muito sobre português e não é o meu plano
Português	Acho interessante
Educação Física	Eu não gosto muito dessas áreas porque não entendo muito
Português	gosto mas é muito dificil
Educação Física	acho uma boa profissao ate porque irei ter que ir para essa profissao ou irei seguir
História, Francês.	acho bom .
Português	eu gosto dessa areas, até porque eu gostaria de fazer uma dessas areas , são areas que ajudam a ter mais conhecimentos , de integracã
Inglês	turismo,porque voce pode ver o mundo
Educação Visual , Educação Física	acho interessante podes aprender varias línguas
Eu diria que nenhuma, a escola me traz muito descontentan	Acho legais, umas são mais importantes que outras, mas todas tem seu valor, eu entendo quem gosta e ou trabalha com essas áreas, mi
Educação Física	Gosto de turismo, porque dá-me uma sensação de que se conhece melhor o mundo e há mais liberdade
Matemáticas e Ciências Naturais	gosto de linguas para poder viajar e conhecer paises e culturas. historia dos paises portugal para turismo
nenhuma	não acho nada, não uma coisa que me interessa
Inglê	nao sei
Matemática	n sei nada disso me interessa
nao tenho disciplina preferida	acho normal algumas corresponde as aulas na escola
não tenho	sao boas
n sei	n sei
não tenho	nsei
TIC	a minha atividade e línguas porque eu gosto de aprender línguas novas
nenhuma	não acho nada sou faço por que sou obrigada pela a escola
História	Eu acho bom
História e Inglês	eu vou tentar entrar para línguas e humanidades então acho interessante
Ciências Naturais	so gosto do turismo
Inglês	acho interessante, turismo é a área que pretendo seguir
Educação Visual	Acho bom , não sei .
Educação Física.	Gosto
Matemática	Não me dão muito interesse

Figura 14- Amostra do *dataset*

4.1.1 Análise Estatística

Das 267 respostas ao questionário, 146 são de alunos da Escola Secundária de Sacavém, AE Eduardo Gageiro (9 turmas), 77 da Escola Secundária Eng.º Acácio Calazans Duarte, AE de Marinha Grande Poente (5 turmas) e 44 da Escola Secundária de Coruche, AE de Coruche (4 turmas), sendo 137 do sexo Feminino (51%), 122 do sexo Masculino (46%) e 8 (3%) selecionaram a opção ‘Prefiro não responder’.

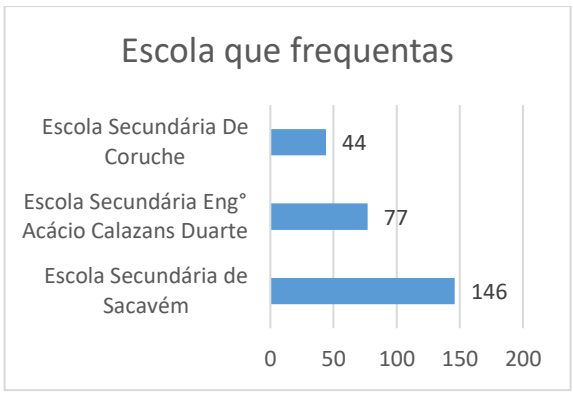


Figura 15-Gráfico com o número de alunos por escola

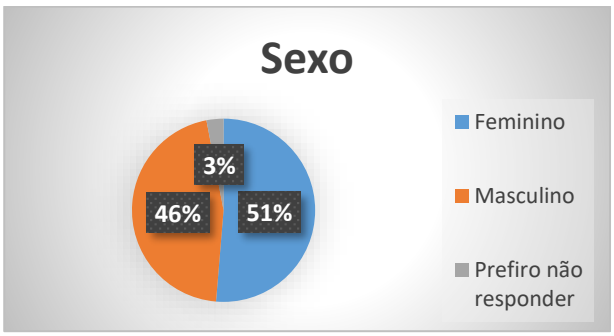


Figura 16- Gráfico com percentagem de alunos por sexo

Tabela 9- Sexo dos participantes

	Sexo
Feminino	137
Masculino	122
Prefiro não responder	8

Como podemos observar na tabela 9, responderam ao questionário, 137 participantes do sexo feminino, 122 do sexo masculino e 8 assinalaram a opção 'Prefiro não responder'. A idade dos participantes varia entre os 14 e os 19, conforme a tabela 10 e a figura 17, sendo indicativa do número elevado de alunos com mais de 14/15 anos:

Tabela 10- Idade dos participantes

	Que idade tens?
15	123
14	82
16	42
17	15
18	4
19	1

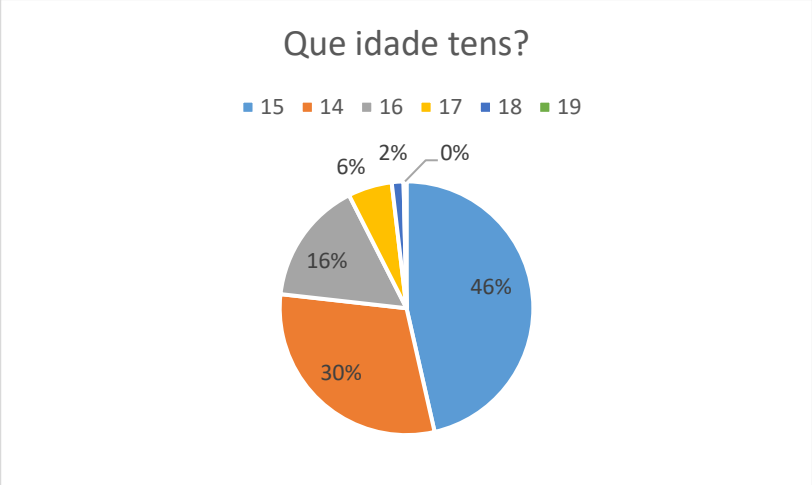


Figura 17 - Percentagem das idades dos participantes

4.1.2 Análise da Pergunta com o número 15

A pergunta '15. Gostavas de tirar um curso superior? Porquê? Qual seria?' foi analisada à parte, visto não ter relevância para o âmbito do estudo, contudo é interessante em termos estatísticos, para averiguar apetência para seguir um curso superior. (Anexo I) O script criado para análise da Pergunta com o número 15 lê o ficheiro CSV, categoriza as respostas e fornece a contagem de respostas positivas, neutras, negativas e não categorizáveis por idade, sexo e escola, além de plotar as imagens e guardar os resultados em Excel. No script foram também definidos léxicos para respostas positivas, neutras e negativas. Podemos observar os resultados por sexo, escola e idade, nas figuras 18, 19 e 20.

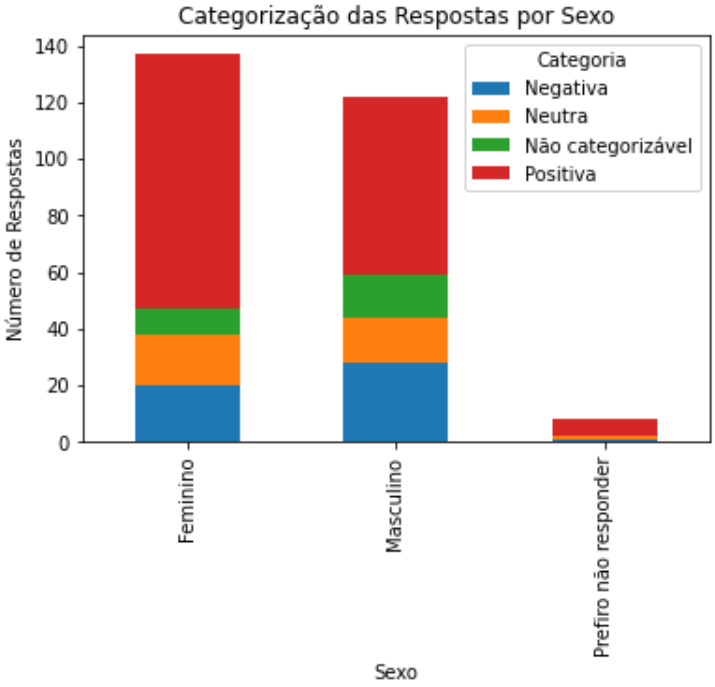


Figura 18- Categorização das respostas à pergunta 15 por sexo

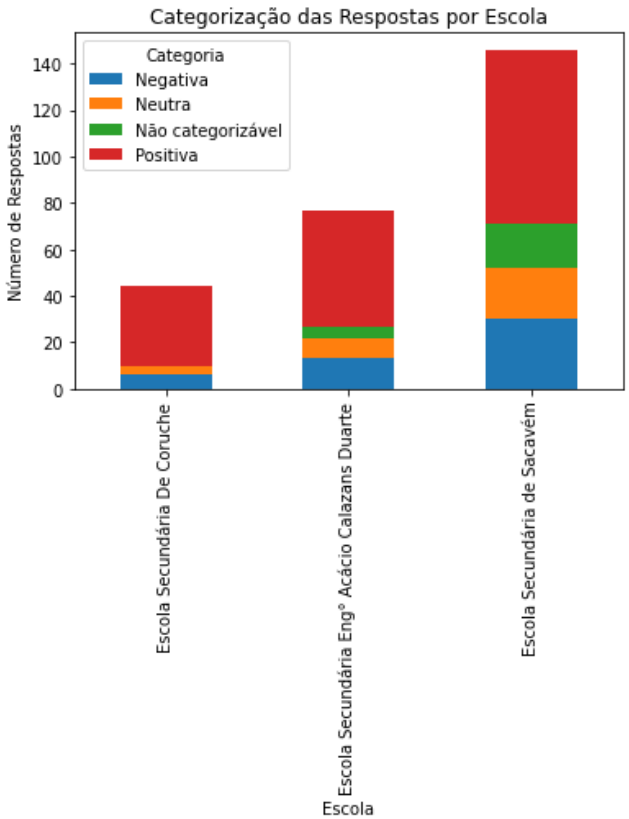


Figura 19- Categorização das respostas à pergunta 15 por escola

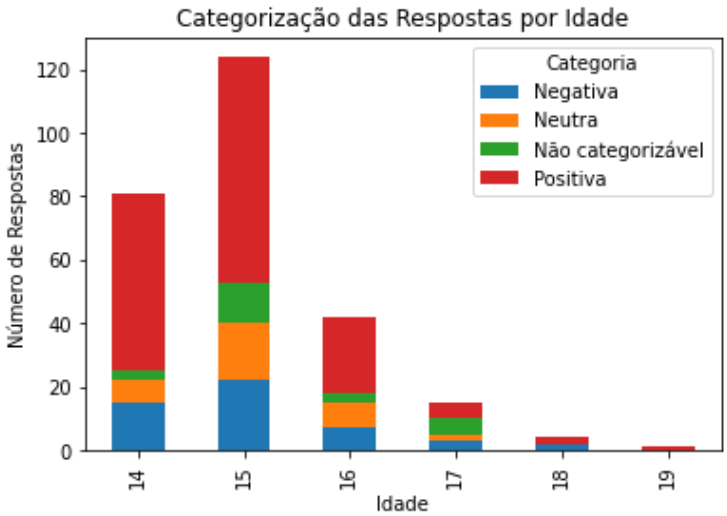


Figura 20- Categorização das respostas à pergunta 15 por idade

Tabela 11- Resultados gerais da pergunta 15

	Categoria
Positiva	159
Negativa	49
Neutra	35
Não categorizável	24

Podemos observar que de acordo com a tabela 11, resultante do script para a pergunta com o número 15, que 159 alunos obtiveram resultado positivo na hipótese de ingressar num curso superior, 49 alunos obtiveram um resultado negativo e 35 alunos obtiveram um resultado neutro e 24 obtiveram um resultado não categorizável.

Tabela 12- Resultados das respostas à pergunta 15 por sexo

Sexo	Negativa	Neutra	Não categorizável	Positiva
Feminino	20	18	9	90
Masculino	28	16	15	63
Prefiro não responder	1	1		6

De acordo com a tabela 12, os participantes do sexo feminino obtiveram o maior número de resultados positivos à pergunta com o número 15, ou seja, vontade de seguir um curso superior, com um total de 90 resultados positivos.

Tabela 13- Resultados da pergunta 15 por escola

Escola que frequentas	Negativa	Neutra	Não categorizável	Positiva
Escola Secundária De Coruche	6	4		34
Escola Secundária Eng° Acácio Calazans Duarte	13	9	5	50
Escola Secundária de Sacavém	30	22	19	75

Na tabela 13, podemos observar que o maior número de alunos que pretende ir para o ensino superior é na Escola Secundária de Sacavém, contudo, há que ter em conta que nessa escola o número de participantes foi muito superior às outras duas escolas. No entanto podemos analisar que em todas as escolas o número de registos positivos é superior aos negativos, pelo que é um indicador de apetência para seguir os estudos a nível superior.

Tabela 14- Resultados da pergunta 15 por idade

Que idade tens?	Negativa	Neutra	Não categorizável	Positiva
14	15	7	3	56
15	22	18	13	71
16	7	8	3	24
17	3	2	5	5
18	2			2
19				1

Em relação ao fator idade, podemos observar que os alunos com 15 anos obtiveram o maior número de resultados positivos na categoria de estudar no ensino superior.

4.2 Pré-Processamento dos Dados

O pré-processamento dos dados envolveu a recolha, limpeza dos dados, remoção de stopwords, stemming, importação de bibliotecas, lematização, tokenização e a normalização, guardados num ficheiro csv. Por exemplo, alguns alunos eram oriundos de países de língua não portuguesa e houve o cuidado de tradução dessas respostas para obtenção de um ficheiro normalizado, ou ainda colocar as disciplinas escritas de várias formas, como por exemplo ‘EV, ev, Ed. Visual’ passarem a ser ‘Educação Visual’. Dividi as perguntas em dois ficheiros python separados, visto que as perguntas com número do 1 ao 8 foram analisadas com léxicos criados por mim, visto serem perguntas não específicas de uma área vocacional e as perguntas do número 9 ao 14 serem específicas da área. A pergunta com o número 15 é

vocacionada para o ensino superior foi analisada à parte, conforme demostrado anteriormente. Foram criados léxicos de uma palavra, duas palavras e três palavras de modo a avaliar-se a conotação das palavras nos diferentes casos. Utilizei para esta investigação as seguintes áreas vocacionais: as 4 do ensino regular (Ciências e Tecnologias, Ciências Socioeconómicas, Línguas e Humanidades, Artes Visuais) e escolhi 3 mais abrangentes do ensino profissional (Profissional de informática eletrónica e comunicações, Profissional turismo e lazer, Profissional de cultura, património e gestão de conteúdos), sendo no total 7 áreas vocacionais. Criei um ficheiro para os léxicos, onde se pode observar os vários léxicos, (Anexo J) como o léxico por área, léxicos de várias palavras, léxico de disciplina, etc, sendo essencial para analisar as perguntas de 1 a 8 analisadas em outro ficheiro, importando o mesmo. Além de análise com léxicos e a métrica, analisei também as perguntas com os números 2, 3 e 5, colunas 7, 8 e 10 do ficheiro dos dados, com análise de sentimento através do *Valence Aware Dictionary and Sentiment Reasoner* (VADER), e da métrica personalizada, visto que estas perguntas induzem um sentimento, sendo as perguntas: '2. Qual é a profissão da tua mãe e planeias seguir o mesmo caminho? Porquê?', '3. Qual é a profissão do teu pai e planeias seguir o mesmo caminho? Porquê?', '5. Que profissão é que os teus pais gostariam que tu tivesses? Concordas com eles? Porquê?'. Contudo não incluí estes resultados na análise, visto a maior parte dos sentimentos obtidos serem negativos, e para o que se pretendia com o ficheiro, averiguar a área vocacional com base em léxicos, ficaram somente guardados no ficheiro de Excel resultante. (Anexo K) Como se pode observar, as funções 'calculate_custom_score' e 'calculate_vader_score', para calcular o score da métrica e o score do VADER, só se aplicaram às perguntas das colunas 7, 8 e 10. Neste ficheiro temos ainda o dicionário de valores por área que atribui um valor numérico a cada área para ser usado por classificadores posteriormente:

```
valores_por_area = {
    "Ciências e Tecnologias": 1,
    "Ciências Socioeconómicas": 2,
    "Línguas e Humanidades": 3,
    "Artes Visuais": 4,
    "Profissional informatica eletronica e comunicacoes": 5,
    "Profissional turismo e lazer": 6,
    "Profissional cultura, patrimonio e gestão de conteudos": 7,
    "Área não encontrada": 0
}
```

Em relação aos resultados do número de alunos por escola e área vocacional, verificou-se que na Escola Secundária de Coruche 3 alunos obtiveram a área de Artes Visuais, 1 aluno a área de Ciências Socioeconómicas, 26 a área de Ciências e Tecnologias e 14 área não encontrada. Na Escola Secundária de Sacavém, 5 alunos obtiveram a área de Artes Visuais, 11 a área de Ciências Socioeconómicas, 64 a área de Ciências e Tecnologias, 6 a área de Línguas e Humanidades e 60 área não encontrada. Na Escola Secundária Engº Acácio Calazans Duarte, 2 alunos obtiveram a área de Artes Visuais, 37 a área de Ciências e Tecnologias, 2 a área de Línguas e Humanidades e 36 obtiveram área não encontrada. (Anexo AB) Verificou-se ainda que na Escola Secundária de Coruche o número de alunos do sexo feminino que obteve a área vocacional de Ciências e Tecnologias é superior ao sexo masculino, bem como na Escola Secundária Engº Acácio Calazans Duarte e na Escola Secundária de Sacavém, o número de alunos que obteve a área de Ciências e Tecnologias é muito idêntico, em relação ao sexo. (Anexo AC) Criei ainda uma tabela com os resultados com o número de alunos por área vocacional, sexo e idade, que pode ser verificada no Anexo AD.

As principais funcionalidades do código de ana´lise de sentimento são a importação das bibliotecas pandas, nltk, lexicos2, carregamento dos léxicos para análise de conotação de palavras positivas e negativas para a língua portuguesa a partir do nltk opinion lexicon, pré-processamento, através de etapas como a conversão do texto para minúsculas, remoção da pontuação, remoção de palavras irrelevantes (stopwords), leitura e processamento dos dados, análise de conotação, através de funções para avaliação da conotação das palavras em diferentes áreas, análise de sentimento nas colunas das perguntas com os números de 9 a 14, print dos resultados e armazenamento dos resultados num ficheiro Excel. (Anexo L) Deste modo através de léxicos neutros, pode-se obter resultados para as perguntas específicas das áreas. Através de funções obteve-se a conversão das respostas em scores, o cálculo da média dos sentimentos do VADER e da métrica personalizada, em 6 colunas, o cálculo da área mais comum, o seu valor numérico, a contagem da área mais comum, entre outros.

Este ficheiro também inclui o mapeamento das perguntas por áreas correspondentes em que a pergunta ‘9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve’ corresponde às áreas de "Línguas e Humanidades" e de "Profissional turismo e lazer", a pergunta ‘10. O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve’ corresponde à área de "Ciências Socioeconómicas", a pergunta ‘11. O que achas da área das

ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve' corresponde às áreas de "Ciências e Tecnologias" e de "Profissional informatica eletronica e comunicacoes", a pergunta '12. O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve' corresponde às áreas de "Artes Visuais" e de "Profissional cultura, patrimonio e gestão de conteudos", a pergunta '13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê?' corresponde às áreas de "Profissional informatica eletronica e comunicacoes, de "Profissional cultura, patrimonio e gestão de conteudos" e de "Profissional turismo e lazer", a pergunta '14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve' corresponde à área de "Profissional turismo e lazer".

Nos resultados das tabelas dos anexos AE e AF, podemos verificar que há um número significativo de resultados com valor 0, ou seja área não encontrada, resultados esses que podem melhorar com a modelagem de texto.

4.3 Modelagem de Texto

Para a criação de modelos foi necessário a preparação dos dados, a escolha dos algoritmos, a divisão dos dados em conjuntos de treino e teste, treino e avaliação dos modelos. A avaliação dos modelos utilizou métricas de avaliação como a exatidão, precisão, recall e F1-score, foi implementado ajuste de hiperparâmetros para encontrar a melhor combinação para cada algoritmo e realizou-se uma comparação dos modelos utilizados.

4.3.1 Classificação

A classificação é usada quando a variável de destino (saída) é discreta e categórica, ou seja, consiste em classes ou rótulos. O objetivo da classificação é atribuir um rótulo ou classe a um objeto com base nas suas características. As métricas de avaliação comuns para problemas de classificação incluem a AUC-ROC (Área sob a Curva da Característica de Operação do Receptor), a precisão, a sensibilidade (recall), o F1-score, entre outras, que medem a qualidade das previsões em termos de verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos.

4.3.1.1 Classificador Naive Bayes Gaussiano

Os procedimentos no uso do classificador NB Gaussiano foram o carregamento dos dados, a preparação dos dados relacionados com os scores, a divisão em treino e testes, inicialização do classificador, importação de bibliotecas, treino, previsão, mapeamento, cálculo das métricas de avaliação, matriz de confusão, previsões, curva ROC, plotagem de gráfico e armazenamento dos resultados num ficheiro Excel. Dei um peso igual aos scores da métrica personalizada e VADER, através da sua média, sendo o x essas 6 variáveis, as colunas de scores combinados das perguntas com os números de 9 a 14 e o y foi a coluna da área do valor numérico do ficheiro de análise das perguntas com os números de 1 a 8. (Anexo M) Como muitos valores obtidos no x foram 0, optei por mapear os valores negativos para (-1) e os positivos (incluindo o 0), para 1, mantendo a direção do sentimento. Esta abordagem mantém a direção do sentimento e evita divisões por 0, fornecendo resultados mais precisos do que a divisão por valor absoluto.

Os resultados com o classificador NB foram os seguintes:

Accuracy: 0.43, Precision: 0.53, Recall: 0.43, F1-Score: 0.38

Confusion Matrix - Naive Bayes:

[[14 8 0 0 0]
[13 9 0 0 0]
[3 1 0 0 0]
[2 2 0 0 0]
[0 2 0 0 0]]

Class 'Ciências e Tecnologias': TP = 9, FN = 13, FP = 13

Class 'Ciências Socioeconómicas': TP = 0, FN = 4, FP = 0

Class 'Línguas e Humanidades': TP = 0, FN = 4, FP = 0

Class 'Artes Visuais': TP = 0, FN = 2, FP = 0

Class 'Profissional informatica eletronica e comunicacoes': TP = 0, FN = 0, FP = 0

Class 'Profissional turismo e lazer': TP = 0, FN = 0, FP = 0

Class 'Profissional cultura, patrimonio e gestão de conteudos': TP = 0, FN = 0, FP = 0

Class 'Área não encontrada': TP = 14, FN = 8, FP = 18

O NB Gaussiano obteve o valor de 0.43 de exatidão, ou seja, acertou 43% das classificações, 0.53% de precisão, ou seja, das instâncias classificadas como positivas, 53% eram realmente positivas, o recall, também chamado de sensibilidade foi de 0.42,

ou seja, o classificador capturou cerca de 43% dos exemplos verdadeiramente positivos, o F1-score combina a precisão com o recall e é útil quando existe desequilíbrio de classes, foi de 0.38.

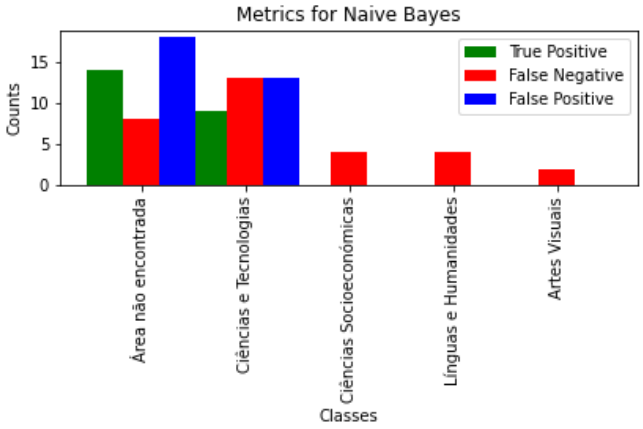


Figura 21- Métricas para o Naive Bayes

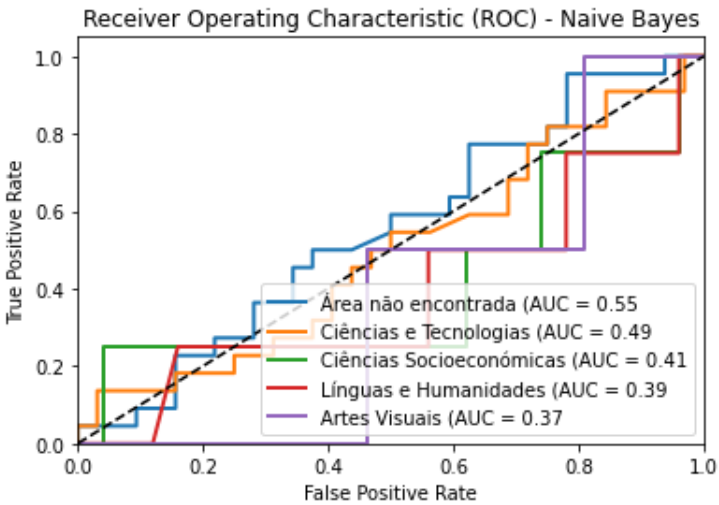


Figura 22- Curva ROC -Naive Bayes

Na Matriz de confusão os verdadeiros positivos (TP) significam o número de exemplos que foram corretamente classificados como positivos. Ou seja, a previsão é positiva e a classe real também é positiva, os falsos negativos (FN) significam o número de exemplos que foram erradamente classificados como negativos, ou seja a previsão é negativa, enquanto a classe real é positiva. Os falsos positivos (FP) são os exemplos que foram classificados erradamente como positivos, a previsão é positiva, mas a classe real

é negativa e os verdadeiros negativos (TN) referem-se aos exemplos que foram corretamente classificados como negativos, a previsão é negativa, bem como a classe real.

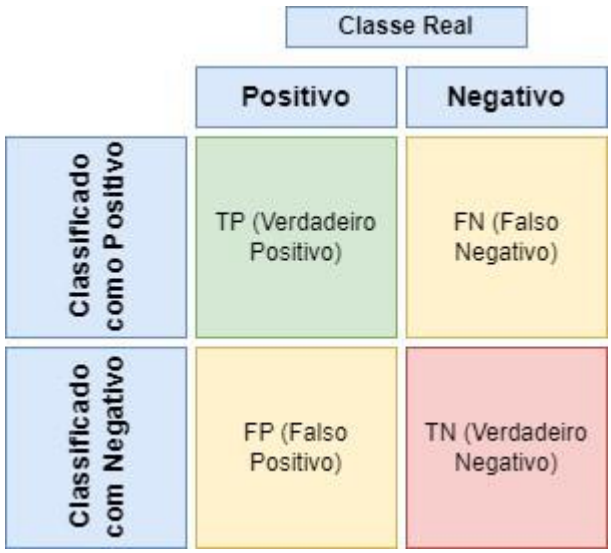


Figura 23- Matriz de Confusão, autoria própria

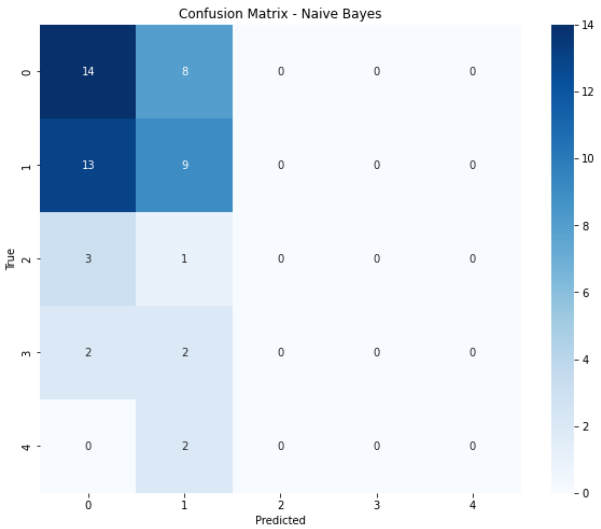


Figura 24- Matriz de confusão NB

Podemos observar por exemplo, na classe 'Ciências e Tecnologias', o classificador teve 9 verdadeiros positivos (acertos), 13 falsos negativos (instâncias que pertenciam à classe, mas foram classificadas como negativas) e 13 falsos positivos (instâncias que não pertenciam à classe, mas foram classificadas como positivas).

4.3.1.2. Classificador Decision Tree

Os procedimentos no uso do classificador Decision Tree (Árvore de Decisão) foram o carregamento dos dados, a preparação dos dados relacionados com os scores, importação de bibliotecas, a divisão em treino e testes, inicialização do classificador, cross-validation (validação cruzada), treino, previsão, mapeamento, cálculo das métricas de avaliação, matriz de confusão, curva ROC, AUC, previsões, plotagem de gráfico e armazenamento dos resultados num ficheiro Excel. Dei um peso igual aos scores da métrica personalizada e VADER, através da sua média, sendo o x essas 6 variáveis, as colunas de scores combinados das perguntas com os números de 9 a 14 e o y foi a coluna da área do valor numérico do ficheiro de análise das perguntas com os números de 1 a 8. Com a validação cruzada divide-se o conjunto de dados em 5 folds e treina-se e avalia-se o modelo em cada fold, armazenando-se as métricas para cada fold e assim obtém-se as pontuações (scores) para cada fold, bem como a média e o desvio padrão dos scores. (Anexo M) O código criado para classificar com uma árvore encontra-se no Anexo N.

Os resultados para este classificador foram os seguintes:

Accuracy: 0.44
Precision: 0.48
Recall: 0.44
F1-Score: 0.42
Fold 1: Accuracy = 0.44
Fold 2: Accuracy = 0.39
Fold 3: Accuracy = 0.42
Fold 4: Accuracy = 0.34
Fold 5: Accuracy = 0.40
Média da Exatidão: 0.40
Desvio Padrão da Exatidão: 0.03
Confusion Matrix - Decision Tree:
[[5 17 0 0 0]
 [6 16 0 0 0]
 [0 4 0 0 0]
 [1 3 0 0 0]
 [0 2 0 0 0]]
Class 'Ciências e Tecnologias': TP = 16, FN = 6, FP = 26
Class 'Ciências Socioeconómicas': TP = 0, FN = 4, FP = 0

Class 'Línguas e Humanidades': TP = 0, FN = 4, FP = 0
Class 'Artes Visuais': TP = 0, FN = 2, FP = 0
Class 'Profissional informatica eletronica e comunicacoes': TP = 0, FN = 0, FP = 0
Class 'Profissional turismo e lazer': TP = 0, FN = 0, FP = 0
Class 'Profissional cultura, patrimonio e gestão de conteudos': TP = 0, FN = 0, FP = 0
Class 'Área não encontrada': TP = 5, FN = 17, FP = 7
AUC for class 'Área não encontrada': 0.46
AUC for class 'Ciências e Tecnologias': 0.50
AUC for class 'Línguas e Humanidades': 0.50
AUC for class 'Ciências Socioeconómicas': 0.50
No positive or only positive examples for class 'Artes Visuais'

Em cada fold da validação cruzada, as métricas de exatidão foram calculadas e os valores variam de 0.34 a 0.44, mostrando como o desempenho do modelo varia em diferentes divisões dos dados, sendo a média da exatidão em todos os folds de aproximadamente 0.40, com um desvio padrão de cerca de 0.03. A matriz de confusão mostra os resultados detalhados da classificação para cada classe, como por exemplo, na classe 'Ciências e Tecnologias', houve 16 verdadeiros positivos (TP), 6 falsos negativos (FN) e 26 falsos positivos (FP). A classe 'Artes Visuais' não teve nenhum exemplo positivo detetado pelo modelo (TP = 0), e não houve erros de classificação para as classes 'Ciências Socioeconómicas', 'Línguas e Humanidades', 'Profissional informatica eletronica e comunicacoes', 'Profissional turismo e lazer', 'Profissional cultura, patrimonio e gestão de conteudos'.

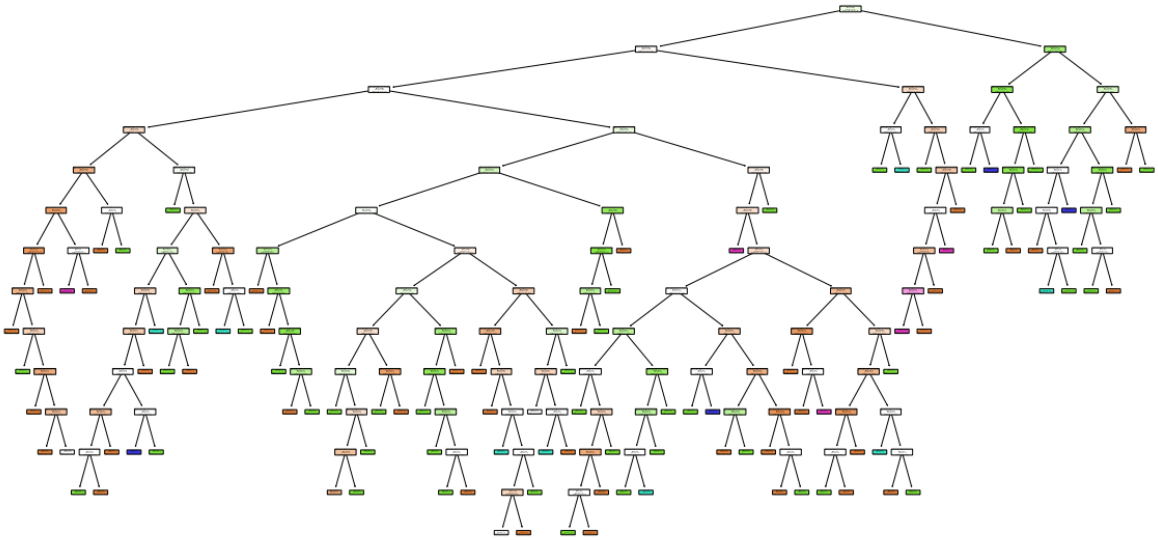


Figura 25- Árvore de Decisão

Alterando a linha de código abaixo obtém-se uma árvore mais rasa, com menos ramos, e mais legível, com menos profundidade, sendo que a legibilidade da árvore melhora:

```
# Inicializar o Decision Tree classifier com max_depth menor
tree_classifier = DecisionTreeClassifier(max_depth=3)
```

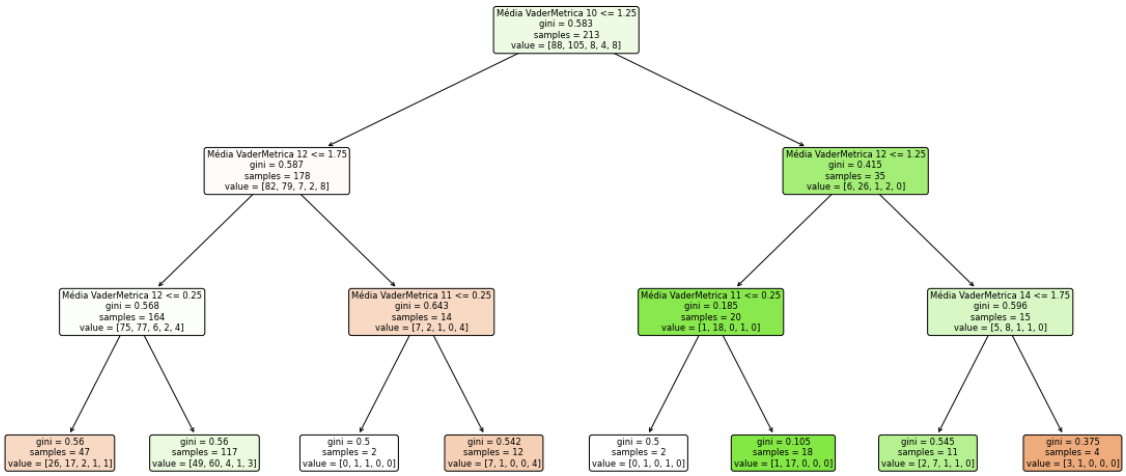


Figura 26- Árvore de Decisão depth 3

Quando se põe menos níveis na árvore, corre-se o risco de baixar a precisão, reduzindo a profundidade, visto que, árvores mais rasas são menos capazes de capturar relações complexas nos dados. No entanto, ao reduzir a profundidade também poderá ajudar a reduzir o overfitting, tornando o modelo mais generalizável para novos dados.

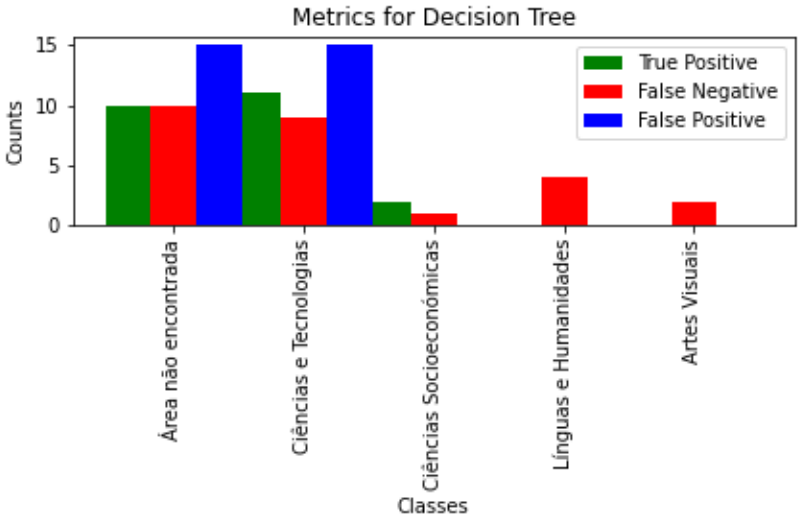


Figura 27- Métricas para o Decision Tree

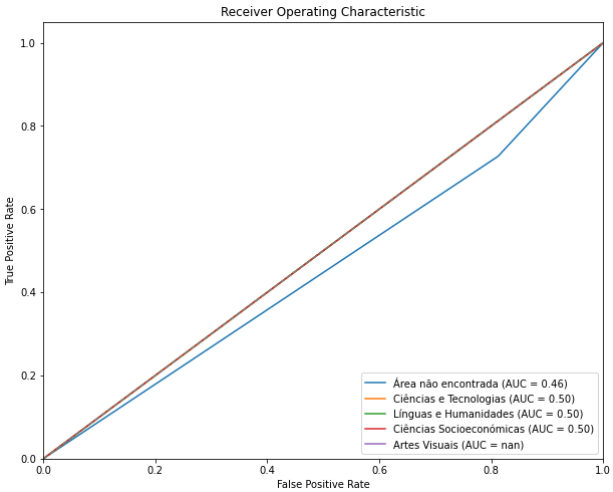


Figura 28-Curva ROC da Decision Tree

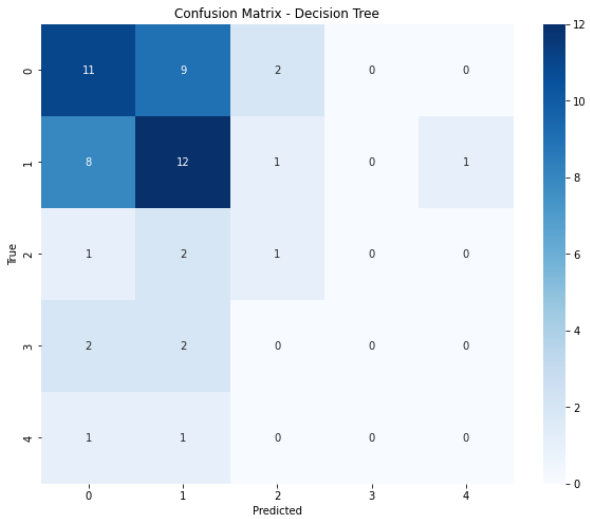


Figura 29-Matriz de confusão da Decision Tree

A AUC é calculada para cada classe e representa a área sob a curva da curva ROC (Receiver Operating Characteristic) para essa classe. Ela mede a capacidade do modelo de distinguir entre classes positivas e negativas. Por exemplo, a AUC para a classe 'Ciências e Tecnologias' é de 0.50. A métrica AUC para a classe 'Artes Visuais' não foi calculada porque não houve exemplos positivos detetados pelo modelo para essa classe (TP = 0). Isso pode indicar que essa classe possui desafios específicos de classificação.

4.3.1.3 Classificador Random Forest

Os procedimentos no uso do classificador Random Forest foram o carregamento dos dados, a preparação dos dados relacionados com os scores, importação de bibliotecas, a divisão em treino e testes, inicialização do classificador, treino, previsão, mapeamento, cálculo das métricas de avaliação, matriz de confusão, curva ROC, AUC, previsões, plotagem de gráfico e armazenamento dos resultados num ficheiro Excel. Dei um peso igual aos scores da métrica personalizada e VADER, através da sua média, sendo o x essas 6 variáveis, as colunas de scores combinados das perguntas com os números de 9 a 14 e o y foi a coluna da área do valor numérico do ficheiro de análise das perguntas com os números de 1 a 8. (Anexo N) O código para este classificador encontra-se no Anexo O.

Os resultados obtidos foram os seguintes:

Accuracy: 0.54

Precision: 0.62

Recall: 0.54

F1-Score: 0.50

Confusion Matrix - Random Forest:

[[12 10 0 0 0]

 [6 16 0 0 0]

 [0 3 1 0 0]

 [2 2 0 0 0]

 [1 1 0 0 0]]

Class 'Ciências e Tecnologias': TP = 16, FN = 6, FP = 16

Class 'Ciências Socioeconómicas': TP = 1, FN = 3, FP = 0

Class 'Línguas e Humanidades': TP = 0, FN = 4, FP = 0

Class 'Artes Visuais': TP = 0, FN = 2, FP = 0

Class 'Profissional informatica eletronica e comunicacoes': TP = 0, FN = 0, FP = 0

Class 'Profissional turismo e lazer': TP = 0, FN = 0, FP = 0

Class 'Profissional cultura, patrimonio e gestão de conteudos': TP = 0, FN = 0, FP = 0

Class 'Área não encontrada': TP = 12, FN = 10, FP = 9

AUC for class 'Área não encontrada': 0.60

AUC for class 'Ciências e Tecnologias': 0.55

AUC for class 'Línguas e Humanidades': 0.46

AUC for class 'Ciências Socioeconómicas': 0.54

AUC for class 'Artes Visuais': 0.17

O valor de 0.54 de exatidão significa que o modelo classificou corretamente cerca de 54% de todas as instâncias no conjunto de dado e 0.62 de precisão representa a proporção de exemplos classificados como positivos que eram realmente positivos. Um F1-Score de 0.50 indica um equilíbrio entre precisão e recall.

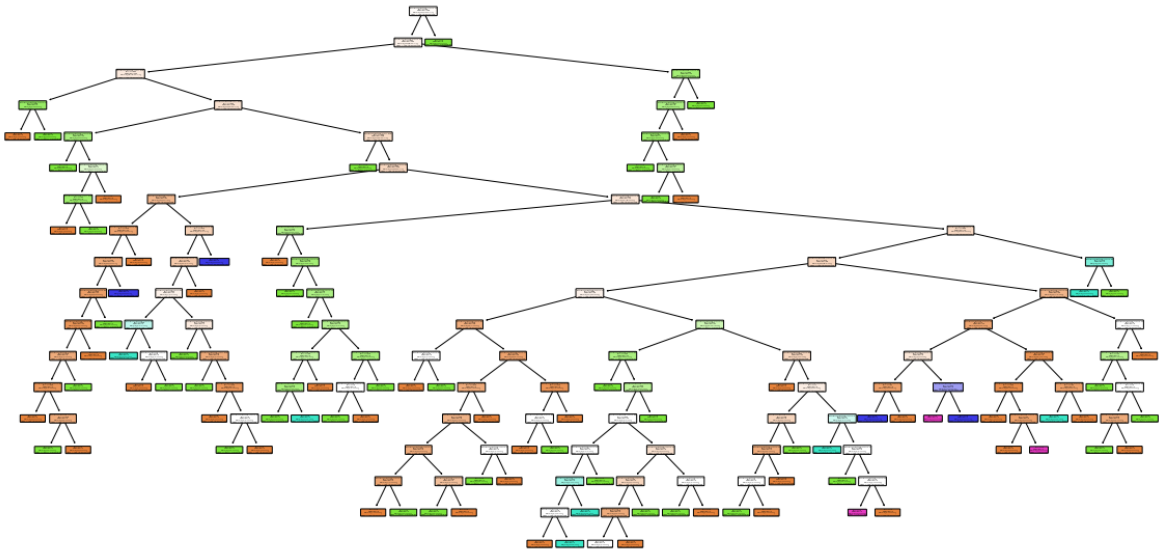


Figura 30- Random Forest

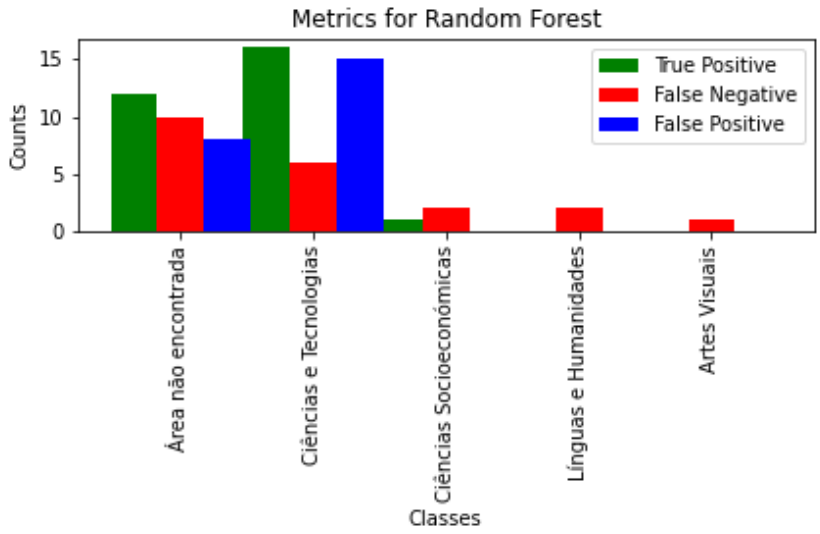


Figura 31- Métricas para o Random Forest

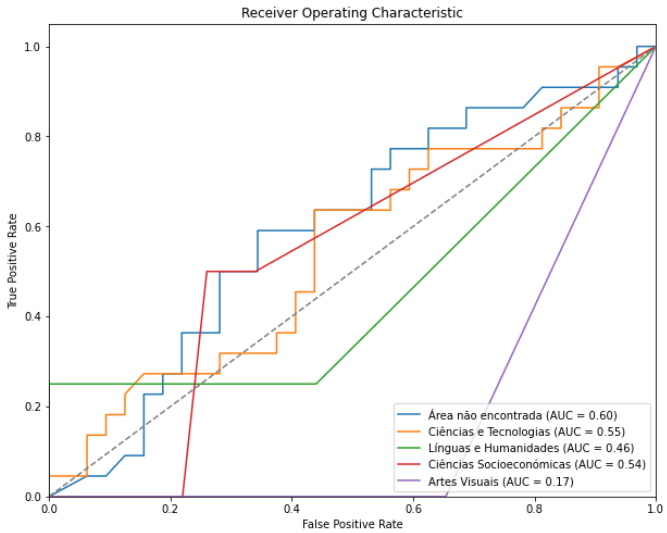


Figura 32-Curva ROC da Random Forest

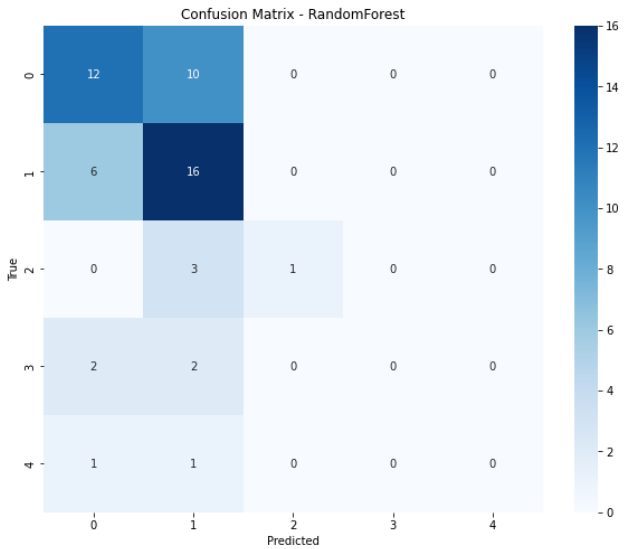


Figura 33- Matriz de confusão da Random Forest

A matriz de confusão mostra os resultados detalhados da classificação para cada classe, como por exemplo, na classe 'Ciências e Tecnologias', houve 16 verdadeiros positivos (TP), 6 falsos negativos (FN) e 16 falsos positivos (FP). A classe 'Artes Visuais' não teve nenhum exemplo positivo detetado pelo modelo (TP = 0), e não houve erros de classificação para as classes 'Ciências Socioeconómicas', 'Línguas e Humanidades', 'Profissional informatica eletronica e comunicacoes', 'Profissional turismo e lazer', 'Profissional cultura, patrimonio e gestão de conteudos'.

4.3.1.4 Classificador K-NN

Para este classificador desenvolvi um script em python que realiza a classificação usando o algoritmo K-Nearest Neighbors (KNN) com diferentes valores de k (vizinhos mais próximos) e avalia o desempenho do modelo usando várias métricas. O script carrega os dados x das colunas com os resultados da análise de sentimento do ficheiro Excel resultante da análise das perguntas com os números de 9 a 14 “Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14" e o y "Área mais frequente Valor Numérico" da coluna do ficheiro Excel resultante do ficheiro de análise das perguntas com os números de 1 a 8. Como se pode observar no código, (Anexo P) os dados são divididos em conjuntos de treino e teste usando train_test_split, e o código cria um loop que itera por diferentes valores de k (1, 3, 5, 7) para o classificador K-NN. Para cada valor de k, ele faz o seguinte: Inicializa o classificador KNN com o valor atual de k, treina o classificador com os dados de treino, realiza previsões nos dados de teste, calcula e imprime as métricas de desempenho, incluindo exatidão, precisão, recall e F1-Score, calcula e plota as curvas ROC para cada classe e calcula a AUC para cada classe. Cria uma matriz de confusão e a exibe como um mapa de calor, calcula e imprime os valores de verdadeiros positivos (TP), falsos negativos (FN) e falsos positivos (FP) para cada classe. Depois de completar o loop para todos os valores de k, o código concatena os resultados das previsões com os dados originais num dataframe e guarda os resultados em Excel.

Os resultados obtidos para o K-NN foram os seguintes:

Para K=1:

Accuracy (K=1): 0.44

Precision (K=1): 0.48

Recall (K=1): 0.44

F1-Score (K=1): 0.42

Confusion Matrix - KNN (K=1):

[[9 11 1 0 1]

 [8 14 0 0 0]

 [0 3 1 0 0]

[1 2 0 0 1]

[1 0 1 0 0]]

Class 'Área não encontrada' - K=1: TP = 9, FN = 13, FP = 10

Class 'Ciências e Tecnologias' - K=1: TP = 14, FN = 8, FP = 16

Class 'Ciências Socioeconómicas' - K=1: TP = 1, FN = 3, FP = 2

Class 'Línguas e Humanidades' - K=1: TP = 0, FN = 4, FP = 0

Class 'Artes Visuais' - K=1: TP = 0, FN = 2, FP = 2

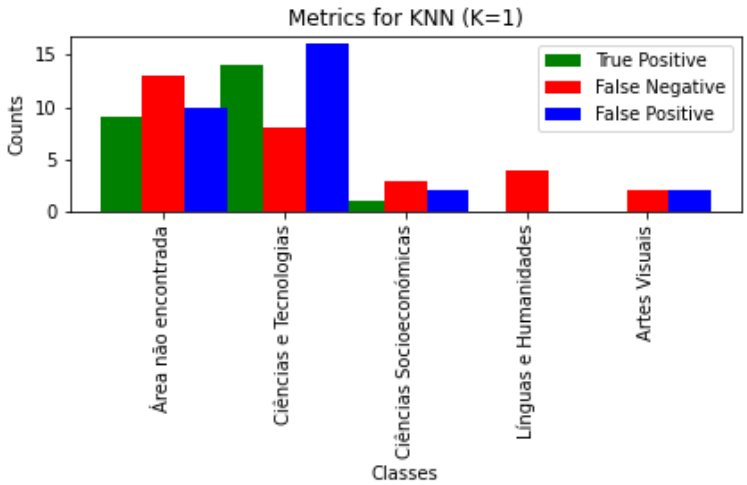


Figura 34- Métricas K=1

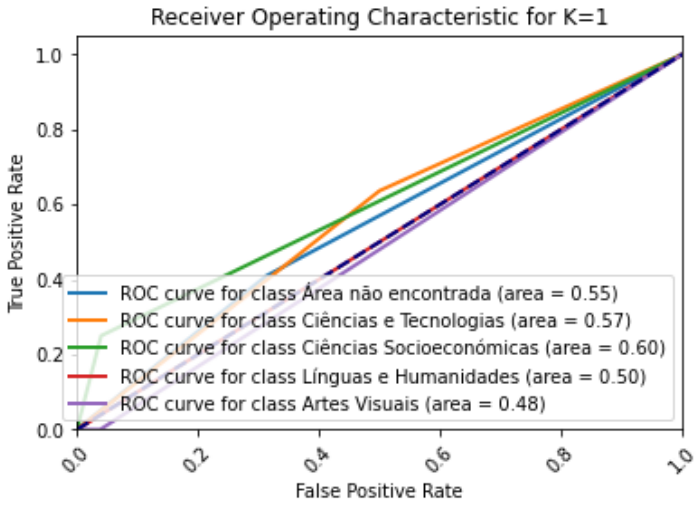


Figura 35- Curva ROC do K-NN(K=1)

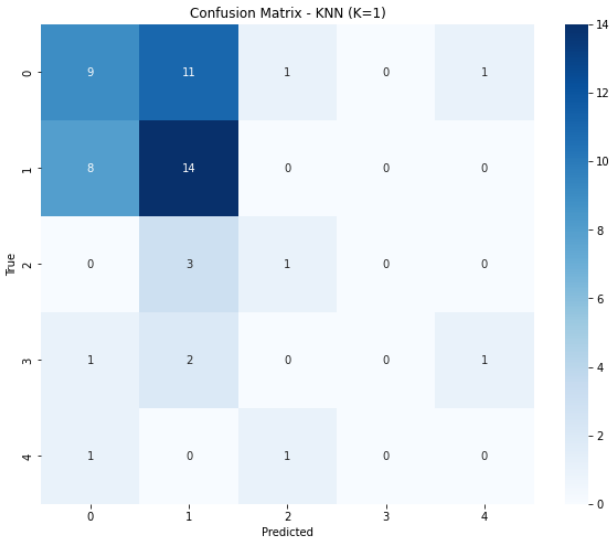


Figura 36- Matriz de confusão do K-NN(K=1)

Pode-se observar, por exemplo, que o modelo errou em 13 exemplos (falsos negativos - FN) e classificou incorretamente 10 exemplos como falsos positivos (FP).

Para K=3:

Accuracy (K=3): 0.44

Precision (K=3): 0.55

Recall (K=3): 0.44

F1-Score (K=3): 0.40

Confusion Matrix - KNN (K=3):

[[11 11 0 0 0]

 [9 13 0 0 0]

 [3 1 0 0 0]

 [2 2 0 0 0]

 [0 2 0 0 0]]

Class 'Área não encontrada' - K=3: TP = 11, FN = 11, FP = 14

Class 'Ciências e Tecnologias' - K=3: TP = 13, FN = 9, FP = 16

Class 'Ciências Socioeconómicas' - K=3: TP = 0, FN = 4, FP = 0

Class 'Línguas e Humanidades' - K=3: TP = 0, FN = 4, FP = 0

Class 'Artes Visuais' - K=3: TP = 0, FN = 2, FP = 0

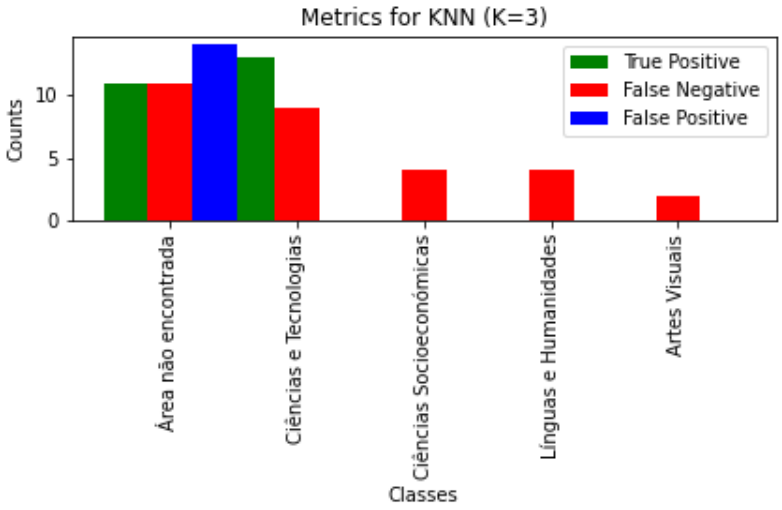


Figura 37- Métricas K=3

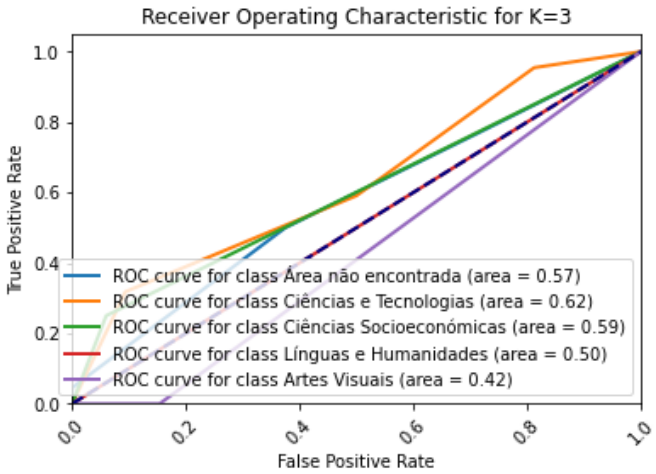


Figura 38-Curva ROC do K-NN(K=3)

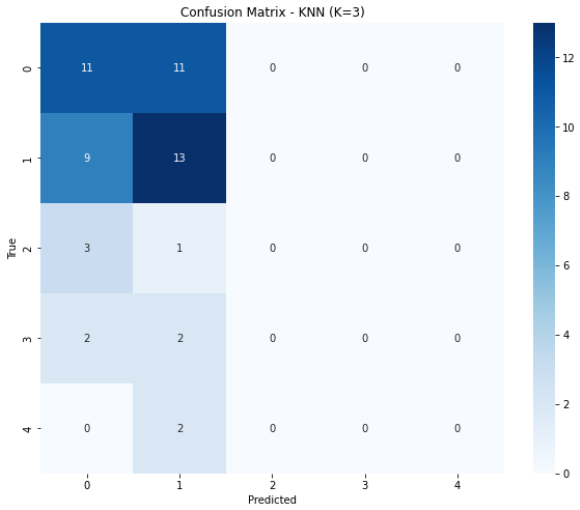


Figura 39- Matriz de confusão (K=3)

Com k=3, os resultados são semelhantes aos do k=1 em termos de exatidão e recall, mas a precisão é ligeiramente melhor (55%), o que significa que, ao considerar três vizinhos mais próximos, o modelo tem uma precisão maior nas suas previsões positivas.

Para K=5:

Accuracy (K=5): 0.48

Precision (K=5): 0.58

Recall (K=5): 0.48

F1-Score (K=5): 0.43

Confusion Matrix - KNN (K=5):

[[12 10 0 0 0]

[8 14 0 0 0]

[3 1 0 0 0]

[2 2 0 0 0]

[0 2 0 0 0]]

Class 'Área não encontrada' - K=5: TP = 12, FN = 10, FP = 13

Class 'Ciências e Tecnologias' - K=5: TP = 14, FN = 8, FP = 15

Class 'Ciências Socioeconómicas' - K=5: TP = 0, FN = 4, FP = 0

Class 'Línguas e Humanidades' - K=5: TP = 0, FN = 4, FP = 0

Class 'Artes Visuais' - K=5: TP = 0, FN = 2, FP = 0

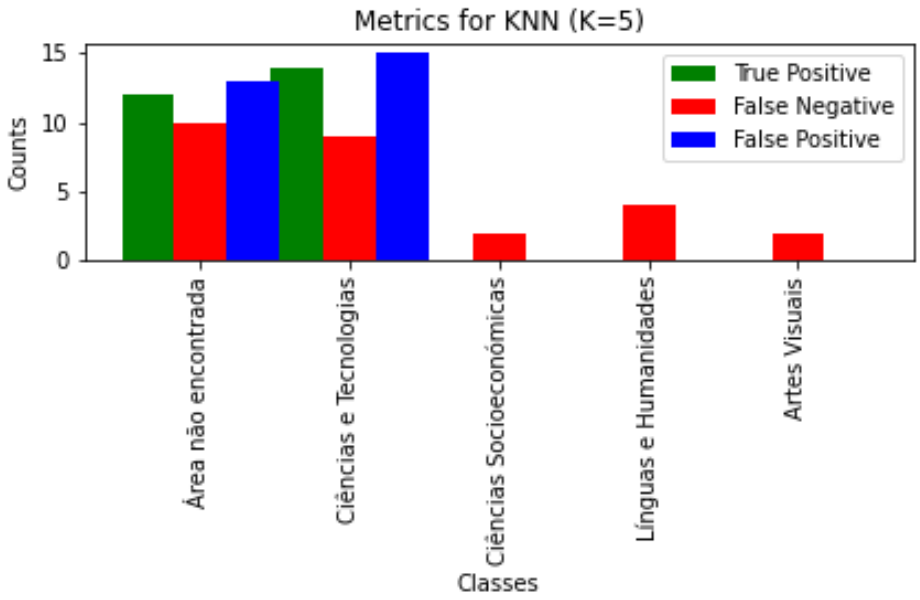


Figura 40- Métricas do K-NN (K=5)

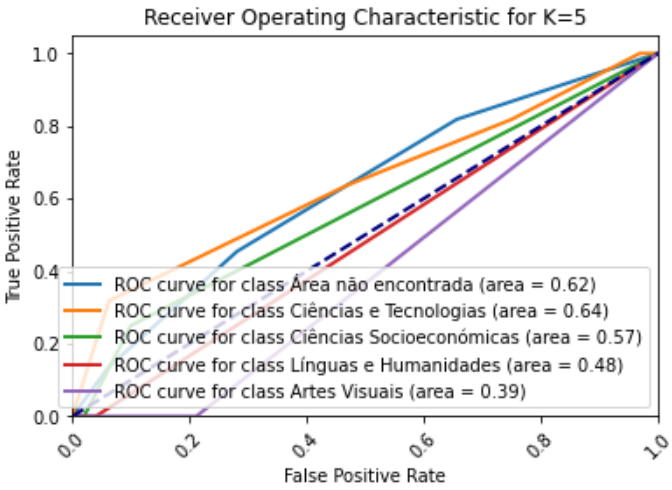


Figura 41- Curva ROC do K-NN(K=5)

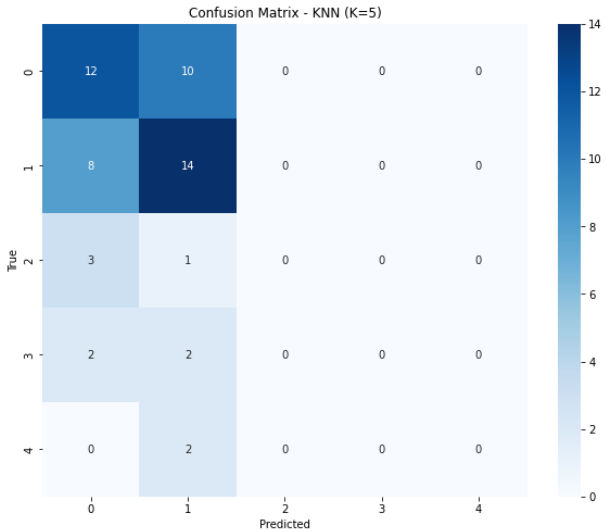


Figura 42-Matriz de confusão K=5

Com k=5, a exatidão e o recall permanecem semelhantes aos valores anteriores, mas a precisão melhora ainda mais para 58%, portanto, ao considerar cinco vizinhos mais próximos, o modelo está a fazer previsões mais precisas em relação à classe positiva.

Para K=7:

Accuracy (K=7): 0.39

Precision (K=7): 0.50

Recall (K=7): 0.39

F1-Score (K=7): 0.35

Confusion Matrix - KNN (K=7):

[[9 13 0 0 0]

 [10 12 0 0 0]

 [2 2 0 0 0]

 [3 1 0 0 0]

 [1 1 0 0 0]]

Class 'Área não encontrada' - K=7: TP = 9, FN = 13, FP = 16

Class 'Ciências e Tecnologias' - K=7: TP = 12, FN = 10, FP = 17

Class 'Ciências Socioeconómicas' - K=7: TP = 0, FN = 4, FP = 0

Class 'Línguas e Humanidades' - K=7: TP = 0, FN = 4, FP = 0

Class 'Artes Visuais' - K=7: TP = 0, FN = 2, FP = 0

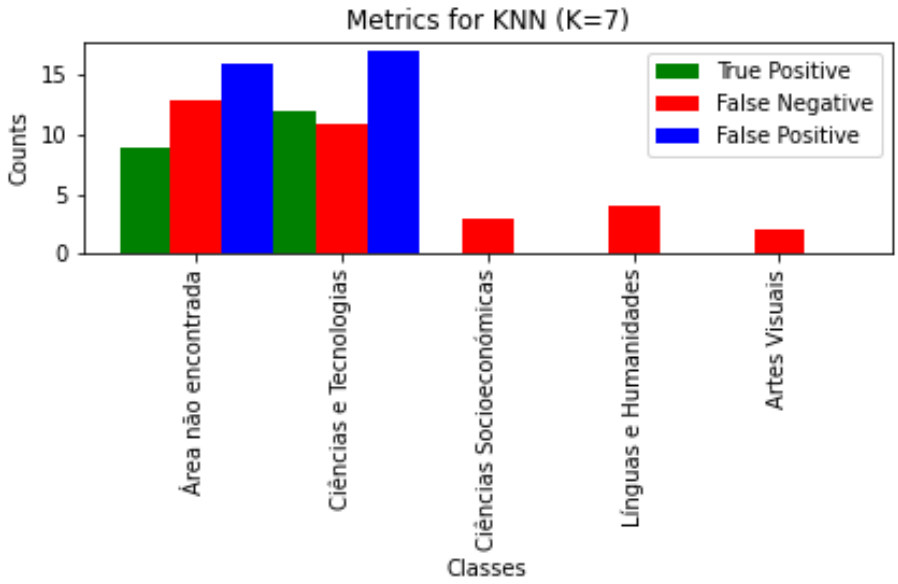


Figura 43- Métricas do K-NN(k=7)

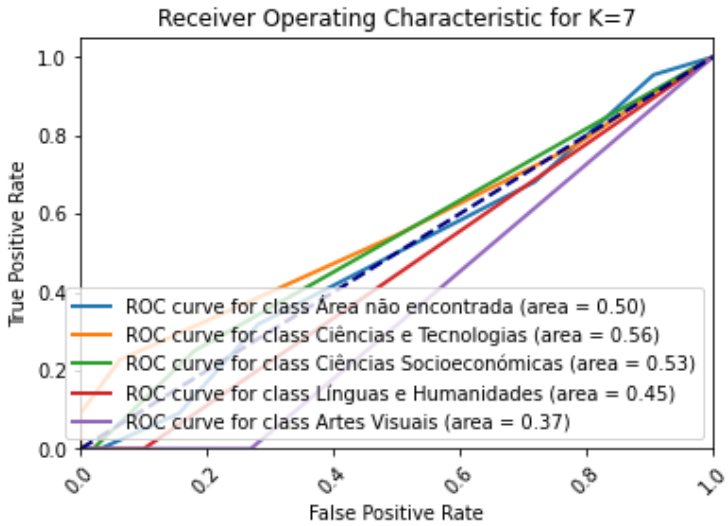


Figura 44- Curva ROC do K-NN(K=7)

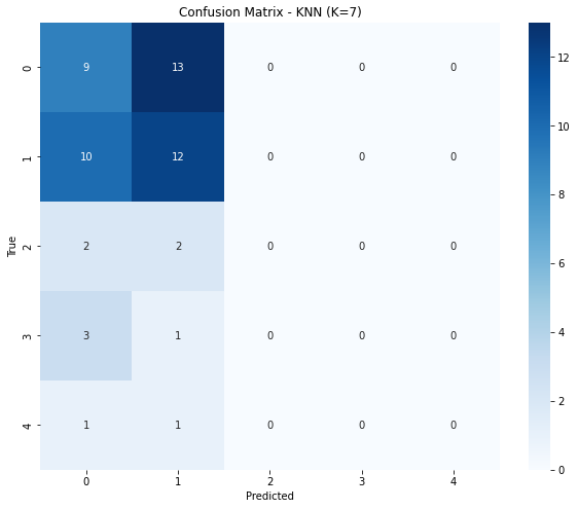


Figura 45- Matriz de confusão K=7

Com k=7, a exatidão cai para 39%, e tanto a precisão quanto o recall são menores em comparação com os valores anteriores, o que indica que considerar sete vizinhos mais próximos pode resultar numa perda de precisão e recall em relação à classe positiva. A precisão diminuiu para 0.50, o recall diminuiu para 39% e o F1-Score diminuiu para 0.35, sendo assim, o K=7 obteve o desempenho mais fraco entre os valores de k testados, com a exatidão mais baixa e valores de precisão, recall e F1-Score também mais baixos.

Em resumo, com base nos resultados apresentados e considerando que a exatidão não é excecionalmente alta para nenhum valor de k, pode-se argumentar que K=5 parece oferecer o melhor equilíbrio entre precisão, recall e F1-Score, com uma exatidão razoável.

4.3.1.5 Classificador Adaboost

O AdaBoost é um algoritmo pertence à categoria de "ensemble learning" que envolve a combinação de vários modelos de ML para criar um modelo mais robusto e geralmente mais preciso. Os procedimentos para este classificador foram o carregamento dos dados das colunas que representam o x e o y, a divisão dos dados em conjuntos de treino e teste e inicialização do classificador com a classe AdaBoostClassifier da biblioteca Scikit-Learn, treino (conjunto X_train e y_train). Durante o treino, o AdaBoost constrói vários classificadores fracos e combina-os de forma ponderada para formar um classificador final mais forte. O código faz previsões, avalia o desempenho do classificador calculando várias métricas de avaliação, como exatidão, precisão, recall, F1-Score e criação de curvas ROC para cada classe. Cria ainda visualizações das curvas ROC para cada classe, bem como uma matriz de confusão para avaliar o desempenho do modelo nas diferentes classes e

imprime os resultados de avaliação, incluindo as métricas de desempenho e a matriz de confusão, além de guardar as previsões do modelo num ficheiro Excel chamado "previsoes_AdaBoost.xlsx".

O código que utilizei para classificar com o Adaboost encontra-se no Anexo Q

Os resultados obtidos com o Adaboost foram os seguintes:

Confusion Matrix - Ada Boost:

[[3 15 2 1 1]

 [2 18 1 1 0]

 [0 3 0 1 0]

 [0 3 0 0 1]

 [0 1 1 0 0]]

Class 'Ciências e Tecnologias': TP = 18, FN = 4, FP = 22

Class 'Ciências Socioeconómicas': TP = 0, FN = 4, FP = 4

Class 'Línguas e Humanidades': TP = 0, FN = 4, FP = 3

Class 'Artes Visuais': TP = 0, FN = 2, FP = 2

Class 'Profissional informatica eletronica e comunicacoes': TP = 0, FN = 0, FP = 0

Class 'Profissional turismo e lazer': TP = 0, FN = 0, FP = 0

Class 'Profissional cultura, patrimonio e gestão de conteudos': TP = 0, FN = 0, FP = 0

Class 'Área não encontrada': TP = 3, FN = 19, FP = 2

AUC for class 'Área não encontrada': 0.53

AUC for class 'Ciências e Tecnologias': 0.60

AUC for class 'Ciências Socioeconómicas': 0.42

AUC for class 'Línguas e Humanidades': 0.43

AUC for class 'Artes Visuais': 0.93

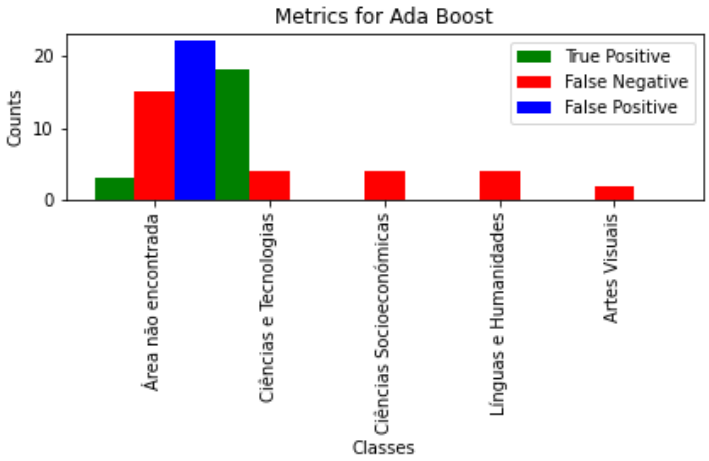


Figura 46- Métricas AdaBoost

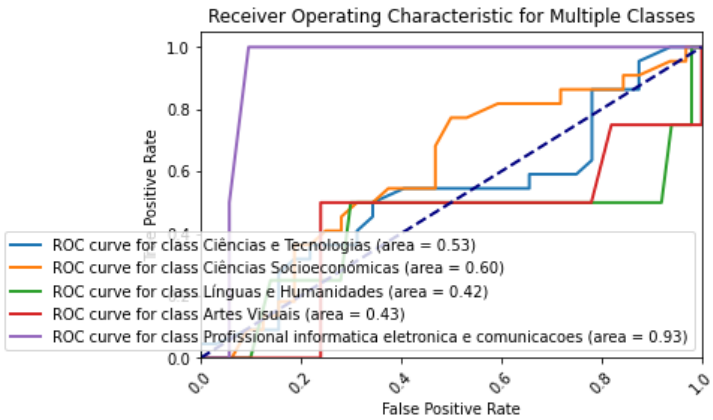


Figura 47-ROC do AdaBoost

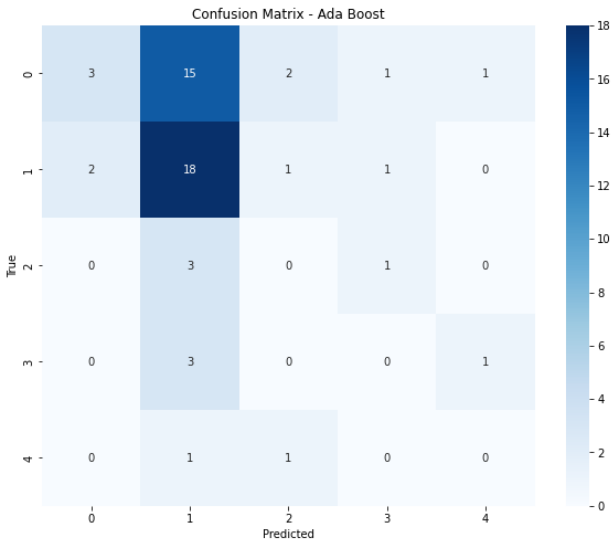


Figura 48-Matriz de confusão AdaBoost

Na Matriz de confusão, na classe 'Ciências e Tecnologias', o modelo classificou corretamente 18 instâncias (True Positives - TP), mas errou 4 instâncias (False Negatives - FN) e fez 22 previsões incorretas (False Positives - FP). Nas outras classes, como 'Ciências Socioeconómicas', 'Línguas e Humanidades' e 'Artes Visuais', o modelo teve desempenho inferior, com algumas classes tendo apenas previsões incorretas (FP) e nenhuma previsão correta (TP). A AUC é uma medida que avalia o desempenho do modelo em classificar classes positivas e negativas. Quanto maior a AUC, melhor o desempenho do modelo. A AUC mais elevada (0,93) para a classe 'Artes Visuais' indica que o modelo é muito bom em distinguir essa classe das outras. No geral, o modelo AdaBoost apresenta um desempenho razoável, com destaque para a classe 'Artes Visuais', mas pode ser melhorado em outras classes, como 'Ciências Socioeconómicas' e 'Línguas e Humanidades'.

4.3.1.6 Classificador Multilayer Perceptron (MLP)

O MLP é um tipo de rede neuronal artificial que consiste em múltiplas camadas de neurónios divididas em três tipos principais: Input layer, Hidden layers e Output layers. O script para utilizar este classificador carrega os dois dataframes com os resultados das métricas, do VADER e dos léxicos, prepara os dados, selecionando o x e o y e divide os dados em treino e teste. O MLP é inicializado com determinados hiperparâmetros, como o número de neurónios em cada camada oculta e o número máximo de iterações, sendo que será treinado e irá realizar previsões, avaliar o desempenho utilizando várias métricas, como a Curva ROC, a AUC, a exatidão, previsão, recall e F1-score, imprime uma matriz de confusão, obtém e imprime os epochs e guarda as previsões num ficheiro Excel. O código desenvolvido para implementação do classificador MLP encontra-se no Anexo R. Neste caso, ‘hidden_layer_sizes=(6, 6)2’ define a arquitetura da rede neuronal, tendo o modelo duas camadas ocultas, cada uma com 6 neurónios. Isso significa que a rede tem 2 camadas intermediárias, com 6 neurónios em cada camada e a configuração das camadas intermediárias afeta a capacidade da rede neuronal de aprender padrões complexos nos dados. ‘max_iter=1000’ define o número máximo de épochs (iterações) durante o treino. Se a convergência (parada) não for alcançada após 1000 épochs, o treino será interrompido. ‘random_state=42’: define a semente (seed) para a geração de números pseudoaleatórios, garantindo que os resultados sejam reproduzíveis.

Resultados obtidos com o classificador MLP:

Accuracy: 0.44

Precision: 0.56

Recall: 0.44

F1-Score: 0.38

Número de epochs: 848

Confusion Matrix - Multilayer Perceptron:

[[7 15 0 0 0]

 [5 17 0 0 0]

 [1 3 0 0 0]

 [1 3 0 0 0]

 [0 2 0 0 0]]

Class 'Ciências e Tecnologias': TP = 17, FN = 5, FP = 23

Class 'Ciências Socioeconómicas': TP = 0, FN = 4, FP = 0

Class 'Línguas e Humanidades': TP = 0, FN = 4, FP = 0

Class 'Artes Visuais': TP = 0, FN = 2, FP = 0

Class 'Profissional informatica eletronica e comunicacoes': TP = 0, FN = 0, FP = 0

Class 'Profissional turismo e lazer': TP = 0, FN = 0, FP = 0

Class 'Profissional cultura, patrimonio e gestão de conteudos': TP = 0, FN = 0, FP = 0

Class 'Área não encontrada': TP = 7, FN = 15, FP = 7

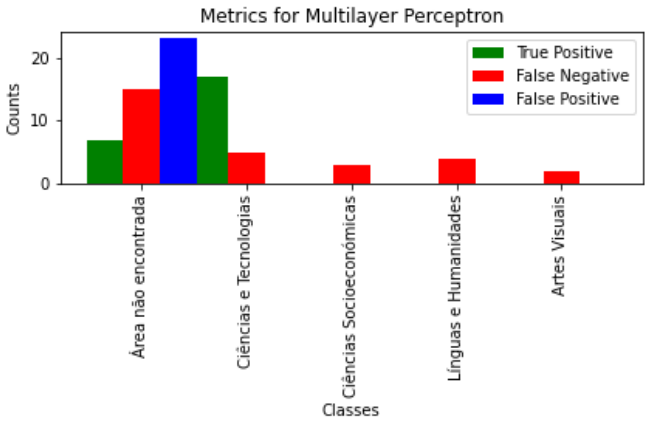


Figura 49- Métricas do MLP

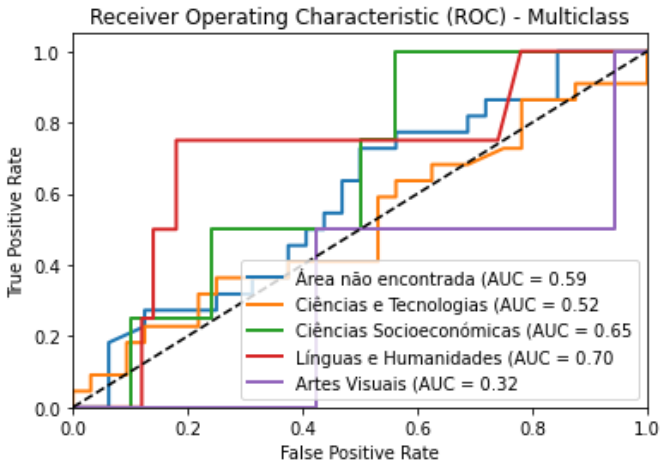


Figura 50- Curva ROC do MLP

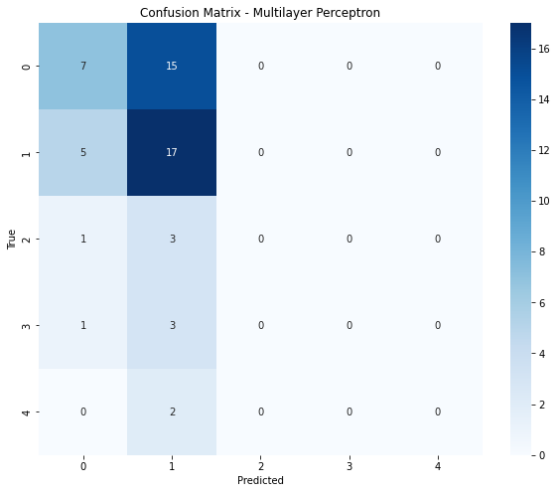


Figura 51- Matriz de confusão do MLP

Neste caso, a exatidão é de 0,44, o que significa que o MLP classificou corretamente 44% das amostras de teste e a precisão é de 0,56, o que significa que, das amostras classificadas

como positivas pelo modelo, 56% eram verdadeiras positivas. O MLP classificou corretamente 17 amostras como 'Ciências e Tecnologias' (Classe 1), mas errou 5 amostras dessa classe. Para 'Ciências Socioeconómicas', todas as amostras classificadas estavam corretas (precisão perfeita). Da mesma forma, todas as amostras de 'Línguas e Humanidades' foram classificadas corretamente (precisão perfeita). 'Artes Visuais' também foi classificada corretamente, com uma precisão perfeita. Não houve classificações para as outras classes ('Profissional informatica eletronica e comunicacoes', 'Profissional turismo e lazer', 'Profissional cultura, patrimonio e gestão de conteudos'), portanto, a precisão, revocação e F1-Score para essas classes são zero. Em resumo, o MLP obteve uma exatidão moderada de 44%, mas a precisão, revocação e F1-Score são relativamente baixos, o que sugere que o modelo pode não estar desempenhando tão bem na classificação das classes, especialmente para 'Ciências e Tecnologias' e 'Área não encontrada'. Pode ser necessário ajustar o modelo ou considerar outras abordagens para melhorar o desempenho.

Em resumo, o MLPClassifier criado tem duas camadas ocultas com 6 neurónios cada uma e treina até 1000 epochs, usando uma semente aleatória de 42 para inicialização. Estes hiperparâmetros podem ser ajustados de acordo com os resultados desejados. Um valor de 848 epochs significa que o modelo foi treinado por 848 iterações completas sobre o conjunto de treino. Em termos simples, cada epoch representa uma passagem completa por todo o conjunto de treino. O treino do modelo com 848 epochs pode ter sido uma escolha apropriada com base no comportamento do treino e nos resultados obtidos. Em geral, o desempenho do modelo de MLP com base nas métricas fornecidas pode ser considerado abaixo do ideal. Um F1-Score de 0.38 indica que o modelo não está a obter um equilíbrio satisfatório entre precisão e recall. No entanto, a avaliação de um modelo não é apenas sobre números, depende muito do contexto e dos requisitos do problema. Em alguns cenários, um F1-Score de 0.38 pode ser aceitável, dependendo das consequências dos falsos positivos e falsos negativos. Para obtenção das imagens com os gráficos das métricas para cada classificador foi utilizado o script do Anexo S.

4.3.1.7. Comparação entre modelos de classificação

Foram criados três scripts para comparar os modelos e as suas métricas, através dos dados das matrizes de confusão, obtendo-se um gráfico por modelo, um gráfico que representasse

todos os modelos e a curva ROC com todos os modelos. O script para obter a comparação das métricas num só gráfico encontra-se no Anexo T.

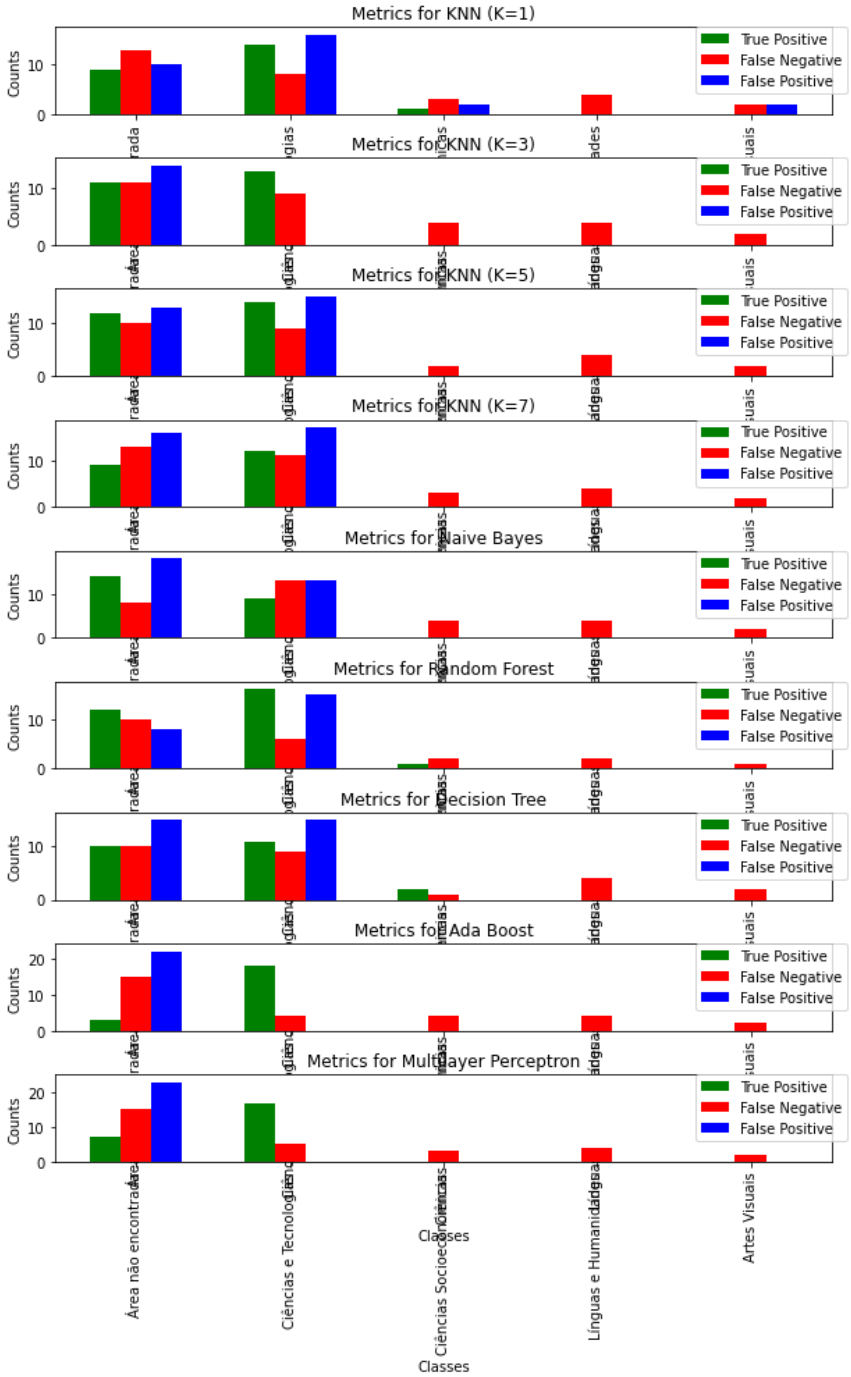


Figura 52- Comparação de métricas de todos os classificadores

Para obtenção de um gráfico com as curvas ROC de todos os classificadores utilizou-se o script presente no Anexo U.

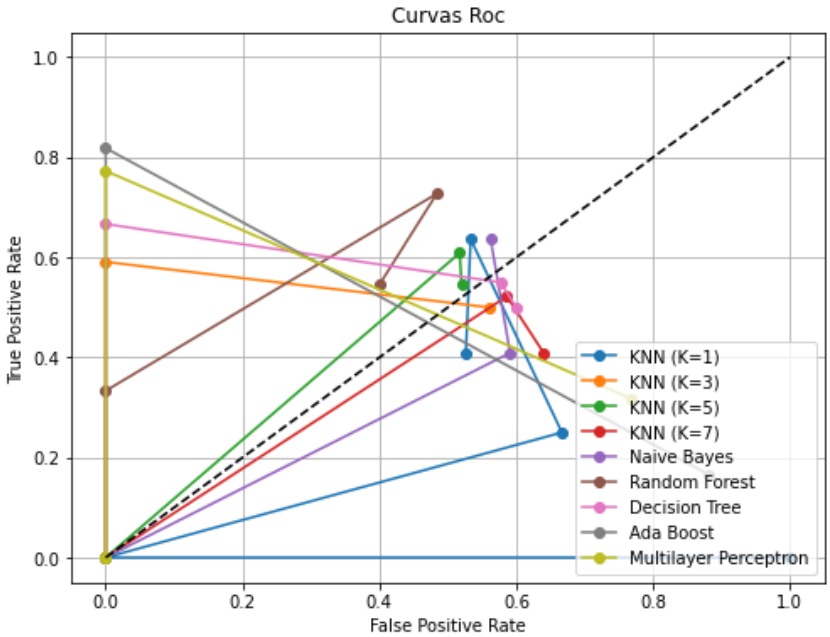


Figura 53- Curva ROC com todos os classificadores

O AUC (Área sob a Curva) da curva ROC (Receiver Operating Characteristic) é uma métrica importante para avaliar o desempenho de classificadores binários, visto que fornece informações valiosas sobre a capacidade do classificador de distinguir entre as classes positiva e negativa. Quanto maior o valor do AUC, melhor a capacidade do classificador de discriminar entre as classes, por exemplo, um AUC de 1.0 indica um classificador perfeito, enquanto um AUC de 0.5 indica um classificador que não é melhor do que uma escolha aleatória. Observando os resultados, os valores da AUC-ROC para cada classificador são os seguintes:

Naive Bayes: 0.3654

Decision Tree (dividido por classe):

'Área não encontrada': 0.46

'Ciências e Tecnologias': 0.50

'Línguas e Humanidades': 0.50

'Ciências Socioeconómicas': 0.50

Multilayer Perceptron (MLP): 0.3173

Random Forest (dividido por classe):

'Área não encontrada': 0.66

'Ciências e Tecnologias': 0.58

'Línguas e Humanidades': 0.42

'Ciências Socioeconómicas': 0.43

'Artes Visuais': 0.18

AdaBoost (dividido por classe):

'Área não encontrada': 0.53

'Ciências e Tecnologias': 0.60

'Ciências Socioeconómicas': 0.42

'Línguas e Humanidades': 0.43

'Artes Visuais': 0.93

KNN (K=1, K=3, K=5, K=7): Todos têm o mesmo valor de 0.37.

Para determinar qual dos classificadores obteve os melhores resultados gerais com base nas AUC-ROC médias, pode-se calcular a média das AUC-ROC de todas as classes e escolher o classificador com a média mais alta. As médias das AUC-ROC para cada classificador são as seguintes:

Naive Bayes: 0.3654

Decision Tree (média de todas as classes): 0.48

Multilayer Perceptron (MLP): 0.3173

Random Forest (média de todas as classes): 0.454

AdaBoost (média de todas as classes): 0.48

KNN (K=1, K=3, K=5, K=7): 0.37 (mesmo valor para todos os K)

Com base nas AUC-ROC médias, os classificadores Decision Tree e AdaBoost tiveram o melhor desempenho geral, com uma média de AUC-ROC de 0.48, tendo também o Adaboost obtido um valor muito bom (0.93), com a melhor curva ROC para a classe 'Artes Visuais'.

Tabela 15- Comparação das métricas dos classificadores

Classificador	Accuracy	Precision	Recall	F1-Score
Naive Bayes	0,43	0,53	0,43	0,38
Decision Tree	0,44	0,48	0,44	0,42
Random Forest	0,54	0,62	0,54	0,5
KNN (K=1)	0,44	0,48	0,44	0,42
KNN (K=3)	0,44	0,55	0,44	0,4
KNN (K=5)	0,48	0,58	0,48	0,43
KNN (K=7)	0,44	0,43	0,44	0,42
AdaBoost	0,39	0,43	0,39	0,33
Multilayer Perceptron	0,44	0,56	0,44	0,38

Podemos observar que com base nos resultados apresentados, o classificador Random Forest obteve a maior exatidão (0,54) e o maior F1-Score (0,50), indicando um desempenho geralmente melhor em comparação com os outros classificadores. Por outro lado, o AdaBoost teve a exatidão mais baixa (0,39) e o F1-Score mais baixo (0,33). O classificador K-NN (K=5) também obteve resultados competitivos com uma exatidão de 0,48 e um F1-Score de 0,43.

4.3.2 Regressão

O objetivo da regressão é prever ou estimar um valor numérico com base num conjunto de características (variáveis independentes) e as suas métricas de avaliação comuns incluem o Erro Quadrático Médio (MSE), o Erro Médio Absoluto (MAE), o Coeficiente de Determinação (R^2) que medem a precisão das previsões em relação aos valores reais.

4.3.2.1 Regressão Linear

Foi criado um script com 6 variáveis independentes (entradas) para prever uma variável dependente (saída). As variáveis independentes são as médias das métricas do VADER e da métrica personalizada para as diferentes perguntas (9 a 14), e a variável dependente é o valor numérico da área mais frequente associada a essas respostas, obtida na análise das perguntas (1 a 8). O modelo de regressão linear tenta encontrar uma relação linear entre essas variáveis independentes e a variável dependente, de modo a fazer previsões para novos conjuntos de

variáveis independentes. O código para criar a Regressão Linear realiza as seguintes tarefas: (Anexo V)

-Importa as bibliotecas necessárias, incluindo NumPy, pandas, scikit-learn e matplotlib.

-Define um dicionário chamado ‘class_names’ que mapeia os valores numéricos para nomes de classes.

-Carrega dois ficheiros Excel em DataFrames usando a biblioteca pandas: ‘df_vader_metric’ e ‘df_area_scores’.

-Seleciona as variáveis independentes (características) X e a variável dependente y a partir dos DataFrames carregados.

-Divide os dados em conjuntos de treino e teste usando a função ‘train_test_split’ do scikit-learn.

-Inicializa um modelo de regressão linear e o treina os dados de treino usando a função ‘fit’.

-Faz previsões nos dados de teste usando a função ‘predict’.

-Avalia o desempenho do modelo calculando o erro quadrático médio (RMSE) e o coeficiente de determinação (R^2).

-Plota um gráfico de dispersão das previsões em relação aos valores reais com uma reta de regressão.

-Arredonda as previsões para valores inteiros de classes (de 1 a 7).

-Cria um DataFrame para armazenar os valores reais e previstos da área.

-Mapeia os valores numéricos das áreas previstas e reais para nomes de classes usando o dicionário ‘class_names’.

-Calcula o RMSE novamente para as previsões arredondadas.

-Guarda as previsões num ficheiro Excel chamado 'predictions_RegressaoLinear.xlsx'.

-Guarda o modelo treinado num ficheiro chamado 'modelo_regressao_linear.pkl' usando a biblioteca joblib.

Os valores obtidos foram:

RMSE: 1.04

R2 Score: 0.01

Coeficientes: [0.06627046 0.2175203 -0.12983224 0.11675394 -0.10872344 -0.02485705]

Intercept: 0.7106150856912423

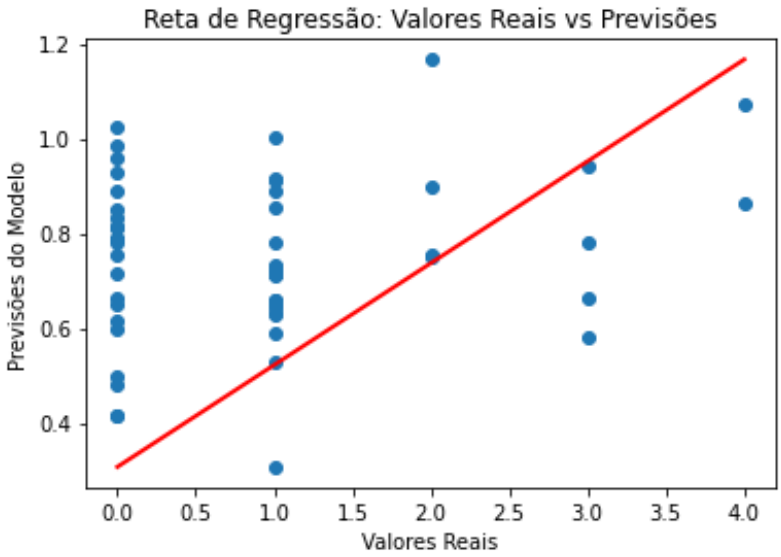


Figura 54- Reta da Regressão Linear

O valor do RMSE é 1.04, o que significa que, em média, as previsões do modelo estão a cerca de 1.04 unidades de distância dos valores reais das áreas. O valor do R^2 Score é 0.01. O R^2 Score varia de 0 a 1 e representa a proporção da variabilidade nos dados de destino que é explicada pelo modelo. Os coeficientes do modelo representam a contribuição relativa de cada variável independente (característica) na previsão da variável dependente (área). Um coeficiente positivo indica uma relação positiva com a área, enquanto um coeficiente negativo indica uma relação negativa. O valor da intercetação (intercept) é 0.7106150856912423, que representa o valor previsto da área quando todas as variáveis independentes (características) são iguais a zero.

4.3.2.2. Regressão Polinomial

A regressão polinomial é uma extensão da regressão linear, que permite capturar relacionamentos não lineares entre as variáveis independentes e dependentes. Em modelos polinomiais, o grau do polinómio refere-se ao nível de complexidade do modelo. Um grau de polinómio maior do que 1 indica um modelo polinomial de grau superior, que é mais

complexo do que um modelo linear (grau 1), o que significa que a relação entre as características e a variável de destino é modelada por um polinómio de grau superior. Por exemplo, um modelo polinomial de grau 2 é representado por uma parábola, enquanto um modelo polinomial de grau 3 é representado por uma curva cúbica. O código criado para a Regressão Polinomial encontra-se no Anexo W.

Os resultados obtidos foram:

Size of y_test: 54

Size of y_train: 213

Polynomial Degree: 1

RMSE: 1.04

R-squared: 0.01

Polynomial Degree: 2

RMSE: 1.11

R-squared: -0.12

Polynomial Degree: 3

RMSE: 2.30

R-squared: -3.77

Polynomial Degree: 4

RMSE: 38.97

R-squared: -1372.61

Polynomial Degree: 5

RMSE: 8.49

R-squared: -64.23

Polynomial Degree: 6

RMSE: 9.40

R-squared: -78.90

Polynomial Degree: 7

RMSE: 13.95

R-squared: -175.11

Polynomial Degree: 8

RMSE: 25.00

R-squared: -564.15

Polynomial Degree: 9

RMSE: 48.79

R-squared: -2151.64

Polynomial Degree: 10

RMSE: 96.69

R-squared: -8455.24

Best Polynomial Degree: 1

Best RMSE: 1.04

RMSE: 96.69

Coeficientes: [-5.06904468e-13 -5.81461682e-03 -5.32897820e-01 ... 2.36003707e-02

-1.60894937e-01 -1.46128447e-01]

Intercept: -4.5152770411505116e-13

Polynomial Degree: 10

RMSE: 96.69

R-squared: -8455.24

Best Polynomial Degree: 1

Best RMSE: 1.04

Os resultados apresentados indicam o desempenho dos diferentes modelos de regressão polinomial com diferentes graus de polinómio no ajuste dos dados de treino e teste. Pode-se interpretar os resultados da seguinte forma:

-‘Size of y_test e Size of y_train’: Estes valores indicam o tamanho dos conjuntos de dados de teste e treino, respetivamente. Neste caso, há 213 amostras no conjunto de treino e 54 amostras no conjunto de teste.

-‘Polynomial Degree’: O grau do polinómio representa o nível de complexidade do modelo polinomial. A análise começa com um grau 1 (equivalente à regressão linear) e aumenta gradualmente até o grau 10 para avaliar diferentes níveis de complexidade.

-‘RMSE (Root Mean Squared Error)’: O RMSE é uma métrica de avaliação que mede a média dos erros quadrados entre as previsões do modelo e os valores reais. Quanto menor o valor do RMSE, melhor o ajuste do modelo aos dados. É uma medida de precisão, onde valores mais baixos indicam um melhor ajuste. Para um grau de polinómio de 1 (equivalente a uma regressão linear), o RMSE é 1.04 e o R^2 é 0.01, o que indica um ajuste fraco aos dados. À medida que o grau do polinómio aumenta, o RMSE tende a aumentar, e o R^2 tende a diminuir drasticamente.

-‘R-squared (Coeficiente de Determinação)’: O R^2 é uma métrica que varia de 0 a 1 e representa a proporção da variabilidade nos dados de destino que é explicada pelo modelo.

No final, o "Best Polynomial Degree" é relatado como 1, o que significa que o modelo de grau 1 (regressão linear) é considerado o melhor com um RMSE de 1.04. Num modelo de regressão polinomial de grau 1, a interpretação é mais simples do que em graus mais altos, porque não há termos elevados ao quadrado ou interações complexas para considerar. Cada coeficiente está associado diretamente a uma variável de entrada e representa o efeito linear dessa variável na variável de saída.

Em resumo, os resultados indicam que um modelo de regressão linear simples (grau 1) parece ser o mais apropriado para os meus dados, já que modelos de grau superior estão apresentando um ajuste pobre aos dados de teste, com RMSEs muito altos e R^2s muito baixos. O princípio da parcimónia sugere que, em igualdade de condições, o modelo mais simples é preferível, e, neste caso, o modelo de regressão linear (grau 1) é mais simples e está fornecendo um ajuste aceitável aos dados.

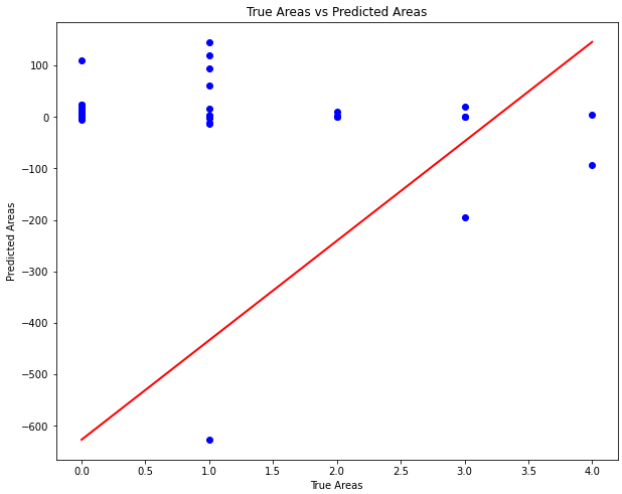


Figura 55- Previsões Regressão Polinomial

4.3.3. Aplicação Web

Uma aplicação prática baseada na regressão linear é uma parte fundamental do processo de modelagem, importante para fazer previsões em situações reais e avaliar o seu desempenho, fazendo parte do processo de validação e teste do modelo. Assim usa-se o modelo para fazer previsões em novos dados ou em amostras de dados que não foram usados no treino do mesmo e assim pode-se avaliar o modelo. Foi criado um script em python, um em javascript e outro em html. O código em python criado para o modelo encontra-se no Anexo X.

```
(base) C:\Users\Admin>cd Previsao

(base) C:\Users\Admin\previsao>flask run
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [09/Sep/2023 21:35:29] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [09/Sep/2023 21:35:29] "GET /static/previsao.js HTTP/1.1" 304 -
```

Figura 56- Execução da App

Este código Python utiliza a framework Flask para criar uma aplicação web simples que faz previsões com base num modelo de regressão linear previamente treinado. As bibliotecas necessárias são importadas, incluindo o Flask para criar a aplicação web, o ‘joblib’ para carregar e guardar modelos, e outras bibliotecas para manipulação de dados e treino de modelos de regressão linear. O modelo de regressão linear é carregado a partir de um arquivo chamado 'modelo_regressao_linear.pkl', através do ‘joblib.load()’. A aplicação Flask é criada e chamada de "app". Quando um utilizador acessa a página inicial (rota '/'), é

redirecionado para um modelo HTML chamado 'modelo.html'. A Rota '/predict' (Método POST) lida com previsões, os valores dos scores são enviados através de uma solicitação POST do formulário web, são extraídos da solicitação usando o ‘request.form’ e são convertidos em números. A previsão é formatada como uma resposta JSON, retornada ao cliente (navegador) como um resultado da previsão e antes de iniciar a aplicação web, o código realiza o treino do modelo de regressão linear. Os dados são carregados a partir de ficheiros Excel, os conjuntos de treino e teste são divididos, e o modelo é treinado nos dados de treino. As previsões do modelo são arredondadas para valores de classe, entre 1 e 7 e a aplicação Flask é iniciada usando o ‘app.run()’. O código do ficheiro da previsão em javascript encontra-se no Anexo Y e é utilizado para realizar uma solicitação POST para um servidor Flask quando um formulário HTML é submetido. A expressão ‘var score1 = parseFloat(document.getElementById('score1').value);’ obtém o valor do campo de entrada com o ID "score1" e converte-o num número decimal (float). Isso é feito para cada um dos campos "score" (score1 a score6) e ‘var dados = { ... }’ cria um objeto chamado "dados" que contém os valores dos scores com chaves associadas aos nomes das variáveis que serão usadas no lado do servidor, que será enviado como dados JSON na solicitação POST. A função ‘fetch’ envia uma solicitação POST para o servidor Flask na rota "/predict". Em resumo, este código JavaScript é usado para recolher valores de campos de entrada num formulário HTML, enviá-los como uma solicitação POST para o servidor Flask na rota "/predict" e tratar a resposta do servidor. O código do formulário em html encontra-se no Anexo Z.

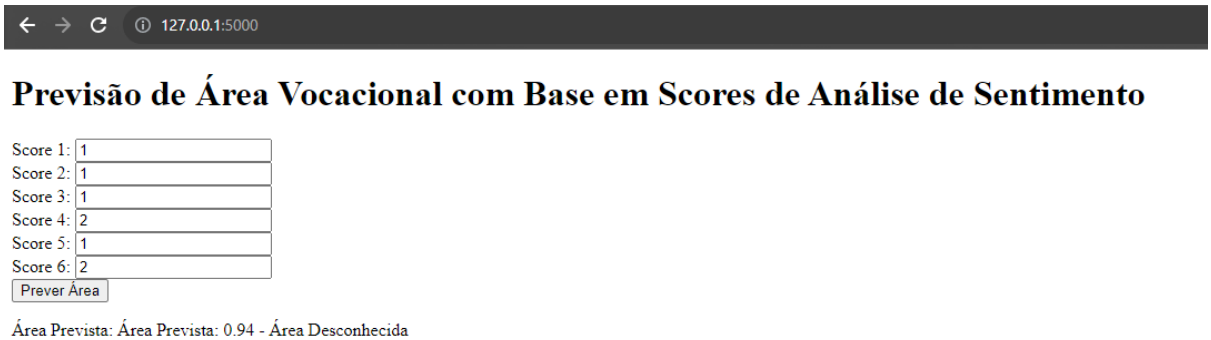


Figura 57- Aplicação local – Resultado Área Desconhecida

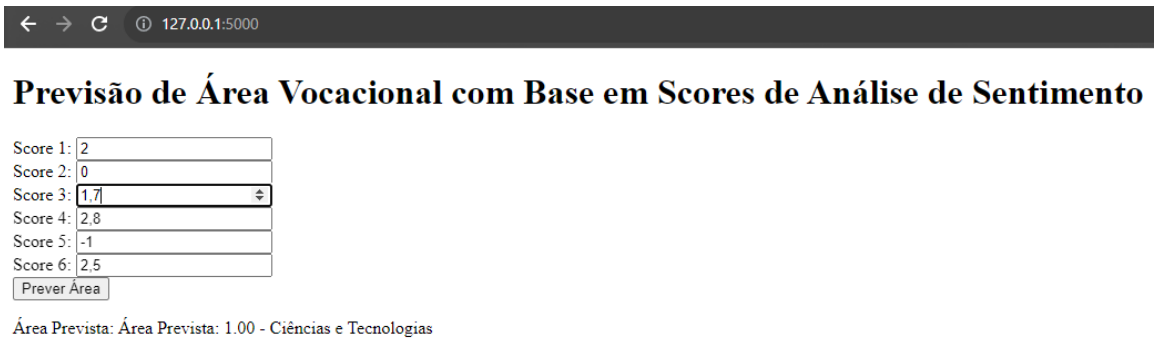


Figura 58- Aplicação local – Resultado Ciências e Tecnologias

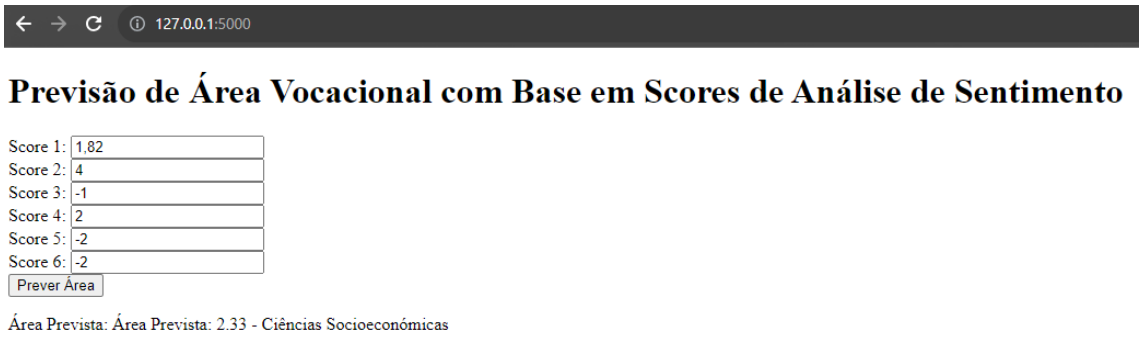


Figura 59- Aplicação local – Resultado Ciências Socioeconómicas

5. DISCUSSÃO

Os meus resultados não foram tão elevados como os que mencionei na revisão de literatura, contudo, comparar os resultados de um trabalho com os resultados de outros investigadores, é uma tarefa complexa, visto que os dados, os algoritmos, os objetivos, o pré-processamento, o contexto, a aplicação e cenários são muito diferentes, o que influencia os resultados, contudo é uma etapa importante na discussão dos mesmos.

Começo por discutir os resultados em relação ao classificador Naive Bayes Gaussiano, em que obtive de exatidão 0.43, de precisão 0.53, **(**Pang, B., Lee, L., & Vaithyanathan, S., 2002) obtiveram de precisão 65,6%, (Al-Dossari et al., 2020) obtiveram uma precisão de 70.20%, (Socher et al., 2013), uma precisão de 67.2%, % (Aman, S., & Szpakowicz, S., 2007) obtiveram uma precisão de 72,08%, (Sultana et al., 2018) obtiveram 70.20 de exatidão com o Bayes Net e (Kularbphettong, K., & Tongsiri, C., 2014), uma precisão de 0.93 com o Bayesian Networks. Quanto ao classificador Decision Tree, obtive uma precisão de 0.48, (Al-Dossari et al., 2020) obtiveram de precisão 60,15%, (Sultana et al., 2018) obtiveram uma exatidão de 75.83 e (Kularbphettong, K., & Tongsiri, C., 2014) obtiveram uma precisão de 0.84, num algoritmo J48. Em relação ao Random Forest, obtive de exatidão 0.54 e de precisão 0.62 e (Sultana et al., 2018) obtiveram de exatidão 76.66. Quanto ao K-NN(k=1), obtive de exatidão 0.44 e de precisão 0.48 e (Al-Dossari et al., 2020) obtiveram uma exatidão de 49.2%. Em relação ao K-NN com outros vizinhos, como não há referências na revisão de literatura que fiz, não posso comparar os resultados que tive com k=3, k=5 e k=7. Em relação ao MLP obtive de exatidão 0.44 e de precisão 0.56, (Sultana et al., 2018) obtiveram 78.33 de exatidão, (Elangovan, D., & Subedha, V., 2020) obtiveram de exatidão num algoritmo FF-MLP, 97.14 e (Sultana et al., 2018) obtiveram uma precisão de 78.33% para o MLP. Para o classificador AdaBoost obtive de exatidão 0.39, de precisão 0.43 e (Al-Dossari et al., 2020) obtiveram para um algoritmo de boosting, o XGBoost, uma exatidão de 70,47% e para o algoritmo Gradient Boosting, 61.22%.

O meu trabalho apresenta uma variedade de classificadores, mas os resultados são mistos em comparação com a literatura. Alguns dos meus classificadores têm desempenho semelhante aos da literatura, enquanto outros têm desempenho inferior. É importante destacar que os resultados podem variar dependendo dos dados, configurações do modelo, pré-processamento e outros fatores. Os valores de precisão variam de acordo com o contexto e a aplicação, e é importante considerar que diferentes conjuntos de dados e cenários podem

influenciar os resultados. No caso do NB Gaussiano, obtive uma precisão de 0.53 e estudos anteriores relataram valores de precisão variando entre 65.6% a 72.08%. Em relação à Decision Tree, a minha precisão foi de 0.48, sendo que observo que estudos anteriores alcançaram valores mais elevados, como 0.60 e 0.84. Em relação à Random Forest, alcancei uma precisão de 0.62, que é comparável a um valor de 0.76 encontrado em um estudo anterior. Este classificador geralmente é conhecido por ser robusto e fornecer bons resultados numa variedade de situações. Quanto ao K-NN, a escolha de k pode afetar significativamente o desempenho do classificador, e estudos anteriores podem ter usado valores diferentes de k. Quanto ao MLP, a minha precisão foi de 0.56, enquanto um estudo anterior alcançou 78.33%. As redes neuronais, como o MLP, podem ser sensíveis à arquitetura, hiperparâmetros e tamanho do conjunto de dados, o que pode explicar as diferenças nos resultados. Com o AdaBoost obtive uma precisão de 0.43, enquanto um estudo anterior alcançou 70.47% com o XGBoost e 61.22% com Gradient Boosting. O AdaBoost é uma técnica de boosting, e os resultados podem variar dependendo do algoritmo base e dos parâmetros usados.

A percentagem baixa de precisão e exatidão obtida, poderá ser justificada pela pouca quantidade e qualidade dos dados no meu dataset. É importante notar que a precisão pode variar normalmente entre diferentes tentativas, mesmo com o mesmo modelo e conjunto de dados, devido a fatores aleatórios, portanto, não é incomum que resultados ligeiramente diferentes sejam obtidos em diferentes estudos, mesmo quando os métodos são semelhantes.

A comparação do meu estudo com estudos anteriores pode fornecer insights valiosos sobre como melhorar os meus modelos. Para trabalho futuro poderei investigar quais as técnicas, pré-processamento, ajuste de hiperparâmetros, variantes dos classificadores ou estratégias específicas é que foram utilizadas nos estudos anteriores que alcançaram precisões mais altas, alterar os meus modelos de modo a obter melhores resultados e até construir novos modelos com outros classificadores.

Em relação às minhas questões de investigação, penso que, em resposta à primeira questão, ‘Como um modelo de Inteligência Artificial aplicado a análise sentimental textual pode ajudar os estudantes a escolher a sua decisão de percurso vocacional futuro?’, os modelos de IA podem ajudar, na medida em que os alunos podem saber as áreas vocacionais mais adequadas aos seus perfis, com base nas suas respostas ao questionário, contudo, deverão ser verdadeiros nas suas respostas e o seu texto ser claro e completo, de escrita

longa, de modo a que o classificador possa ser eficaz. Caso assim seja, a área vocacional obtida poderá ajudar o aluno a decidir que caminho seguir.

Em relação à questão ‘Quais os melhores algoritmos para solucionar o problema com melhor eficácia, precisão e exatidão?’ concluo que com base nos meus resultados são o classificador Random Forest e o K-NN (K=5). Apesar dos resultados dos modelos não serem elevados, penso que tiveram sucesso nas suas previsões. Quanto à questão ‘Pode a solução final ser aplicada a nível nacional e ser uma referência nesta problemática?’, penso que sim, seria útil a utilização dos modelos a nível nacional, se as respostas dos alunos forem completas e verdadeiras e o pré-processamento dos dados seja eficiente.

Em relação à hipótese colocada no inicio desta dissertação ‘O uso de modelos de IA fornecem uma exatidão de elevada percentagem nos casos estudados na revisão da literatura e são possíveis de ser usadas neste campo de investigação e aplicação.’ é verdade que nos casos estudados os modelos obtiveram uma exatidão elevada, contudo nenhum dos casos foi com alunos de 3º ciclo, nem o âmbito do estudo foi a previsão de áreas vocacionais, devido a não existirem estudos nessa área, no entanto, os modelos de IA criados são possíveis de ser usados neste campo de investigação e aplicação.

O meu estudo teve algumas limitações, como a demora na aprovação do questionário pela DGE/MIME (Direção Geral de Educação / Monitorização de Inquéritos em Meio Escolar), cerca de um mês, sendo que não pude enviar o questionário para os agrupamentos sem ter a aprovação e o número de registo do questionário, a pouca quantidade de alunos que responderam ao questionário, 267, e também a pouca qualidade nas respostas, visto que muitas são incompletas, ambíguas, não categorizáveis. Lamento a falta de adesão dos Agrupamentos de Escolas contactados para participarem no meu estudo, sendo que apenas 3 dos 60 AE contactados participaram, o que limitou a qualidade do meu dataset. Outra limitação encontrada foi o pré-processamento dos dados, penso que poderia ter contornado os problemas que iam surgindo de forma mais eficiente.

6. CONCLUSÃO

Este estudo desempenha um papel significativo na interseção entre análise de sentimento textual e orientação vocacional para estudantes do 3º ciclo. Apesar dos resultados terem revelado uma precisão e exatidão moderadas nos classificadores avaliados, eles ainda fornecem uma base sólida para futuras pesquisas e aplicações práticas. Ao explorar a eficácia de diversos algoritmos de ML, identificamos que o Random Forest e o K-NN (K=5) se destacaram como os classificadores mais promissores para prever as áreas vocacionais dos alunos. Apesar dos meus resultados não terem atingido os níveis de precisão e exatidão relatados em alguns estudos anteriores, é importante lembrar que a comparação direta com esses estudos é complexa, dada a diversidade de fatores influentes. As minhas descobertas enfatizam a necessidade de considerar o contexto específico, a qualidade dos dados e as particularidades do público-alvo ao aplicar modelos de IA para orientação vocacional. Este trabalho responde à minha primeira questão de investigação, demonstrando que modelos de IA podem ser valiosos na identificação de áreas vocacionais adequadas para os estudantes, desde que as respostas dos alunos sejam completas e verdadeiras e o pré-processamento dos dados seja eficiente. Além disso, identifiquei que o Random Forest e o K-NN (K=5) são escolhas promissoras para abordar esta questão com eficácia. Apesar da minha hipótese inicial sugerir que os modelos de IA podem alcançar altas taxas de precisão, observei que o contexto único deste estudo apresenta desafios próprios. A baixa adesão das escolas e a qualidade variável das respostas dos alunos representam limitações que devem ser abordadas em pesquisas futuras. Em relação à hipótese inicial, não foi totalmente confirmada, mas, esta pesquisa estabelece uma base sólida para futuras investigações e melhorias.

Concluo que, este estudo representa um avanço importante na aplicação de IA para orientação vocacional de estudantes do 3º ciclo. Embora haja espaço para melhorias, as minhas descobertas destacam as oportunidades e desafios nesta área de investigação em rápido crescimento, indicando um caminho promissor para futuros desenvolvimentos e aplicações práticas. A minha abordagem pioneira fornece um trampolim para estudos subsequentes, visando aprimorar os modelos e estender a sua aplicação a nível nacional, envolvendo mais escolas e algumas sugestões para trabalho futuro poderiam ser a introdução de uma nova variável no ficheiro de análise de sentimento textual para avaliar a subjetividade ou ainda a intensidade, bem como a análise com outros classificadores, com diferentes combinações e abordagens.

BIBLIOGRAFIA

A Machine Learning Approach to Career Path Choice for Information Technology Graduates. *Engineering, Technology & Applied Science Research*.

Acervo Lima. (s.d.). Classificadores Naive Bayes. Obtido de https://acervolima.com/classificadores-naive-bayes/

Agrawal, Ameeta; An, Aijun (2012), Unsupervised Emotion Detection from Text Using Se-mantic and Syntactic Relations. Em: 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology.

Agrupamento de Escolas Eduardo Gageiro. (2022). Plano Plurianual de Melhoria. Acedido em https://aeeg.pt/wp-content/uploads/2022/02/Plano-Plurianual-de-Melhoria.pdf

Al- Dossari, H., Nughaymish, F. A., Al-Qahtani, Z., Alkahlifah, M., & Alqahtani, A. (2020).

Aman, S., & Szpakowicz, S. (2007). Identifying expressions of emotion in text. In *International Conference on Text, Speech and Dialogue*. Springer, Berlin, Heidelberg.

Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media, Inc.

Câmara Municipal de Loures. (s.d.). Rede de Bibliotecas de Loures. Obtido de https://www.cm-loures.pt/Ligacao.aspx?DisplayId=88

Cardoso, José, (2020), Profile Matcher, Dissertação de Mestrado, Instituto Superior de Engenharia do Porto, Politécnico do Porto.

Direção-Geral de Estatísticas da Educação e Ciência. (s.d.). DGEEC. Obtido de https://www.dgeec.mec.pt/np4/dgeec/

Direção-Geral de Estatísticas da Educação e Ciência. (s.d.). MIME - Monitorização de Inquéritos em Meio Escolar. Obtido de http://mime.dgeec.mec.pt/Default.aspx

Eckman, P. (1994). All Emotions are basic, Ekman,P &Davison, R. (ed.) The nature of emotion: fundamental questions. New York: Oxford University.

Elangovan, D., & Subedha, V. (2020). An Effective Feature Selection Based Classification model using Firefly with Levy and Multilayer Perceptron based Sentiment Analysis. In *2020 International Conference on Inventive Computation Technologies (ICICT)*, IEEE.

Faria, E., (2018), Redes Neurais convolucionais e máquinas de Aprendizado Extremo aplicadas ao Mercado financeiro brasileiro, Tese de Doutoramento, Universidade Federal do Rio de Janeiro.

Gao, J., Yao, R., Lai, H., & Chang, T. C. (2019). Sentiment analysis with CNNs built on LSTM on tourists comments. In *2019 IEEE Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability (ECBIOS)*, IEEE.

GeeksforGeeks. (s.d.). Naive Bayes Classifiers. Obtido de https://www.geeksforgeeks.org/naive-bayes-classifiers/

Graves, A. (2012). Long short-term memory. *Supervised sequence labelling with recurrent neural networks*.

Haque, M. R., Lima, S. A., & Mishu, S. Z. (2019). Performance analysis of different neural networks for sentiment analysis on IMDb movie reviews. In *2019 3rd International Conference on Electrical, Computer & Telecommunication Engineering (ICECTE)*, IEEE.

Kularbphettong, K., & Tongsiri, C. (2014). Mining educational data to support students' major selection. *International Journal of Educational and Pedagogical Sciences*, *8*(1), 21-23.

Labone Consultoria. (s.d.). Teorema de Bayes. Obtido de https://www.laboneconsultoria.com.br/teorema-de-bayes/

Lerner, Jennifer S (2004). Emotions and Decision Making, p., Annual Review of Psychology.

Liddy, E.D. (2001). Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc.

microblogs. *arXiv preprint arXiv:1103.2903*.

Minhaneli, Gustavo (2018), Análise de dados reconhecendo emoções em textos com objetivo de obter informações úteis para vendas utilizando as ferramentas Naive Bay Bayes e NLTK com Python, Instituto Ensinar Brasil, Faculdade Doctum Caratinga.

Ministério da Educação. (2017). Programa TEIP 2017-2021: territórios educativos de intervenção prioritária. Lisboa: Direção-Geral da Educação. Disponível em http://www.dge.mec.pt/sites/default/files/Noticias/programa_teip_2017_2021.pdf

Moreira, João, (2020), Deteção de Emoções em argumentos trocados no contexto de tomada de decisão em grupo, Tese de Mestrado, Politécnico do Porto.

Nielsen, F. Å. (2011). A new ANEW: Evaluation of a word list for sentiment analysis. Nordic Journal of Linguistics, 33(2).

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*.

Picard, Rosalind W. (2000). Affective Computing. MIT Press. 308 pp. isbn: 978-0-262-66115-7. Google Books: GaVncRTcb1gC.

Picard, Rosalind, Affective Computing, MIT Press, Cambridge, MA, 1997.

Plano Plurianual de Melhoria (2018-2021). (s.d.). Acedido em 18 de julho de 2023, em https://age-mgpoente.pt/images/2122/maio/PLANO_PLURIANUAL_DE_MELHORIA_VF2_18-21.pdf

Plano Plurianual de Melhorias (2018-2022). (s.d.). Acedido em 17 de julho de 2023, em https://mcctic.ese.ipsantarem.pt/corucheae/wp-content/uploads/2022/05/PPM-18-22-VF.pdf

Pong-Inwong, Chakrit; Kaewmak, Konpusit (2016). [IEEE 2016 2nd IEEE International Conference on Computer and Communications (ICCC) - Chengdu, China (2016.10.14-2016.10.17)] 2016 2nd IEEE International Conference on Computer and Communications (ICCC) - Improved sentiment analysis for teaching evaluation using feature selection and voting ensemble learning integration., (), 1222–1225. doi:10.1109/CompComm.2016.7924899

Rodrigues, Carlos Augusto S. et al. (2011), Mineração de Opinião/ Análise de Sentimentos. UFSC.

Sah, U. K., & Singh, (2022), M. A. Student Career Prediction Using Machine Learning, International Journal of Scientific Development and Research (IJSDR) Department of Computer Science & Engineering, Kalinga University, Raipur (C.G.), India.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*.

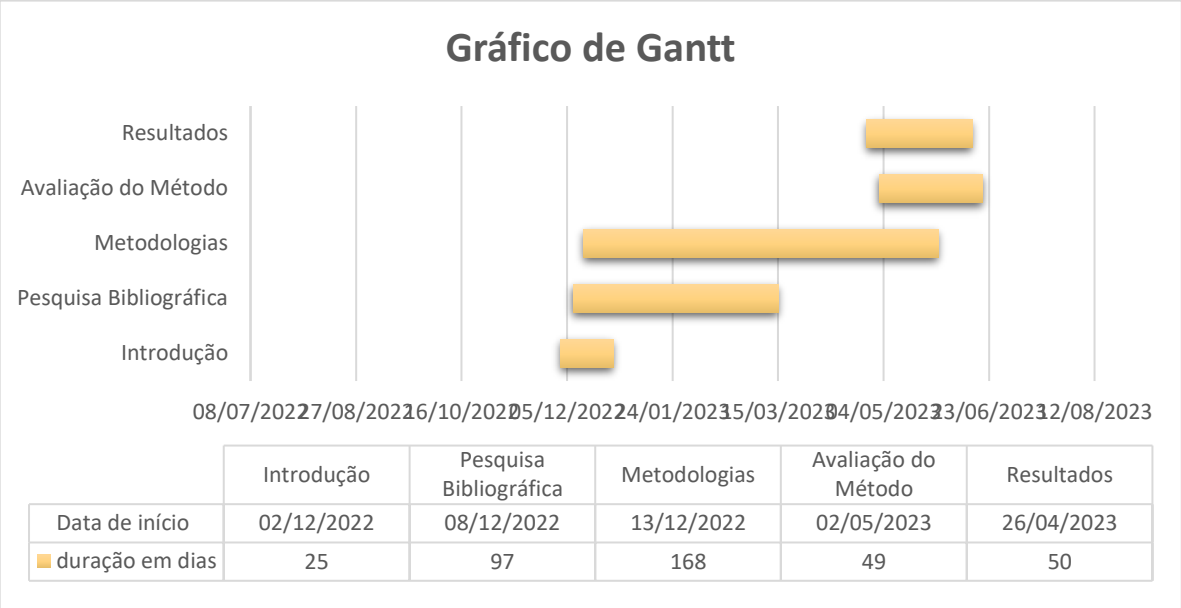
Sultana, J., Sultana, N., Yadav, K., & AlFayez, F. (2018). Prediction of sentiment analysis on educational data based on deep learning approach. In *2018 21st Saudi computer society national computer conference (NCC)*. IEEE.

Suraj Pandey;Md. Shad Akhtar;Tanmoy Chakraborty; (2021). *Syntactically Coherent Text Augmentation for Sequence Classification . IEEE Transactions on Computational Social Systems, (),* –. doi:10.1109/TCSS.2021.3075774

Techopedia. (s.d.). Multilayer Perceptron (MLP) - Definition. Obtido de https://www.techopedia.com/definition/20879/multilayer-perceptron-mlp

ANEXOS

Anexo A - Planeamento



	Introdução	Pesquisa Bibliográfica	Metodologias	Avaliação do Método	Resultados
Data de início	02/12/2022	08/12/2022	13/12/2022	02/05/2023	26/04/2023
duração em dias	25	97	168	49	50

Anexo B - Nota Metodológica

No âmbito da Dissertação do Mestrado em Engenharia de Tecnologias e Sistemas Web "Análise de sentimento textual para correspondência de perfis vocacionais", sob orientação do Professor Doutor Ricardo Rosa Vardasca, do ISLA-Santarém, com o objetivo de obter perfis vocacionais futuros para alunos de 9º ano através de tecnologias de Inteligência Artificial de análise de sentimento textual, elabora-se a seguinte nota metodológica, com as seguintes fases:

1. Revisão de literatura;
2. Construção de léxico;
3. Construção de questionários, aplicação, validação e análise dos mesmos a alunos de 9º ano; Os questionários serão realizados online através de link https://forms.gle/hDGPDZryePw8mrVY6 enviado por email aos diretores de 3 agrupamentos de escolas (Agrupamento de Escolas Eduardo Gageiro, Sacavém, Agrupamento de Escolas Dr. Ginestal Machado, Santarém e Agrupamento de Escolas do Bairro Padre Cruz, Lisboa). O questionário demorará cerca de 15 minutos a preencher e será realizado apenas com autorização de recolha de dados dos alunos pelo Encarregado de Educação, através do consentimento informado enviado previamente e assinado pelo mesmo. A aluna de Mestrado deslocar-se-á às escolas previamente à aplicação dos mesmos para entregar as declarações de Consentimento Informado aos Diretores para serem assinadas pelos Encarregados de Educação. Esta fase decorrerá de 15 de março a 28 de abril de 2023.
4. Treino de algoritmos de *Machine Learning* e *Deep Learning* e selecionar os mais fiáveis e com maior percentagem de rigor; Os algoritmos irão analisar sentimentos positivos e negativos nas palavras escritas pelos alunos nas respostas ao questionário e assim obter-se-ão resultados indicativos de uma área vocacional.
5. Construção de modelo de Inteligência Artificial aplicável à problemática.
6. Validação e análise dos resultados obtidos.

01 de março de 2023

Ana Paula da Conceição Silva

Anexo C – Declaração de Consentimento informado

Eu,_____ (nome completo do encarregado de educação), autorizo a aluna Ana Paula da Conceição Silva do Mestrado em Engenharia de Tecnologias e Sistemas Web, do Instituto Superior de Gestão e Administração (ISLA) de Santarém a proceder à recolha e tratamento dos meus dados pessoais e do meu educando _____(nome completo do aluno) fornecidos num questionário, no âmbito da Dissertação de Mestrado "Análise de sentimento textual para correspondência de perfis vocacionais", sob orientação do Professor Doutor Ricardo Rosa Vardasca, com o objetivo de obter perfis vocacionais futuros para alunos de 9º ano através de tecnologias de Inteligência Artificial de análise de sentimento textual.

Declaro, ainda que fui informado(a) do seguinte:

1. **Dados pessoais recolhidos**

   Através de um questionário feito aos alunos, serão apenas recolhidos dados relevantes para efeitos do referido estudo. Serão recolhidos dados demográficos de alunos e encarregados de educação, tais como, idade, género, grau de escolaridade, entre outros e serão feitas perguntas de opinião aos alunos acerca dos próprios. O questionário é anónimo, identificando pessoalmente apenas a turma e a escola dos alunos.

2. **Finalidade e metodologia da recolha de dados**

   O questionário será realizado online, sendo os dados pessoais utilizados de modo anónimo e apenas para o estudo da Dissertação referida, sendo os resultados divulgados na apresentação pública da mesma no ISLA de Santarém em junho/julho de 2023. A recolha de informação ocorrerá em março/abril de 2023.

3. **Tratamento e armazenamento dos dados pessoais**

   O ISLA de Santarém garante que serão implementadas as medidas necessárias para garantir o anonimato das respostas de todos os participantes. Os dados serão tratados de forma coletiva, não particularizando o aluno, família, professor ou escola.

4. **Conservação dos dados pessoais**

   Os dados pessoais serão armazenados apenas pelo período necessário ao cumprimento das finalidades referidas no ponto 2 da presente declaração.

5. **Direitos dos titulares dos dados pessoais**

   Os titulares de dados pessoais poderão, em relação aos mesmos, pedir informações e solicitar limitações ao seu tratamento ou ainda opor-se ao mesmo. Os titulares de dados pessoais poderão exercer os seus direitos junto do responsável de dados pessoais, enviando um email para: a22109656@islasantarem.pt .

_____

(data e assinatura)

Anexo D –Declaração do Professor Orientador da Dissertação

DECLARAÇÃO

Santarém, 15 de fevereiro de 2023

Serve a presente declaração para confirmar que a estudante **Ana Paula da Conceição Silva** com o número de estudante **22109656** está a realizar sob minha orientação a Dissertação intitulada "Análise de sentimento textual para correspondência de perfis vocacionais" de final de curso de **Mestrado em Engenharia de Tecnologias e Sistemas Web** e mais informo que concordo com a metodologia utilizadas para atingir o objetivo de suportar a tomada de decisão vocacional, instrumentos a aplicar que por mim foram aprovados e com o modelo de consentimento informado e esclarecido do titular dos dados.

Indico estar disponível para quaisquer esclarecimentos adicionais necessários.

Atentamente,

Professor Doutor Ricardo Ângelo Rosa Vardasca
ISLA Santarém
Diretor curso de Mestrado em Engenharia de Tecnologias e Sistemas Web

ISLA Santarém
Largo Cândido dos Reis | 2000 - 241 Santarém | Tel: 243 305 880 | info@islasantarem.pt
www.islasantarem.pt

Anexo E - Questionário de Análise de Sentimento Textual

A presente investigação de Dissertação de Mestrado tem como principal objetivo obter resultados acerca da vocação de aluno(a)s de 9º ano através de análise por um modelo de Inteligência Artificial de sentimento textual e insere-se no Mestrado em Engenharia de Tecnologias e Sistemas Web do ISLA de Santarém.
Para que este estudo seja possível, necessito da tua colaboração!
A tua participação é voluntária, contudo, o teu contributo é da maior importância para o sucesso desta investigação, que pretende obter a a tua vocação/área de prosseguimentos de estudos a seguir ao 9º ano.
A tarefa consiste em preencheres o questionário que se segue procurando ser o mais sincero(a) possível nas tuas respostas, e, não há respostas certas nem erradas, quanto mais escreveres mais possibilidades tem este estudo de ser bem-sucedido.
O questionário demora cerca de 15 minutos a ser preenchido.
O questionário é anónimo, não tens que escrever o teu nome em lado nenhum, e confidencial, mais ninguém terá acesso a ele.
No fim do questionário confirma se respondeste a todas as questões, incluindo o preenchimento dos dados nesta folha.

OBRIGADA pela tua colaboração!

A aluna de Mestrado Ana Paula da Conceição Silva

*Obrigatório

Li a informação acima apresentada e aceito participar neste estudo.*
Sexo*

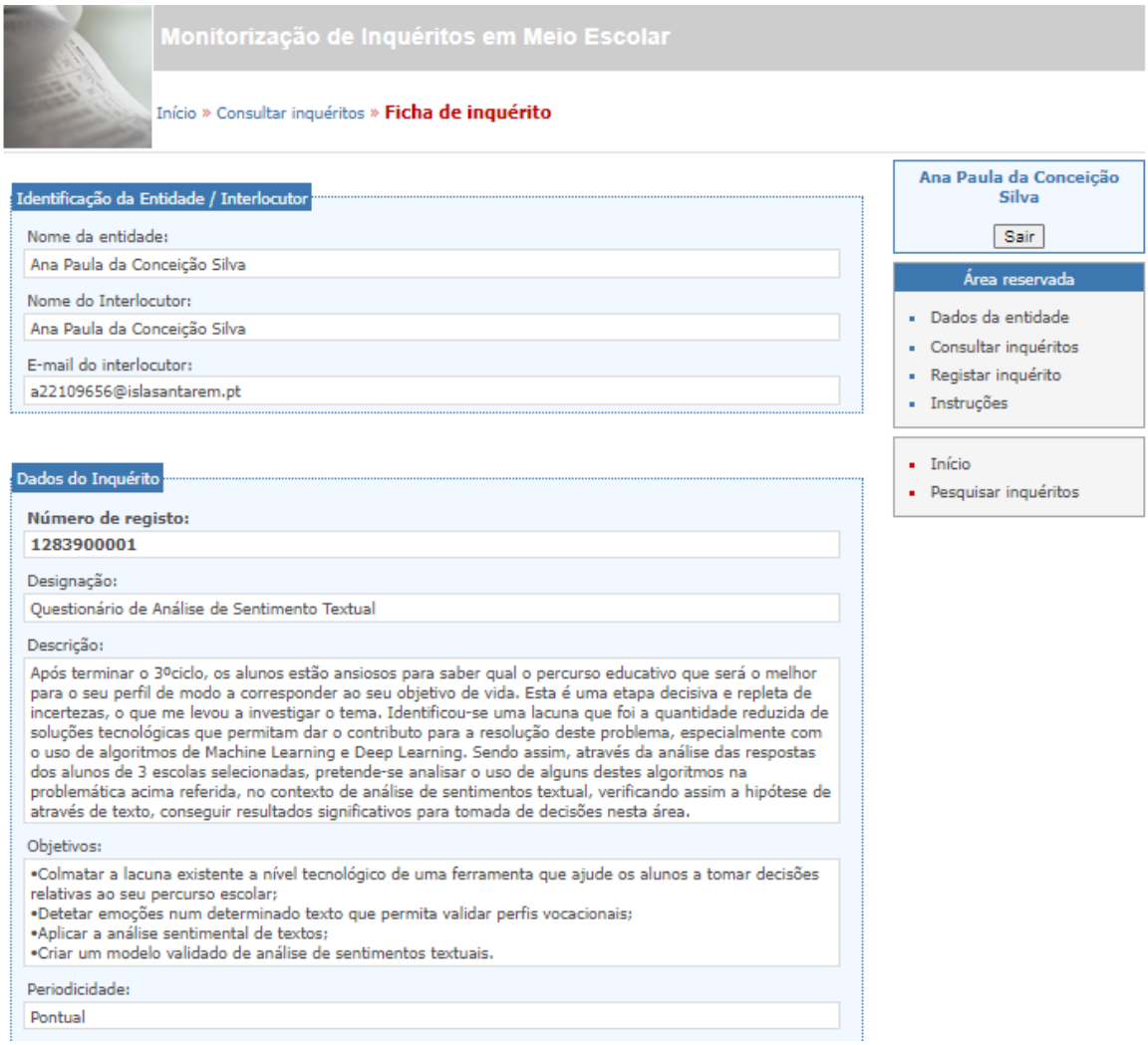
Questionário de Análise de Sentimento Textual
Seguem-se perguntas de resposta longa, do género composição. Sê sincero(a) para que o estudo seja realizado com sucesso e obtenha resultados verdadeiros. Escreve respostas completas. Obrigada.
A análise de sentimento textual processa-se através da polaridade nas palavras, ou seja fornece sentimentos positivos e negativos a determinadas palavras e assim é possível obter-se resultados vocacionais positivos e negativos através do que vais escrever.

Escola que frequentas*
Ano de escolaridade e turma que frequentas*
Que idade tens?*
1. O que esperas alcançar na tua vida adulta? Descreve.*
2.Qual é a profi ssão da tua mãe e planeias seguir o mesmo caminho? Porquê?*
3.Qual é a profissão do teu pai e planeias seguir o mesmo caminho? Porquê?*
4. Qual é a profissão dos teus sonhos? Descreve.*
5. Que profissão é que os teus pais gostariam que tu tivesses? Concordas com eles? Porquê?*
6. Descreve as atividades que gostas mais de fazer nas férias.*
7.Descreve as atividades que gostas mais de fazer nos teus tempos livres.*
8. Qual é a tua disciplina preferida?*
9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve.*

10. O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve.*
11. O que achas da área das ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve.*
12. O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve.*
13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê?*
14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve.*
15. Gostavas de tirar um curso superior? Porquê? Qual seria? *

https://forms.gle/GTBoZVYrXf7czbaq8

Anexo F – Ficha de Inquérito



Data do início do período de recolha de dados:

15-03-2023

Data do fim do período de recolha de dados:

28-04-2023

Universo:

Turmas de 9ºano

Unidade de observação:

Alunos de turmas do 9º ano de 3 Agrupamentos de Escolas (Agrupamento de Escolas Eduardo Gageiro, Sacavém, Agrupamento de Escolas Dr. Ginestal Machado,

Método de recolha de dados:

Questionário online com exportação de dados para ficheiro excel https://forms.gle/hDGPDZryePw8mrVY6

Inquérito registado no Sistema Estatístico Nacional:

Não

Inquérito aplicado pela entidade:

Sim

Instrumento de inquirição:

12839_202303011633_Documento1.pdf (PDF - 112,38 KB)

Nota metodológica:

12839_202303011633_Documento2.pdf (PDF - 109,47 KB)

Outros documentos:

12839_202303011633_Documento3.pdf (PDF - 379,92 KB)

Data de registo:

01-03-2023

Versão:

1 (1)

Dados adicionais

Estado:

Pendente

Avaliação:

Observações:

Outras observações:

Anexo G – Email de Aprovação e Registo do Questionário

mime.noreply@min-educ.pt quarta, 29/03, 09:46

para mim

Exmo(a)s. Sr(a)s.

O pedido de autorização do inquérito n.º 1283900001, com a designação *Questionário de Análise de Sentimento Textual*, registado em 01-03-2023, foi aprovado.

Avaliação do inquérito:

Exmo.(a) Senhor(a) Ana Paula da Conceição Silva

Cumpre-nos informar que o pedido de realização de inquérito em meio escolar é aprovado uma vez que, submetido a análise, cumpre os requisitos, devendo atender-se às observações aduzidas, devendo, contudo, ser dado conhecimento das mesmas ao/à orientador/a responsável pelo estudo/investigação académica.

Com os melhores cumprimentos

José Carlos Sousa

Diretor de Serviços

DGE

Observações:

a) A realização dos Inquéritos fica sujeita a autorização das Direções dos Agrupamentos de Escolas do ensino público a contactar para a realização do estudo. Merece especial atenção o modo, o momento e condições de aplicação dos instrumentos de recolha de dados em meio escolar, devendo fazer-se em estreita articulação com as Direções dos Agrupamento de Escolas e com os encarregados de educação ou quem tutele os menores.
b) Deve considerar-se o disposto legal em matéria de garantia de anonimato dos sujeitos e da sua não identificabilidade, confidencialidade, proteção e segurança dos dados pessoais a recolher e tratar no presente estudo, devendo prever-se medidas adequadas e específicas para a defesa dos direitos fundamentais e dos interesses do titular dos dados. Deste modo, procura-se garantir o tratamento lícito dos mesmos e a conformidade com os termos procedimentais indicados e legislação em vigor. Considerados os documentos que foram anexados e para efeitos da proteção de dados a recolher junto dos inquiridos resultam obrigações que o responsável se propõe cumprir, enunciadas nos documentos apresentados. Destas deve dar conhecimento a todos os inquiridos e a quem intervenha na recolha e tratamento de dados. É obrigatório recolher o consentimento inequívoco,

informado e esclarecido, junto dos inquiridos, titulares dos dados, no caso presente alunos menores, através dos seus representantes legais. Recomenda-se que, dado o exposto, para efeitos de proteção de dados e cumprimento do disposto legal, o orientador da investigação académica e o/a Encarregado/a de Proteção de Dados da entidade de ensino superior responsável pelo estudo possam apoiar todo o processo, ponderando obviar dados que possam ser considerados excessivos face aos objetivos e finalidades declaradas para o estudo e não garantem a não identificabilidade do respondente (ie: Escola de frequência - substituir por um código; dado turma), e bem assim acionar medidas de salvaguarda previstas na lei para segurança dos dados pessoais e devida proteção dos titulares.

c) Deve ser dado conhecimento das observações acima aduzidas ao/à orientador/a responsável pelo estudo/investigação académica.
d) Atendendo à data de início do período de recolha de dados indicada, 15-03-2023, já vencida, informa-se que a aplicação dos inquéritos só deverá ocorrer em data posterior à sua aprovação.

Pode consultar na Internet toda a informação referente a este pedido no endereço http://mime.dgeec.mec.pt. Para tal terá de se autenticar fornecendo os dados de acesso da entidade.

Anexo H – Email enviado às Escolas

Estimada Direção / Membros de Direção,

O meu nome é Ana Paula Silva e sou aluna do Mestrado em Engenharia de Tecnologias e Sistemas Web, no ISLA de Santarém.

No âmbito do mestrado, estou a desenvolver uma investigação para a minha Dissertação intitulada "Análise de Sentimento Textual para Correspondência de Perfis Vocacionais", com os objetivos de colmatar a lacuna existente a nível tecnológico de uma ferramenta que ajude os alunos a tomar decisões relativas ao seu percurso escolar, detetar emoções num determinado texto que permita validar perfis vocacionais, aplicar a análise sentimental de textos e criar um modelo validado de análise de sentimentos textuais, numa amostra de alunos do 9ºano. Escolher o percurso educativo após o 3º ciclo é uma etapa decisiva e repleta de incertezas, e o meu contributo será, através do uso de algoritmos de Machine Learning e Deep Learning, analisar as respostas dos alunos de 3 escolas selecionadas, no contexto de análise de sentimentos textual, verificando a hipótese de conseguir resultados significativos para tomada de decisões nesta área. Ou seja, a investigação incide sobre os alunos do 9º ano e as suas decisões vocacionais futuras, através de análise das respostas dos questionários com algoritmos de Inteligência artificial.

Este inquérito foi aprovado pela **DGE/MIME** (Direção Geral de Educação / Monitorização de Inquéritos em Meio Escolar), tendo sido registado com o número 1283900001, e o processo pode ser consultado em http://mime.dgeec.mec.pt .

Solicito a aplicação dos questionários online nas turmas de 9º ano que tenham no vosso Agrupamento/Escola e a partilha deste email com todos os Diretores de Turma de turmas de 9º ano do vosso agrupamento/escola.

Reforço que nunca foi realizado um estudo neste âmbito, utilizando tecnologias de inteligência artificial de análise de sentimento textual em Portugal, pelo que os resultados obtidos com as respostas aos questionários, poderão ser muito relevantes para obtenção de resultados que podem vir a ser significativos nesta investigação e investigações futuras. Assim, a participação da sua escola revela-se imprescindível para o sucesso deste estudo.

A participação é voluntária e os questionários são anónimos e poderão interromper os mesmos a qualquer momento. Devido à metodologia de análise de resultados ser de análise de sentimento em texto, solicito que seja pedido aos alunos que escrevam o mais possível, dando respostas completas, descritivas e detalhadas e verdadeiras. Sugiro que os questionários possam ser respondidos em aulas com computadores para todos os alunos, como por exemplo em aulas de TIC, solicitando a colaboração aos docentes de TIC na aplicação dos mesmos.

Os dados fornecidos são absolutamente confidenciais e anónimos e serão exclusivamente utilizados para fins desta investigação. O preenchimento deste questionário demora cerca de 15-20 minutos.

Em anexo segue o documento de consentimento informado para entrega aos Encarregados de Educação, de modo a tomarem conhecimento e autorizarem a participação dos seus educandos no referido estudo. Esta autorização deve ser recolhida o mais breve possível para se aplicarem os questionários aos alunos ainda no mês de abril. Pode ser enviada por email aos EE pelos Diretores de Turma ou impressas para os mesmos assinarem e devolverem para se poder analisar as respostas dos alunos ao questionário. Poderei deslocar-me à Escola/Agrupamento e deixar com a Direção o documento com o consentimento informado que envio em anexo, em formato impresso, necessitando de saber o número de alunos do 9º ano para saber a quantidade de consentimentos informados a imprimir. Se possível, poderia ir nas duas primeiras semanas de aulas após a Páscoa para deixar esses documentos, em data a agendar, ou se preferirem poderão imprimir na vossa escola e eu irei recolher posteriormente. Segue também em anexo um documento com a metodologia a utilizar na análise das respostas, bem como uma declaração de aprovação da metodologia pelo meu orientador de Dissertação, o Professor Doutor Ricardo Vardasca.

Para aceder ao inquérito, clique na seguinte hiperligação: https://forms.gle/hDGPDZryePw8mrVY6

Melhores cumprimentos,

Ana Paula Silva

a22109656@islasantarem.pt

Anexo I- Código da Pergunta com o número 15

```
import pandas as pd
import matplotlib.pyplot as plt

# Ler o ficheiro CSV num DataFrame
df = pd.read_csv('dadosquestnormalizado.csv', sep=';', encoding='unicode_escape')

# Identificar o nome das colunas
pergunta_coluna = '15. Gostavas de tirar um curso superior? Porquê? Qual seria?'
escola_coluna = 'Escola que frequentas'
sexo_coluna = 'Sexo'
idade_coluna = 'Que idade tens?'

# Função para verificar se uma resposta contém palavras positivas, neutras ou negativas
def categorizar_resposta(resposta):
    resposta_lower = resposta.lower()
    if any(palavra in resposta_lower for palavra in palavras_positivas):
        return 'Positiva'
    elif any(palavra in resposta_lower for palavra in palavras_neutras):
        return 'Neutra'
    elif any(palavra in resposta_lower for palavra in palavras_negativas):
        return 'Negativa'
    else:
        return 'Não categorizável'

# Palavras-chave para identificar respostas positivas
palavras_positivas = [
    "sim", "claro", "com certeza", "gostava", "gostaria", "adorava", "sim, gostaria", "gostava sim", "acho que sim", "amava", "eu adoraria", "curso de", "política", "desporto", "saída para o futuro", "universidade de", "mecânica", "educação", "secretariado", "designer", "é uma boa forma", "engenharia", "pode ajudar-me", "quero", "penso que sim", "agora quero", "gestão", "tiraria", "eu gostaria", "queria", "medica"
]

# Palavras-chave para identificar respostas neutras
palavras_neutras = [
```

```
    "talvez", "indeciso", "não sei"
]

# Palavras-chave para identificar respostas negativas
palavras_negativas = [
    "não"
]

# Aplicar a função para categorizar respostas
df['Categoria'] = df[pergunta_coluna].apply(categorizar_resposta)

# Contagem das categorias
contagem_categorias = df['Categoria'].value_counts()

# Guardar os resultados num Excel para análise da pergunta 15
resultado_excel = pd.ExcelWriter('resultados_questao15.xlsx', engine='openpyxl')

# Guardar a contagem das categorias
contagem_categorias.to_excel(resultado_excel, sheet_name='Contagem_Categorias',
index=True)

# Contagem por sexo
contagem_sexo = df.groupby([sexo_coluna, 'Categoria']).size().unstack()

# Contagem por escola
contagem_escola = df.groupby([escola_coluna, 'Categoria']).size().unstack()

# Contagem por idade
contagem_idade = df.groupby([idade_coluna, 'Categoria']).size().unstack()

# Plotar gráficos de barras
contagem_sexo.plot(kind='bar', stacked=True)
plt.title('Categorização das Respostas por Sexo')
plt.xlabel('Sexo')
plt.ylabel('Número de Respostas')
```

```
    plt.show()

    contagem_escola.plot(kind='bar', stacked=True)
    plt.title('Categorização das Respostas por Escola')
    plt.xlabel('Escola')
    plt.ylabel('Número de Respostas')
    plt.show()

    contagem_idade.plot(kind='bar', stacked=True)
    plt.title('Categorização das Respostas por Idade')
    plt.xlabel('Idade')
    plt.ylabel('Número de Respostas')
    plt.show()

    # Guardar a contagem por sexo, escola e idade em folhas separadas
    contagem_sexo.to_excel(resultado_excel, sheet_name='Contagem_Sexo', index=True)
    contagem_escola.to_excel(resultado_excel, sheet_name='Contagem_Escola',
index=True)
    contagem_idade.to_excel(resultado_excel, sheet_name='Contagem_Idade', index=True)

    # Fechar o Excel
    resultado_excel.save()

    # Exibir uma mensagem indicando que o ficheiro foi guardado
    print("Os resultados da pergunta 15 foram guardados em 'resultados_questao15.xlsx'")
```

Anexo J- Código dos Léxicos

```
# -*- coding: utf-8 -*-
"""
Created on Thu Jul 27 17:35:15 2023

@author: Admin
"""
import nltk
from nltk.stem import RSLPStemmer
from nltk.tokenize import word_tokenize

# Instalar os recursos do nltk para a língua portuguesa (somente na primeira execução)
nltk.download('rslp')
nltk.download('punkt')

# Criar um objeto de lematizador para o português
stemmer = RSLPStemmer()

# Função para lematizar o texto
def lematizar_texto(texto):
    palavras = word_tokenize(texto.lower())  # Tokenização e conversão para minúsculas
    lematizado = [stemmer.stem(palavra) for palavra in palavras]
    return lematizado

# Função para lematizar o léxico
def lematizar_lexico(lexico):
```

```
    lematizado = [stemmer.stem(palavra) for palavra in lexico]
    return lematizado

# Função para avaliar a conotação de um conjunto de palavras numa área específica
def avaliar_conotacao(conjunto_palavras, lexico):
    conotacao = 0
    if len(conjunto_palavras) == 1:
        if conjunto_palavras[0] in lexico:
            conotacao = 1  # Sentimento positivo
    elif len(conjunto_palavras) == 2:
        if all(palavra in lexico for palavra in conjunto_palavras):
            conotacao = 0  # Sentimento neutro
    elif len(conjunto_palavras) == 3:
        if all(palavra in lexico for palavra in conjunto_palavras):
            conotacao = -1  # Sentimento negativo
    return conotacao

# Definir a lista de perguntas na ordem correspondente
perguntas = [
    "1. O que esperas alcançar na tua vida adulta? Descreve",
    "2. Qual é a profissão da tua mãe e planeias seguir o mesmo caminho? Porquê?",
    "3. Qual é a profissão do teu pai e planeias seguir o mesmo caminho? Porquê?",
    "4. Qual é a profissão dos teus sonhos? Descreve",
    "5. Que profissão é que os teus pais gostariam que tu tivesses? Concordas com eles? Porquê?",
    "6. Descreve as atividades que gostas mais de fazer nas férias",
```

```
    "7. Descreve as atividades que gostas mais de fazer nos teus tempos livres",
    "8. Qual é a tua disciplina preferida?",

]

# Palavra única
lexico_ciencias_tecnologias_1 = [
    "tecnologia", "sismologo", "computador", "aviação", "matemática", "zootécnica", "veterinária", "matemática", "programar", "computadores", "informática", "médico", "cirurgia", "morgue", "medicina", "saúde", "biologia", "ciências", "engenharia", "cardiologista", "forense", "médica", "bombeiro", "informatico", "arqueologia", "equestre", "saude", "fisioterapeuta", "cirurgiã", "Engenharia", "neurociencia", "Biólogo"
]

# Grupos de 2 palavras
lexico_ciencias_tecnologias_2 = [
    "ciências tecnológicas", "detetive", "natureza", "Médico Veterinário", "Saúde Biologia", "cirurgia plástica", "Ciências Naturais", "ciências tecnologias", "saúde pública", "engenharia civil", "engenharia informática", "arquitetura moderna", "medicina veterinária"
]
# Grupos de 3 palavras
lexico_ciencias_tecnologias_3 = [
    "conhecimentos de informática", "profissão do futuro", "mexer no computador", "inovação e tecnologia", "engenharia de software", "Técnico de informática"
]
# Palavra única
lexico_ciencias_socioeconomicas_1 = [
```

"ciências", "empresaria", "Contabilista", "economia", "psicólogo", "imobiliária", "sociologia", "contabilidade", "psicologia", "negócios", "consumo", "comércio", "política", "educação", "cultura", "voluntariado", "advogado", "advogada", "Gestor", "empresário", "banco", "psicóloga", "negócio", "empreendedor", "Empreendedor", "loja"

]

Grupos de 2 palavras

lexico_ciencias_socioeconomicas_2 = [

"ciências sociais", "economia contabilidade", "sociologia psicologia", "gestão empresarial", "negócios empresariais", "consumo e marketing", "técnico comercial", "marketing publicidade", "recursos humanos", "ciências económicas", "ciências socioeconómicas", "Marketing Digital"

]

Grupos de 3 palavras

lexico_ciencias_socioeconomicas_3 = [

"publicidade e marketing", "gestão de recursos humanos", "mulher de negócios", "Gestão de empresas"

]

Palavra única

lexico_linguas_humanidades_1 = [

"línguas", "humanidades", "comunicação", "direito", "professor", "história", "educação", "escrita", "leitura", "turismo", "sociedade", "escrever", "arqueologia", "professora", "Jornalista"

]

Grupos de 2 palavras

lexico_linguas_humanidades_2 = [

"línguas estrangeiras", "história antiga", "educação inclusiva", "leitura crítica", "pesquisa educacional"

]

Grupos de 3 palavras

lexico_linguas_humanidades_3 = [

"turismo e cultura", "sociedade e cultura", "línguas e humanidades", "educadora de infância"

]

Palavra única

lexico_artes_visuais_1 = [

"artes", "arquiteta", "Pediatra", "desenho", "moda", "artística", "cores", "texturas", "desenhar", "moldes", "pintura", "arquitetura", "criatividade", "escultura", "geometria", "fotografia", "design", "multimédia", "arquitecta", "estilista"

]

Grupos de 2 palavras

lexico_artes_visuais_2 = [

"artes visuais", "desenho geométrico", "escultura contemporânea", "design gráfico", "multimédia interativa",

]

Grupos de 3 palavras

lexico_artes_visuais_3 = [

"fotografia de moda", "teatro de fantoches", "artes visuais multimédia", "designer de moda", "Pintor de quadros", "empresa de roupa"

]

```
# Palavra única
lexico_profissional_informatica_eletronica_comunicacoes_1 = [
    "informática", "eletrónica", "telecomunicações", "computadores", "mecatrónica", "programação", "computador", "mecânico", "informatico", "Programador", "eletricista", "Tecnologia"
]

# Grupos de 2 palavras
lexico_profissional_informatica_eletronica_comunicacoes_2 = [
    "programação web", "suporte técnico", "desenvolvimento mobile", "técnico informática", "programação informática"
]

# Grupos de 3 palavras
lexico_profissional_informatica_eletronica_comunicacoes_3 = [
    "análise de sistemas", "segurança da informação", "Programador de computadores", "técnico de informática", "operador de eletrónica", "sistemas de informação", "programador de informática", "Mecânico de automóvel", "Técnico de informática"
]

# Palavra única
lexico_profissional_turismo_lazer_1 = [
    "turismo", "desporto", "jogador", "jogadora", "natureza", "jogar", "Militar", "atleta", "cozinheiro", "cozinhar", "futebol", "animação", "hotelaria", "restauração", "champions", "equestre", "futebolistas", "internacional", "recepcionista", "clube", "equipe", "treinador", "treinadora", "Hospedeira", "hospedeira", "futebolista", "aeromoça", "cozinheira"
]
```

Grupos de 2 palavras

lexico_profissional_turismo_lazer_2 = [

"gestão hoteleira", "animação turística", "Técnico desportivo", "treinador futebol", "educação física", "Lutador Profissional"

]

Grupos de 3 palavras

lexico_profissional_turismo_lazer_3 = [

"cozinheiro de primeira", "Empregado de Restaurante", "treinador de futebol", "técnico de turismo", "jogador de futebol", "jogar no benfica", "jogador de basquete", "profissional de futebol", "Assistente de bordo"

]

Palavra única

lexico_profissional_cultura_patrimonio_conteudos_1 = [

"cultura", "actor", "cinema", "músico", "ator", "actriz", "atriz", "multimédia", "moda", "Modelo", "música", "dança", "fotografia", "audiovisuais", "artes", "designer", "influenciador", "fotógrafo", "Gestora de Redes Sociais"

]

Grupos de 2 palavras

lexico_profissional_cultura_patrimonio_conteudos_2 = [

"dança contemporânea", "fotografia documental", "produtos multimédia", "artes gráficas", "vídeo", "edição gráfica", "som e imagem", "designer multimédia", "designer gráfico", "Marketing Digital"

]

Grupos de 3 palavras

lexico_profissional_cultura_patrimonio_conteudos_3 = [

"desenvolvimento de conteúdos", "operador de câmara", "designer de interiores", "técnico de multimédia", "Produção de som", "produção de vídeo", "designer de jogos"

]

Adicionar 'None' para os léxicos que não têm valores

lexico_ciencias_socioeconomicas_1 = None

lexico_ciencias_socioeconomicas_2 = None

lexico_ciencias_socioeconomicas_3 = None

lexico_linguas_humanidades_1 = None

lexico_linguas_humanidades_2 = None

lexico_linguas_humanidades_3 = None

lexico_artes_visuais_1 = None

lexico_artes_visuais_2 = None

lexico_artes_visuais_3 = None

lexico_profissional_informatica_eletronica_comunicacoes_1 = None

lexico_profissional_informatica_eletronica_comunicacoes_2 = None

lexico_profissional_informatica_eletronica_comunicacoes_3 = None

lexico_profissional_turismo_lazer_1 = None

lexico_profissional_turismo_lazer_2 = None

lexico_profissional_turismo_lazer_3 = None

lexico_profissional_cultura_patrimonio_conteudos_1 = None

```
lexico_profissional_cultura_patrimonio_conteudos_2 = None
lexico_profissional_cultura_patrimonio_conteudos_3 = None


# Dicionário com os léxicos de cada área, incluindo os valores 'None'
lexicos = {
    "Ciências e Tecnologias": [
        lexico_ciencias_tecnologias_1,
        lexico_ciencias_tecnologias_2,
        lexico_ciencias_tecnologias_3
    ],
    "Ciências Socioeconómicas": [
        lexico_ciencias_socioeconomicas_1,
        lexico_ciencias_socioeconomicas_2,
        lexico_ciencias_socioeconomicas_3
    ],
    "Línguas e Humanidades": [
        lexico_linguas_humanidades_1,
        lexico_linguas_humanidades_2,
        lexico_linguas_humanidades_3
    ],
    "Artes Visuais": [
        lexico_artes_visuais_1,
        lexico_artes_visuais_2,
        lexico_artes_visuais_3
    ],
    "Profissional informatica eletronica e comunicacoes": [
```

```
    lexico_profissional_informatica_eletronica_comunicacoes_1,
    lexico_profissional_informatica_eletronica_comunicacoes_2,
    lexico_profissional_informatica_eletronica_comunicacoes_3
],
"Profissional turismo e lazer": [
    lexico_profissional_turismo_lazer_1,
    lexico_profissional_turismo_lazer_2,
    lexico_profissional_turismo_lazer_3
],
"Profissional cultura, patrimonio e gestão de conteudos": [
    lexico_profissional_cultura_patrimonio_conteudos_1,
    lexico_profissional_cultura_patrimonio_conteudos_2,
    lexico_profissional_cultura_patrimonio_conteudos_3
],

"1. O que esperas alcançar na tua vida adulta? Descreve".strip(): [
    lexico_ciencias_tecnologias_1,
    lexico_ciencias_tecnologias_2,
    lexico_ciencias_tecnologias_3,
    lexico_ciencias_socioeconomicas_1,
    lexico_ciencias_socioeconomicas_2,
    lexico_ciencias_socioeconomicas_3,
    lexico_linguas_humanidades_1,
    lexico_linguas_humanidades_2,
    lexico_linguas_humanidades_3,
    lexico_artes_visuais_1,
```

lexico_artes_visuais_2,

lexico_artes_visuais_3,

lexico_profissional_informatica_eletronica_comunicacoes_1,

lexico_profissional_informatica_eletronica_comunicacoes_2,

lexico_profissional_informatica_eletronica_comunicacoes_3,

lexico_profissional_turismo_lazer_1,

lexico_profissional_turismo_lazer_2,

lexico_profissional_turismo_lazer_3,

lexico_profissional_cultura_patrimonio_conteudos_1,

lexico_profissional_cultura_patrimonio_conteudos_2,

lexico_profissional_cultura_patrimonio_conteudos_3

],

"2. Qual é a profissão da tua mãe e planeias seguir o mesmo caminho? Porquê?".strip():

[

lexico_ciencias_tecnologias_1,

lexico_ciencias_tecnologias_2,

lexico_ciencias_tecnologias_3,

lexico_ciencias_socioeconomicas_1,

lexico_ciencias_socioeconomicas_2,

lexico_ciencias_socioeconomicas_3,

lexico_linguas_humanidades_1,

lexico_linguas_humanidades_2,

lexico_linguas_humanidades_3,

lexico_artes_visuais_1,

lexico_artes_visuais_2,

lexico_artes_visuais_3,

lexico_profissional_informatica_eletronica_comunicacoes_1,

lexico_profissional_informatica_eletronica_comunicacoes_2,

lexico_profissional_informatica_eletronica_comunicacoes_3,

lexico_profissional_turismo_lazer_1,

lexico_profissional_turismo_lazer_2,

lexico_profissional_turismo_lazer_3,

lexico_profissional_cultura_patrimonio_conteudos_1,

lexico_profissional_cultura_patrimonio_conteudos_2,

lexico_profissional_cultura_patrimonio_conteudos_3

],

"3. Qual é a profissão do teu pai e planeias seguir o mesmo caminho? Porquê?".strip():

[

lexico_ciencias_tecnologias_1,

lexico_ciencias_tecnologias_2,

lexico_ciencias_tecnologias_3,

lexico_ciencias_socioeconomicas_1,

lexico_ciencias_socioeconomicas_2,

lexico_ciencias_socioeconomicas_3,

lexico_linguas_humanidades_1,

lexico_linguas_humanidades_2,

lexico_linguas_humanidades_3,

lexico_artes_visuais_1,

lexico_artes_visuais_2,

lexico_artes_visuais_3,

lexico_profissional_informatica_eletronica_comunicacoes_1,

lexico_profissional_informatica_eletronica_comunicacoes_2,

```
    lexico_profissional_informatica_eletronica_comunicacoes_3,
    lexico_profissional_turismo_lazer_1,
    lexico_profissional_turismo_lazer_2,
    lexico_profissional_turismo_lazer_3,
    lexico_profissional_cultura_patrimonio_conteudos_1,
    lexico_profissional_cultura_patrimonio_conteudos_2,
    lexico_profissional_cultura_patrimonio_conteudos_3
],
"4. Qual é a profissão dos teus sonhos? Descreve".strip(): [
    lexico_ciencias_tecnologias_1,
    lexico_ciencias_tecnologias_2,
    lexico_ciencias_tecnologias_3,
    lexico_ciencias_socioeconomicas_1,
    lexico_ciencias_socioeconomicas_2,
    lexico_ciencias_socioeconomicas_3,
    lexico_linguas_humanidades_1,
    lexico_linguas_humanidades_2,
    lexico_linguas_humanidades_3,
    lexico_artes_visuais_1,
    lexico_artes_visuais_2,
    lexico_artes_visuais_3,
    lexico_profissional_informatica_eletronica_comunicacoes_1,
    lexico_profissional_informatica_eletronica_comunicacoes_2,
    lexico_profissional_informatica_eletronica_comunicacoes_3,
    lexico_profissional_turismo_lazer_1,
    lexico_profissional_turismo_lazer_2,
```

lexico_profissional_turismo_lazer_3,

lexico_profissional_cultura_patrimonio_conteudos_1,

lexico_profissional_cultura_patrimonio_conteudos_2,

lexico_profissional_cultura_patrimonio_conteudos_3

],

"5. Que profissão é que os teus pais gostariam que tu tivesses? Concordas com eles? Porquê?".strip(): [

lexico_ciencias_tecnologias_1,

lexico_ciencias_tecnologias_2,

lexico_ciencias_tecnologias_3,

lexico_ciencias_socioeconomicas_1,

lexico_ciencias_socioeconomicas_2,

lexico_ciencias_socioeconomicas_3,

lexico_linguas_humanidades_1,

lexico_linguas_humanidades_2,

lexico_linguas_humanidades_3,

lexico_artes_visuais_1,

lexico_artes_visuais_2,

lexico_artes_visuais_3,

lexico_profissional_informatica_eletronica_comunicacoes_1,

lexico_profissional_informatica_eletronica_comunicacoes_2,

lexico_profissional_informatica_eletronica_comunicacoes_3,

lexico_profissional_turismo_lazer_1,

lexico_profissional_turismo_lazer_2,

lexico_profissional_turismo_lazer_3,

lexico_profissional_cultura_patrimonio_conteudos_1,

```
    lexico_profissional_cultura_patrimonio_conteudos_2,
    lexico_profissional_cultura_patrimonio_conteudos_3
],
"6. Descreve as atividades que gostas mais de fazer nas férias".strip(): [
    lexico_ciencias_tecnologias_1,
    lexico_ciencias_tecnologias_2,
    lexico_ciencias_tecnologias_3,
    lexico_ciencias_socioeconomicas_1,
    lexico_ciencias_socioeconomicas_2,
    lexico_ciencias_socioeconomicas_3,
    lexico_linguas_humanidades_1,
    lexico_linguas_humanidades_2,
    lexico_linguas_humanidades_3,
    lexico_artes_visuais_1,
    lexico_artes_visuais_2,
    lexico_artes_visuais_3,
    lexico_profissional_informatica_eletronica_comunicacoes_1,
    lexico_profissional_informatica_eletronica_comunicacoes_2,
    lexico_profissional_informatica_eletronica_comunicacoes_3,
    lexico_profissional_turismo_lazer_1,
    lexico_profissional_turismo_lazer_2,
    lexico_profissional_turismo_lazer_3,
    lexico_profissional_cultura_patrimonio_conteudos_1,
    lexico_profissional_cultura_patrimonio_conteudos_2,
    lexico_profissional_cultura_patrimonio_conteudos_3
],
```

```
    "7. Descreve as atividades que gostas mais de fazer nos teus tempos livres".strip(): [
        lexico_ciencias_tecnologias_1,
        lexico_ciencias_tecnologias_2,
        lexico_ciencias_tecnologias_3,
        lexico_ciencias_socioeconomicas_1,
        lexico_ciencias_socioeconomicas_2,
        lexico_ciencias_socioeconomicas_3,
        lexico_linguas_humanidades_1,
        lexico_linguas_humanidades_2,
        lexico_linguas_humanidades_3,
        lexico_artes_visuais_1,
        lexico_artes_visuais_2,
        lexico_artes_visuais_3,
        lexico_profissional_informatica_eletronica_comunicacoes_1,
        lexico_profissional_informatica_eletronica_comunicacoes_2,
        lexico_profissional_informatica_eletronica_comunicacoes_3,
        lexico_profissional_turismo_lazer_1,
        lexico_profissional_turismo_lazer_2,
        lexico_profissional_turismo_lazer_3,
        lexico_profissional_cultura_patrimonio_conteudos_1,
        lexico_profissional_cultura_patrimonio_conteudos_2,
        lexico_profissional_cultura_patrimonio_conteudos_3
    ],

}
```

```
# Dicionário que mapeia disciplinas preferidas às áreas correspondentes
disciplina_lexico = {
    "Matemática": ["Ciências e Tecnologias", "Profissional informatica eletronica e comunicacoes"],
    "História": ["Ciências Socioeconómicas", "Profissional cultura, patrimonio e gestão de conteudos", "Línguas e Humanidades"],
    "Geografia": ["Ciências Socioeconómicas", "Ciências e Tecnologias", "Profissional turismo e lazer"],
    "Português": ["Línguas e Humanidades", "Profissional cultura, patrimonio e gestão de conteudos"],
    "Inglês": ["Línguas e Humanidades", "Profissional turismo e lazer"],
    "Françês": ["Línguas e Humanidades", "Profissional turismo e lazer"],
    "Educação Visual": ["Artes Visuais", "Profissional cultura, patrimonio e gestão de conteudos"],
    "Educação Tecnológica": ["Artes Visuais", "Profissional cultura, patrimonio e gestão de conteudos"],
    "TIC": ["Ciências e Tecnologias", "Profissional informatica eletronica e comunicacoes", "Profissional cultura, patrimonio e gestão de conteudos"],
    "Educação Física": ["Ciências e Tecnologias", "Profissional turismo e lazer"],
    "Físico-Química": ["Ciências e Tecnologias"],
    "Ciências Naturais": ["Ciências e Tecnologias", "Profissional turismo e lazer"],
    "Cidadania e Desenvolvimento": ["Ciências Socioeconómicas"],
    "Espanhol": ["Línguas e Humanidades", "Profissional turismo e lazer"],
}

# Exemplo para a tokenização de léxicos
def tokenizar_lexico(lexico):
    if lexico is not None:
```

```
            lexico_texto = ' '.join(lexico)
            palavras_tokenizadas = nltk.word_tokenize(lexico_texto)
            return palavras_tokenizadas
        else:
            return []

    # Tokenização dos léxicos de Ciências e Tecnologias
    token_ciencias_tecnologias_1 = tokenizar_lexico(lexico_ciencias_tecnologias_1)
    token_ciencias_tecnologias_2 = tokenizar_lexico(lexico_ciencias_tecnologias_2)
    token_ciencias_tecnologias_3 = tokenizar_lexico(lexico_ciencias_tecnologias_3)

    # Tokenização dos léxicos de Ciências Socioeconómicas
    token_ciencias_socioeconomicas_1 = tokenizar_lexico(lexico_ciencias_socioeconomicas_1)
    token_ciencias_socioeconomicas_2 = tokenizar_lexico(lexico_ciencias_socioeconomicas_2)
    token_ciencias_socioeconomicas_3 = tokenizar_lexico(lexico_ciencias_socioeconomicas_3)

    # Tokenização dos léxicos de Línguas e Humanidades
    token_linguas_humanidades_1 = tokenizar_lexico(lexico_linguas_humanidades_1)
    token_linguas_humanidades_2 = tokenizar_lexico(lexico_linguas_humanidades_2)
    token_linguas_humanidades_3 = tokenizar_lexico(lexico_linguas_humanidades_3)

    # Tokenização dos léxicos de Artes Visuais
    token_artes_visuais_1 = tokenizar_lexico(lexico_artes_visuais_1)
    token_artes_visuais_2 = tokenizar_lexico(lexico_artes_visuais_2)
```

```
    token_artes_visuais_3 = tokenizar_lexico(lexico_artes_visuais_3)

    # Tokenização dos léxicos de Profissional informatica eletronica comunicacoes
    token_profissional_informatica_eletronica_comunicacoes_1 = tokenizar_lexico(lexico_profissional_informatica_eletronica_comunicacoes_1)
    token_profissional_informatica_eletronica_comunicacoes_2 = tokenizar_lexico(lexico_profissional_informatica_eletronica_comunicacoes_2)
    token_profissional_informatica_eletronica_comunicacoes_3 = tokenizar_lexico(lexico_profissional_informatica_eletronica_comunicacoes_3)

    # Tokenização dos léxicos de Profissional turismo_lazer
    token_profissional_turismo_lazer_1 = tokenizar_lexico(lexico_profissional_turismo_lazer_1)
    token_profissional_turismo_lazer_2 = tokenizar_lexico(lexico_profissional_turismo_lazer_2)
    token_profissional_turismo_lazer_3 = tokenizar_lexico(lexico_profissional_turismo_lazer_3)

    # Tokenização dos léxicos de Profissional cultura patrimonio conteudos
    token_profissional_cultura_patrimonio_conteudos_1 = tokenizar_lexico(lexico_profissional_cultura_patrimonio_conteudos_1)
    token_profissional_cultura_patrimonio_conteudos_2 = tokenizar_lexico(lexico_profissional_cultura_patrimonio_conteudos_2)
    token_profissional_cultura_patrimonio_conteudos_3 = tokenizar_lexico(lexico_profissional_cultura_patrimonio_conteudos_3)
```

```
# Exemplos de uso
print("Léxico de Ciências e Tecnologias (Palavra única):", token_ciencias_tecnologias_1)
print("Léxico de Ciências e Tecnologias (Grupos de 2 palavras):", token_ciencias_tecnologias_2)
print("Léxico de Ciências e Tecnologias (Grupos de 3 palavras):", token_ciencias_tecnologias_3)
print("Léxico de Ciências Socioeconómicas (Palavra única):", token_ciencias_socioeconomicas_1)
print("Léxico de Ciências Socioeconómicas (Grupos de 2 palavras):", token_ciencias_socioeconomicas_2)
print("Léxico de Ciências Socioeconómicas (Grupos de 3 palavras):", token_ciencias_socioeconomicas_3)
print("Léxico de artes visuais (Palavra única):", token_artes_visuais_1)
print("Léxico de artes visuais (Grupos de 2 palavras):", token_artes_visuais_2)
print("Léxico de Artes Visuais (Grupos de 3 palavras):", token_artes_visuais_3)
print("Léxico de linguas_humanidades (Palavra única):", token_linguas_humanidades_1)
print("Léxico de linguas_humanidades (Grupos de 2 palavras):", token_linguas_humanidades_2)
print("Léxico de linguas_humanidadess (Grupos de 3 palavras):", token_linguas_humanidades_3)
print("Léxico de informatica_eletronica_comunicacoes (Palavra única):", token_profissional_informatica_eletronica_comunicacoes_1)
print("Léxico de informatica_eletronica_comunicacoes (Grupos de 2 palavras):", token_profissional_informatica_eletronica_comunicacoes_2)
print("Léxico de informatica_eletronica_comunicacoes (Grupos de 3 palavras):", token_profissional_informatica_eletronica_comunicacoes_3)
```

```
print("Léxico de profissional_turismo_lazer (Palavra única):", token_profissional_turismo_lazer_1)

print("Léxico de profissional_turismo_lazer (Grupos de 2 palavras):", token_profissional_turismo_lazer_2)

print("Léxico de profissional_turismo_lazer (Grupos de 3 palavras):", token_profissional_turismo_lazer_3)

print("Léxico de profissional_cultura_patrimonio_conteudos (Palavra única):", token_profissional_cultura_patrimonio_conteudos_1)

print("Léxico de profissional_cultura_patrimonio_conteudos (Grupos de 2 palavras):", token_profissional_cultura_patrimonio_conteudos_2)

print("Léxico de profissional_cultura_patrimonio_conteudos (Grupos de 3 palavras):", token_profissional_cultura_patrimonio_conteudos_3)
```

Anexo K – Código do ficheiro de análise das perguntas de 1 a 8

```
import pandas as pd
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

from lexicos_perguntas1a8 import lexicos, disciplina_lexico
from analisesentimento9a14_vaderatual import metrica_personalizada

# Carregar recursos do NLTK
nltk.download('punkt')
nltk.download('stopwords')
stop_words = set(stopwords.words('portuguese'))

# Inicializar o SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()

lexicos = lexicos

perguntas = [
    "1. O que esperas alcançar na tua vida adulta? Descreve",
    "2. Qual é a profissão da tua mãe e planeias seguir o mesmo caminho? Porquê?",
    "3. Qual é a profissão do teu pai e planeias seguir o mesmo caminho? Porquê?",
    "4. Qual é a profissão dos teus sonhos? Descreve",
    "5. Que profissão é que os teus pais gostariam que tu tivesses? Concordas com eles? Porquê?",
    "6. Descreve as atividades que gostas mais de fazer nas férias",
    "7. Descreve as atividades que gostas mais de fazer nos teus tempos livres",
    "8. Qual é a tua disciplina preferida?",

]
# Inicializar o dicionário de resultados por área
area = {
    "Ciências e Tecnologias": [],
```

```
    "Ciências Socioeconómicas": [],
    "Línguas e Humanidades": [],
    "Artes Visuais": [],
    "Profissional informatica eletronica e comunicacoes": [],
    "Profissional turismo e lazer": [],
    "Profissional cultura, patrimonio e gestão de conteudos": [],
}

valores_por_area = {
    "Ciências e Tecnologias": 1,
    "Ciências Socioeconómicas": 2,
    "Línguas e Humanidades": 3,
    "Artes Visuais": 4,
    "Profissional informatica eletronica e comunicacoes": 5,
    "Profissional turismo e lazer": 6,
    "Profissional cultura, patrimonio e gestão de conteudos": 7,
    "Área não encontrada": 0
}

# Read the CSV file into a DataFrame with 'utf-32' encoding
df = pd.read_csv('dadosquestnormalizado.csv', sep=';', encoding='unicode_escape')

# Function to process the text and get connotation using a specific lexicon
def get_connotation(text, lexicons):
    if isinstance(text, int):
        text = str(text)  # Convert integer to string
    tokens = word_tokenize(text.lower())
    conotacao = 0
    for lexicon in lexicons:
        if lexicon in tokens:
            conotacao += 1
    return conotacao

# Função para calcular o score de sentimento usando léxicos personalizados
def get_custom_sentiment_score(text, lexicon):
```

```
    score = 0
    for word in word_tokenize(text.lower()):
        if word in lexicon:
            score += lexicon[word]
    return score

# Criar um novo DataFrame para armazenar os resultados por área
resultados_por_area = pd.DataFrame(columns=["Área"])

# Função para calcular a métrica personalizada
def calculate_custom_score(text):
    words = text.lower().split()  # Converter para minúsculas
    score = sum([metrica_personalizada.get(word, 0) for word in words])
    return score

# Função para calcular o score VADER
def calculate_vader_score(text):
    sentiment = analyzer.polarity_scores(text)
    vader_score = sentiment['compound']
    return vader_score

# Colunas do VADER Score e métrica personalizada
vader_metric_columns = [df.columns[7], df.columns[8], df.columns[10]]

# Loop para calcular os scores VADER e métrica personalizada para as colunas especificadas
for col in vader_metric_columns:
    vader_col_name = col + ' VADER Score'
    df[vader_col_name] = df[col].apply(calculate_vader_score)
    df[col + ' Custom Metric Score'] = df[col].apply(calculate_custom_score)

# Função para calcular a média dos scores da métrica personalizada e do VADER Score
def calculate_combined_score(row):
    metric_score = sum([row[col + ' Custom Metric Score'] for col in vader_metric_columns])
    vader_score = sum([row[col + ' VADER Score'] for col in vader_metric_columns])
```

```
    combined_score = (metric_score + vader_score) / len(vader_metric_columns)
    return combined_score

# Coluna para armazenar os scores combinados
df['Combined Sentiment Score'] = df.apply(calculate_combined_score, axis=1)

# Função para calcular a pontuação de área usando léxicos personalizados e valores por área
def calcular_pontuacao_area_resposta(resposta, lexico, valores_por_area):
    resposta_lower = resposta.lower()  # Convert to lowercase
    areas = lexico.get(resposta_lower, [])
    pontuacao = sum(valores_por_area.get(area, 0) for area in areas)
    return pontuacao
#funcao determinar area

def determinar_area_por_resposta(resposta, lexicos):
    if resposta in disciplina_lexico:  # Check for exact match
        return disciplina_lexico[resposta][0]  # Return the first associated area
    else:
        resposta_lower = resposta.lower()
        for area, lexicon_list in lexicos.items():
            if lexicon_list is not None:
                for lexicon in lexicon_list:
                    if lexicon is not None:
                        if any(word in resposta_lower for word in lexicon):
                            return area
        return "Área não encontrada"

# Loop sobre as perguntas de 1 a 7 e calcular as áreas correspondentes e as pontuações
for pergunta in perguntas[:-1]:
    coluna_respostas = df[pergunta]  # Acessar a coluna correspondente à pergunta
    df[f"{pergunta} Área"] = coluna_respostas.apply(lambda resposta: determinar_area_por_resposta(resposta, lexicos))
    df[f"{pergunta} Pontuação"] = coluna_respostas.apply(lambda resposta: valores_por_area.get(determinar_area_por_resposta(resposta, lexicos), 0))
```

```
# Processar respostas da pergunta 8 (disciplina preferida) e área correspondente
coluna_disciplina = df[perguntas[-1]]  # Access the column for the preferred discipline
df[f"{perguntas[-1]} Área"] = coluna_disciplina.apply(lambda disciplina: determinar_area_por_resposta(disciplina, lexicos))
df[f"{perguntas[-1]} Pontuação"] = df[f"{perguntas[-1]} Área"].apply(lambda disciplina_area: valores_por_area.get(disciplina_area, 0))

# Concatenar a pontuação com a área na pergunta 8
df["8. Qual é a tua disciplina preferida?"] = df["8. Qual é a tua disciplina preferida?"] + "; " + df["8. Qual é a tua disciplina preferida? Área"]

def calcular_score_area_resposta(resposta, lexico, disciplina_lexico):
    contagem_areas = {}  # Dicionário para contar a frequência de cada área

    for palavra in resposta.split():
        if palavra in lexico:
            areas = lexico[palavra]
            for area in areas:
                if area not in contagem_areas:
                    contagem_areas[area] = 1
                else:
                    contagem_areas[area] += 1

        if palavra in disciplina_lexico:
            areas_disciplina = disciplina_lexico[palavra]
            for area in areas_disciplina:
                if area not in contagem_areas:
                    contagem_areas[area] = 1
                else:
                    contagem_areas[area] += 1

    if contagem_areas:
        area_resultante = max(contagem_areas, key=contagem_areas.get)
    else:
        area_resultante = "Área não encontrada"
```

```
    return area_resultante

def calculate_most_frequent_area(row):
    area_counts = {}  # Dictionary to store the count of each area

    for col_num in range(8):
        area = row[f"{perguntas[col_num]} Área"]
        if area != "Área não encontrada":
            if area in area_counts:
                area_counts[area] += 1
            else:
                area_counts[area] = 1

    if area_counts:
        most_frequent_area = max(area_counts, key=area_counts.get)
        most_frequent_area_count = area_counts[most_frequent_area]
    else:
        most_frequent_area = "Área não encontrada"
        most_frequent_area_count = 0

    return most_frequent_area, most_frequent_area_count

# Apply the function to each row of the DataFrame to calculate the most frequent area and its count
df[["Área mais frequente", "Contagem da Área mais frequente"]] = df.apply(calculate_most_frequent_area, axis=1, result_type="expand")

# Add a new column that displays the number of times the most frequent area appears
df["Contagem da Área mais frequente"] = df["Contagem da Área mais frequente"].astype(str) + " vezes"

# Concatenate the area name and its count with the discipline area column
df["8. Qual é a tua disciplina preferida?"] = df["8. Qual é a tua disciplina preferida?"] + f"; {df['Área mais frequente']} ({df['Contagem da Área mais frequente']})"
```

```
# Print the first few rows of the DataFrame
print("First few rows of the DataFrame:")
print(df.head())
print()

# Função para obter o valor numérico da área mais frequente
def get_most_frequent_area_numeric_value(row):
    area = row["Área mais frequente"]
    return valores_por_area.get(area, 0)

# Aplicar a função para obter o valor numérico da área mais frequente
df["Área mais frequente Valor Numérico"] = df.apply(get_most_frequent_area_numeric_value, axis=1)

# Save the results to an Excel file
excel_output_file = 'respostas_com_areas_e_scores1a8_maissentimento_235.xlsx'
df.to_excel(excel_output_file, index=False)

print("Processamento concluído. Resultados guardados no ficheiro:", excel_output_file)
```

Anexo L- Código do ficheiro área, idade e escola

```
import pandas as pd


# Lê o ficheiro Excel com os dados originais

sentimento1a8_df = pd.read_excel('respostas_com_areas_e_scores1a8.xlsx')


# Limpeza e padronização da coluna "Escola que frequentas"

sentimento1a8_df['Escola que frequentas'] = sentimento1a8_df['Escola que frequentas'].str.strip().str.upper()


# Dicionário de mapeamento de valores numéricos para nomes de áreas

valores_por_area = {

    1: "Ciências e Tecnologias",

    2: "Ciências Socioeconómicas",

    3: "Línguas e Humanidades",

    4: "Artes Visuais",

    5: "Profissional informatica eletronica e comunicacoes",

    6: "Profissional turismo e lazer",

    7: "Profissional cultura, patrimonio e gestão de conteudos",

    0: "Área não encontrada"

}

# Cria uma coluna adicional com nomes de áreas com base nos valores numéricos

sentimento1a8_df['Área Vocacional'] = sentimento1a8_df['Área mais frequente Valor Numérico'].map(valores_por_area)

# Agrupa os dados por 'Escola que frequentas', 'Área Vocacional', 'Sexo' e 'Idade', contando a quantidade de alunos em cada grupo
```

```
    grouped = sentimento1a8_df.groupby(['Escola que frequentas', 'Área Vocacional', 'Sexo', 'Que idade tens?']).size().reset_index(name='Número de Alunos')



    # Verificar se o DataFrame 'grouped' possui dados

    if not grouped.empty:

        # Cria um objeto ExcelWriter para guardas os resultados num ficheiro Excel

        with pd.ExcelWriter('area_escola_idade.xlsx', engine='xlsxwriter') as writer:



            # Guarda o DataFrame contendo o número de alunos por área, escola, sexo e idade num ficheiro Excel

            grouped.to_excel(writer, sheet_name='area_escola_idade', index=False)



        print("Número de alunos por área, escola, sexo e idade guardado em 'area_escola_idade.xlsx'")

    else:

        print("O DataFrame 'grouped' está vazio. Verificar se os dados originais e o mapeamento estão corretos.")
```

Anexo M- Código do ficheiro de análise das perguntas 9 a 14

```
import pandas as pd
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Import valores_por_area from analise_sentimento1a8
from analise_sentimento1a8 import valores_por_area

# Carregar recursos do NLTK
nltk.download('punkt')
nltk.download('stopwords')
stop_words = set(stopwords.words('portuguese'))

# Criar uma instância do SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()

def preprocess_text(text):
    # Converte para minúsculas
    text = text.lower()
    # Tokenização
    tokens = word_tokenize(text)
    # Remoção de stop words
    tokens = [word for word in tokens if word not in stopwords.words('portuguese')]
    return ' '.join(tokens)

# Definir métricas personalizadas
metrica_personalizada = {
    "não gosto nada": -2,
    "não gosto muito": -2,
    "desnecessário": -2,
    "não pretendo seguir": -1,
    "não fazem o meu estilo": -1,
    "não gosto": -1,
```

"não quero": -1,
"chatas": -1,
"nada": -1,
"desinteressante": -2,
"não": -1,
"Difícil": -1,
"entediante": -1,
"complicada": -1,
"não gosto tanto": -1,
"não acho nada": -1,
"não me identifico": -1,
"seca": -1,
"secante": -1,
"não é para mim": -2,
"acho normal": 1,
"importante": 1,
"bacano": 1,
"não me importo": 1,
"mais ou menos": 1,
"não sei": 0,
"Ainda não sei": 0,
"interessante": 1,
"segunda opção": 1,
"divertida": 1,
"gosto": 2,
"são boas": 2,
"sim": 2,
"gostava": 2,
"boa": 2,
"gostaria": 2,
"importantes": 1,
"muito bom": 3,
"acho bom": 2,
"acho que sim": 1,
"talvez": 1,

```
        "gosto bastante": 3,
        "é fixe": 2,
        "boa opção": 2,
        "gosto muito": 3,
        "muito interessante": 3,
        "favoritas": 3,
        "muito fixe": 3,
        "espetacular": 3,
        "favorita": 3,
        "5 maravilhas": 3,
        "amo": 3,
        "amo muito": 3,
        "muito interessantes": 3,
        "Adoro": 3,
        "amava": 3,
        "adoro": 3,
        "É o que vou seguir": 3
    }
    # Definir perguntas e áreas correspondentes
    perguntas = [
        "9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve",
        "10.  O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve",
        "11.  O que achas da área das ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve",
        "12.  O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve",
        "13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê?",
        "14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve",

    ]
```

```
# Mapear perguntas para áreas correspondentes
areas_correspondentes = {
    "9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve": ["Línguas e Humanidades", "Profissional turismo e lazer"],
    "10.  O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve": ["Ciências Socioeconómicas"],
    "11.  O que achas da área das ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve": ["Ciências e Tecnologias", "Profissional informatica eletronica e comunicacoes"],
    "12.  O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve": ["Artes Visuais", "Profissional cultura, patrimonio e gestão de conteudos"],
    "13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê?": ["Profissional informatica eletronica e comunicacoes", "Profissional cultura, patrimonio e gestão de conteudos", "Profissional turismo e lazer"],
    "14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve": ["Profissional turismo e lazer"],

}

# Ler o ficheiro CSV
df = pd.read_csv('dadosquestnormalizado.csv', sep=';', encoding='unicode_escape')

# Colunas com texto
text_columns = df.columns[14:20].tolist()

def calculate_vader_score(text):
    sentiment = analyzer.polarity_scores(text)
    vader_score = sentiment['compound']
    return vader_score

# Definir a função para ajustar o VADER Score mantendo a direção do sentimento
def adjust_vader_score(score):
    if score < 0:
        adjusted_score = -1
    else:
        adjusted_score = 1
    return adjusted_score
```

```
# Função para calcular o score personalizado ajustado para a nova escala de -2 a 3
def calculate_custom_score(text):
    words = text.lower().split()  # Converter para minúsculas e dividir em palavras
    score = sum([metrica_personalizada.get(word, 0) for word in words])  # Somar os scores personalizados
    adjusted_score = min(max(score, -2), 3)  # Ajustar para o intervalo de -2 a 3
    return adjusted_score


# Loop para calcular os scores VADER e personalizado nas colunas desejadas
for col in text_columns:
    df[col + ' Vader Score'] = df[col].apply(calculate_vader_score)
    df[col + ' Metrica Score'] = df[col].apply(calculate_custom_score)

# Aplicar a função para calcular os scores personalizados nas colunas desejadas
text_columns = df.columns[14:20].tolist()
for col in text_columns:
    df[col + ' Metrica Score'] = df[col].apply(calculate_custom_score)


# Função para converter respostas em scores
def converter_resposta_em_score(resposta):
    return metrica_personalizada.get(resposta, 0)

# Função para processar o texto e obter conotação usando um léxico específico
def get_connotation(text, lexicons):
    if isinstance(text, int):
        text = str(text)  # Convert integer to string
    tokens = word_tokenize(text.lower())
    conotacao = 0
    for lexicon in lexicons:
        if lexicon in tokens:
            conotacao += 1
    return conotacao
```

```
# Função para calcular o score de sentimento usando léxicos personalizados
def get_custom_sentiment_score(text, lexicon):
    score = 0
    for word in word_tokenize(text.lower()):
        if word in lexicon:
            score += lexicon[word]
    return score



# Loop para calcular os scores VADER e personalizado nas colunas desejadas
for col in text_columns:
    df[col + ' Vader Score'] = df[col].apply(calculate_vader_score)
    df[col + ' Metrica Score'] = df[col].apply(calculate_custom_score)


    # Ajustar os valores do VADER Score para a nova escala
    df[col + ' Vader Score'] = df[col + ' Vader Score'].apply(adjust_vader_score)


# Calcular a média dos sentimentos do VADER e da métrica personalizada para cada resposta
df['Média VaderMetrica 9'] = (df['9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve Vader Score'] + df['9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve Metrica Score']) / 2

df['Média VaderMetrica 10'] = (df['10.  O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve Vader Score'] + df['10.  O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve Metrica Score']) / 2

df['Média VaderMetrica 11'] = (df['11.  O que achas da área das ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve Vader Score'] + df['11.  O que achas da área das ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve Metrica Score']) / 2

df['Média VaderMetrica 12'] = (df['12.  O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve Vader Score'] + df['12.  O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve Metrica Score']) / 2

df['Média VaderMetrica 13'] = (df['13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê? Vader Score'] + df['13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê? Metrica Score']) / 2
```

```
df['Média VaderMetrica 14'] = (df['14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve Vader Score'] + df['14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve Metrica Score']) / 2


def determinar_area_vocacional(row):
    areas = []
    valores = []


    # Lista de nomes de colunas para análise
    columns_to_analyze = [
        '9. O que achas da área das línguas, humanidades, história, educação, escrita, leitura, turismo e sociedade? Descreve',
        '10.  O que achas da área das ciências socioeconómicas, comércio, psicologia, política, educação, cultura e economia? Descreve',
        '11.  O que achas da área das ciências e tecnologias, informática, saúde, biologia, engenharia e arquitetura? Descreve',
        '12.  O que achas da área das artes visuais, desenho, pintura, multimédia, geometria e fotografia? Descreve',
        '13. Qual é a tua opinião acerca de cursos profissionais? Gostavas de fazer algum? Qual e porquê?',
        '14. Gostas de desporto? Praticas algum ou gostavas de praticar algum? Descreve',


    ]


    for col in df.columns[df.columns.isin(columns_to_analyze)]:
        if row[col] > 0:
            area_textual = areas_correspondentes.get(col)  # Use get para evitar KeyError
            if area_textual:  # Verifique se a área está definida
                areas.extend(area_textual)
                valores.extend([valores_por_area[area] for area in area_textual])


    if areas:  # Verificar se há áreas encontradas antes de retornar
        return '', ''.join(areas), valores
    else:
        return "", []  # Retorna valores vazios se nenhuma área foi encontrada
```

```
    # Aplicar as funções nas colunas de índice 14 a 20
    colunas_selecionadas = df.columns[14:20].tolist()

    # Aplicar a função de conversão nas colunas selecionadas
    df[colunas_selecionadas] = df[colunas_selecionadas].applymap(converter_resposta_em_score)

    # Aplicar a função para determinar a área vocacional nas colunas selecionadas
    df[["Área Vocacional", "Valores Área"]] = df[colunas_selecionadas].apply(determinar_area_vocacional, axis=1, result_type='expand')

    # Loop para calcular as áreas vocacionais para cada aluno
    for index, row in df.iterrows():
        area_vocacional, valores_area = determinar_area_vocacional(row)
        df.at[index, "Área Vocacional"] = area_vocacional
        df.at[index, "Valores Área"] = valores_area

    # Calcular a área mais comum para cada aluno
    most_common_areas = []
    most_common_numeric_values = []
    area_counts = []

    for values in df["Valores Área"]:
        if values:
            most_common_numeric_value = max(values, key=values.count)
            most_common_area = [area for area, numeric_value in valores_por_area.items() if numeric_value == most_common_numeric_value][0]

            most_common_areas.append(most_common_area)
            most_common_numeric_values.append(most_common_numeric_value)
            area_counts.append(values.count(most_common_numeric_value))
        else:
            most_common_areas.append("")
            most_common_numeric_values.append(0)
            area_counts.append(0)
```

```
    # Adicionar colunas com a área mais comum, o seu valor numérico e a contagem da área para cada aluno
    df["Área Mais Comum"] = most_common_areas
    df["Valor Numérico"] = most_common_numeric_values
    df["Contagem da Área Mais Comum"] = area_counts

    # Guardar os resultados no mesmo ficheiro Excel com as novas colunas
    excel_output_file = 'analiseperguntas9a14_VADER.xlsx'
    df.to_excel(excel_output_file, index=False)
```

Anexo N - Código do classificador Naïve Bayes

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_curve, auc, confusion_matrix
import matplotlib.pyplot as plt
from analise_sentimento1a8 import valores_por_area
import numpy as np
import seaborn as sns


# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')


# Select the columns related to scores from df_vader_metric
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]


# Select the target column from df_area_scores
y = df_area_scores["Área mais frequente Valor Numérico"]


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Initialize the Gaussian Naive Bayes classifier
nb_classifier = GaussianNB()


# Train the classifier
nb_classifier.fit(X_train, y_train)


# Predict the labels for the test data
y_pred = nb_classifier.predict(X_test)
```

```
# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")

# Calculate the precision of the classifier
precision = precision_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"Precision: {precision:.2f}")

# Calculate the recall of the classifier
recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"Recall: {recall:.2f}")

# Calculate the F1-Score of the classifier
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"F1-Score: {f1:.2f}")

# Calculate the probabilities for each class
y_probs = nb_classifier.predict_proba(X_test)

# Plot ROC curves for each class
plt.figure()
for i in range(y_probs.shape[1]):
    fpr, tpr, _ = roc_curve(y_test == i, y_probs[:, i])
    roc_auc = auc(fpr, tpr)

    class_name                                                    =
list(valores_por_area.keys())[list(valores_por_area.values()).index(i)]
    plt.plot(fpr, tpr, lw=2, label=f'{class_name} (AUC = {roc_auc:.2f}')

plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
```

```
plt.title('Receiver Operating Characteristic (ROC) - Naive Bayes')
plt.legend(loc='lower right')
plt.show()

# Add the predicted labels to the DataFrame
df_vader_metric.loc[X_test.index, "Predicted Área"] = y_pred

# Calculate the confusion matrix for the Naive Bayes classifier
conf_matrix_nb = confusion_matrix(y_test, y_pred)

# Create a heatmap of the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix_nb, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Naive Bayes')
plt.show()

# Print the confusion matrix
print("Confusion Matrix - Naive Bayes:")
print(conf_matrix_nb)

# Define o mapeamento de nomes das classes usando o dicionário 'valores_por_area'
classes_nomes = [nome_da_classe for valor_da_classe, nome_da_classe in valores_por_area.items()]

# Calculate the confusion matrix for the Naive Bayes classifier
conf_matrix_nb = confusion_matrix(y_test, y_pred, labels=classes_nomes)

# Calculate TP, FN, FP using numpy functions
tp = np.diag(conf_matrix_nb)  # Verdadeiros positivos na diagonal principal
fn = np.sum(conf_matrix_nb, axis=1) - tp  # Falsos negativos somando a linha e subtraindo TP
fp = np.sum(conf_matrix_nb, axis=0) - tp  # Falsos positivos somando a coluna e subtraindo TP
```

```
for i in range(len(tp)):
    class_name = list(valores_por_area.keys())[i]
    print(f"Class '{class_name}': TP = {tp[i]}, FN = {fn[i]}, FP = {fp[i]}")

# Save the DataFrame with the predicted labels in an Excel file
excel_output_file_with_predictions = 'previsoes_NB.xlsx'
df_vader_metric.to_excel(excel_output_file_with_predictions, index=False)

# Print a message indicating the completion and the file name
print("Processamento concluído. Resultados e previsões guardados no ficheiro:", excel_output_file_with_predictions)
```

Anexo O- Código do classificador Decision Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_curve, auc, confusion_matrix
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
from sklearn.preprocessing import label_binarize
import seaborn as sns
import numpy as np
from analise_sentimento1a8 import valores_por_area


# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')


# Select the columns related to scores from df_vader_metric
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]


# Select the target column from df_area_scores
y = df_area_scores["Área mais frequente Valor Numérico"]


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


class_names = {
    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
    6: "Profissional turismo e lazer",
```

```
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# Initialize the Decision Tree classifier
tree_classifier = DecisionTreeClassifier()

# Definir o número de folds para a validação cruzada
num_folds = 5

# Realizar a validação cruzada e obter os resultados para diferentes folds
scores = cross_val_score(tree_classifier, X, y, cv=num_folds, scoring='accuracy')

# Exibir os resultados de cada fold
for fold, score in enumerate(scores, start=1):
    print(f"Fold {fold}: Accuracy = {score:.2f}")

# Calcular a média e o desvio padrão dos resultados
mean_accuracy = scores.mean()
std_accuracy = scores.std()

print(f"Média da Exatidão: {mean_accuracy:.2f}")
print(f"Desvio Padrão da Exatidão: {std_accuracy:.2f}")

# Train the classifier
tree_classifier.fit(X_train, y_train)

# Predict the labels for the test data
y_pred = tree_classifier.predict(X_test)

class_names = df_area_scores["Área mais frequente"].unique()

# Binarize the true labels
y_test_bin = label_binarize(y_test, classes=[1, 2, 3, 4, 5, 6, 7])
```

```
# Compute ROC curve and ROC area for each class
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(len(class_names)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_pred == i+1)
    roc_auc[i] = auc(fpr[i], tpr[i])

# Plot the decision tree
plt.figure(figsize=(20, 10))
plot_tree(tree_classifier, feature_names=X.columns, filled=True, rounded=True)
plt.show()

# Plot ROC curves for each class
plt.figure(figsize=(10, 8))
for i in range(len(class_names)):
    plt.plot(fpr[i], tpr[i], label=f'{class_names[i]} (AUC = {roc_auc[i]:.2f})')

plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic')
plt.legend(loc="lower right")

plt.show()

# Print a message indicating the completion and the file name
print("Processamento concluído. Árvore de Decisão guardada como 'decision_tree.pdf'")

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
```

```
# Calculate the precision of the classifier
precision = precision_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"Precision: {precision:.2f}")

# Calculate the recall of the classifier
recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"Recall: {recall:.2f}")

# Calculate the F1-Score of the classifier
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"F1-Score: {f1:.2f}")

# Add the predicted labels to the DataFrame
df_vader_metric.loc[X_test.index, "Predicted Área"] = y_pred

# Create the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Create a heatmap of the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Decision Tree')
plt.show()

# Print the confusion matrix
print("Confusion Matrix - Decision Tree:")
print(conf_matrix)

# Define o mapeamento de nomes das classes com o dicionário 'valores_por_area'
classes_nomes = [nome_da_classe for valor_da_classe, nome_da_classe in valores_por_area.items()]
```

```
# Calculate the confusion matrix for DT
conf_matrix_nb = confusion_matrix(y_test, y_pred, labels=classes_nomes)


# Calculate TP, FN, FP using numpy functions
tp = np.diag(conf_matrix_nb)  # Verdadeiros positivos na diagonal principal
fn = np.sum(conf_matrix_nb, axis=1) - tp  # Falsos negativos somando a linha e subtraindo TP
fp = np.sum(conf_matrix_nb, axis=0) - tp  # Falsos positivos somando a coluna e subtraindo TP


for i in range(len(tp)):
    class_name = list(valores_por_area.keys())[i]
    print(f"Class '{class_name}': TP = {tp[i]}, FN = {fn[i]}, FP = {fp[i]}")


# Compute ROC curve and ROC area for each class
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(len(class_names)):
    if np.sum(y_test_bin[:, i]) > 0 and np.sum(y_test_bin[:, i]) < len(y_test_bin[:, i]):
        fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_pred == i+1)
        roc_auc[i] = auc(fpr[i], tpr[i])
        print(f"AUC for class '{class_names[i]}': {roc_auc[i]:.2f}")
    else:
        print(f"No positive or only positive examples for class '{class_names[i]}'")


# Save the DataFrame with the predicted labels in an Excel file
excel_output_file_with_predictions = 'previsoes_DT.xlsx'
df_vader_metric.to_excel(excel_output_file_with_predictions, index=False)


# Print a message indicating the completion and the file name
print("Processamento concluído. Resultados e previsões guardados no ficheiro:", excel_output_file_with_predictions)
```

Anexo P – Código do classificador Random Forest

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_curve, auc, confusion_matrix
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
import seaborn as sns
import numpy as np
from analise_sentimento1a8 import valores_por_area


# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')


# Select the columns related to scores from df_vader_metric
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]


# Select the target column from df_area_scores
y = df_area_scores["Área mais frequente Valor Numérico"]


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Dictionary with class names
class_names = {

    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
```

```
    6: "Profissional turismo e lazer",
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# Initialize the Random Forest classifier
rf_classifier = RandomForestClassifier()

# Train the classifier
rf_classifier.fit(X_train, y_train)
# Choose one of the trees from the Random Forest
tree_index = 0  # Change this index to choose a different tree

# Predict the labels for the test data
y_pred = rf_classifier.predict(X_test)
class_names = df_area_scores["Área mais frequente"].unique()

# Get the predicted probability scores for each class
y_pred_proba = rf_classifier.predict_proba(X_test)

# Define feature names for the tree plot
feature_names = ["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]

# Plot the selected tree
plt.figure(figsize=(20, 10))
plot_tree(rf_classifier.estimators_[tree_index], feature_names=feature_names, filled=True, rounded=True,
          class_names=[class_names[i] for i in rf_classifier.classes_])
plt.show()

# Plot ROC curves for each class
plt.figure(figsize=(10, 8))
for class_index in range(len(class_names)):
    fpr, tpr, _ = roc_curve(y_test == class_index, y_pred_proba[:, class_index])
    roc_auc = auc(fpr, tpr)
```

```
    plt.plot(fpr, tpr, label=f'{class_names[class_index]} (AUC = {roc_auc:.2f})')

plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic')
plt.legend(loc="lower right")
plt.show()

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")

# Calculate the precision of the classifier
precision = precision_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"Precision: {precision:.2f}")

# Calculate the recall of the classifier
recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"Recall: {recall:.2f}")

# Calculate the F1-Score of the classifier
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"F1-Score: {f1:.2f}")

# Align the index of y_pred with the index of X_test
y_pred_series = pd.Series(y_pred, index=X_test.index)

# Add the predicted labels to the DataFrame
df_vader_metric.loc[X_test.index, "Predicted Área"] = y_pred

# Create the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
    # Create a heatmap of the confusion matrix
    plt.figure(figsize=(10, 8))
    sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.title('Confusion Matrix - RandomForest')
    plt.show()

    # Print the confusion matrix
    print("Confusion Matrix - Random Forest:")
    print(conf_matrix)

    # Define o mapeamento dos nomes das classes com o dicionário 'valores_por_area'
    classes_nomes = [nome_da_classe for valor_da_classe, nome_da_classe in valores_por_area.items()]

    # Calculate the confusion matrix for the Random Forest
    conf_matrix_nb = confusion_matrix(y_test, y_pred, labels=classes_nomes)

    # Calculate TP, FN, FP using numpy functions
    tp = np.diag(conf_matrix_nb)  # Verdadeiros positivos na diagonal principal
    fn = np.sum(conf_matrix_nb, axis=1) - tp  # Falsos negativos somando a linha e subtraindo TP
    fp = np.sum(conf_matrix_nb, axis=0) - tp  # Falsos positivos somando a coluna e subtraindo TP

    for i in range(len(tp)):
        class_name = list(valores_por_area.keys())[i]
        print(f"Class '{class_name}': TP = {tp[i]}, FN = {fn[i]}, FP = {fp[i]}")

    # Calculate and print the AUC for each class
    for class_index in range(len(class_names)):
        fpr, tpr, _ = roc_curve(y_test == class_index, y_pred_proba[:, class_index])
        roc_auc = auc(fpr, tpr)
        print(f"AUC for class '{class_names[class_index]}': {roc_auc:.2f}")
```

```
# Save the DataFrame with the predicted labels in an Excel file
excel_output_file_with_predictions = 'previsoes_RForest.xlsx'
df_vader_metric.to_excel(excel_output_file_with_predictions, index=False)


# Print a message indicating the completion and the file name
print("Processamento concluído. Resultados e previsões guardados no ficheiro:", excel_output_file_with_predictions)
```

Anexo Q- Código do classificador K-NN

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_curve, auc, confusion_matrix
import matplotlib.pyplot as plt
import numpy as np
from analise_sentimento1a8 import valores_por_area
import seaborn as sns


# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')


# Select the columns related to scores from df_vader_metric
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]


# Select the target column from df_area_scores
y = df_area_scores["Área mais frequente Valor Numérico"]


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Dictionary with class names
class_names = {
    0: "Área não encontrada",
    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
    6: "Profissional turismo e lazer",
```

```
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# List of k values to try
k_values = [1, 3, 5, 7]

for k in k_values:
    # Initialize the KNN classifier
    knn_classifier = KNeighborsClassifier(n_neighbors=k)

    # Train the classifier
    knn_classifier.fit(X_train, y_train)

    # Predict the labels for the test data
    y_pred = knn_classifier.predict(X_test)

    # Calculate the accuracy of the classifier
    accuracy = accuracy_score(y_test, y_pred)
    print(f"Accuracy (K={k}): {accuracy:.2f}")

    # Calculate the precision of the classifier
    precision = precision_score(y_test, y_pred, average='weighted', zero_division=1)
    print(f"Precision (K={k}): {precision:.2f}")

    # Calculate the recall of the classifier
    recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)
    print(f"Recall (K={k}): {recall:.2f}")

    # Calculate the F1-Score of the classifier
    f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
    print(f"F1-Score (K={k}): {f1:.2f}")

    # Calculate ROC curve and AUC for each class
    y_pred_proba = knn_classifier.predict_proba(X_test)
    unique_classes = np.unique(y_test)
```

```
        class_indices = {label: idx for idx, label in enumerate(unique_classes)}

        all_fpr = []
        all_tpr = []
        all_roc_auc = []

        for label in unique_classes:
            class_index = class_indices[label]
            y_test_binary = np.where(y_test == label, 1, 0)
            y_pred_positive_class = y_pred_proba[:, class_index]

            fpr, tpr, _ = roc_curve(y_test_binary, y_pred_positive_class)
            roc_auc = auc(fpr, tpr)

            all_fpr.append(fpr)
            all_tpr.append(tpr)
            all_roc_auc.append(roc_auc)

    # Plot ROC curves for all classes with class names
        plt.figure()
        for idx, label in enumerate(unique_classes):
            class_name = class_names[label]
            plt.plot(all_fpr[idx], all_tpr[idx], lw=2, label='ROC curve for class %s (area = %0.2f)' % (class_name, all_roc_auc[idx]))

        plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
        plt.xlim([0.0, 1.0])
        plt.ylim([0.0, 1.05])
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.title(f'Receiver Operating Characteristic for K={k}')
        plt.legend(loc="lower right")

        # Rotate x-axis labels for better visibility
        plt.xticks(rotation=45)
```

```
        plt.show()

        # Create the confusion matrix
        conf_matrix = confusion_matrix(y_test, y_pred)

        # Create a heatmap of the confusion matrix
        plt.figure(figsize=(10, 8))
        sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
        plt.xlabel('Predicted')
        plt.ylabel('True')
        plt.title(f'Confusion Matrix - KNN (K={k})')
        plt.show()

        # Print the confusion matrix
        print(f"Confusion Matrix - KNN (K={k}):")
        print(conf_matrix)

        # Define the list of classes using 'class_names' dictionary
        classes_list = list(class_names.values())

        # Calculate TP, FN, FP using numpy functions
        tp = np.diag(conf_matrix)  # True positives on the diagonal
        fn = np.sum(conf_matrix, axis=1) - tp  # False negatives by summing the rows and subtracting TP
        fp = np.sum(conf_matrix, axis=0) - tp  # False positives by summing the columns and subtracting TP

        # Print TP, FN, FP for each class
        for i in range(len(tp)):
            class_name = classes_list[i]
            print(f"Class '{class_name}' - K={k}: TP = {tp[i]}, FN = {fn[i]}, FP = {fp[i]}")

    # Save the predicted labels in a DataFrame
    y_pred_df = pd.DataFrame(y_pred, columns=[f"Predicted Área (K={k})"])
```

```
    # Concatenate the original DataFrame, y_test, and y_pred_df horizontally
    result_df = pd.concat([df_area_scores, y_test, y_pred_df], axis=1)

    # Define o mapeamento de nomes das classes usando o dicionário 'valores_por_area'
    classes_nomes = [nome_da_classe for valor_da_classe, nome_da_classe in valores_por_area.items()]

    # Calculate the confusion matrix for the Random Forest
    conf_matrix_nb = confusion_matrix(y_test, y_pred, labels=classes_nomes)

    # Save the DataFrame with the predicted labels in an Excel file
    excel_output_file_with_predictions = f'previsoes_knn_{k}.xlsx'
    result_df.to_excel(excel_output_file_with_predictions, index=False)

       # Print a message indicating the completion and the file name
    print("Processamento concluído. Resultados e previsões guardados no ficheiro:", excel_output_file_with_predictions)
```

Anexo R – Código do classificador AdaBoost

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_curve, auc, confusion_matrix
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from analise_sentimento1a8 import valores_por_area

# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')

# Select the columns related to scores from df_vader_metric
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]

# Select the target column from df_area_scores
y = df_area_scores["Área mais frequente Valor Numérico"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the AdaBoost classifier
ada_classifier = AdaBoostClassifier()

# Train the classifier
ada_classifier.fit(X_train, y_train)

# Predict the labels for the test data
y_pred = ada_classifier.predict(X_test)
```

```
# Calculate the predicted probabilities
y_pred_proba = ada_classifier.predict_proba(X_test)

# Get the unique classes from y_test
unique_classes = np.unique(y_test)

# Dictionary with class names
class_names = {
    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
    6: "Profissional turismo e lazer",
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# Create a dictionary to store the class indices and labels
class_indices = {}
for idx, label in enumerate(unique_classes):
    class_indices[label] = idx

# Initialize variables to store fpr, tpr, and roc_auc for each class
all_fpr = []
all_tpr = []
all_roc_auc = []

# Calculate ROC curve and AUC for each class
for label in unique_classes:
    class_index = label - 1  # Adjust class index
    y_test_binary = np.where(y_test == label, 1, 0)
    y_pred_positive_class = y_pred_proba[:, class_index]

    fpr, tpr, _ = roc_curve(y_test_binary, y_pred_positive_class)
    roc_auc = auc(fpr, tpr)
```

```
        all_fpr.append(fpr)
        all_tpr.append(tpr)
        all_roc_auc.append(roc_auc)

    # Plot ROC curves for all classes with class names
    plt.figure()
    for idx, label in enumerate(unique_classes):
        class_name = class_names[label + 1]  # Adjust the index by adding 1
        plt.plot(all_fpr[idx], all_tpr[idx], lw=2, label='ROC curve for class %s (area = %0.2f)'
% (class_name, all_roc_auc[idx]))

    plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver Operating Characteristic for Multiple Classes')
    plt.legend(loc="lower right")

    # Rotate x-axis labels for better visibility
    plt.xticks(rotation=45)

    plt.show()

    # Calculate the accuracy of the classifier
    accuracy = accuracy_score(y_test, y_pred)
    print(f"Accuracy: {accuracy:.2f}")

    # Calculate the precision of the classifier
    precision = precision_score(y_test, y_pred, average='weighted', zero_division=1)
    print(f"Precision: {precision:.2f}")

    # Calculate the recall of the classifier
    recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)
```

```
print(f"Recall: {recall:.2f}")

# Calculate the F1-Score of the classifier
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
print(f"F1-Score: {f1:.2f}")

# Align the index of y_pred with the index of X_test
y_pred_series = pd.Series(y_pred, index=X_test.index)

# Add the predicted labels to the DataFrame
df_vader_metric.loc[X_test.index, "Predicted Área"] = y_pred

# Create the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Create a heatmap of the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Ada Boost')
plt.show()

# Print the confusion matrix
print("Confusion Matrix - Ada Boost:")
print(conf_matrix)

# Define o mapeamento de nomes das classes usando o dicionário 'valores_por_area'
classes_nomes = [nome_da_classe for valor_da_classe, nome_da_classe in valores_por_area.items()]

# Calculate the confusion matrix for the Random Forest
conf_matrix_nb = confusion_matrix(y_test, y_pred, labels=classes_nomes)
```

```
# Calculate TP, FN, FP using numpy functions
tp = np.diag(conf_matrix_nb)  # Verdadeiros positivos na diagonal principal
fn = np.sum(conf_matrix_nb, axis=1) - tp  # Falsos negativos somando a linha e subtraindo TP
fp = np.sum(conf_matrix_nb, axis=0) - tp  # Falsos positivos somando a coluna e subtraindo TP


for i in range(len(tp)):
    class_name = list(valores_por_area.keys())[i]
    print(f"Class '{class_name}': TP = {tp[i]}, FN = {fn[i]}, FP = {fp[i]}")


# Calculate and print the AUC for each class
for label, roc_auc in zip(unique_classes, all_roc_auc):
    if label == 0:
        class_name = "Área não encontrada"
    else:
        class_name = class_names[label]
    print(f"AUC for class '{class_name}': {roc_auc:.2f}")


# Save the DataFrame with the predicted labels in an Excel file
excel_output_file_with_predictions = 'previsoes_AdaBoost.xlsx'
df_vader_metric.to_excel(excel_output_file_with_predictions, index=False)


# Print a message indicating the completion and the file name
print("Processamento concluído. Resultados e previsões guardados no ficheiro:", excel_output_file_with_predictions)
```

Anexo S – Código do classificador MLP

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_curve, auc, confusion_matrix
import matplotlib.pyplot as plt
from analise_sentimento1a8 import valores_por_area
import numpy as np
import seaborn as sns

# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')

# Select the columns related to scores from df_vader_metric
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]

# Select the target column from df_area_scores
y = df_area_scores["Área mais frequente Valor Numérico"]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the MLP classifier
mlp_classifier = MLPClassifier(hidden_layer_sizes=(6, 6), max_iter=1000, random_state=42)

# Train the classifier
mlp_classifier.fit(X_train, y_train)

# Predict the probabilities for each class
y_probs = mlp_classifier.predict_proba(X_test)
```

```
# Dictionary with class names
class_names = {

    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
    6: "Profissional turismo e lazer",
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# Plot ROC curves for each class
plt.figure()
for i in range(y_probs.shape[1]):
    fpr, tpr, _ = roc_curve(y_test == i, y_probs[:, i])
    roc_auc = auc(fpr, tpr)

    class_name = list(valores_por_area.keys())[list(valores_por_area.values()).index(i)]
    plt.plot(fpr, tpr, lw=2, label=f'{class_name} (AUC = {roc_auc:.2f}')

plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - Multiclass')
plt.legend(loc='lower right')
plt.show()

# Predict the labels for the test data
y_pred = mlp_classifier.predict(X_test)
```

```
# Calculate and print evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted', zero_division=1)
recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1-Score: {f1:.2f}")

# Save the predicted labels in a DataFrame
y_pred_df = pd.DataFrame(y_pred, columns=["Predicted Área"])

# Concatenate the original DataFrame, y_test, and y_pred_df horizontally
result_df = pd.concat([df_area_scores, y_test.reset_index(drop=True), y_pred_df], axis=1)

# Substituir NaN por um valor específico (por exemplo, "N/A")
result_df.fillna("N/A", inplace=True)

# Create the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Create a heatmap of the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Multilayer Perceptron')
plt.show()

# Print the confusion matrix
print("Confusion Matrix - Multilayer Perceptron:")
print(conf_matrix)
```

```
# Definir o mapeamento de nomes das classes usando o dicionário 'valores_por_area'
classes_nomes = [nome_da_classe for valor_da_classe, nome_da_classe in valores_por_area.items()]

# Calculate the confusion matrix for the Random Forest
conf_matrix_nb = confusion_matrix(y_test, y_pred, labels=classes_nomes)

# Calculate TP, FN, FP using numpy functions
tp = np.diag(conf_matrix_nb)  # Verdadeiros positivos na diagonal principal
fn = np.sum(conf_matrix_nb, axis=1) - tp  # Falsos negativos somando a linha e subtraindo TP
fp = np.sum(conf_matrix_nb, axis=0) - tp  # Falsos positivos somando a coluna e subtraindo TP

for i in range(len(tp)):
    class_name = list(valores_por_area.keys())[i]
    print(f"Class '{class_name}': TP = {tp[i]}, FN = {fn[i]}, FP = {fp[i]}")

# Save the DataFrame with the predicted labels in an Excel file
excel_output_file_with_predictions = 'previsoes_MLP.xlsx'
result_df.to_excel(excel_output_file_with_predictions, index=False)

# Obter o número de epochs
num_epochs = mlp_classifier.n_iter_

# Imprimir o número de epochs
print(f"Número de epochs: {num_epochs}")

# Print a completion message with the file name
print("Processamento concluído. Resultados e previsões guardados no ficheiro:", excel_output_file_with_predictions)
```

Anexo T- Cálculo das métricas para cada Classificador

```
import os

import numpy as np

import matplotlib.pyplot as plt


classifiers = ["KNN (K=1)", "KNN (K=3)", "KNN (K=5)", "KNN (K=7)", "Naive Bayes", "Random Forest", "Decision Tree", "Ada Boost", "Multilayer Perceptron"]

classes = ["Área não encontrada", "Ciências e Tecnologias", "Ciências Socioeconómicas", "Línguas e Humanidades", "Artes Visuais"]


# Dados fornecidos (valores de TP, FN, FP)

tp_values = [

    [9, 14, 1, 0, 0],

    [11, 13, 0, 0, 0],

    [12, 14, 0, 0, 0],

    [9, 12, 0, 0, 0],

    [14, 9, 0, 0, 0],

    [12, 16, 1, 0, 0],

    [10, 11, 2, 0, 0],

    [3, 18, 0, 0, 0],

    [7, 17, 0, 0, 0]

]


fn_values = [

    [13, 8, 3, 4, 2],

    [11, 9, 4, 4, 2],
```

```
    [10, 9, 2, 4, 2],
    [13, 11, 3, 4, 2],
    [8, 13, 4, 4, 2],
    [10, 6, 2, 2, 1],
    [10, 9, 1, 4, 2],
    [15, 4, 4, 4, 2],
    [15, 5, 3, 4, 2]
]

fp_values = [
    [10, 16, 2, 0, 2],
    [14, 0, 0, 0, 0],
    [13, 15, 0, 0, 0],
    [16, 17, 0, 0, 0],
    [18, 13, 0, 0, 0],
    [8, 15, 0, 0, 0],
    [15, 15, 0, 0, 0],
    [22, 0, 0, 0, 0],
    [23, 0, 0, 0, 0]
]
# Pasta para guardar os plots
output_folder = "plots"
if not os.path.exists(output_folder):
    os.makedirs(output_folder)
```

```
# Preparação dos dados para os gráficos
x = np.arange(len(classes))
width = 0.35  # Largura das barras

for i, classifier in enumerate(classifiers):
    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width, tp_values[i], width, label='True Positive', color='g')
    rects2 = ax.bar(x, fn_values[i], width, label='False Negative', color='r')
    rects3 = ax.bar(x + width, fp_values[i], width, label='False Positive', color='b')

    ax.set_xlabel('Classes')
    ax.set_ylabel('Counts')
    ax.set_title(f'Metrics for {classifier}')  # Título com nome do classificador
    ax.set_xticks(x)
    ax.set_xticklabels(classes, rotation='vertical')
    ax.legend()

    output_path = os.path.join(output_folder, f"{classifier}.png")
    plt.tight_layout()
    plt.savefig(output_path)
    plt.close()  # Fecha o plot atual
    plt.tight_layout()
    plt.show()

print("Plots saved in the output folder.")
```

Anexo U - Código de Comparação das métricas

```
import numpy as np

import matplotlib.pyplot as plt


classifiers = ["KNN (K=1)", "KNN (K=3)", "KNN (K=5)", "KNN (K=7)", "Naive Bayes", "Random Forest", "Decision Tree", "Ada Boost", "Multilayer Perceptron"]

classes = ["Área não encontrada", "Ciências e Tecnologias", "Ciências Socioeconómicas", "Línguas e Humanidades", "Artes Visuais"]



# Dados fornecidos (valores de TP, FN, FP)

tp_values = [

    [9, 14, 1, 0, 0],

    [11, 13, 0, 0, 0],

    [12, 14, 0, 0, 0],

    [9, 12, 0, 0, 0],

    [14, 9, 0, 0, 0],

    [12, 16, 1, 0, 0],

    [10, 11, 2, 0, 0],

    [3, 18, 0, 0, 0],

    [7, 17, 0, 0, 0]

]


fn_values = [

    [13, 8, 3, 4, 2],

    [11, 9, 4, 4, 2],
```

```
    [10, 9, 2, 4, 2],
    [13, 11, 3, 4, 2],
    [8, 13, 4, 4, 2],
    [10, 6, 2, 2, 1],
    [10, 9, 1, 4, 2],
    [15, 4, 4, 4, 2],
    [15, 5, 3, 4, 2]
]

fp_values = [
    [10, 16, 2, 0, 2],
    [14, 0, 0, 0, 0],
    [13, 15, 0, 0, 0],
    [16, 17, 0, 0, 0],
    [18, 13, 0, 0, 0],
    [8, 15, 0, 0, 0],
    [15, 15, 0, 0, 0],
    [22, 0, 0, 0, 0],
    [23, 0, 0, 0, 0]
]

x = np.arange(len(classes))
width = 0.2  # Largura das barras
```

```
    fig, axs = plt.subplots(len(classifiers), figsize=(10, 15))

    for i, classifier in enumerate(classifiers):
        rects1 = axs[i].bar(x - width, tp_values[i], width, label='True Positive', color='g')
        rects2 = axs[i].bar(x, fn_values[i], width, label='False Negative', color='r')
        rects3 = axs[i].bar(x + width, fp_values[i], width, label='False Positive', color='b')

        axs[i].set_xlabel('Classes')
        axs[i].set_ylabel('Counts')
        axs[i].set_title(f'Metrics for {classifier}')
        axs[i].set_xticks(x)
        axs[i].set_xticklabels(classes, rotation='vertical')
        axs[i].legend(fontsize='small')  # Reduz o tamanho da fonte da legenda

        # Ajuste manual do espaçamento vertical para evitar sobreposição
        axs[i].margins(y=0.1)
        axs[i].title.set_position([0.5, 1.05])  # Posição do título
        axs[i].legend(loc='upper right', bbox_to_anchor=(1.02, 1.0), borderaxespad=0.)  # Posição da legenda

    # Ajuste manual do espaçamento vertical entre subplots
    plt.subplots_adjust(hspace=0.5)
    plt.tight_layout()  # Organiza os subplots
    plt.savefig("combined_plots.png")
    plt.show()
```

Anexo V – Código com as curvas ROC de todos os classificadores

```
import matplotlib.pyplot as plt


classifiers = ["KNN (K=1)", "KNN (K=3)", "KNN (K=5)", "KNN (K=7)", "Naive Bayes", "Random Forest", "Decision Tree", "Ada Boost", "Multilayer Perceptron"]


# Dados fornecidos (valores de TP, FN, FP)
tp_values = [
    [9, 14, 1, 0, 0],
    [11, 13, 0, 0, 0],
    [12, 14, 0, 0, 0],
    [9, 12, 0, 0, 0],
    [14, 9, 0, 0, 0],
    [12, 16, 1, 0, 0],
    [10, 11, 2, 0, 0],
    [3, 18, 0, 0, 0],
    [7, 17, 0, 0, 0]
]


fn_values = [
    [13, 8, 3, 4, 2],
    [11, 9, 4, 4, 2],
    [10, 9, 2, 4, 2],
    [13, 11, 3, 4, 2],
    [8, 13, 4, 4, 2],
    [10, 6, 2, 2, 1],
```

```
    [10, 9, 1, 4, 2],
    [15, 4, 4, 4, 2],
    [15, 5, 3, 4, 2]
]

fp_values = [
    [10, 16, 2, 0, 2],
    [14, 0, 0, 0, 0],
    [13, 15, 0, 0, 0],
    [16, 17, 0, 0, 0],
    [18, 13, 0, 0, 0],
    [8, 15, 0, 0, 0],
    [15, 15, 0, 0, 0],
    [22, 0, 0, 0, 0],
    [23, 0, 0, 0, 0]
]

plt.figure(figsize=(8, 6))

for i, classifier in enumerate(classifiers):
    tpr = [tp / (tp + fn + 1e-6) for tp, fn in zip(tp_values[i], fn_values[i])]
    fpr = [fp / (fp + tn + 1e-6) for fp, tn in zip(fp_values[i], tp_values[i])]

    plt.plot(fpr, tpr, marker='o', label=classifier)
```

```
plt.plot([0, 1], [0, 1], 'k--')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Curvas Roc')

plt.legend(loc='lower right')

plt.grid(True)

plt.show()

plt.savefig('curvasRocTodos.png')
```

Anexo W – Código da Regressão Linear

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import joblib

# Dictionary with class names
class_names = {
    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
    6: "Profissional turismo e lazer",
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')

# X são as variáveis independentes, Y é a variável dependente
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]
y = df_area_scores["Área mais frequente Valor Numérico"]

# Divisão dos dados em conjunto de treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inicialização e treino do modelo de regressão linear
model = LinearRegression()
```

```
    model.fit(X_train, y_train)

    # Previsão nos dados de teste
    y_pred = model.predict(X_test)

    # Avaliação do modelo
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    r2 = r2_score(y_test, y_pred)
    # -*- coding: utf-8 -*-

    print(f'RMSE: {rmse:.2f}')
    print(f'R2 Score: {r2:.2f}')

    # Coeficientes da regressão
    coefficients = model.coef_
    intercept = model.intercept_

    print('Coeficientes:', coefficients)
    print('Intercept:', intercept)

    plt.scatter(y_test, y_pred)
    plt.plot([min(y_test), max(y_test)], [min(y_pred), max(y_pred)], color='red',
linewidth=2)
    plt.xlabel("Valores Reais")
    plt.ylabel("Previsões do Modelo")
    plt.title("Reta de Regressão: Valores Reais vs Previsões")
    plt.show()

    # Arredondar as previsões para os valores das classes (1 a 7)
    predicted_areas = np.clip(np.round(y_pred), 1, 7)
    # Create a DataFrame to store the true and predicted area values
    predictions_df = pd.DataFrame({
        "True_Area": y_test,
        "Predicted_Area": predicted_areas
    })
```

```
    # Mapear áreas previstas para nomes de classe
    predictions_df["True_Area_Name"] = predictions_df["True_Area"].map(class_names)
    predictions_df["Predicted_Area_Name"] = predictions_df["Predicted_Area"].map(class_names)

    # Calcular o RMSE
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))

    print("Previsões das áreas:")
    for _, row in predictions_df.iterrows():
        print(f"Área real: {row['True_Area_Name']}, Área prevista: {row['Predicted_Area_Name']}")

    print(f"RMSE: {rmse:.2f}")

    # Guardar as previsões num Excel
    predictions_df.to_excel('predictions_RegressaoLinear.xlsx', index=False)

    # Atribuir o modelo treinado à variável 'modelo'
    modelo = model

    # Especificar o nome do ficheiro para guardar modelo
    nome_ficheiro_modelo = 'modelo_regressao_linear.pkl'

    # Guardar o modelo num ficheiro usando joblib
    joblib.dump(modelo, nome_ficheiro_modelo)

    print(f"Modelo treinado guardado em {nome_ficheiro_modelo}")
```

Anexo X – Código da Regressão Polinomial

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load the Excel files into DataFrames
df_vader_metric = pd.read_excel('analiseperguntas9a14_VADER.xlsx')
df_area_scores = pd.read_excel('respostas_com_areas_e_scores1a8_maissentimento_235.xlsx')

# Extract features (X) and target (y) columns
X = df_vader_metric[["Média VaderMetrica 9", "Média VaderMetrica 10", "Média VaderMetrica 11", "Média VaderMetrica 12", "Média VaderMetrica 13", "Média VaderMetrica 14"]]
y = df_area_scores["Área mais frequente Valor Numérico"]

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("Size of y_test:", len(y_test))
print("Size of y_train:", len(y_train))
best_degree = None
best_rmse = float('inf')

for degree in range(1, 11):
    poly = PolynomialFeatures(degree)
    X_train_poly = poly.fit_transform(X_train)
    X_test_poly = poly.transform(X_test)

    model = LinearRegression()
    model.fit(X_train_poly, y_train)
    y_pred = model.predict(X_test_poly)
```

```
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))

    if rmse < best_rmse:
        best_rmse = rmse
        best_degree = degree

    print(f'Polynomial Degree: {degree}')
    print(f'RMSE: {rmse:.2f}')
    print(f'R-squared: {r2_score(y_test, y_pred):.2f}')

print(f'Best Polynomial Degree: {best_degree}')
print(f'Best RMSE: {best_rmse:.2f}')

class_names = {
    0: "Área não encontrada",
    1: "Ciências e Tecnologias",
    2: "Ciências Socioeconómicas",
    3: "Línguas e Humanidades",
    4: "Artes Visuais",
    5: "Profissional informatica eletronica e comunicacoes",
    6: "Profissional turismo e lazer",
    7: "Profissional cultura, patrimonio e gestão de conteudos"
}

# Arredondar as previsões para os valores das classes (1 a 7)
predicted_areas = np.clip(np.round(y_pred), 1, 7)

# Mapear áreas previstas para nomes de classe
predicted_class_names = [class_names[area] for area in predicted_areas]

# Calculate the RMSE
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

print("Predicted Areas:")
```

```
    for true_area, predicted_area in zip(y_test, predicted_areas):
        if true_area == 0:
            true_area_name = "Área não encontrada"
        else:
            true_area_name = class_names[true_area]

        predicted_area_name = class_names[predicted_area]

        print(f"True Area: {true_area_name}, Predicted Area: {predicted_area_name}")

    print(f'RMSE: {rmse:.2f}')

    # Scatter plot of true vs predicted areas
    plt.figure(figsize=(10, 8))
    plt.scatter(y_test, y_pred, color='blue')
    plt.plot([min(y_test), max(y_test)], [min(y_pred), max(y_pred)], color='red', linewidth=2)
    plt.xlabel("True Areas")
    plt.ylabel("Predicted Areas")
    plt.title("True Areas vs Predicted Areas")
    plt.show()

    # Save the DataFrame with the predicted areas in an Excel file
    result_df = pd.DataFrame({'True Area': [class_names[area] for area in y_test], 'Predicted Area': predicted_class_names})
    excel_output_file_with_predictions = 'predicted_areas_polynomial.xlsx'
    result_df.to_excel(excel_output_file_with_predictions, index=False)

    print("Processing completed. Predicted areas saved in the Excel file:", excel_output_file_with_predictions)

    # Mapear áreas previstas para nomes de classe
    predicted_class_names = [class_names[area] for area in predicted_areas]

    # Criar um DataFrame com as previsões e as áreas reais
    result_df = pd.DataFrame({
```

```
    'True Area': [class_names[area] if area in class_names else "Área não encontrada" for area in y_test],
    'Predicted Area': predicted_class_names
})

# Visualizar todas as previsões
pd.set_option('display.max_rows', None)
print(result_df)

# Guardar o DataFrame num novo ficheiro Excel
result_df.to_excel('areas_previstas_regressao_polinomial.xlsx', index=False)

# Coeficientes da regressão
coefficients = model.coef_
intercept = model.intercept_

print('Coeficientes:', coefficients)
print('Intercept:', intercept)



print(f'Polynomial Degree: {degree}')
print(f'RMSE: {rmse:.2f}')
print(f'R-squared: {r2_score(y_test, y_pred):.2f}')

print(f'Best Polynomial Degree: {best_degree}')
print(f'Best RMSE: {best_rmse:.2f}')
```

Anexo Y – Código em python da aplicação web

```
# Importar todas as bibliotecas necessárias
from flask import Flask, jsonify, request, render_template
import joblib
import numpy as np

app = Flask(__name__)

# Carregar o modelo de regressão linear
modelo = joblib.load('modelo_regressao_linear.pkl')

# Configurar a rota para a página HTML
@app.route('/')
def index():
    return render_template('index.html', resultado='')

# Configurar a rota para a previsão via POST
@app.route('/predict', methods=['POST'])
def predict():
    # Receber os scores dos utilizadores através de uma solicitação POST
    score1 = float(request.json['Média VaderMetrica 9'])
    score2 = float(request.json['Média VaderMetrica 10'])
    score3 = float(request.json['Média VaderMetrica 11'])
    score4 = float(request.json['Média VaderMetrica 12'])
    score5 = float(request.json['Média VaderMetrica 13'])
    score6 = float(request.json['Média VaderMetrica 14'])

    # Realizar a previsão com base nos scores
    previsao = modelo.predict([[score1, score2, score3, score4, score5, score6]])

    # Formatar a previsão como resposta JSON
    return jsonify({'previsao': previsao[0]})

if __name__ == '__main__':
    app.run()
```

Anexo Z- Código da previsão em javascript

```
document.getElementById('sentiment-form').addEventListener('submit', function(e) {
    e.preventDefault();

    // Recolha dos valores dos scores de análise de sentimento
    var score1 = parseFloat(document.getElementById('score1').value);
    var score2 = parseFloat(document.getElementById('score2').value);
    var score3 = parseFloat(document.getElementById('score3').value);
    var score4 = parseFloat(document.getElementById('score4').value);
    var score5 = parseFloat(document.getElementById('score5').value);
    var score6 = parseFloat(document.getElementById('score6').value);

    // Criar um objeto com os scores
    var dados = {
        'Média VaderMetrica 9': score1,
        'Média VaderMetrica 10': score2,
        'Média VaderMetrica 11': score3,
        'Média VaderMetrica 12': score4,
        'Média VaderMetrica 13': score5,
        'Média VaderMetrica 14': score6
    };

    // Enviar uma solicitação POST para o servidor Flask
    fetch('/predict', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
    },
    body: JSON.stringify(dados),
})
.then(function(response) {
    if (response.status === 200) {

        // Por exemplo, exibir uma mensagem de sucesso
        console.log('Previsão bem-sucedida');
```

```
    } else {
        // Tratar erros de resposta do servidor aqui
        console.error('Erro na previsão: ' + response.status);
    }
})
.catch(function(error) {
    // Tratar erros de rede aqui
    console.error('Erro de rede:', error);
});
```

Anexo AA- Código do formulário html

```
<!DOCTYPE html>
<html>
<head>
    <title>Previsão de Área Vocacional</title>
</head>
<body>
    <h1>Previsão de Área Vocacional com Base em Scores de Análise de Sentimento</h1>
    <form id="sentiment-form">
        <label for="score1">Score 1:</label>
        <input type="number" id="score1" step="0.01" required><br>

        <label for="score2">Score 2:</label>
        <input type="number" id="score2" step="0.01" required><br>

        <label for="score3">Score 3:</label>
        <input type="number" id="score3" step="0.01" required><br>

        <label for="score4">Score 4:</label>
        <input type="number" id="score4" step="0.01" required><br>

        <label for="score5">Score 5:</label>
        <input type="number" id="score5" step="0.01" required><br>

        <label for="score6">Score 6:</label>
        <input type="number" id="score6" step="0.01" required><br>

        <input type="submit" value="Prever Área">
    </form>
    <p>Área Prevista: <span id="predicted-area"></span></p>
    <script src="/areaprevista/previsao.js"></script>

</body>
</html>
```

Anexo AB – Tabela dos resultados com o número de alunos por área vocacional

Escola que frequentas	Área Vocacional	Número de Alunos
ESCOLA SECUNDÁRIA DE CORUCHE	Artes Visuais	3
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências Socioeconómicas	1
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	26
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	14
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	5
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	11
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	64
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	6
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	60
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	37
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Línguas e Humanidades	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	36
Total		267

Anexo AC – Tabela dos resultados com o número de alunos por área Vocacional e Sexo

Escola que frequentas	Área Vocacional	Sexo	Número de Alunos
ESCOLA SECUNDÁRIA DE CORUCHE	Artes Visuais	Feminino	1
ESCOLA SECUNDÁRIA DE CORUCHE	Artes Visuais	Prefiro não responder	2
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências Socioeconómicas	Feminino	1
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Feminino	17
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Masculino	9
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	9
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	5
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Masculino	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Feminino	5
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Masculino	6
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Feminino	31
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Masculino	32
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Prefiro não responder	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	5
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Masculino	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	28
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	31
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Prefiro não responder	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	Feminino	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	Masculino	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Feminino	20

ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Masculino	15
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Prefiro não responder	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Línguas e Humanidades	Feminino	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Línguas e Humanidades	Masculino	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	14
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	20
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Prefiro não responder	2
Total			267

Anexo AD – Tabela dos resultados com o número de alunos por área vocacional, sexo e idade

Escola que frequentas	Área Vocacional	Sexo	Que idade tens?	Número de Alunos
ESCOLA SECUNDÁRIA DE CORUCHE	Artes Visuais	Feminino	16	1
ESCOLA SECUNDÁRIA DE CORUCHE	Artes Visuais	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA DE CORUCHE	Artes Visuais	Prefiro não responder	16	1
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências Socioeconómicas	Feminino	15	1
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Feminino	14	8
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Feminino	15	8
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Feminino	16	1
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Masculino	14	3
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências e Tecnologias	Masculino	15	6
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	14	5
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	15	2
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	16	2
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	14	1
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	15	2
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	16	2
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	14	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	17	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	19	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Masculino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Feminino	14	2

ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Feminino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Feminino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Feminino	17	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Masculino	15	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Masculino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Masculino	17	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Feminino	14	6
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Feminino	15	20
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Feminino	16	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Feminino	17	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Masculino	14	12
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Masculino	15	11
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Masculino	16	5
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Masculino	17	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	14	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	17	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	18	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Masculino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	14	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	15	12
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	16	8

ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	17	3
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	18	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	14	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	15	16
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	16	7
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	17	3
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	18	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Prefiro não responder	14	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	Feminino	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	Masculino	14	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Feminino	14	12
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Feminino	15	8
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Masculino	14	6
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Masculino	15	8
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Masculino	16	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Prefiro não responder	14	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Ciências e Tecnologias	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Línguas e Humanidades	Feminino	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Línguas e Humanidades	Masculino	15	1

ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	14	6
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	15	5
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	16	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	18	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	14	7
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	15	11
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	16	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Prefiro não responder	16	1
Total				267

Anexo AE – Tabela com os resultados por área, sexo, escola e idade com uso de métrica personalizada e o VADER

Escola que frequentas	Área Vocacional	Sexo	Que idade tens?	Número de Alunos
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências Socioeconómicas	Masculino	14	1
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	14	13
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	15	11
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Feminino	16	4
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	14	3
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	15	8
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Masculino	16	2
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	Prefiro não responder	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Feminino	17	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	Masculino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Feminino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	Masculino	16	2
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Masculino	14	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	Feminino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional informatica eletronica e comunicacoes	Feminino	14	1

ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional informatica eletronica e comunicacoes	Feminino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional informatica eletronica e comunicacoes	Feminino	16	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional informatica eletronica e comunicacoes	Masculino	15	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional turismo e lazer	Feminino	14	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	14	12
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	15	32
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	16	12
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	17	6
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	18	2
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Feminino	19	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	14	15
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	15	30
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	16	12
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	17	8
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Masculino	18	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	Prefiro não responder	14	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	Feminino	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Profissional turismo e lazer	Masculino	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Profissional turismo e lazer	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	14	18

ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	15	14
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	16	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Feminino	18	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	14	14
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	15	19
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Masculino	16	3
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Prefiro não responder	14	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Prefiro não responder	15	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	Prefiro não responder	16	1
Total				267

Anexo AF – Tabela dos resultados por área vocacional e escola com o VADER e métrica personalizada

Escola que frequentas	Área Vocacional	Número de Alunos
ESCOLA SECUNDÁRIA DE CORUCHE	Ciências Socioeconómicas	1
ESCOLA SECUNDÁRIA DE CORUCHE	Área não encontrada	43
ESCOLA SECUNDÁRIA DE SACAVÉM	Artes Visuais	3
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências Socioeconómicas	3
ESCOLA SECUNDÁRIA DE SACAVÉM	Ciências e Tecnologias	2
ESCOLA SECUNDÁRIA DE SACAVÉM	Línguas e Humanidades	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional informatica eletronica e comunicacoes	4
ESCOLA SECUNDÁRIA DE SACAVÉM	Profissional turismo e lazer	1
ESCOLA SECUNDÁRIA DE SACAVÉM	Área não encontrada	132
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Artes Visuais	1
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Profissional turismo e lazer	2
ESCOLA SECUNDÁRIA ENG° ACÁCIO CALAZANS DUARTE	Área não encontrada	74
Total		267