Instituto Politécnico de Coimbra

Instituto Superior de Engenharia de Coimbra

Departamento de Engenharia Informática e de Sistemas

Ferramentas Open Source de Data Mining

Tânia Gomes

Mestrado em Informática e Sistemas

Coimbra, Dezembro, 2014

Instituto Politécnico de Coimbra

Instituto Superior de Engenharia de Coimbra

Departamento de Engenharia Informática e de Sistemas

Ferramentas Open Source de Data Mining

Dissertação apresentada para obtenção do grau de Mestre em Informática e Sistemas

Autor

Tânia Gomes

Orientador

Prof. Doutor Viriato M. Marques

ISEC - DEIS

Coimbra, Dezembro, 2014

AGRADECIMENTOS

A conclusão desta dissertação representa o cumprimento de um grande objetivo pessoal. Este foi o percurso mais difícil e desafiante que tive, e que só foi possível graças ao apoio das pessoas que passo a agradecer:

Ao Professor Doutor Viriato Marques que me orientou neste projeto e sempre me acompanhou e motivou, com a sua boa disposição e otimismo, tornando possível a conclusão deste curso.

Aos meus pais, irmã e avós por todo o esforço que fazem para que seja feliz e consiga cumprir os meus objetivos pessoais.

Ao meu namorado que esteve sempre presente nos momentos bons e maus, e sempre me deu a confiança necessária para continuar.

A todos os professores e colegas do Mestrado, pela forma como me receberam, pela amizade demonstrada, pela compreensão e paciência ao longo deste percurso!

E por fim, a todos os amigos, colegas de trabalho e colegas de equipa que sempre me acompanharam, representando um apoio fundamental neste percurso.

"Some men see things as they are and say why.
I dream things that never were and say why not."

Robert.F.Kennedy

RESUMO

Em época de crise financeira, as ferramentas *open source* de *data mining* representam uma nova tendência na investigação, educação e nas aplicações industriais, especialmente para as pequenas e médias empresas. Com o *software open source*, estas podem facilmente iniciar um projeto de *data mining* usando as tecnologias mais recentes, sem se preocuparem com os custos de aquisição das mesmas, podendo apostar na aprendizagem dos seus colaboradores. Os sistemas *open source* proporcionam o acesso ao código, facilitando aos colaboradores a compreensão dos sistemas e algoritmos e permitindo que estes o adaptem às necessidades dos seus projetos. No entanto, existem algumas questões inerentes ao uso deste tipo de ferramenta. Uma das mais importantes é a diversidade, e descobrir, tardiamente, que a ferramenta escolhida é inapropriada para os objetivos do nosso negócio pode ser um problema grave. Como o número de ferramentas de *data mining* continua a crescer, a escolha sobre aquela que é realmente mais apropriada ao nosso negócio torna-se cada vez mais difícil. O presente estudo aborda um conjunto de ferramentas de *data mining*, de acordo com as suas características e funcionalidades. As ferramentas abordadas provém da listagem do KDnuggets referente a *Software Suites* de *Data Mining*. Posteriormente, são identificadas as que reúnem melhores condições de trabalho, que por sua vez são as mais populares nas comunidades, e é feito um teste prático com *datasets* reais. Os testes pretendem identificar como reagem as ferramentas a cenários diferentes do tipo: *performance* no processamento de grandes volumes de dados; precisão de resultados; etc. Nos tempos que correm, as ferramentas de *data mining open source* representam uma oportunidade para os seus utilizadores, principalmente para as pequenas e médias empresas, deste modo, os resultados deste estudo pretendem ajudar no processo de tomada de decisão relativamente às mesmas.

ABSTRACT

In times of financial crisis, the open source data mining tools represent a new trend in research, education and industrial applications, especially for small and medium enterprises. With open source software, they can easily initiate a data mining project using the latest technologies without worrying about the costs of acquiring them, and may invest in employees' learning. The *open source* systems provide access to the code, making it easier for employees to understand the systems and algorithms, and giving them the ability to adapt the system for their project needs. However, there are some issues inherent with the use of this types of tools. One of the most important is diversity, and finding out, too late, that the chosen tool is inappropriate for the objectives of our business can be a real problem. As the number of data mining tools continues to grow, the choice on which is more appropriate to our business tends to become increasingly difficult. This study presents a set of data mining tools, according to their features and functionality. The tools discussed come from a list of KDnuggets regarding Software Suites for Data Mining. Subsequently, we gather the ones with better working conditions, which in turn are also the most popular in the communities, and make a practical test with real datasets. The tests are intended to identify how the tools react to different scenarios such as: performance in processing large volumes of data; accuracy of results; and so on. Nowadays, the *open source* data mining tools represent an opportunity for its users, especially for small and medium enterprises, thus the results of this study are intended to help in the decision making process regarding them.

PALAVRAS-CHAVE

- *Open source*
- Livre
- *Data mining*
- Conhecimento
- Descoberta de conhecimento
- Análise de dados

KEYWORDS

- Open source
- Free
- Data mining
- Knowledge
- Knowledge discovery
- Data Analysis

ABREVIATURAS

ADaM – Algorithm Development and Mining System

AGPL – Affero General Public License

AHC – Agglomerative Hierarchical Clustering

API - Application Programming Interface

BD – Base de dados

BI – Business Intelligence

BSD – Berkeley Sofware Distribution

CART – Classification and Regression Trees

CMSR – Cramer Modeling Segmentation & Rules

CRAMER – Classificaçaão de árvores de decisão e segmentação

CRAN – Comprehensive R Archive Network

CRISP-DM – Cross Industry Standard Process for Data Mining

CSV – Comma-Separated Values

DBMS – Database Management Systems

DBSCAN – Density-Based Spatial Clustering of Applications with Noise

DCBD – Descoberta de Conhecimento em Bases de Dados

DM – Data Mining

EA – Evolucionary Algorithms

ELKI – Environment for Developing KDD-Applications supported by Index-Structures

EM – Expectation – Maximization

ESML – Earth Science Markup Language

ESOM – Emergent Self-Organizing Maps

ETI – E-business Technology Institute

ETL – Extract Transform and Load

FP-Growth – Frequent Pattern Growth

FSF – Free Software Foundation

GCC – GNU Compiler Colection

GiST – Generalized Search Tree

GPL – General Public License

GUI – Guide User *Interface*

HTML – Hypertext Markup Language

I/O – Input/ Output

IDE – Integrated Development Environment

JDBC – Java Database Connectivity

JRE – Java Runtime Environment

KDD – Knowledge Discovery in Databases

KDnuggets – Knowledge Discovery Nuggets

KEEL – Knowledge Extraction for Evolucionary Learning

KNIME – Kontanz Information Miner

KNN – K-Nearest Neighbors

LGPL – Lesser General Public License

MJL – Machine learning in Java

ODBC – Open Database Connectivity

OpenNN – Open Neural Networks Library

OS – Open Source

OSI – Open Source Initiative

PME – Pequenas e Médias Empresas

Rattle – R Analytical Tool to Learn Easy

RDS – Relational Database System

ROC – Receiver Operating Characteristic

SAS – Statistical Analysis System

SCaVis – Scientific Computation and Visualization Environment

SEMMA – Sample, Explore, Modify, Model and Assess

SGBD – Sistema de Gestão de Base de dados

SIG – Silicom Graphics International

SIGKDD – Special Interest Group on Knowledge Discovey in Databases

SPSS – Spatial Package for the Social Sciences

SQL – Structured Query Language

SVM – Support Vector Machines

TCL/TK – Tool Command Language/ Toolkit

URL – Uniform Resource Locator

USB – Universal Serial Bus

XML – Extensible Markup Language

WEKA – Waikato Environment for Knowledge Analysis

YALE – Yet Another Learning Environment

ÍNDICE

CAPÍTULO 5 – FERRAMENTAS OPEN SOURCE DE DATA MINING

CAPÍTULO 6 – AVALIAÇÃO PRÁTICA DAS FERRAMENTAS

CAPÍTULO 7 – IMPLEMENTAÇÃO PRÁTICA

CAPÍTULO 8 - CONCLUSÕES E TRABALHO FUTURO

CAPÍTULO 9 - REFERÊNCIAS BIBLIOGRÁFICAS

ANEXO A

ANEXO B

ANEXO C

ANEXO D

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

1 INTRODUÇÃO

A crise financeira vivida atualmente incita á poupança de todos os recursos possíveis por parte das empresas. Com orçamentos controlados ao milímetro, estas são obrigadas a continuar a apresentar resultados e a manter uma posição na linha da frente dos negócios. A falta de investimento em soluções de inovação e desenvolvimento, leva á procura e criação de soluções que permitam maximizar e otimizar os recursos que ainda estão disponíveis. Esta questão afeta sobretudo as pequenas e médias empresas (PME's), na medida em que estas não têm capacidade para continuar a investir na melhoria dos seus processos de negócio, como acontece nas grandes empresas. A quantidade de produção, nestes casos, passa á frente da qualidade de produção, que cada vez mais se torna um fator incómodo, que vai contra a rentabilidade dos negócios. Nas grandes empresas, o cenário é diferente, uma vez que estas têm poder para investir devido á sua forte capacidade económica.

O objetivo deste estudo é, de forma geral, propor algumas soluções para estas questões do dia-a-dia. As soluções propostas podem ajudar no aumento dos lucros das empresas, na melhoria dos processos e, principalmente na diminuição de custos. Com estes fatores melhorados, as empresas poderão começar a investir de uma forma mais rentável.

As técnicas de *data mining* (DM) reúnem métodos de várias áreas que, ao longo dos anos, se têm desenvolvido, como a estatística, inteligência artificial e *machine learning*. Estas técnicas permitem a transformação de informação em conhecimento potencialmente útil. Qualquer empresa, na sua atividade diária ou mensal, produz dados como resultado dos seus processos de trabalho. Dados que se encontram num estado cru, em grandes volumes, dispersos por ficheiros, vários repositórios e bases de dados e que não representam qualquer tipo de conhecimento. Para tirar todas as vantagens desses dados, a consulta de dados não é suficiente, sendo que é necessário o uso de uma ferramenta de *software* para sumarização automática de dados, extração da essência da informação armazenada e descoberta de padrões. As ferramentas de DM possibilitam a análise desses dados, para prever ou descrever comportamentos, de forma a assistir os gestores nas suas tomadas de decisão. O uso deste tipo de ferramenta permite aos gestores tomar conhecimento de informações como o comportamento dos clientes (p.ex. num supermercado, perceber que produtos são comprados conjuntamente); desvios significativos nos dados (p. ex. detetar fraudes no uso de cartões de crédito), entre muitas outras. Assim os gestores podem tomar decisões baseadas nestas análises, de forma a criar novas soluções que sejam mais rentáveis para os seus negócios (p. ex. através de técnicas de *marketing*, aproximar os produtos que são comprados conjuntamente na disposição das estantes, ou criar promoções sobre eles) ou de forma a prevenir possíveis comportamentos indesejados (p.ex. localizar o cartão de crédito que está a ser utilizado de forma indevida).

Uma vez que as ferramentas comerciais de *software* implicam um grande custo de utilização e manutenção, torna-se impossível para as empresas com menos capacidades financeiras arriscar num investimento do género. Com as ferramentas de *data mining*, acresce o problema da falta de conhecimento relativamente às técnicas de *data mining*, ou do próprio sistema de funcionamento, o que implica mais tempo e dinheiro na formação dos colaboradores. Assim, o nosso estudo incide nas ferramentas *data mining open source*, uma vez que estas representam uma grande oportunidade para as empresas com menos capacidades financeiras. Os sistemas *open source* são desenvolvidos por comunidades de programadores que produzem *software*, de forma não lucrativa, de acordo com um conjunto de princípios definidos pela *Open Source Initiative* (OSI). Este tipo de produtos, usualmente disponível a custo zero, possibilita que os colaboradores compreendam o sistema através da consulta livre do código, possibilitando a sua alteração conforme as necessidades dos seus projetos. O *software open source* está a desempenhar um papel muito importante nos sectores público e privado por todo o mundo. Uma grande quantidade de ferramentas de *data mining* está agora disponível para estudantes, profissionais, investigadores, etc. (PAKDD'09, 2009). No entanto, o impacto e investimento envolvido no processo de seleção de uma ferramenta são muito importantes. A seleção de uma ferramenta imprópria para um propósito específico tem consequências como desperdício de recursos financeiros, tempo, recursos humanos e a obtenção de resultados indesejados (Collier, Carey, Sautter, & Marjaniemi, 1999).

Depois de um estudo às ferramentas *open source* de DM disponíveis no mercado, decidimos analisar um conjunto de ferramentas eleito pela KDnuggets *(Knowledge Discovery nuggets)*, como *Software Suites for Data mining, Analytics, and Knowledge Discovery,* selecionando posteriormente as melhores e mais populares no mercado para análise de um conjunto de dados reais. Esta análise prática pretende discernir como as ferramentas se comportam em vários cenários relativamente á performance, precisão, etc., comprovando assim a sua capacidade para operar no mundo empresarial.

As ferramentas *open source* de *data mining* são uma solução viável para a modernização dos processos de trabalho, inovação e desenvolvimento de uma empresa. No entanto, é necessário perceber que estas implicam algum conhecimento na sua implementação e utilização. Atualmente, novas ferramentas ou novas versões são constantemente desenvolvidas e estas mostram-se cada vez mais intuitivas e amigáveis, principalmente ao nível de *interface*s de utilizador.

1.1 Principais contribuições deste trabalho

O objetivo central deste estudo é contribuir para a evolução do conhecimento no uso das ferramentas *open source* de *data mining*. Assim, as principais contribuições deste trabalho são as seguintes:

- Demonstrar a possibilidade de redução de custos das empresas com menos capacidade de investimento, com o uso das ferramentas *open source*;
- Esclarecer o processo de Descoberta de Conhecimento em Bases de Dados (DCBD);
- Clarificar o conceito de *data mining* e os métodos abrangidos nele;
- Divulgar as ferramentas *open source*;
- Descrever as ferramentas *open source* de *data mining*, provenientes da listagem do KDnuggets;
- Comparar e avaliar as ferramentas *open source* de *data mining* mais populares;
- Ajudar no processo de seleção de uma ferramenta *open source* adequada às necessidades.

Assim pretendemos contribuir de forma proactiva para o desenvolvimento do tema da nossa investigação.

1.2 Estrutura do relatório

O presente estudo está organizado em 8 capítulos:

- Capítulo 1 – O presente capítulo, refere-se á parte introdutória, onde é descrito o tema em estudo e a sua estrutura.
- Capítulo 2 – O objetivo deste capítulo é apresentar uma análise do estado da arte de *data mining*. É feita um breve enquadramento histórico e serão apresentadas algumas contribuições na área.
- Capítulo 3 – O capítulo 3 aborda todo o processo de *Knowledge Discovey in Databases (KDD).* Aqui são descritas as várias fases que fazem parte do seu ciclo de vida. Seguidamente é apresentado o conceito de *data mining* em pormenor e todas as fases do seu ciclo de vida. São enumeradas algumas vantagens e desvantagens da técnica e ainda os principais métodos usados: classificação, previsão, *clustering*, regressão e deteção de desvios.
- Capítulo 4 – O capítulo 4 refere-se aos sistemas *open source*. Aqui são apresentadas algumas vantagens e desvantagens do uso deste tipo de sistema e são ainda mencionados alguns repositórios, onde é possível encontrar este tipo de sistema. Relativamente às várias licenças de desenvolvimento são abordadas as mais usadas, segundo a OSI, e quais os seus princípios gerais.

- Capítulo 5 – No capítulo 5 são abordadas e descritas 23 ferramentas *open source* de *data mining*, consideradas como *suites* de *data mining* pelo KDnuggets. Aqui são abordadas as principais características de cada uma, o objetivo para que foram criadas e algumas informações base sobre as mesmas. Por fim é apresentado uma tabela, que resume as principais características das 23 ferramentas.
- Capítulo 6 – O objetivo do capítulo 6 é proporcionar um resumo comparativo das potencialidades das quatro ferramentas em análise. Aqui são demonstradas as características de cada uma ao nível de requisitos, funcionalidades e compatibilidade.
- Capítulo 7 – No capítulo 7 são apresentados os resultados dos testes práticos aplicados às quatro ferramentas. A comparação dos resultados obtidos pelas tarefas de classificação, regressão, *clustering* e associação são o objetivo principal.
- Capítulo 8 – Por fim, o capítulo 8 apresenta as conclusões práticas e teóricas de todo o trabalho.
- Capítulo 9 e 10 – Os últimos capítulos referem-se às referências bibliográficas e anexos respetivamente.

2 ESTADO DA ARTE

O trabalho desenvolvido no âmbito das ferramentas de *data mining* é vasto e tem sido diversificado ao longo dos tempos. Existem contribuições de vários tipos, umas que abordam conceitos e técnicas mais próximas da área de *data mining* e outras que cruzam essas questões com o uso de ferramentas de *software*. Os contributos mais comuns abordam comparações entre os vários *software*s e sistemas de *data mining*.

De seguida iremos apresentar algumas das abordagens estudadas neste domínio, e que pensamos terem sido os principais marcos na evolução das ferramentas *open source* de *data mining*.

Segundo (Mikut & Reischl, 2011), a expressão *data mining* surge nos anos 80 pelo investigador Lovell. Esta área evoluiu em simultâneo com o desenvolvimento de ferramentas de *software* para fins estatísticos e de análise de dados.

Em 1996, os autores (Fayyad, Piatetsky-Shapiro, & Smyth, 1996) publicam um estudo em que abordam a distinção e relação de *data mining* e *Knowledge Discovery in Databases* (KDD), evidenciando que estas estão também relacionadas com outras áreas como a estatística, *machine learning* e as bases de dados. Para além dessa questão, explicam a importância da evolução das técnicas e ferramentas desta área emergente, demonstrando algumas áreas de aplicação em que já são usadas com frequência, como o *marketing*, finanças, manufatura, saúde, ciência, retalho, etc.

No seu estudo, (Mikut & Reischl, 2011) definem *data mining* como uma das tarefas que constitui a descoberta de conhecimento em bases de dados (DCBD), que consiste na análise de dados e descoberta de algoritmos, para produzir uma enumeração específica de padrões sobre os dados analisados. Também a DCBD (ou KDD) é definido como o processo não trivial de identificar padrões novos, potencialmente úteis e compreensíveis nos dados. Neste sentido, os autores clarificam que esta última definição é a que todas as ferramentas de *software* suportam, uma vez que estas são responsáveis por todo o processo de KDD e não só pela tarefa de *data mining*.

Relativamente às técnicas de *data mining*, estas também têm as suas origens em métodos já existentes. Os métodos estatísticos clássicos, onde o objetivo central passou da confirmação de potenciais hipóteses para a criação de novas hipóteses, são exemplo de métodos de onde terão origem algumas técnicas de *data mining*, como p.ex. os métodos de *Bayes* e Regressão. Outro grupo de métodos estará associado á inteligência artificial, como são exemplo as árvores de decisão e os sistemas baseados em regras. Por fim, o *machine learning* também ele deu origem a métodos como por exemplo, as *support vector machines* e as redes neuronais.

Ao longo dos tempos muitas ferramentas de *data mining* têm sido criadas, e algumas duraram um período muito curto de tempo. Essas questões estão relacionadas muitas vezes com decisões de *marketing*. As ferramentas de *data mining* comerciais que tiveram sucesso, resultaram da integração

inteligente de métodos de *data mining* em ferramentas estatísticas que já se encontravam estabelecidas, são exemplo as empresas SPSS Inc., fundada em 1975 e a SAS Inc. em 1976. Estas ferramentas foram adaptadas a computadores pessoais e soluções de cliente/servidor para grandes clientes. Com o aumento da popularidade da *data mining*, algoritmos como redes neuronais ou árvores de decisão foram integrados nos produtos principais das empresas especializadas de *data mining* como a *Integrated Solutions, Ltd*., (adquirida pela SPSS em 1998). Esta foi adquirida para facultar o acesso a ferramentas de *data mining* como o *Clementine*, popular na época. Nessa altura, muitas empresas de *software* mudaram de nome, sendo que, por exemplo, o *Clementine* (SPSS) passou a chamar-se *PASW Modeler* e agora encontra-se disponível como *IBM SPSS Modeler*, depois da aquisição da SPSS pela IBM, em 2009. Atualmente as empresas que integram soluções de *business intelligence* também incorporam métodos de *data mining* para os seus produtos, como é o exemplo da *Oracle data mining* (Mikut & Reischl, 2011).

Relativamente às soluções *open source*, também ficaram muito populares nos anos 90, sendo que, o caso mais conhecido foi o aparecimento da WEKA. Esta iniciou-se como uma biblioteca escrita em C++, e a sua primeira versão foi lançada em 1996. Mais tarde foi desenvolvida de origem mas, desta vez, em linguagem java e a partir daí tem sido atualizada regularmente. Com esta mudança tornou-se muito popular e algumas das suas componentes foram usadas para integrar outras ferramentas *open source* de *data mining*, como é o caso do RapidMiner e KNIME. As ferramentas de *data mining* continuam a crescer a bom ritmo e a escolha de uma em particular torna-se cada vez mais difícil. Nos últimos anos a área de *data mining* tornou-se uma tecnologia muito importante em vários sectores, como são exemplo, as investigações de genética onde foi essencial, e está a mostrar-se também muito significativa na investigação da semântica e do *text mining*.

No que diz respeito a análises comparativas, metodologias ou propostas para escolha de ferramentas de *data mining*, existem algumas que constam na revisão bibliográfica deste estudo, e que são publicadas quase de ano para ano:

- No ano de 1999, os autores (Collier, Carey, Sautter, & Marjaniemi, 1999) lançam uma metodologia para ajudar na escolha do *software* de *data mining*, uma vez que se trata de um processo relativamente difícil e que, se for feito da forma errada pode trazer consequências pouco desejáveis. Assim, desenvolveram uma metodologia e uma *framework* para avaliação de ferramentas de *data mining*. Apesar das ferramentas em estudo serem comerciais, o estudo mostra-se muito interessante, uma vez que são considerados vários critérios de análise, onde é tido em conta que não existe uma ferramenta de *data mining*, que seja melhor em todos os propósitos. Baseada em *scoring*, esta metodologia atribui pesos aos critérios que considera importantes e assim, num quadro geral apresenta as comparações com os resultados quantitativos das várias ferramentas. Esta metodologia era usada até á data pelo *Center of Data Insight* e servia para prestar apoio aos utilizadores que estivessem á procura de orientação na seleção de uma ferramenta de *data mining*. Os resultados e o *feedback* eram muito positivos e úteis.

- Mais tarde em 2006, também os autores (Britos, et al., 2006) desenvolvem uma metodologia para a seleção de uma ferramenta de *data mining*, baseando-se na mesma questão, ou seja uma escolha inapropriada de ferramenta de *data mining* pode significar perda de tempo e de dinheiro. Neste caso, as ferramentas em causa para aplicação da metodologia são ambas comerciais e *open source*. Esta metodologia baseia-se em atribuição de pesos aos critérios *(weighting)* e, neste caso, os critérios podem ser definidos previamente pelo utilizador ou pela empresa, o que torna este método muito mais flexível e adaptável às necessidades de cada um.
- Em 2007, os autores (Chen, Williams, & Xu, 2007) comparam 12 ferramentas de *data mining* que se encontram disponíveis na *internet*. O objetivo é comparar estas ferramentas em vários aspetos: características gerais, acessibilidade da fonte de dados, funcionalidades de *data mining* e usabilidade. Por fim, são discutidas algumas vantagens e desvantagens do uso deste tipo de sistemas *open source*. As ferramentas incluídas no estudo são: KNIME, YALE, WEKA, MLC++, RATTLE, TANAGRA, MiningMart, Orange, ADaM, AlphaMiner, Gnome Data Miner e Databionic ESOM. Neste estudo concluiu-se que, quase todos os sistemas apresentavam boas funcionalidades, e que disponibilizam ferramentas muito poderosas para aplicar em campos como os da investigação e da educação. No entanto, para usos comerciais ainda haviam questões a ser melhoradas, para que estas estivessem a altura e fossem acessíveis. Fatores como suporte de várias fontes de dados; alta performance na execução de tarefas de *data mining*, no que diz respeito a tratamento de largos volumes de dados; escalabilidade; segurança; e confiança foram sugeridos para integrar e melhorar o *deployment* das ferramentas de *data mining* em estudo.
- No ano seguinte, (Zupan & Demsar, 2008) fazem um estudo relativamente á evolução das abordagens usadas no desenvolvimento das ferramentas de *data mining*, analisando em particular os seus *interface*s de utilizador. As ferramentas em análise foram as seguintes: R, TANAGRA, WEKA, YALE, KNIME, Orange, GGobi. Nesse mesmo estudo, os autores abordam ainda algumas das vantagens do uso dos sistemas *open source* relativamente aos comerciais e concluem com uma *wishlist* de características, que uma ferramenta de *data mining* deve incluir para atuar no campo da biomédica. Em suma, concluíram que as ferramentas de *data mining* oferecem um bom *interface* gráfico, que se foca na usabilidade, interatividade e flexibilidade. Para além disso, as ferramentas encontram-se bem documentadas e são usados fóruns ou grupos de discussão para troca de ideias ou suporte de utilizadores. Em consonância com os estudos anteriores, não foi encontrada uma ferramenta que fosse claramente melhor que outras, uma vez que a escolha depende da área que se analisa e também do analista que vai trabalhar no projeto logo, fatores como a linguagem de desenvolvimento ou a simplicidade de *interface* de utilizador podem ser mais importantes para uns projetos que outros.
- Em 2010, a (Informatics Research and Development Unit of Public Health Informatics & Technology Program Office, 2010) publicou um relatório, no qual faz uma avaliação geral de 5 ferramentas de *data mining open source*. As ferramentas em estudo são o RapidMiner (antigo

YALE), WEKA, Orange, RATTLE e KNIME, e são analisadas relativamente a três critérios: informação geral, *features* de sistema e funcionalidades de *data mining*. A comparação das *features* de sistema foi feita com a atribuição de um valor de "0-5", e as funcionalidades de *data mining* apenas com um "Sim" ou "Não". Em conclusão, para encontrar a melhor ferramenta de *data mining* é necessário perceber quais os objetivos do projeto, e qual o *background* dos profissionais que vão operar nele, relativamente aos níveis de conhecimento de programação e tarefas de *data mining*.

No que diz respeito á literatura evidenciam-se alguns autores com experiência nas áreas de *data mining*, que publicaram algumas obras de referência. Em 2011, foi lançada a 3ª edição da obra "*Data mining*: *practical machine learning tools and techniques*" pelos autores (Witten, Frank, & Hall, 2011). Esta obra foi escrita inicialmente para acompanhar o desenvolvimento da ferramenta *open source* WEKA e, atualmente, encontra-se na 3ª versão, a par da atualização da WEKA para a sua versão 3.6.11. Trata-se de uma obra de referência para a área de *data mining*, uma vez que os autores têm muitos anos de experiência em investigação e foram percursores deste tipo de estudo. Mark Hall foi um dos fundadores da ferramenta *open source* WEKA, Ian Witten e o Eibe Frank pertencem também á Universidade de Waikato, como professores e investigadores tendo ambos também um vasto currículo na área.

O estudo que fazemos neste relatório incide essencialmente nas ferramentas *open source* de *data mining*. O motivo central que está na escolha deste tipo de ferramentas é o facto de não implicarem custos, relativamente às ferramentas comerciais que apresentam elevados custos de aquisição, suporte e manutenção, aos quais ainda acresce o custo das licenças. Atualmente estamos perante um mercado fortemente liderado por PME's, e estas não apresentam capacidades financeiras para investir em ferramentas de *software* de *data mining*, sem conhecer o retorno do investimento *a priori*. Nesse sentido, a solução é enveredar pelos sistemas *open source*.

O trabalho existente no que diz respeito ao estado da arte das ferramentas de *data mining* incide essencialmente na comparação, análises de critérios e estudos práticos às ferramentas disponíveis e propostas de metodologias para a escolha das mesmas. No entanto, trata-se de um campo onde todos os anos existem alterações e inovações, onde muitas ferramentas ficam obsoletas e outras se superam.

3 KNOWLEDGE DISCOVERY IN DATABASES

3.1 Enquadramento

A Descoberta de Conhecimento em Bases de Dados (DCBD), tradução da expressão inglesa *Knowledge Discovery in Databases* (KDD), surge em 1989 no primeiro *workshop* da área para enfatizar que o conhecimento é o produto final da análise de dados (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). Mais tarde, foi popularizada na inteligência artificial e nos campos do *machine learning*, como procura de conhecimento em bases de dados, sob a forma de padrões processando-se de forma automática.

Na década de 70, definiu-se que o conhecimento podia ser representado através de regras lógicas do tipo "se y, então z". Para lidar com estas regras, bastava criar um sistema capaz de as armazenar e de as escolher de forma conveniente e útil. Apesar de automático, este tipo de sistema requeria sempre a intervenção de um analista que, pelo menos, criasse as regras a identificar pelo programa. Este processo tinha algumas limitações pois as intervenções dos analistas eram frequentemente baseadas na sua intuição, passando alguma subjetividade para as regras criadas, e para além disso, existia ainda, a dificuldade em verbalizar e exteriorizar o conhecimento. Como acontece com o cérebro humano, o ideal seria que os sistemas adquirissem o conhecimento através de aprendizagem. Esta aprendizagem, proveniente de experiências (definidas pelas alterações de sistema) e, em termos matemáticos, na perceção de conjuntos de dados. Assim surgiu a "aprendizagem automática". Esta passa então a possibilitar a extração de conhecimento, ou seja, a partir de um conjunto de dados específico, passa a ser retirada a informação previamente desconhecida e potencialmente útil, para as tomadas de decisão. A descoberta de conhecimento em bases de dados inclui uma série de etapas e técnicas constituintes do seu processo, onde a principal é o *data mining* (Santos & Azevedo, 2005).

Em várias áreas, os dados estão a ser agrupados e acumulados a um ritmo galopante. Desta forma, existe a necessidade da criação de sistemas e teorias computacionais que consigam fazer a gestão deste amontoado de dados, para que seja possível a extração de informação útil (conhecimento). Assim surgiu então a descoberta de conhecimento em base de dados, como um processo que opera no sentido de combater esta emergência. O KDD está preocupado com o desenvolvimento de métodos e técnicas para trabalhar com os dados. A sua tarefa mais importante é a de *data mining*, como já foi referido anteriormente, que se encarrega da descoberta e extração de padrões nos dados. O método tradicional de conversão de dados em conhecimento baseia-se na análise manual de dados e da sua respetiva interpretação. Como em qualquer abordagem tradicional de análise de dados, em qualquer ciência, o analista ou investigador começa por conhecer os dados que estão em estudo. Esta abordagem não só é lenta, mas também traiçoeira devido á subjetividade do analista, que opera o estudo e, hoje em dia, praticamente impossível devido aos grandes volumes de dados que necessitam de tratamento. A automatização da análise de dados é praticamente obrigatória nos dias

que correm (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). No mundo dos negócios, existem várias áreas conhecidas por usarem o processo de KDD, como são exemplo o *marketing*, as finanças, a deteção de fraudes, as telecomunicações, a manufatura, entre muitas outras. No *marketing*, por exemplo, as principais necessidades são a previsão do comportamento dos clientes, que se processa através da segmentação dos mesmos por grupos de interesse. Um dos exemplos de análise aplicada neste sentido é o *market-basket*, que decifra comportamentos como: "se o cliente compra x, também é provável que compre y e z" (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). Assim é possível encontrar padrões nas compras dos clientes, permitindo às empresas fazer promoções ou outro tipo de ofertas de *marketing*, que lhes sejam convenientes, baseadas nos resultados da análise feita. Esta análise produz informação útil a partir de grandes conjuntos de dados, que até ao momento era desconhecida.

Muitas vezes confundida com a expressão *data mining*, KDD refere-se a todo o processo de descoberta de conhecimento útil, a partir de um conjunto de dados. *Data mining* refere-se a um passo dentro de todo esse processo. A distinção entre o processo de KDD e *data mining* é importante. Todos os outros passos constituintes do processo de KDD são também essenciais para assegurar a aquisição de conhecimento. O KDD evoluiu e continua a evoluir a partir da interseção de campos de investigação como *machine learning*, reconhecimento de padrões, bases de dados, estatística, inteligência artificial, aquisição de conhecimento para sistemas *experts*, visualização de dados, e computação de alta *performance*. O objetivo comum é a extração de conhecimento de alto nível, a partir de dados de baixo nível, em grandes conjuntos de dados.

Segundo (Fayyad, Piatetsky-Shapiro, & Smyth, 1996), KDD é então o processo não trivial de identificação de padrões válidos, novos, potencialmente úteis e compreensíveis nos dados. Neste contexto, os dados representam um conjunto de factos, e um padrão representa uma expressão em algum tipo de linguagem que descreve um subconjunto de dados ou modelos aplicáveis a um outro conjunto. Dentro deste processo existem um conjunto de tarefas associadas, que trabalham de acordo com um ciclo de vida.

No ponto seguinte, vamos passar a explicar como se processa todo o ciclo de vida do KDD.

3.2 Ciclo de vida

O processo de KDD inicia-se, como em outras áreas, com a compreensão do domínio da aplicação e dos objetivos a atingir. É necessário compreender que tipo de conhecimento e necessidade fazem parte dos objetivos do utilizador e que tipo de dados existem para análise. Trata-se de um processo iterativo, pois é composto por várias fases e também interativo, uma vez que é orientado por tomadas de decisão por parte do analista/utilizador.

Após esta fase inicial de compreensão, passa-se a uma fase mais prática que se inicia com o agrupamento organizado de um volume de dados. Dado um conjunto de dados, vamos criar um alvo e selecionar um conjunto onde se pretende incidir a análise.

Seguidamente é necessário efetuar uma limpeza ao conjunto escolhido. Aqui, é necessário organizar os dados que vão para análise, uma vez que estes apresentam incoerências e/ou ruído e possivelmente também será necessário decidir estratégias para colmatar possíveis falhas e faltas nos dados.

Depois de organizados os dados torna-se necessário uniformizá-los e transformá-los. Dependendo do objetivo da análise dos dados são aplicados métodos de redução de dimensionalidade ou de transformação, para reduzir algumas variáveis em análise.

Neste momento, estão prontos a ser analisados e podem ser aplicados os métodos de *data mining*, como classificação, previsão, associação, etc., de acordo com os objetivos do processo de KDD. Aqui aplicam-se os processos de análise, seleção de modelos, as hipóteses e a consequente escolha de algoritmos de *data mining* para a localização de padrões de interesse.

Depois de analisados, os resultados devem ser visualizados de várias formas e deviamente interpretados pelo analista/utilizador. Por fim, deve atuar-se sobre o conhecimento obtido, e criar relatórios ou documentar os resultados para que possam ser aplicados na prática.

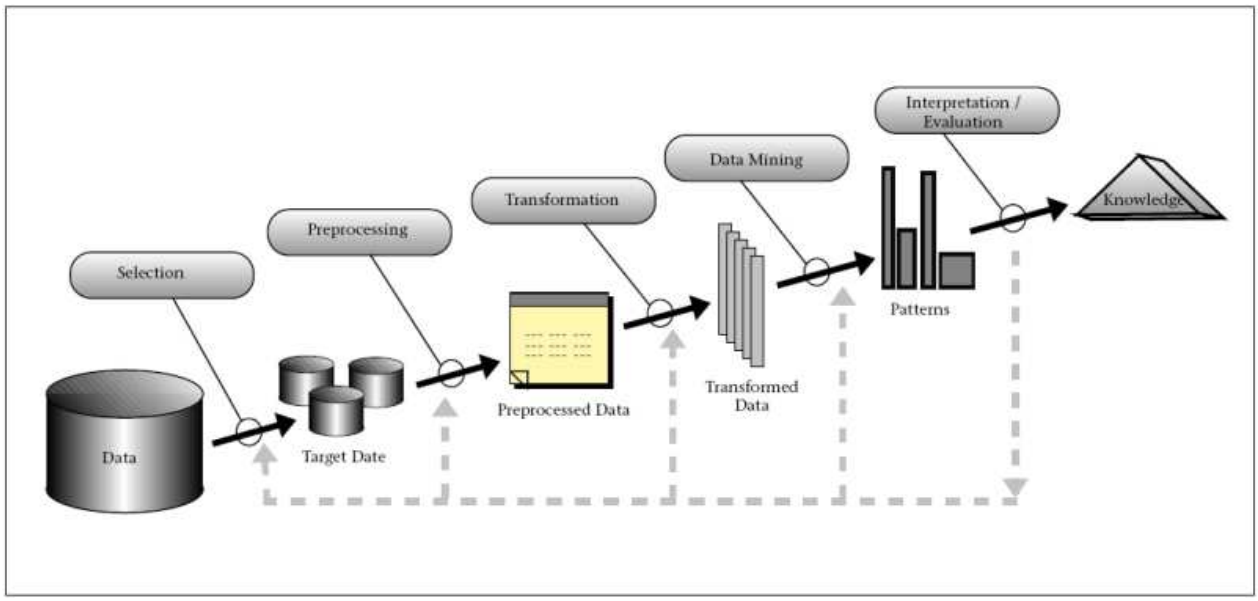

Figura 3.1 - Ciclo de vida do KDD ou DCBD (Fayyad, Piatetsky-Shapiro, & Smyth, 1996)

O processo de KDD que acabámos de descrever inicia-se com um conjunto de dados em formato cru e termina com um bloco de conhecimento potencialmente útil. Este processo, visível na Figura 3.1**Erro! A origem da referência não foi encontrada.** é composto pelas seguintes fases:

a) Seleção

Esta é a fase onde já se conhecem os objetivos do estudo e são escolhidos os dados a submeter a análise. O conjunto de dados a trabalhar é selecionado e recolhido para começar a ser moldado.

b) Pré-processamento

A fase de pré-processamento de dados prende-se com o tratamento preliminar dos mesmos. Os dados encontram-se em estado cru, provenientes de várias fontes de dados. Deste modo podem apresentar características diferentes, que tem de ser tratadas antes de se proceder á análise. Problemas como ruído, incoerências e redundâncias são comuns e necessitam de correção e limpeza. Alguns exemplos de casos que necessitam de ser pré-processados são dados mal introduzidos ou mal representados que revelam falta de normalização, onde por exemplo, definimos que o atributo "sexo = f/m" logo, não é legível que "sexo = f/masculino"; omissão de dados, onde existem *records* com faltas de preenchimento; dados com ruído, que apresentam valores muito diferentes dos esperados, etc.

c) Transformação

A fase de transformação é a fase onde se transformam os dados, para que estes possam ser analisados. Utilizam-se por exemplo as *data warehouses,* para agregação de grandes volumes de dados, de forma a reduzir algumas variáveis em análise. Estas ajudam especialmente na limpeza e acesso aos dados, uma vez que são responsáveis pela recolha e limpeza de dados transacionais. A finalidade é torná-los disponíveis para análise *online* e para facultar suporte a decisões. Para além das *data warehouses*, também se aplicam outros métodos como: seleção de atributos - para remoção daqueles que não são relevantes ao estudo; generalização de atributos - responsáveis por substituir um valor particular por um mais geral, dentro do mesmo atributo e ainda, métodos de discretização, que operam na conversão de atributos numéricos em atributos nominais.

d) Data mining

Segundo (Fayyad, Piatetsky-Shapiro, & Smyth, 1996) *data mining* é a aplicação de algoritmos específicos, que sobre algumas circunstâncias computacionais produzem uma enumeração de padrões sobre os dados. Esta fase é responsável pela aplicação de algoritmos, para descobrir padrões e tendências no conjunto de dados em análise. Para esse efeito são aplicados métodos de *data mining* provenientes das técnicas de *machine learning*, estatística, reconhecimento de padrões como classificação, *clustering*, regressão e associação de regras.

e) Interpretação e avaliação

A última fase é a de interpretação dos resultados obtidos na análise. Aqui é necessário fazer uma verificação e avaliar se estes têm interesse, relativamente aos objetivos definidos inicialmente. Os resultados são apresentados sobre diversas formas e devem ser explícitos para a compreensão do utilizador. A qualidade da verificação depende da maneira como os dados são visualizados. "Pode considerar-se uma fase que inclui um pós-processamento pois são determinados os valores de confiança e suporte das regras aplicadas sobre os dados." Pode haver a necessidade de não seguir

estas tarefas de forma linear. Muitas vezes é necessário voltar atrás e corrigir algo ou refazer um teste (Santos & Azevedo, 2005).

Existem algumas variações da designação das fases constituintes deste processo, no entanto, todas elas significam a execução das mesmas tarefas. Por exemplo, segundo (Santos & Azevedo, 2005) pode-se resumir o processo de KDD em três fases essenciais: pré-processamento, *data mining* e pós-processamento. Neste caso, a fase de pré-processamento inclui todas tarefas associadas aos processos de seleção, e transformação dos dados e a fase de pós-processamento representa as mesmas questões relativas aos processos de interpretação e avaliação final dos dados.

3.3 Conceito de Data Mining

Nas últimas décadas tem-se verificado um grande aumento dos dados armazenados em base de dados pelas organizações. Esta questão deve-se à modernização dos sistemas de informação que permitem que se guarde informação em bases de dados seguras, e por muito tempo, e também devido à diminuição constante do custo dos mesmos. Com esta evolução surgiram outros problemas que precisavam de resolução. A informação passou a ter um valor crucial, responsável por otimizar as tomadas de decisão. No entanto, esta informação, potencialmente útil, encontrava-se crua, sobre a forma de dados fechados ou escondidos nas bases de dados.

As análises de dados inicialmente faziam-se com recorrência à estatística e folhas de cálculo, o que com a evolução dos meios e do volume de informação criada se tornou impraticável. Estes métodos tradicionais são agora incapazes de, sozinhos, detetarem padrões, pois só utilizam métodos estatísticos. Para fazer face às dificuldades enunciadas pelos métodos tradicionais, surgiu a área de *data mining*, para extrair conhecimento útil, padrões e tendências de mercado de forma semiautomática (Santos & Azevedo, 2005)

Segundo (Padhy, Mishra, & Panigrahi, 2012) *data mining* deriva o seu nome das semelhanças existentes entre a procura de informação de negócio em grandes bases de dados e a mineração de uma montanha por um objeto valorizado. Ambos processos requerem a verificação de um grande volume de material ou a análise de tudo para descobrir onde reside exatamente o valor. Dadas bases de dados com tamanho e qualidade suficientes, a tecnologia de *data mining* pode gerar novas oportunidades de negócio.

As definições de *data mining* são várias sendo que, nenhuma está exatamente mais correta que outra, mas de alguma forma todas convergem para o mesmo sentido. Algumas definições que consideramos importantes:

- *Data mining*, trata-se do processo de recolha e extração automática de padrões de dados e transformação dos mesmos em conhecimento novo e útil (com interesse, implícito e previamente desconhecido) a partir de largas quantidades de dados. (Chen, Williams, & Xu, 2007);

- *Data mining* trata-se do processo de extração de padrões a partir de dados (Auza, 2010).
- *Data mining* trata-se da extração de informação implícita, previamente desconhecida e potencialmente útil, a partir de conjuntos de dados (Witten, Frank, & Hall, 2011);
- *Data mining* define-se como a aplicação de algoritmos de *machine learning* para a extração de informação automática ou semi-automática dos dados armazenados nas bases de dados (Konjevoda & Stambuk, 2011).
- *Data mining* é a extração de informação preditiva escondida em grandes bases de dados, é uma forte tecnologia com grande potencial para ajudar organizações a focar-se na informação mais importante (Padhy, Mishra, & Panigrahi, 2012).
- *Data mining* trata-se da extração não trivial de informação implícita previamente desconhecida e potencialmente útil a partir de dados digitais (Jailia & Tyagi, 2013).

Condensando todas as opiniões dos autores mencionados, consideramos então que *data mining* se trata do processo de extração de informação útil, que até então era desconhecida, a partir de grandes conjuntos de dados, que se processa de forma automática ou semi-automática através da aplicação de técnicas e métodos específicos de análise de dados.

As técnicas de *data mining* tornam-se indispensáveis a várias áreas, devido à grande necessidade de analisar o conhecimento nos dados, p.ex. nos supermercados é importante conhecer as tendências dos clientes e onde é que existem padrões de consumo. O objetivo do negócio não pode ser estático, e com as técnicas de *data mining* aplicadas sobre as transações obtidas diariamente, é possível para os gestores impulsionarem o seu negócio. Juntando as técnicas de *data mining* oriundas da inteligência artificial com as técnicas de aprendizagem automáticas e os conceitos estatísticos é possível: “estudar, investigar e desenvolver processos que permitam, a partir de grandes volumes de dados, extrair conhecimento, implicitamente incluído, que se revele inovador, útil e válido, e representá-lo de forma acessível e legível para o utilizador” (Santos & Azevedo, 2005).

Data mining é definido como o processo de descoberta de padrões nos dados. Os padrões descobertos devem ter significado de forma a representar alguma vantagem, principalmente ao nível económico, como por exemplo, servir para fazer previsões de comportamentos futuros. Os dados estão presentes de forma invariável em quantidades substanciais e os padrões permitem-nos fazer predições não triviais a partir dos novos dados. *Data mining* é um tópico que envolve a aprendizagem de uma forma prática e não teórica. O interesse reside nas técnicas para descobrir e descrever padrões estruturais nos dados, como uma ferramenta para ajudar a explicar esses dados e fazer predições a partir deles. Os dados detêm a forma de um conjunto de exemplos, como os clientes que escolhem certos tipos de produtos. O *output* toma a forma de predição sobre novos exemplos, este pode ainda conter descrições de uma estrutura que pode ser usada para classificar exemplos desconhecidos (Witten, Frank, & Hall, 2011).

Quando o termo de *data mining* surgiu, os outros campos como a estatística, *machine learning*, visualização de dados e engenharia do conhecimento já estavam desenvolvidos. No entanto, a maioria dos sistemas funcionavam em linhas de comando. A evolução do *software* e dos paradigmas dos *interfaces* tornou possível a criação de um sistema de *data mining* que oferecesse simplicidade, integração de visualização das ferramentas para exploração e ainda flexibilidade de descobrir novas formas para analisar os dados e adaptar algoritmos (Zupan & Demsar, 2008).

Segundo o mesmo autor, um modelo moderno de *data mining* deve proporcionar as seguintes características: facilidade de uso de *interfaces*; visualização de modelos e dados; ferramentas de análise de dados; exploração interativa; simplicidade; flexibilidade; extensibilidade e compreensão.

3.3.1 Vantagens

As vantagens da implementação das ferramentas de *data mining* são as seguintes (Santos & Azevedo, 2005):

- O âmbito das consultas feitas em *data mining* é muito mais alargado relativamente às consultas tradicionais feitas nas bases de dados (*query reports, SQL*), que apenas lidam com questões diretas do tipo “qual foi o meu rendimento total nos últimos 5 anos?”;
- O uso de algoritmos específicos para deteção de padrões e tendências nos dados, inferindo regras para os mesmos. Esta análise permite ir além do conhecimento que se tem sobre o negócio;
- O sistema responsabiliza-se pela geração de hipóteses. Existem duas abordagens possíveis de análise dos dados (Santos & Azevedo, 2005):
    - *Topdown* – serve como modelo de verificação, quando se sabe o que se vai pesquisar e apenas se querem confirmar hipóteses. O analista questiona o sistema e verifica se a hipótese é correta ou não.
    - *Bottom-up* – serve como modelo de descoberta, quando não sabemos o que contêm os dados.
- Previsão de tendências futuras através da deteção de padrões nos dados;
- Ajuda no processo de tomada de decisão, onde as decisões podem ser tomadas de forma proactiva de forma a evitar contextos menos desejados;
- Análise de dados mais rápida e rentável em relação às técnicas tradicionais, que eram lentas e dispendiosas;
- Variedade de algoritmos e técnicas disponíveis para análise.

3.3.2 Desvantagens

Para além das inúmeras vantagens do uso das ferramentas de *data mining*, existem também algumas desvantagens inerentes. De seguida, enumeramos as principais:

- Variedade de algoritmos para usar, torna-se uma vantagem mas também uma desvantagem, uma vez que a oferta é grande, é necessário perceber previamente quais as necessidades inerentes ao estudo. No caso de se tratar de um utilizador que não domine a área, será um problema a resolver;
- Possibilidade de inferir falsos resultados. Nos anos 60, surgiu a questão de estarem a ser encontrados falsos padrões nos dados, onde estatisticamente pareciam relevantes mas que de facto não eram. É necessário aplicar as técnicas da melhor forma, para que não hajam situações de falsos resultados;
- Dificuldade em escolher uma ferramenta de *data mining* específica para as necessidades do projeto em causa;
- Custos elevados das ferramentas proprietárias e possível falta de suporte das ferramentas *open source* disponíveis (inerente á dificuldade na escolha de uma ferramenta);
- Complexidade dos sistemas. Para utilização de alguns sistemas específicos de *data mining* é requerido que a pessoa tenha algum conhecimento de base em áreas como programação, estatística, *data mining*, etc.

3.3.3 Ciclo de vida

Para uma melhor compreensão de todos os processos incluídos na fase de *data mining* vamos analisar o seu ciclo de vida.

A Figura 3.2 representa o ciclo de vida de um processo de *data mining*, conhecido também como o modelo CRISP-DM (*Cross Industry Standard Process for Data Mining*). O modelo CRISP-DM inclui seis fases. Passamos a descrevê-las (Padhy, Mishra, & Panigrahi, 2012):

1. *Business Understanding* – Compreensão do negócio

Esta fase foca-se na compreensão de todo o projeto e dos seus objetivos e requisitos, da perspetiva do negócio. Esse conhecimento é convertido numa definição de problema de *data mining* e é criado um plano preliminar, desenhado para alcançar esses objetivos.

2. *Data Understanding* – Compreensão dos dados

Esta fase começa com o acesso a uma coleção de dados, para nos começarmos a familiarizar. O objetivo é identificar inicialmente os problemas na qualidade dos dados, verificar as primeiras particularidades sobre o conjunto e detetar *subsets*, possivelmente interessantes, para formar hipóteses sobre a informação escondida.

3. *Data Preparation* - Preparação dos dados

Nesta fase, reúnem-se todos os conjuntos de dados, para aplicar tarefas de pré-processamento, tendo em conta a forma como estes foram encontrados inicialmente (sem qualquer tratamento). Aqui são aplicadas técnicas de limpeza e transformação dos dados.

4. *Modeling* – Modelação dos dados

Aqui são selecionadas e aplicadas várias técnicas de modelação. As técnicas de modelação são responsáveis por extrair os resultados que pretendemos dos dados, e consistem na implementação de algoritmos de *data mining* e outras operações necessárias (classificação, predição, *clustering*, associação de regras, etc.).

5. *Evaluation* – Avaliação dos dados

De seguida o modelo é avaliado e revisto. Para serem corretos, os passos executados para a construção de um modelo, devem ir ao encontro dos objetivos do negócio. No fim desta fase, deve ser alcançada uma decisão relativamente ao uso dos resultados atingidos com a aplicação das técnicas de *data mining* e, se necessário, comparar e selecionar novos modelos de *data mining*.

6. *Deployment* - Entrega

O objetivo do modelo é melhorar o conhecimento sobre os dados. O conhecimento adquirido irá ser organizado e apresentado de forma a ser utilizado pelo cliente. Esta fase é tão simples como gerar um relatório ou tão complexa como implementar um processo de *data mining* que seja executável pela empresa.

Figura 3.2 - Ciclo de vida de data mining

Estas fases são as principais e não se considera que devam seguir um fluxo linear, pelo contrário, seguir em frente ou voltar atrás faz parte deste ciclo de vida, dependendo dos resultados atingidos em cada fase.

A Figura 3.3 representa um ciclo de vida de um processo de *data mining* alternativo, conhecido como o modelo SEMMA (*Sample*, *Explore*, *Modify*, *Model*, *Assess*).

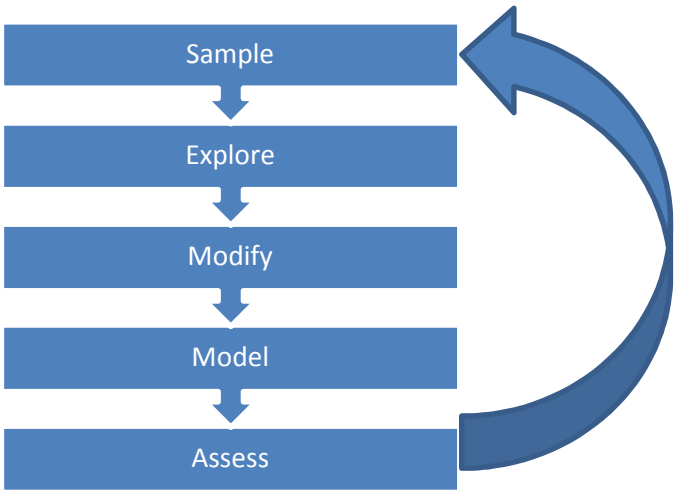

Figura 3.3 - Modelo SEMMA

Como é visível na figura anterior, o ciclo de vida de um processo de *data mining* com o modelo SEMMA inclui cinco fases. Passamos a descrevê-las (SAS Institute Inc., 1998):

1. *Sample* – Amostragem

A fase de amostragem pressupõe que os dados fiquem prontos para análise. Assim as amostras devem ser extensivas o suficiente para conter a informação necessária, mas pequenas para facilitar o processamento dos dados.

2. *Explore* – Exploração

Na fase de exploração devem procurar-se possiveis tendências, relações e anomalias nos dados de forma a permitir a sua compreensão.

3. *Modify* – Modificação

A fase da modificação surge na sequência dos resultados da fase anterior e serve para criar, selecionar e transformar os atributos da amostra de forma a prepará-los para a fase da modelação.

4. *Model* – Modelação

Na fase da modelação serão aplicados os algoritmos de *data mining* onde os dados vão ser modelados de forma automática, para encontrar resultados ou combinações úteis.

5. *Assess* – Avaliação

Por fim, os dados são avaliados de forma a extrair os modelos com melhor *performance* e que realmente refletem os objetivos da análise.

3.3.4 Tipos de algoritmos de data mining

Os objetivos centrais da aplicação dos algoritmos de *data mining* são a previsão ou descrição de dados (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). Estes objetivos são alcançados com a aplicação dos principais métodos que vamos passar a descrever nos pontos seguintes.

3.3.4.1 Classificação

O objetivo da classificação é encontrar uma função capaz de associar um conjunto de valores de atributos a uma classe específica. Dado um conjunto de atributos com uma classe atribuída, vamos designar uma função que associe, de forma mais precisa possível, uma classe a atributos de valor desconhecidos. Este novo exemplo, definido por um conjunto de atributos vai ser classificado de forma autónoma, de acordo com o modelo de classificação criado. Na classificação são usadas técnicas que utilizam conjuntos de treino, que se encontram pré-classificados, de modo a criarem-se modelos que serão utilizados mais tarde quando lhes forem aplicados a dados por classificar.
Exemplo: Estabelecemos que um mamífero é definido por características como: beber leite e ser vertebrado. Então todos os exemplos que cumpram estas características serão classificados como mamíferos (Santos & Azevedo, 2005).

3.3.4.2 Previsão

O objetivo da previsão é definir valores futuros, que ainda não são conhecidos. Este método tem como base exemplos e tenta descobrir padrões e tendências nos dados.
Exemplo: Com base nas minhas habilitações académicas e os padrões da indústria e a economia do país pode prever-se o salário a receber no ano de interesse (Santos & Azevedo, 2005).

3.3.4.3 Regressão

O objetivo da regressão é encontrar uma função para a previsão do comportamento de uma variável o mais aproximado possível (Santos & Azevedo, 2005). Ou seja, trata-se da previsão do valor de uma variável contínua, tendo em conta os valores de outras variáveis e assumindo uma dependência linear ou não linear.
Exemplo: Prever a velocidade do vento como uma função de temperatura, humidade, pressão atmosférica, etc.

3.3.4.4 Clustering

O *clustering* representa a identificação de segmentos, que contêm dados da mesma espécie. Os *clusters* são assim grupos homogéneos que pertencem à mesma classe, logo os dados que estes contêm devem apresentar algum tipo de semelhança, relativamente àqueles que pertencem a outras classes (Santos & Azevedo, 2005). Um conjunto de objetos são definidos por um conjunto de atributos que, posteriormente, formam grupos de acordo com as suas características. Os objetos

incluídos no mesmo *cluster* são mais semelhantes e os objetos incluídos em clusters diferentes são menos semelhantes.
Exemplo: Encontrar grupos de documentos semelhantes baseados na frequência de ocorrência de alguns termos linguísticos especificados.

3.3.4.5 Associação

O método de associação identifica dependências entre variáveis e cria um modelo de associações. As associações surgem quando várias ocorrências estão ligadas num único evento ou seja, podem ser identificadas através das tendências e padrões. Dado um conjunto de transações compostas por vários itens de uma coleção é possível criar regras capazes de prever a ocorrência de um item numa transação futura, em função da presença de outros itens particulares nessa transação (Santos & Azevedo, 2005).
Exemplo: Identificado um conjunto de padrões frequentes nas transações dos clientes de um supermercado é possível prever que tipo de itens se compram em conjunto. Ou seja, se o cliente leva fraldas e leite, então é provável que também leve cerveja.

3.3.4.6 Visualização

O método de visualização é o responsável pela visualização gráfica dos resultados de uma análise, sejam estes intermédios ou finais. Através de diagramas e gráficos são representados os dados, de forma explícita, que entre si têm alguma complexidade (Santos & Azevedo, 2005).
Exemplo: Depende muito do tipo de ferramenta que se usa, e dos meios que esta oferece para visualização de dados mas são comuns: histogramas, gráficos de quantis, gráficos de dispersão, gráficos de curvas, gráficos 3D, mapas sensoriais, gráficos de superfície, entre muitas outras possibilidades.

3.3.4.7 Deteção de desvios (outliers)

A deteção de desvios identifica alterações significativas nos dados a partir de previsões anteriores ou valores normativos. Dentro de um conjunto de dados, a deteção de desvios localiza possíveis anormalidades no comportamento dos dados.
Exemplo: A deteção de fraudes é um dos casos de aplicação deste método, pois dentro de uma tendência pré-estabelecida é possível identificar a variável "que não seguiu o mesmo caminho", e que pode ser um potencial *outlier* (Santos & Azevedo, 2005).

3.3.5 Conclusão

Os métodos de *data mining* usam vários tipos de algoritmos e dados. Os algoritmos podem derivar das técnicas das Árvores de Decisão, *Nearest Neighbours*, Redes neuronais, Redes Bayesianas, Associação de regras, *Support Vector Machines*, etc.

Em 2009, (Wu & Kumar, 2009) editaram um livro onde abordam os 10 algoritmos mais populares nas comunidades de *data mining*. São eles:

1. C4.5
2. K-Means
3. SVM – *Support Vector Machines*
4. Apriori
5. EM – *Expectation-maximization*
6. PageRank
7. AdaBoost
8. KNN – *K-Nearest Neighbors*
9. Naïve Bayes
10. CART – *Classification and Regression Trees*.

A escolha dos algoritmos está sempre dependente de dois fatores muito importantes: tipos de *datasets* e tipo de requisitos de utilizador (Padhy, Mishra, & Panigrahi, 2012).

No capítulo seguinte vamos abordar o *software open source*, onde são referidas as características inerentes a este tipo de sistema, as principais vantagens e desvantagens e ainda algumas informações úteis para encontrar ferramentas que incluem este tipo de sistema.

4 SOFTWARE OPEN SOURCE

Neste capítulo abordamos o conceito de *software open source* e as suas origens, e são ainda referidas algumas das principais vantagens e desvantagens do uso deste tipo de modelo relativamente a outros.

O *open source (OS)* surgiu em 1998 pela *Open Source Iniciative* (OSI), e tornou-se um desafio sério para o *software* proprietário. A designação surge para quebrar a ambiguidade com a expressão *free software*, que significava *software* livre e/ou *software* gratuito. Constituindo-se por políticas de desenvolvimento específicas estipuladas pela OSI e reconhecidas pela *Free Software Foundation* (FSF), distingue-se assim, do *free software*.

O termo *free software* foi o primeiro a surgir, em 1983. Este respeita as liberdades essenciais dos utilizadores como a liberdade de processar o *software*, de o estudar, de o modificar e de redistribuir cópias com ou sem alterações. Trata-se de uma questão de liberdade e não de preços, "*free speech, not free beer!* (Stallman, 2009)*." Open source* trata-se de uma metodologia de desenvolvimento e *free software* um movimento social, que se prende com a liberdade de partilha e cooperação, promovendo a solidariedade social (Stallman, 2009). Parecem muito semelhantes, mas no fundo são duas comunidades de desenvolvimento que surgem de pontos de vista diferentes.

A expressão *open source* refere-se a acesso livre, distribuição livre e modificação de código fonte livre (Zhu & Zhou, 2011). Trata-se de *software* desenvolvido sobre uma licença que permite a inspeção, uso, modificação e redistribuição do código do programa de *software*. Pelo ponto de vista económico, o *open source* pode ser encarado como um processo de inovação: um processo novo e revolucionário de produção de *software* baseado no acesso sem restrições do código fonte relativamente á abordagem tradicional fechada do mundo do *software* proprietário.

Segundo a OSI, uma aplicação *open source* deverá garantir as seguintes características: (Open Source Initiative, s.d.)

- Distribuição livre

A licença não deve restringir qualquer membro pela venda ou pela distribuição gratuita da aplicação, como ainda não deve restringir a distribuição do *software*, como uma componente de outra aplicação. Não é possível a implementação de taxas para esse propósito.

- Código fonte

A aplicação deve incluir o código fonte, e deve permitir a sua distribuição na sua forma compilada. Quando isto não for possível é necessária a justificação de onde este pode ser obtido, de forma simples, clara e gratuita. O código deve ser o meio pelo qual o programador começa a criar alterações no programa, por isso deve ser explícito e ser apresentado no seu estado original e não convertido em *output.*

- Trabalhos derivados

A licença deve permitir modificações e a criação de trabalhos derivados, e deve permitir que estes sejam distribuídos sobre os mesmos termos da licença para o *software* original.

- Integridade do autor do código fonte

A licença pode restringir o código fonte de ser distribuído na sua forma modificada, no caso desta estabelecer que a distribuição é feita por *patch files*, de forma a evitar modificações na aplicação no momento em que está a ser construída.

- Sem discriminações relativas a pessoas ou grupos

A licença não deve discriminar o uso da aplicação por qualquer pessoa ou grupos de pessoas.

- Sem discriminações relativas a grupos de trabalho específicos

A licença não deve restringir ninguém de fazer uso da aplicação num contexto de trabalho específico.

- Distribuição da licença

Os direitos da aplicação devem ser aplicados a todos que usufruírem da redistribuição da mesma. Ou seja, a licença suporta a totalidade do produto sem a necessidade de adicionar novas licenças depois da sua distribuição.

- A licença não deve ser específica de um produto

Os direitos da aplicação não devem depender do facto do programa fazer parte de uma distribuição específica de *software*. Ou seja, se o programa for redistribuído, todas as partes devem possuir os mesmos direitos aplicados á distribuição do *software* original.

- A licença não deve restringir outros *software*s

A licença não deve criar restrições a outros *software*s que sejam distribuídos de forma licenciada. Por exemplo, as licenças não devem obrigar que outros *software*s usados sejam também eles *open source*.

- A licença deve ser tecnologicamente neutra

A licença não deve criar modelos padrão ou ser aplicada a programas com estilos específicos. Não deve haver tendências.

Um projeto *open source* é definido como qualquer grupo de pessoas a desenvolver *software* e apresentando os seus resultados ao público tudo sob uma licença *open source* (Evers, 2000). O grupo de pessoas que está a trabalhar no projeto e a licença sob a qual o código fonte é realizado são as chaves para esta definição. Um projeto bem-sucedido tipicamente começa por um indivíduo que encontra um problema, que precisa de um *software* específico, para ser resolvido. Procurando soluções na rede social encontram-se pessoas que partilham o mesmo problema. Assim se forma

um grupo embriónico que começa a trabalhar na resolução do problema que todos partilham (Bonaccorsi & Rossi, 2003).

O primeiro objetivo do *open source* é criar um sistema que seja útil e interessante para aqueles que estão a trabalhar nele e não para preencher uma necessidade comercial. Os programadores são muitas vezes voluntários e não recebem pelo seu trabalho, contribuindo como de um *hobby* se tratasse, e em retorno recebem reconhecimento e a satisfação pessoal sobre os resultados atingidos. O autor do projeto inicialmente dá-lhe vida sozinho e tem o poder de escolher quem quer integrar na sua equipa e quais os contornos do produto que vai ser criado (Godfrey & Tu, 2001).

Depois da abordagem a todo o conceito de *open source* consideramos que existem uma série de vantagens e desvantagens perante o *software* proprietário. Os modelos *open source* podem não ser tão estáveis e completos como os comerciais, mas oferecem alternativas úteis e inovadoras, relativamente a novas *interfaces* e implementação de protótipos, com as técnicas mais recentes (Zupan & Demsar, 2008). Nas secções seguintes serão descritas as principais vantagens e desvantagens do uso deste tipo de metodologia.

4.1 Vantagens do software open source

Consideramos que as principais vantagens do uso de *software open source* são:

- Código fonte: o código é disponibilizado por forma a ser usado, estudado, melhorado ou alterado para outros fins.
- Suporte: existe uma comunidade que tem o objetivo de apoiar os projetos *open source*, no esclarecimento de dúvidas ou na contribuição de novas ideias. Cada projeto é geralmente apoiado por fóruns e *mailing lists*.
- Atualização: através da comunidade são também fornecidas novas ideias e contribuições para resolução de problemas, o que leva a uma atualização constante do *software open source* e das metodologias de desenvolvimento.
- Capacidade de adaptação: o *software open source* pode ser adaptado de forma a responder às necessidades de cada organização. A partir do código fonte podem construir-se várias hipóteses conforme as necessidades do utilizador.
- Heterogeneidade: os projetos *open source* reúnem colaboradores com um *know-how* muito diversificado, o que aumenta o valor daquilo que é produzido. O conjunto de perfis de pessoas que se juntam para programar enriquece os projetos. (Bonaccorsi & Rossi, 2003)
- Redundância: as tarefas pendentes são concretizadas por mais que um indivíduo, o que gera redundância de código, que por sua vez melhora a qualidade do produto permitindo a escolha entre um conjunto de soluções disponíveis (Bonaccorsi & Rossi, 2003).

- Reconhecimento profissional: os projetos OS são considerados bens públicos e a participação no desenvolvimento dos mesmos pode tornar-se um benefício para uma carreira profissional (Zhu & Zhou, 2011).
- Custo: não existem custos de entrada no mercado, ao contrário do *software* proprietário, o *software open source* não se comporta de forma estratégica e não usa grandes descontos para entrar no mercado (Zhu & Zhou, 2011).
- Licenças: existem licenças que evitam a monopolização da distribuição de *software* OS, sendo esta livre, o que assegura que ninguém pode vender o produto e mais tarde pedir dinheiro pelo seu *upgrade* (Zhu & Zhou, 2011).
- *Lock-in*: o *software open source* reduz a dependência criada pelo *software* proprietário relativamente ao suporte do sistema, através do apoio da comunidade. Não existe perigo de *Lock-in*, que se trata da situação em que o cliente está dependente do vendedor para novos produtos ou serviços com altos custos (Zhu & Zhou, 2011).
- Seletividade: existe a possibilidade de trabalhar com um *software open source* antes de adotá-lo, sem qualquer custo. Esta capacidade evita tempo e dinheiro mal gastos na aquisição de um *software* menos indicado.
- Acessibilidade: existem repositórios específicos onde é possível encontrar o *software open source*, ou seja, a sua localização é rápida e simples.
- Rapidez na correção de *bugs*, inovação de métodos, integração com outras ferramentas, devido às comunidades de desenvolvimento (Mikut & Reischl, 2011).
- Incorporação de técnicas experimentais, incluindo protótipos, que mostram os problemas emergentes mais cedo que as ferramentas comerciais.

Para as PME's as ferramentas *open source* representam uma grande aposta (pesquisas, educação, aplicações industriais, etc.). Este tipo de produtos ajudam os colaboradores na compreensão do sistema através da consulta do código e permitem ainda ajustar os algoritmos o outros parâmetros ao projeto em causa (Chen, Williams, & Xu, 2007).

4.2 Desvantagens do software open source

Consideramos que as principais desvantagens do uso de *software open source* são:

- Fatores sociais: os projetos OS dependem da auto-motivação dos colaboradores que trabalham de forma gratuita, e da coordenação de equipas em prol do mesmo projeto, que muitas vezes se encontram espalhadas geograficamente.

- Documentação: por vezes os projetos OS não são muito bem documentados. É uma questão que já não é tao frequente mas que ainda existe. É necessário neste caso recorrer aos fóruns para esclarecimento de potenciais questões de instalação e de suporte (Zupan & Demsar, 2008).
- Qualidade do *software*: assim como os projetos comerciais a qualidade é sempre uma característica que pode falhar. A questão nos projetos OS é o facto de estes serem desenvolvidos por uma comunidade, e isso suscitar sempre dúvidas relativamente á qualidade daquilo que é produzido. Ao longo dos anos, esta questão tem melhorado significativamente.
- Maturidade: o constante desenvolvimento e melhoramento dos projetos não é encarado com muita confiança por parte das organizações, pois mostra que ainda não se atingiu o nível de maturidade apropriado.
- Instabilidade: a possibilidade dos métodos usados pela comunidade ainda não estarem bem testados, por serem muito recentes ou versões *beta*, pode levar á criação de um sistema instável e pouco fiável para os seus utilizadores (Zupan & Demsar, 2008).
- Escalabilidade: o facto de os sistemas serem desenhados com propósitos muito específicos ou sobre condições próprias, leva a que estes nem sempre se encontrem preparados para crescer ou para suportar grandes quantidades de dados (Chen, Williams, & Xu, 2007).

Existem cada vez mais projetos *open source* bem sucedidos. Entre eles conhecemos alguns nomes pelo grau de popularidade que atingiram, por exemplo os sistemas *Linux*, *Mozilla Firefox* e *Apache*.

4.3 Onde encontrar software open source?

Para encontrar este tipo de aplicações e esclarecer várias dúvidas, existem repositórios específicos para o efeito. A convergência de todos os projetos *open source* na mesma localização dá aos utilizadores a possibilidade de aceder, de forma simples e rápida, ao *software*. Também às comunidades é possível alocar e ao mesmo tempo divulgar os produtos que são desenvolvidos e atualizados por elas.

Para além destes repositórios de caráter geral existem programas que criam o seu próprio repositório, onde disponibilizam todo o tipo de informação necessária á compreensão do *software*. O acesso aos repositórios permite aos utilizadores e comunidades entrar em sessões de discussão e esclarecimento de questões, por forma a resolver alguns pormenores menos explícitos na documentação. Para além disso, a maior parte dos repositórios facultam outro tipo de *features* de sistema como visualizações, notificações por *email*, API (*Application Programming Interface*), gestão de documentos, modelos 3D, etc.

Na Tabela 4.1 são enunciados alguns repositórios onde é possível obter o *software open source*, bem como a sua localização através da *web* (Wikipédia, 2014):

Tabela 4.1 - Repositórios de software open source

Repositório	Website	Notas
GitHub	https://github.com/	*Free for public, paid for private.*
Gitorious	https://gitorious.org/	*Free for open-source projects.*
Gna!	https://gna.org/	*Only for projects with a GPL compatible license*
GNU Savannah	http://savannah.gnu.org/	*Project by the Free Software Foundation.*
Google Code	https://code.google.com/	*Free. For open-source projects only*
JavaForge	http://www.javaforge.com/login.spr	*Free. For open-source projects only*
Ourproject.org	http://ourproject.org/	*For free software,free culture and free knowledge projects*
SourceForge	http://sourceforge.net/	*Free. For open-source projects only.*

A Tabela 4.1 dá apenas alguns exemplos de repositórios de *software open source*, mas existem muitos mais e com outro tipo de características. Os repositórios mais populares, segundo dados da Wikipédia, são os seguintes:

- GitHub com cerca de 6.700.000 utilizadores e 1.100.000 projetos associados;
- SourceForge com mais de 3.400.000 utilizadores e 324.000 projetos associados;
- GoogleCode com 250.000 projetos associados.

4.4 Licenças open source mais populares

Existem várias ferramentas de *data mining* que usam licenças diferentes, estas podem ser ainda comerciais ou *open source*. No ramo dos negócios, há uma grande procura para as ferramentas comerciais, devido a estabilidade dos sistemas, ao suporte prestado, manutenção e muitas outras questões. Relativamente a outros grupos de trabalho, as soluções *open source* são muito procuradas e, neste caso existem muitas licenças disponíveis.

Dentro do mundo das licenças *open source*, existem algumas que são mais populares que outras. As mais populares são as mais usadas nos projetos *open source* ou, são as que possuem uma comunidade mais forte e desenvolvida. Segundo a OSI, as licenças mais populares são as da Tabela 4.2:

Tabela 4.2 - Licenças open source mais populares (Open Source Initiative, s.d.)

Licenças	Autores
Apache License 2.0	Apache Software Foundation
BSD 3-Clause "New" or "Revised" license	Regents of the University of California
BSD 2-Clause "Simplified" or "FreeBSD" license	Regents of the University of California
GNU General Public License (GPL)	Free Software Foundation
GNU Library or "Lesser" General Public License (LGPL)	Free Software Foundation
MIT license	MIT
Mozilla Public License 2.0	Mozilla Foundation
Common Development and Distribution License	Sun Microsystems
Eclipse Public License	Eclipse Foundation

A licença *open source* mais popular de todas é a *GNU General Public License* da *Free Software Foundation*. De forma resumida, esta pressupõe as seguintes permissões/obrigações (Free Software Foundation, 2014):

- Usar o *software* para qualquer propósito;
- Modificar o *software* conforme as necessidades;
- Partilhar o *software*;
- Partilhar as alterações feitas.

De uma forma geral, as licenças OS são licenças que cumprem com a definição de *open source*, onde é permitido que o *software* desenvolvido seja usado, modificado e partilhado de forma livre e sem restrições. Para serem aprovadas pela OSI, estas devem seguir um processo de revisão específico.

Para além das licenças enunciadas existem ainda outro tipo de licenças que são aprovadas pela OSI, mas que são elaboradas para propósitos específicos, como investigação por exemplo, e que não estão disponíveis para fins comerciais. Para além destas, existem ainda outro tipo de licenças mistas, que são adotadas por exemplo, no caso do desenvolvimento de um *software open source* cujo

objetivo seja aumentar uma ferramenta comercial, como é o caso da conhecida ferramenta *Matlab* (Mikut & Reischl, 2011).

No capítulo seguinte iremos analisar um conjunto de ferramentas *open source* de *data mining*, onde serão abordadas as suas principais características e funcionalidades.

5 FERRAMENTAS OPEN SOURCE DE DATA MINING

Depois de abordados os conceitos de *data mining* e *open source*, vamos passar ao estudo das ferramentas que se enquadram nestes dois conceitos. De uma forma geral, o objetivo deste estudo é compreender e analisar as ferramentas *open source* de *data mining*, ao nível do estado da arte, perceber as suas funcionalidades principais e por fim perceber, entre elas, quais as melhores soluções. Posteriormente, aquelas as melhores irão ser comparadas através de um teste prático com *datasets* reais.

5.1 Suites data mining open source

Existem muitas ferramentas *open source* de *data mining* atualmente disponíveis no mercado. Para este estudo vamos abordar apenas as que constam do ranking do KDnuggets – *Software Suites for Data mining, Analytics, and Knowledge Discovery* – que se intitulam como *Free* e *Shareware*. O KDnuggets é uma página *web* muito conhecida, por ser responsável pela publicação de informações e novidades relacionadas com as áreas de análise de negócios (*busisess analytics*), *big data*, *data mining*, ciência dos dados (*data science*) e muitas outras questões relacionadas com a área. Gregory Piatesky-Shapiro, um especialista na área, criou esta página de referência, com a designação KDnuggets – *Knowledge discovery nuggets*, onde KD se desenvolve como *Knowledge Discovery*, e foi concebido com o objetivo de publicar informações curtas e concisas sobre a área – *nuggets*. Trata-se então de uma página mundialmente conhecida e com vários reconhecimentos, como é exemplo a votação para *Best Big Data Tweeter* em 2013, pela *Big Data Republic*. Além disso tem correntemente mais de 100,000 visitantes mensais e mais de 45,000 subscritores através das redes sociais.

Nesta secção, iremos descrever, de uma forma resumida, as principais ferramentas apontadas pelo *KDnuggets* como *software suites* para *data mining*, análise e descoberta de conhecimento. A designação *suite* pressupõe que as ferramentas sejam constituídas com um conjunto de funcionalidades próprias de uma área específica, neste caso *data mining*.

5.1.1 ADaM

A plataforma ADaM, ou seja, *Algorithm Development and Mining System* foi desenvolvida no centro de Tecnologias de Informação e Sistemas, na Universidade do Alabama, com o objetivo de aplicar tecnologias de *data mining* em dados remotamente detetados e em dados científicos.

O ADaM é constituído por um conjunto de ferramentas de *data mining* desenhadas especialmente para tratamento de dados científicos e de imagens. Para além das ferramentas, é composto por mais de 100 componentes que podem ser configuradas para criar processos personalizados de análise de dados.

Relativamente á arquitetura das componentes do ADaM, esta está desenhada para tirar vantagens em ambientes computacionais emergentes como a *web* e as grelhas de informação. As operações individuais podem executar-se de forma *stand-alone*, facilitando o seu uso em sistemas paralelos e distribuídos. As operações, organizadas como conjuntos de ferramentas, proporcionam:

- Reconhecimento de padrões;
- Processamento de imagens;
- Otimização;
- Capacidade de associação de regras de análise.

A versão 4.0 do ADaM apresenta uma arquitetura diferente das versões anteriores. A versão mais recente, 4.0.2., disponibiliza uma solução que permite a integração de algoritmos desenvolvidos por terceiros e a reutilização das componentes do ADaM por outros sistemas. As componentes são por isso autónomas numa arquitetura distribuída.

- Serviços distribuídos

As componentes do ADaM podem ser acedidas através de vários *interfaces* externos. Esta flexibilidade facilita a implementação de componentes DM e de processamento de imagem como a *web* e serviços em grelha. Os protocolos de execução consistentes e bem documentados suportam a incorporação das componentes do ADaM em aplicações que são desenvolvidas usando as linguagens comuns de *scripting* como PERL e *Python.* A incorporação de tecnologias de troca de dados, como *Earth Science Markup Language (*ESML) gera interoperabilidade distribuída entre os *datasets* heterogéneos científicos.

- Aplicações personalizadas

As aplicações personalizadas podem ser geradas a partir de conjuntos de ferramentas de componentes de análise e processamento de imagem, combinadas por exemplo com outros módulos de *software* especializados. Um exemplo é o uso do ADaM para detetar os ciclones tropicais e para estimar a duração dos seus ventos mais fortes. Esta aplicação operacional combina módulos de análises de imagens de propósito geral, com módulos especiais desenvolvidos especialmente para o problema.

- Serviços de grelha:

ADaM é a primeira aplicação *data mining* que executa na NASA *Information Power Grid*. Os conjuntos de ferramentas do ADaM podem ser disponibilizados como um conjunto de componentes (*Open Grid Services Arquitecture),* que facilmente se transferem para um ambiente em grelha.

O conjunto do ADaM pode ser usado livremente para fins educacionais e de pesquisas, por instituições sem fim lucrativos e agências governamentais dos Estados Unidos apenas. Outras organizações são permitidas mas apenas para fins de avaliação. Outros fins requerem aprovação. O *software* não pode ser vendido ou redistribuído sem aprovação *a priori*. Qualquer um pode fazer

cópias do *software* para seu uso, sabendo que essas cópias não podem ser vendidas ou distribuídas, e são usadas sobre os mesmos termos e condições.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://projects.itsc.uah.edu/datamining/adam/.

5.1.2 Alteryx

O Alteryx é uma ferramenta de *data mining* constituída por 3 plataformas: *Alteryx Designer, Alteryx Server* e o *Alteryx Analytics Gallery*. No entanto, apenas a plataforma *Alteryx Designer* é *free* e por período de 14 dias.

A plataforma *Alteryx Designer* destina-se a grupos de negócio como *marketing*, finanças, etc. pois proporciona um *workflow* intuitivo, rápido e simples para combinação de dados e análise avançada. Reunindo várias funcionalidades num *workflow* apenas, mas muito intuitivo e rápido, o *Alteryx Designer* melhora o desempenho das análises de dados sem recorrer á necessidade de programação:

- Combinação de dados
    - Acesso a várias fontes dados sem a necessidade de ferramentas especiais;
    - Preparação, limpeza e combinação de dados para análise ou visualização;
    - Criação de gráficos 3D a partir dos dados espaciais obtidos.
- Análise preditiva
    - Contém mais de 30 ferramentas de análise de dados baseada na linguagem de programação R, sem necessidade de codificar;
    - Escalabilidade baseada na linguagem R, através do *Revolution Analytics*
    - Codificação direta em R pode ser integrada no *workflow* do *Alteryx* e partilhada com os analistas.
- Análise espacial
    - Acesso e uso dos dados das localizações necessárias;
    - Ferramentas de análise espacial intuitivas mas avançadas (uso de técnicas como *drive time*, *spacial matching* etc.)
    - Visualização e mapeamento dos resultados através de ferramentas como *Tableau, ESRI* ou *MapInfo*;
    - Acesso a dados espaciais e de utilizadores pré-armazenados.

Relativamente ao *Alteryx Server*, este tem objetivo ser fazer análises poderosas de forma a dar apoio e responder ás necessidades dos utilizadores. Apresenta uma solução simples para desenvolver análises, que se prende com a partilha das aplicações de análise de dados com gestores responsáveis pelas tomadas de decisões. O *Alteryx Server* proporciona uma solução de análise de dados através

do processamento de dados baseado num servidor, facultando o acesso e interação á aplicação através da infraestrutura do servidor da organização.

Por fim, o *Alteryx Analytics Gallery* trata-se de uma plataforma de análises na *cloud*. Oferece uma análise muito poderosa baseada na experiência do consumidor, que permite á organização compreender o valor da *Big Data* de forma muito rápida. Para além desta capacidade, o *Alteryx Analyics Gallery* permite a qualquer pessoa, em qualquer lado, o acesso a aplicações de análise a qualquer momento.

Relativamente á sua licença, o *software* Alteryx é da exclusiva propriedade da empresa, onde se rege por princípios próprios.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.alteryx.com/products/alteryx-designer.

5.1.3 AlphaMiner

O Alphaminer é desenvolvido pelo *E-business Tecnhology Institute* (ETI), na Universidade de Hong Kong.

Trata-se de uma plataforma *open source* de *data mining* que proporciona o melhor rácio de custo-*performance* para aplicações de *data mining*. Disponibiliza as seguintes tecnologias:

- Construção de cenários do tipo *workflow*, que permitem aos gestores de negócios fazer operações *drag-and-drop* na construção de uma situação de *data mining*;
- Arquitetura constituída por componentes convertíveis em *plug-ins*, que proporcionam extensibilidade para adicionar novas funções de *business intelligence*, de importação e exportação de dados, transformação de dados, modelação de algoritmos, modelos de avaliação e de desenvolvimento;
- Funções de *data mining* versáteis, que oferecem uma análise poderosa, específica para as indústrias, incluindo análise de clientes, *profiling* e *clustering*, análise de associação de produtos, classificação e predição.

Este programa é distribuído sobre a licença *GNU General Public License* e certificado pela OSI.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.eti.hku.hk/alphaminer/.

5.1.4 CMSR

O CMSR *(Cramer Modeling Segmentation & Rules)* é uma ferramenta de *data mining* desenvolvida em Sydney, na Austrália pela *Rosella Software*, até então conhecida como *StarProbe.*

Proporciona um ambiente integrado para modelação preditiva, segmentação, visualização de dados, análise estatística de dados, e avaliação de modelos baseada em regras. Isto proporciona uma análise integrada e um ambiente com o mecanismo de associação de regras para utilizadores poderosos. As características principais do CMSR são:

- Mapas auto-organizáveis (*SOM – self organizing maps*) – *clustering* neuronal;
- Modelação preditiva com redes neuronais;
- Classificação de árvores de decisão e segmentação (CRAMER);
- *Hotspot drill-down* e análise de perfis;
- Regressão;
- Função *radial basis* (RBF) com funcionamento através de regras;
- Regras de negócio – mecanismos com sistemas *experts* de predição;
- Avaliação de modelos baseados em regras;
- Gráficos poderosos: 3D barras, barras, histogramas, barras de histogramas, dispersão, de caixas (…);
- Segmentação e análise de ganhos;
- Análise de respostas e lucros;
- Análise de correlação;
- Análise de cesto de compras (*Cross-sell)*;
- Estatísticas *drill-down*;
- Tabelas cruzadas com desvios e análise *hotspot*;
- Tabelas em grupos com desvios e análise *hotspot*;
- *SQL query*s e ferramentas do tipo *batch*;
- Estatísticas: Mono, Bi, ANOVA…;
- *Scoring* de bases de dados;
- Conexão a todos os maiores DBMS *(Data base management system*) através de *ODBC/JDBC (Open Database Conectivity/* Java *Database Conectiviy);*
- Tratamento rápido e de grandes quantidades de dados (acima de 2 biliões de registos).

O CMSR funciona com vários sistemas de bases de dados relacionais SQL. Desde a importação de dados à avaliação de modelos, o CMSR proporciona *interface* de utilizador flexível e fácil de usar. Esta ferramenta é desenhada para utilizadores que operam no negócio das aplicações, como a segmentação e avaliação de clientes.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.roselladb.com/starprobe.htm.

5.1.5 CRAN task view

O CRAN *Task view* ou *Comprehensive R Archive Network* é uma ferramenta de *machine learning* e aprendizagem estatística. Trata-se de uma *network* em ftp e *web servers* que se encontra pelo mundo e que armazena versões de código e documentação idênticas, atualizadas para a linguagem R. A última versão saiu em Julho de 2013 e pertence a Torsten Hothorn professor de bioestatística na Universidade de Zurique.

Contém várias *packages* com *add-ons* implementados que representam ideias e métodos desenvolvidos no limite entre a ciência da computação e as estatísticas, que representam o *machine learning*. Estes pacotes de *add-ons* estão estruturados nos seguintes tópicos:

- Redes neuronais;
- Particionamento recursivo;
- *Random forests*;
- Métodos regularizáveis e possíveis de "encolher";
- *Boosting;*
- *Support Vector Machines* e métodos *kernel*;
- Métodos Baysianos;
- Otimização com o uso de algoritmos genéticos;
- Associação de regras;
- Sistema *fuzzy* baseado em regras;
- Seleção e validação de modelos;
- Elementos de aprendizagem estatística;
- *GUI Rattle*, um *interface* gráfico para *data mining* em R.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://cran.r-project.org/web/views/MachineLearning.html.

5.1.6 Databionic ESOM

As ferramentas do Databionic ESOM ou *Emergent Self-Organizing Maps* representam um conjunto de programas cujo objetivo é a realização de tarefas de *data mining* como *clustering*, visualização e classificação.

Os princípios de auto-organização (*self-organizing*) podem ser transferidos para a análise de dados, através da auto-organização de dados multivariados em grupos homogéneos. Uma ferramenta que incorpora esses princípios são os SOM, que iterativamente ajustam estruturas de distância em espaços de altas dimensões para baixas dimensões preservando a topologia do espaço de *input* tanto quanto possível.

Ou seja, assumindo que um conjunto de treino é um conjunto de pontos de um espaço com altas dimensões ao qual chamamos espaço de dados, os SOM (mapas auto-organizáveis) são formas multidimensionais de representação desse espaço de dados. Os dados que se encontram em espaços com altas dimensões (vectores de *input)* são convertidos para espaços multidimensionais baixos (geralmente 1D ou 2D). Estes consistem em componentes chamadas *nodes* ou neurónios, a que está associado um *weight vector,* que representa a mesma dimensão dos vectores de *input* e, ainda uma posição defnida no espaço do mapa (topologia).

O processo de reduzir a dimensão dos vectores é essencialmente uma técnica de compressão de dados chamada *vector quantization,* onde o procedimento para associar um vector a um espaço de dados no mapa é encontrar o *node* com *weitght vector* mais próximo (distância métrica mais pequena) do vector do espaço de dados. Esta função é conduzida por dois algoritmos: o*nline training* e *batch training*. Ambos procuram o vector protótipo mais próximo para cada ponto (*bestmatch*). No algoritmo *online training* os *bestmatches* são constantemente atualizados, enquanto que, no algoritmo *batch training* estes são primeiro armazenados para todos os pontos, e só depois a atualização é feita de forma colectiva. Para além disso estes algoritmos criam uma *network* que armazena informação para que as relações topológicas (forma como os elementos estão fisicamente dispostos) sejam mantidas.

Quanto à topologia, a mais comum é a grelha a duas dimensões, onde cada protótipo (neurónio) tem 4 vizinhos diretos. Apesar de serem também considerados um tipo de redes neuronais, os SOM são treinados através de uma aprendizagem sem supervisão e usam a função de vizinhança para preservar as características das topologias. Assim são necessárias ter em conta duas medidas de distância, uma para cada espaço: euclidiana para o espaço de dados e *cityblock* para o espaço no mapa.

Relativamente á *Emergent*, um bom exemplo de um fenómeno é a onda *La Ola,* nos estádios de futebol. Um grupo grande de pessoas fazem uma simples tarefa de se pôr de pé e levantar os braços num curto espaço de tempo, formando assim uma onda por toda a multidão que só é visível á distância, e não no momento que está a ser feita por uma pessoa específica. Ou seja, a *Emergence* é a capacidade de um sistema desenvolver estruturas de alto nível por cooperação de vários processos elementares.

Este conjunto de programas é composto pelas seguintes *features* de sistema:

- Treino do ESOM com métodos de iniciação diferentes, treino de algoritmos, funções de distância, estratégias de otimização de parâmetros, topologias em grelha do ESOM e *kernels* na vizinhança;
- Visualização dimensional de dados com U-Matrix, P-Matrix, planos de componentes, SDH e mais;
- Visualização animada do processo de treino;

- Análise de dados interativa, explorativa e *clustering* através da ligação do ESOM aos dados de treino, classificações de dados, e descrição de dados;
- Criação de um classificador ESOM e aplicação automática para os novos dados;
- Criação de um *U-Maps* não redundante a partir de ESOM.

As ferramentas ESOM estão a ser desenvolvidas pelo grupo de trabalho Databionics, na Universidade de Marburg, na Alemanha. Estas encontram-se escritas em Java para que haja o máximo de portabilidade, e estão publicadas sobre os termos da licença GPL.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://databionic-esom.sourceforge.net/.

5.1.7 ELKI

A ferramenta ELKI ou seja, *Environment for Developing KDD-Applications Supported by Index-Structures* é uma f*ramework* de *software* desenvolvida especialmente para uso em investigação e ensino. A investigação é orientada pelo professor-investigador Hans-Peter Kriegel, conduzida pela unidade de investigação de sistemas de bases de dados da Universidade Ludwig Maximilian de Munique, na Alemanha. O seu objetivo principal é o desenvolvimento e avaliação de:

- Algoritmos de *data mining* (com enfâse em métodos não supervisionados em análise de *clusters* e deteção de *outliers*);
- Interação com as estruturas de índices das bases de dados.

Como se trata de um projeto de e para investigação, correntemente ainda não integra aplicações de *business intelligence* ou *interfaces* para gestão de bases de dados através do SQL.

Trata-se de uma ferramenta escrita em java, que apresenta uma arquitetura modular á volta de um núcleo de base de dados. Este núcleo usa um *layout* vertical de dados que os armazena em colunas e permite pesquisas de *Nearest Neighbour*, pesquisas por gamas e a funcionalidade de consultas à distância.

Faculta uma larga coleção de algoritmos altamente parametrizáveis, de forma a permitir uma avaliação fácil e verdadeira, e ainda, uma avaliação comparativa dos algoritmos. No ELKI, os algoritmos de *data mining* e as tarefas de gestão de dados são separadas e, por isso, permitem uma avaliação separada. Esta separação torna o ELKI único, no que diz respeito a *frameworks* como a WEKA ou o YALE, ou *frameworks* para estruturas em índices como o GiST. O código-fonte é escrito com as seguintes características em mente: extensibilidade, legibilidade e reutilização. Uma vez que a avaliação experimental dos algoritmos depende de vários fatores, o ELKI tem o objetivo de proporcionar uma base de código partilhada com implementações comparáveis de vários algoritmos.

Ao mesmo tempo, o ELKI está aberto a combinações arbitrárias de vários tipos de dados, distâncias ou medidas similares, ou formatos de ficheiros. A abordagem fundamental é a independência de ficheiros de análise ou conexões às bases de dados, tipos de dados, distâncias, funções de distância, e algoritmos de *data mining*. Assim, ao desenvolver novos algoritmos ou estruturas de índices é possível reutilizar as componentes já existentes e combiná-las.

A ferramenta foi desenhada sobre a licença de funcionamento AGPLv3, que espera servir a comunidade de investigação em *data mining* e bases de dados de forma benéfica. Apresenta os seguintes objetivos de *design*:

- Extensibilidade – com um *design* muito popular onde é permitido várias combinações de tipos de dados, funções de distância, algoritmos, formatos de *input*, estruturas de índices e métodos de avaliação;
- Contribuições – através do seu *design* modular que permite contribuições pequenas como funções de distância únicas e algoritmos individuais, é possível que estudantes e externos contribuam para o progresso do ELKI;
- Plenitude – para uma comparação exaustiva dos métodos, o ELKI pretende abranger o máximo de publicações e trabalho possíveis;
- Justiça – de forma a evitar uma comparação injusta a um programa mal implementado de propósito para o efeito, o ELKI tenta implementar todos os métodos da melhor forma possível, e publica o código fonte para que sejam adicionados melhoramentos por parte de terceiros. Todas as propostas de melhoria são aceites, como são exemplo as estruturas em índice para uma rápida leitura e compreensão da *range* e das *queries* do KNN;
- *Performance* – a arquitetura modular da *framework* permite que se otimizem as versões dos algoritmos e das estruturas em índice, de forma a acelerar os processos;
- Progresso – a cada versão desenvolvida, novas *features* aparecem e a *performance* é melhorada, no entanto existem ainda problemas na API que estão a tentar ser controladas.

O ELKI começou como um objeto de dissertação de doutoramento do Arthur Zimek, que foi premiado como *"SIGKDD Doctoral Dissertation Award 2009 Runner-up"*, pela *Association for Computing Machinery.* Alguns algoritmos foram publicados em simultâneo com a dissertação e agora estão presentes no ELKI.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://elki.dbs.ifi.lmu.de.

5.1.8 Gnome Data mining Tools

As ferramentas Gnome tratam-se de uma coleção de ferramentas de *data mining* em crescimento, que contêm tudo aquilo que é requerido, incluindo o GUI (*Guide User Interface*) e as aplicações de *data mining*. Trata-se de uma coleção de ferramentas desenvolvidas pela Togaware, baseadas em *interface*s de utilizador experimentais, escritos em Python e GTK. Inclui as seguintes funcionalidades:

- Associação de regras com o algoritmo *Apriori* – a aplicação "gdmapriori" é uma funcionalidade para extrair regras de associação a partir de transações de dados. Estão disponíveis inúmeras opções.
- Classificador Bayesiano – a aplicação "gdmbayes" é uma funcionalidade para contruir classificadores bayesianos a partir de dados de treino.
- Árvores de decisão – a aplicação "gdmdtree" é uma funcionalidade para construir árvores de decisão a partir de dados de treino. Existem várias opções disponíveis incluindo *generating rules*.
- Ferramenta de CSV - esta ferramenta proporciona a geração de gráficos *(ploting*) e tabelas (LaTeX).

Este conjunto de ferramentas *open source* encontram-se sobre a licença *GNU General Public Licence* como *software free* e *open source*, na esperança que outros a achem útil e eventualmente a possam melhorar.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.togaware.com/datamining/gdatamine/.

5.1.9 SCaVis

O SCaVis ou *Scientific Computation and Visualization Environment* é um ambiente para computação científica, análise de dados e visualização de dados, desenhado para cientistas, engenheiros e estudantes. Sucessor do popular jHepWork, tem sido desenvolvido intensivamente desde 2005. É retro compatível com a versão 3.9 do jHepWork, ou seja, qualquer código concebido para o antigo programa deve também correr no SCaVis. Pode ser usado em qualquer lugar onde a análise de dados numéricos em grandes volumes, análise estatística e matemática sejam essenciais (ciências naturais, engenharia, modelação e análise de mercados de finanças).

O programa incorpora várias *packages* de *software open source* numa *interface* coerente que usa o conceito de *scripting* dinâmico. Assim, pode ser usada com várias linguagens de *scpriting* para plataformas Java, como o *BeanShell, Jython* (a linguagem de programação *Python*), *Groovy* e *JRuby*

(linguagem de programação *Ruby*). A programação pode ser ainda concebida em Java nativo e além disso, os cálculos simbólicos podem ser feitos através do Matlab/ Octave.

Relativamente á sua portabilidade, esta plataforma não necessita de qualquer tipo de instalação, sendo que o simples *download e unzip* da *package* é suficiente e está pronto a correr. Pode ser corrido a partir do disco externo, de uma USB ou de quaisquer media. É possível trazê-lo em qualquer dispositivo e corrê-lo em qualquer computador com sistema operativo Windows, Mac e Linux.

As bibliotecas *core numerical* e gráficas estão licenciadas pela *GNU General Public Licence v3.* Ou seja, as bibliotecas de documentação, exemplos, instaladores, base de dados de assistência de código e os ficheiros de linguagens integrados no SCaVis IDE não estão licenciados pela licença GPL e encontram-se livres apenas para fins não comerciais (fins académicos, científicos e educacionais).

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://jwork.org/scavis/.

5.1.10 KEEL

O KEEL é uma ferramenta *open source* de *data mining* cujo objetivo é a extração de conhecimento baseado em aprendizagem evolucionária *(Knowledge Extraction based on Evolutionary Learning).* O KEEL inclui algoritmos de extração de conhecimento, técnicas de pré-processamento, aprendizagem de regras evolucionárias, sistemas genéticos *fuzzy* e outros.

O KEEL é um ferramenta escrita em Java, que avalia algoritmos evolucionários para problemas de *data mining* incluindo regressão, classificação, *clustering*, análise de padrões, etc. Tem sido desenvolvido por projetos nacionais espanhóis (s*panish national projects*) com a colaboração de grupos de investigação. A versão do KEEL presentemente disponível proporciona as seguintes funcionalidades:

- Gestão de dados: composta por um conjunto de ferramentas que podem ser usadas para construir novos dados, exportar e importar dados em outros formatos para formatos do KEEL, edição de dados e visualização, aplicação de transformações e particionamento de dados, etc.
- Projeção de experiências: construção de experiências desejadas nos conjuntos de dados selecionados (várias hipóteses: tipo de validação, tipo de aprendizagem, etc.)
- Projeção de experiências não balanceadas: construção de experiências desejadas nos conjuntos de dados não balanceados selecionados. Estas experiências são criadas para *datasets* "5cfo" (*five-fold cross-validation*) e incluem algoritmos específicos para dados não balanceados e algoritmos de classificação geral.

- Experiências com algoritmos de aprendizagem de múltipla instância.
- Testes estatísticos: proporciona ao investigador um conjunto completo de procedimentos estatísticos para comparações a par ou múltiplas.
- Experiências ao nível da educação: a sua estrutura permite-nos projetar uma experiência que pode ser analisada (*debugged*) passo a passo, de forma a ser usada como *guideline* para a compreensão do processo de aprendizagem de um certo modelo pela plataforma.

Para além destas funcionalidades o KEEL disponibiliza as seguintes *features* de sistema:

- Algoritmos evolucionários (EAs);
- Algoritmos de pré-processamento;
- Biblioteca de estatística para análise de resultados dos algoritmos;
- Algoritmos desenvolvidos através da biblioteca de classes de java para computação evolucionária (JCLEC);
- *Interface* de utilizador amigável, orientado para a análise de algoritmos;
- Criação de experiências no modo *online*, visando um apoio educacional, a fim de aprender o funcionamento dos algoritmos incluídos;
- Biblioteca de algoritmos de extração de conhecimento.

O KEEL está desenvolvido sobre a licença GPLv3.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.keel.es/.

5.1.11 KNIME

O KNIME ou, *Kontanz Information Miner* trata-se de uma plataforma líder na análise de dados, que ajuda as empresas a estar um passo á frente da mudança. Como uma plataforma moderna, o KNIME permite o desenvolvimento de estatísticas sofisticadas e operações de *data mining*, de forma a fazer análises de padrões, descobertas tendências e prever potenciais resultados nos dados.

Nasceu em 2004, na Universidade de Konstanz e foi desenvolvido por uma equipa de programadores liderada por Michael Berthold. Inicialmente foi desenhado como um produto proprietário e foi concebido para a indústria farmacêutica, mas posteriormente ficaria disponível no formato *open source*.

A necessidade de processamento e integração de grandes quantidades de dados, levou a que os programadores aderissem a *standards* rigorosos de engenharia de *software* para criar uma plataforma robusta, modular e altamente escalável, que contemplasse vários carregamentos de dados, transformações, análise e modelos de exploração visuais. A primeira versão da aplicação

saiu em 2006, e muitas empresas farmacêuticas começaram a usá-la, logo depois os vendedores do *software* começaram a construir ferramentas baseadas nela. (Wikipédia, KNIME, 2014)

Esta plataforma *open source* integra várias componentes para *machine learning* e *data mining* através de um conceito "modular *data pipelining*". O seu *interface* gráfico de utilizador permite reunir vários nódulos (*nodes*) para pré-processamento de dados, modelação e análise de dados e visualização.

O KNIME é escrito em Java e baseado na plataforma Eclipse onde é facilmente extensível através da sua API modular. No seu conjunto contempla um ambiente de desenvolvimento integrado (IDE) e um sistema de *plug-ins*. Existem vários produtos disponibilizados pelo KNIME:

- *KNIME Analytics Platform*
- *KNIME Personal Productivity*
- *KNIME Partner Productivity*
- *KNIME TeamSpace*
- *KNIME Server Lite*
- *KNIME Server*
- *KNIME Big Data Extension*
- *KNIME Cluster Execution*
- *KNIME Product Matrix*

Os produtos mais conhecidos e importantes são o KNIME *Analytics Platform* e o KNIME *Server*. O KNIME *Analytics Platform* incorpora centenas de *nodes* de processamento para dados I/O, pré-processamento e limpeza, modelação, análise e *data mining*, como também vários tipos de visualizações interativas como gráficos de dispersão, coordenadas paralelas e outras. Para além destas funcionalidades, integra todos os módulos de análise da ferramenta WEKA, e os *pluggins* adicionais que permitem o funcionamento de *scripts* em R, oferecendo acesso a uma biblioteca vasta de rotinas estatísticas.

A plataforma de análise do KNIME está disponível sobre a licença GPLv3, com uma exceção que permite o uso do *node* bem definido da API para adicionar extensões proprietárias. Isto também permite aos vendedores de *software* comercial adicionar extras para que as suas ferramentas possam ser executadas pelo KNIME, para incluir apoio profissional, produtividade e colaboração nas funcionalidades, proporcionando o melhor dos dois mundos.

Relativamente ao KNIME *Server*, este é o coração da configuração do KNIME. Nele é possível ter acesso a todos os *workflows* do KNIME e é possível integrá-los numa arquitetura orientada a serviços caso seja necessário. Assim, é possível fazer o *deployment*, uma vez que este produto

disponibiliza várias *features* de forma colaborativa, que são extensíveis ao KNIME *analytics platform*. As *features* que inclui são: autenticação de utilizador e direitos de utilizador, execução remota ou agendável, geração de relatórios, acesso do portal *web* aos *workflows*, serviços *web, workflow versioning*, repositório de *workflow* partilhado, espaço de dados partilhado, *metanodes* partilhados, acordo de suporte proprietário e *update* de produtos prioritários.

O KNIME proporciona ainda a capacidade de desenvolver relatórios baseados na nossa informação ou a possibilidade de automatizar a aplicação para uma nova visão dos produtos dos sistemas.

Hoje em dia, os utilizadores da aplicação podem ser encontrados em empresas de larga escala, por um conjunto de indústrias várias incluindo ciências da vida, serviços financeiros, editores, retalhistas, etc. em mais de 50 países.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.knime.org/knime.

5.1.12 Machine learning in Java (MJL)

A ferramenta *Machine learning in* Java (MJL) é uma plataforma *open source* de ferramentas desenvolvidas em Java com o objetivo de apoiar na investigação de *Machine learning*. Desenvolvida na Universidade do estado do Kansas, esta plataforma consiste na utilização de várias classes cujo propósito coletivo é o de facilitar experiências através de aprendizagem indutiva.

A plataforma MLJ é orientada sobre a licença de funcionamento *GNU Public License*.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://mldev.sourceforge.net/.

5.1.13 MiningMart

O MiningMart é uma ferramenta *open source* de *data mining* desenvolvida na Universidade de Dortmund na Alemanha. O projeto tem o objetivo de apostar nas novas técnicas, que permitem aos responsáveis pelas tomadas de decisão, o acesso direto á informação armazenada em bases de dados, *data warehouses*, e bases de conhecimento.

O objetivo central desta ferramenta é ajudar os utilizadores a fazerem escolhas inteligentes, através das seguintes funcionalidades que tem disponíveis:

- Operadores para pré-processamento com acesso direto á base de dados;
- Uso de técnicas *machine learning* para pré-processamento;
- Documentação detalhada de casos de sucesso;
- Descoberta de resultado de alta qualidade;
- Escalabilidade a várias bases de dados com tamanho considerável;

- Técnicas que automaticamente selecionam ou mudam representações.

A ideia básica do MiningMart é armazenar os melhores casos de treino de pré-processamento que foram desenvolvidas por utilizadores experientes, seguidamente os dados são descritos no nível *meta* e apresentados nas aplicações. Por fim, os utilizadores escolhem um caso, que seja semelhante às suas necessidades de negócio, e aplicam a transformação correspondente e a aprendizagem á sua aplicação.

As principais vantagens desta ferramenta são:

- Redução do tempo de pré-processamento usado no ambiente de integração;
- *Data documentation beyond meta-data usual in relational databases*;
- Documentação explícita dos passos de pré-processamento;
- Diferentes níveis de abstração para diferentes papéis em *data mining*;
- Suporte na adaptação ou reutilização de casos de sucesso;
- Conhecimento especializado partilhado de forma colaborativa na *internet.*

O *software* é livre para investigação e para aplicações não comerciais. Todas as fontes de dados estão disponíveis.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www-ai.cs.uni-dortmund.de/MMWEB/index.html.

5.1.14 ML-Flex

O ML-Flex é uma *package* de *software open source* desenhada para permitir um processamento de vários conjuntos de dados para análise de *machine learning* (classificação), de forma flexível e eficiente. Na sua essência, usa algoritmos de *machine learning* para derivar modelos de variáveis independentes, com o propósito de prever os valores de uma variável dependente (classe).

Escrito em Java, tem uma estrutura extensível para que os utilizadores possam facilmente adicionar funcionalidades que melhorem as do próprio ML-Flex.

Umas das particularidades da arquitetura desta ferramenta é prevenir o aparecimento de enviesamentos (*bias*). Com uma solução para lidar com as questões da validação cruzada *(cross validation*), que se trata de permitir que a análise seja dividia por múltiplas *threads*, num único computador ou e em vários computadores, levando a tempos de execução substancialmente mais curtos.

Os algoritmos de *machine learning* têm sido desenvolvidos em várias linguagens de programação e oferecem muitas incompatibilidades no que diz respeito as *interfaces*. O ML-Flex torna possível fazer *interface* com qualquer algoritmo que use como *interface* a linha de comandos. Esta

flexibilidade permite aos utilizadores desenvolver experiências *machine learning* através do ML-Flex como um escudo de proteção, enquanto se aplicam algoritmos que podem ter sido desenvolvidos em diferentes linguagens de programação ou que proporcionam *interfaces* diferentes.

O ML-Flex está licenciado pela *GNU General Publica License v3.0.*

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://mlflex.sourceforge.net/.

5.1.15 MLC++

O MLC++ trata-se de uma biblioteca de classes em C++ para *machine learning* supervisionado. O MLC++ encontra-se na versão 1.3X e foi desenvolvido na Universidade de Stanford como um domínio público, liderada por Ronny Kohavi *(SGI - Silicon Graphics International*), com a ajuda de Nils Nilsson e Yoav Shoham.

A versão mencionada é ainda distribuída pela empresa SGI. O produto SGI MLC++ (versão 2.0 e superior) inclui melhoramentos ao MLC++. Estes melhoramentos são apenas de domínio de investigação e estão disponíveis através de código e formatos através do *website*. O SGI MLC++ é usado no produto MineSet da SGI como motor principal para o servidor de *data mining*.

O MLC++ proporciona vários algoritmos de *machine learning* que podem ser usados por utilizadores finais, analistas, profissionais e investigadores. O objetivo central é proporcionar aos utilizadores uma quantidade variada de ferramentas que possam ajudar a analisar dados, a acelerar o desenvolvimento de novos algoritmos de análise, a melhorar a credibilidade do *software*, a proporcionar ferramentas de comparação e mostrar a informação de forma visual.

Mais do que uma coleção de algoritmos, o MLC++ é uma tentativa em extrair pontos em comum dos algoritmos de *machine learning* e decompô-los de forma a resultar uma visualização unificada simples, coerente e extensível.

As árvores de decisão do MLC++ podem ser visualizadas usando o visualizador de árvores SGI MineSet. Este visualizador permite a navegação pela árvore, *zoom* dos nódulos mais interessantes, visualização de cálculos e seleção de pontos de interesse.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.sgi.com/tech/mlc/index.html.

5.1.16 OpenNN

O OpenNN, o seja, *Open Neural Networks Library* é uma biblioteca de classes escritas em C++ que implementa redes neuronais, um modelo clássico na inteligência artificial. Anteriormente, esta biblioteca era conhecida como *Flood*.

O seu desenvolvimento iniciou-se em 2003 no Centro Internacional para métodos numéricos em engenharia (*International Center for Numerical Methods in Engineering).* Mais tarde desfragmentou-se em pequenos projetos e correntemente é a *Intelnics,* uma empresa *start-up*, que está a desenvolver a plataforma.

Relativamente ao seu *design*, o OpenNN está formulado a partir da perspetiva de uma análise funcional e o cálculo de variações. A abordagem para resolver esta questão passa por 3 momentos: escolha de uma rede neuronal que se aproxime da solução; formulação do problema através da seleção de uma função apropriada e por fim desenvolvimento de uma otimização matemática com um algoritmo de forma a encontrar bons parâmetros.

Esta ferramenta é usada por centenas de cientistas por todo o mundo como método de investigação de novos algoritmos de *machine learning*, e foi desenhada com o objetivo de "aprender" a partir de *datasets* e modelos matemáticos. No que diz respeito aos *datasets* opera em funções de regressão, reconhecimento de padrões e predição de séries temporais, relativamente aos modelos matemáticos atua no controlo otimizado e no desenho otimizado de formas. De uma forma geral, juntando as duas categorias, trabalha com problemas invertidos.

O pacote do produto vem com um conjunto de testes unitários, muitos exemplos e documentação extensiva. Em resumo o OpenNN proporciona uma *framework* eficaz para investigação e desenvolvimento de algoritmos de redes neuronais e aplicações.

O OpenNN corre em qualquer computador, encontra-se disponível no *SourceForge* e é desenvolvido sobre a licença *GNU Lesser General Public Licence.*

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://opennn.cimne.com/.

5.1.17 Orange

A ferramenta Orange é uma ferramenta *open source* de análise de dados direcionada para os mais novos na área e também para especialistas. Trata-se de conjunto de *software* compreensível e baseado em componentes para *machine learning* e *data mining*, desenvolvido no laboratório de bioinformática na Faculdade de Ciência da Computação e Tecnologias da Universidade de Ljubljana, na Eslovénia, em conjunto com uma comunidade de apoio *open source*.

A primeira versão da plataforma foi em 1996, tinha o nome de ML*, e tratava-se de uma *framework* de *machine learning* em C++. Em 1997 adicionaram-se vínculos da linguagem *Python* o que levou

á criação da *framework* chamada Orange. Até aos dias de hoje muitas alterações foram aplicadas para melhorar a extensibilidade e capacidades da plataforma, sendo que em 2013 foi redesenhada a *interface* de utilizador. Neste momento encontra-se na sua versão 2.7 para Windows.

Desenhado em C++ e *Python*, permite aplicar técnicas de *data mining* através de programação visual ou *scripting* em *Python*.

Tem componentes para *machine learning* e ainda *add-ons* para bioinformática e *text mining*. Contém as seguintes *features*:

- Programação visual – desenha o processo da análise de dados através de programação visual, relembrando as escolhas e sugerindo as combinações mais usadas, o Orange é uma ferramenta que escolhe de forma inteligente que tipo de canais de comunicação, entre os *widgets*, a usar.
- Visualização – O Orange está equipado com vários tipos de visualizações, desde gráficos de dispersão, gráficos de barras, árvores, dendrogramas, *networks* e mapas sensoriais.
- Interação e análise de dados – as ações propagam-se facilmente através de esquemas de análise de dados. A seleção de um subconjunto de dados num *widget* específico pode automaticamente disparar mudanças noutro. Ao combinar vários *widgets* é possível desenhar uma *framework* de análise de dados de acordo com as nossas necessidades.
- Várias ferramentas – Contem mais de 100 *widgets* e continua a crescer. Cobre todas as tarefas de análise de dados mais importantes, e é ainda especializado em *add-ons* como por exemplo o Bioorange para bioinformática.
- *Interface* para *scripting* – Com uma *interface* para *scripting* com Python, torna-se simples programar novos algoritmos e desenvolver procedimentos de análise de dados complexos, usando e reusando todo o poder da programação visual.
- Extensibilidade – É possível desenvolver os próprios *widgets*, e estender a *interface* de *scripting*, ou ainda criar o próprio contentor de *add-ons*, tudo de forma integrada com a restante aplicação, permitindo a reutilização de código e componentes.
- Documentação – Cobre os primeiros passos no que diz respeito á programação visual e ainda proporciona uma apresentação detalhada dos *widgets* disponíveis, guias através do *scripting*, e apresenta documentação compreensiva.
- *Open source* – O Orange é uma ferramenta *open source* com uma comunidade de apoio ativa. É possível pesquisar e aceder ao código fonte, aumentá-lo e reusá-lo e mesmo participar no seu desenvolvimento enquanto a comunidade oferece o suporte necessário no desenvolvimento. Está desenvolvida sobre a licença GNU GPL.

- Independência da plataforma – O Orange pode correr em Windows, Mac OS X, e numa variedade de sistemas operativos Linux.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://orange.biolab.si/.

5.1.18 PredictionIO

O PredictionIO é um servidor *open source* de *machine learning* para os programadores criarem as suas próprias *features* de predição, como personalização, recomendação e descoberta de conteúdos. Trata-se de uma ferramenta poderosa, escalável e personalizável construída no topo de *frameworks* como o *Hadoop, Scalding e o Cascading*. Permite aos programadores e engenheiros a construção e personalização de aplicações inteligentes. Com esta ferramenta é possível adicionar as seguintes *features* às aplicações de forma instantânea:

- Previsão do comportamento de utilizadores;
- Oferecer vídeos personalizados, novidades, anúncios, novidades de empregos;
- Ajudar os utilizadores a descobrir eventos interessantes, documentos, aplicações e restaurantes;
- Proporcionar serviços de correspondência;

Composto por duas componentes principais: *Event Server e Engine*, recebe os dados a partir de uma aplicação e emite os *outputs* de predição.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://prediction.io.

5.1.19 RapidMiner

O RapidMiner é uma ferramenta líder em análise preditiva, que apresenta uma solução *desktop-to-cloud* muito fácil de usar.

Esta ferramenta foi desenvolvida inicialmente em 2001, com o nome YALE (*Yet Another Learning Environment),* por Ralf Klinkenberg, Ingo Mierswa e Simon Fischer na Unidade de Inteligência Artificial da Universidade Técnica de Dortmund. Em 2006 passou a ser desenvolvida pela Rapid-I, uma empresa fundada pelos investigadores enunciados. Em 2007 mudou o seu nome para RapidMiner.

Atualmente o RapidMiner apoia equipas de colaboradores a trabalhar em tomadas de decisão inteligentes através do uso de inteligência preditiva e *predactions (predictions and actions)* – previsões baseadas em ações – para melhorar as operações de uma organização. Esta tecnologia permite às empresas atingir decisões inteligentes de negócio, ao usar a inteligência preditiva e ações baseadas em predições.

O RapidMiner proporciona *software*, soluções, e serviços na área de análise avançada, incluindo análise preditiva, *data mining*, e *text mining*. Lida com a análise de grandes quantidades de dados incluindo bases de dados e texto. Mais especificamente proporciona operações de *data mining* e procedimentos de *machine learning* tais como: carregamento de dados, transformação de dados *(ETL – Extract, Transform, Load)*, pré-processamento de dados, visualização, análise preditiva, modelação estatística, avaliação e *deployment* (Wikipédia, RapidMiner, 2014).

Segundo o KDnuggets, esta plataforma é a mais avançada no mercado de análise de dados e tem espalhados por mais de 50 países centenas de aplicações, tanto *stand-alone* como integradas nos produtos dos clientes. Atualmente apresenta 3 produtos associados:

- *RapidMiner Studio*
- *RapidMiner Server*
- *RapidMiner Managed Server.*

Relativamente ao *RapidMiner Studio* apresenta as seguintes f*eatures*:

- Assistente de aplicações – O RapidMiner tem os mais recentes assistentes de aplicações para redução de *churn*, análise de sentimentos, manutenção preditiva e *marketing* direto.
- Extensibilidade – Inclui centenas de métodos para integração, transformação, modelação e visualização de dados – com acesso a todas as fontes como *excel, access, oracle, ibm db2, Microsoft sql, sybase, ingres, MySQL, Postgres, SPSS, dBase*, ficheiros de texto e muito mais.
- Vários suportes – O RapidMiner corre em qualquer plataforma e sistema operativo.
- Não necessita de programação – apresenta um *interface* de utilizador poderoso e intuitivo para desenhar os processos de análise. Um ambiente visual fácil de usar permite que se reconheçam erros, aplicação de correções rápidas e ver resultados rápidos e afinados – sem necessidade de recorrer ao código.
- Preferido pelos utilizadores - uma ferramenta muito conhecida no mundo dos negócios, com revisões muito boas a serem executadas periodicamente por novatos e profissionais experientes.

O *RapidMiner Server* trata-se de um ambiente do tipo servidor que permite uma poderosa análise preditiva suportada pelo poder da computação. Assim apresenta as seguintes funcionalidades sem restrições:

- Obtenção de resultados de predições em tempo real, através da aplicação de otimização da *performance*;
- Integração de outras ferramentas, algoritmos e fontes de dados, para além da integração com o *RapidMiner Studio*;

- Colaboração a partir de *dashboards* interativos - através da partilha de repositórios, tarefas, recursos e informação com a equipa de trabalho a partir de qualquer lado do mundo. É possível monitorizar e partilhar as análises de dados a qualquer momento em qualquer lado;
- Processamento remoto das análises de dados – através de um *interface web* flexível é possível correr os processos de trabalho durante 24h/7d, mantê-los atualizados com bases de dados em expansão, calcular os resultados ao minuto e fazer relatórios sobre possíveis alterações.

O *RapidMiner Managed Server* é uma ferramenta que não é *free*, gerida por especialistas que aplicam as configurações necessárias, *backups*, instalações, manutenção, monitorização e atualizações. Desta forma apresenta as seguintes características:

- *Deployment* rápido - pode ser instalado e ficar pronto em poucos minutos;
- Custos reduzidos de propriedade (TCO) – sem a necessidade de ter um servidor *expert* de gestão a operar internamente os clientes podem baixar o custo total do produto.
- Alto rendimento – impulsiona o Amazon RDS *(Relational Database System*) para beneficiar da rápida transferência de dados com as bases de dados *MySQL, Oracle, Microsoft SQL, Server e PostgreSQL.*
- Capacidade de programar – várias opções como calendarização de ações e APIs são programáveis.

Escrito em Java, atualmente encontra-se na sua versão 6.0 e proporciona uma *interface* para desenhar e executar *workflows* de análise de dados. Esses *workflows* no RapidMiner consideram-se processos e são executados por vários operadores que desenvolvem tarefas individualmente. Os operadores trabalham em árvore sendo que o *output* de cada um se torna no *input* do próximo. Para além dos algoritmos e técnicas que contém, integra ainda alguns esquemas, modelos e algoritmos da WEKA e alguns *scripts* em R que podem ser usados.

Esta ferramenta encontra-se disponível no SourceForge, com a certificação da OSI como *#1 business analytics software*, e encontra-se a ser distribuída sobre a licença AGPL. Para além disso, também disponibiliza versões comerciais dos produtos anteriormente referidos.

Considerado pela Gartner como uma das aplicações líderes, foi descrita com as seguintes potencialidades:

- A plataforma suporta uma amplitude e profundidade extensas e com isso chega muito perto dos líderes de mercado;
- As referências reportaram bons níveis de satisfação geral, uma comunidade de utilizadores forte e uma incorporação consistente dos requisitos do produto nas versões futuras;

- Foi quase sempre selecionado com base na facilidade de uso, custo de licença, e velocidade de desenvolvimento de modelos e capacidade de construção de largos números de modelos. Um número de templates orientaram os utilizadores nos casos mais comuns de uso de predição;
- As referências dos clientes citam altos níveis de satisfação com o acesso aos dados, filtro de dados e manipulação, análise preditiva e componentes mais avançadas de análise do produto.

Para além disso foi avaliada pelo KDnuggets como o *software* de análise de dados mais popular com mais de 3 milhões de *downloads* e mais de 20000 utilizadores pelo eBay, Intel, PepsiCo e Kraft Foods. A empresa que anteriormente detinha e fazia a gestão do *software* – Rapid-I, mudou recentemente o seu nome para RapidMiner e considera-se líder de mercado de *software* para serviços de análise de dados preditiva relativamente a empresas como SAS, SQL Server, IBM, etc.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://rapidminer.com/.

5.1.20 R (Rattle)

A ferramenta RATTLE (*R Analytical Tool to Learn Easy*) trata-se de uma *interface* de utilizador gráfica que usa a linguagem R. Através de um *interface* de utilizador gráfico baseado no Gnome, o RATTLE pode ser usado para ele próprio se encarregar de projetos de *data mining*. O RATTLE proporciona ainda o uso sofisticado de técnicas de *data mining* ao usar a linguagem *open source* e *free* R.

O objetivo desta ferramenta é proporcionar uma *interface* intuitiva que nos leve pelos passos básicos de *data mining*, como é ilustrado pela linguagem R. Embora a ferramenta sozinha seja suficiente para todas as necessidades do utilizador, também proporciona um impulso para o processamento e modelação mais sofisticado em R, para *data mining* sofisticado e sem constrangimentos.

RATTLE é usado no dia-a-dia por uma equipa enorme de *data miners* na Austrália, disponível a partir da Togaware, e por uma variedade de empresas do governo e comerciais por todo mundo. Um número de consultores internacionais também usa o RATTLE nos seus negócios diários. O autor do RATTLE recebeu o *Australia Day Medallion 2007*, por liderar e ser mentor de *data mining* nos escritórios da *Australia Taxation* e particularmente citado pelo desenvolvimento e partilha do sistema RATTLE.

O RATTLE é também usado para ensinar a prática de *data mining*. Esta foi a primeira ferramenta de instrução para um *workshop* de *data mining* em Canberra, e no *Harbin Institute of Technology, Shenzhen Graduaete School (*2006), etc. Tem sido usado também nos cursos da Universidade de Yale, entre muitos outros.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://rattle.togaware.com.

5.1.21 TANAGRA

O TANAGRA é um *software* livre de *data mining* com propósitos académicos e de investigação. Propõe vários métodos de *data mining* desde análise exploratória de dados, aprendizagem estatística, *machine learning* e bases de dados.

Este projeto é o sucessor da SIPINA que implementa vários algoritmos de aprendizagem supervisionada. O TANAGRA é mais poderoso, pois contém aprendizagem supervisionada mas também outras componentes como *data source*, visualização, estatística descritiva, seleção de instâncias, seleção de *features*, construção de *features*, regressão, análise factorial, *clustering*, aprendizagem *meta-spv*, avaliação da aprendizagem e associação. Desenvolvido por Ricco Rakotomalala na Universidade de Lumière em Lyon na França, foi lançada a primeira versão do *software* em 2003.

O propósito principal do TANAGRA é servir a comunidade académica e de investigação, facultando-lhes uma ferramenta de *data mining* fácil de usar. Seguidamente, pretende que a sua arquitetura seja facilmente compreendida pelos investigadores, para que estes possam adicionar os seus próprios métodos para cada necessidade. Ou seja, o objetivo do TANAGRA é que os investigadores se possam concentrar nas suas pesquisas sem preocupações com as ferramentas de gestão de dados. Por fim, esta plataforma pretende direcionar-se também aos mais novos, e ser vista como uma ferramenta pedagógica na aprendizagem de técnicas de programação, de forma a difundir uma metodologia possível para a construção deste tipo de *software open source*.

O funcionamento do TANAGRA é semelhante ao das outras ferramentas de *data mining*. O utilizador desenha visualmente um processo de *data mining* numa forma de diagrama. Cada nódulo representa uma técnica estatística ou de *machine learning* e a conexão entre dois nódulos representa a transferência de dados. Por fim os resultados são representados através de um formato em HTML de forma a ser possível exportar os *outputs* para um *browser.*

O TANAGRA não inclui, presentemente, o que torna as ferramentas comerciais mais fortes neste domínio: um conjunto de fontes de dados, acesso direto a *data warehouses* e bases de dados, limpeza de dados, utilização interativa.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://chirouble.univ-lyon2.fr/~ricco/tanagra/en/tanagra.html.

5.1.22 Vowpal Wabbit (Fast Learning)

O Vowpal Wabbit ou VW trata-se de um programa/ biblioteca de aprendizagem de sistema *out-of-core* que foi originalmente desenvolvida na Yahoo! Research e, correntemente pela Microsoft. Dispõe uma implementação eficiente e escalável de *machine learning online*, ponderação e seleção de diferentes funções de perda e otimização de algoritmos.

O Vowpal Wabbit representa a essência da velocidade no *machine learning*, capaz de aprender a partir de *datasets terafeatures* com facilidade. Através de uma aprendizagem paralela, pode-se exceder o resultado de qualquer *interface* de *machine learning* individual ao fazer aprendizagem linear, o primeiro entre alguns algoritmos de aprendizagem.

Este cobre as opções básicas e mais comuns, os formatos de dados para diferentes tipos de problemas, como classificação binária, regressão, classificação de *multi-classes*, *cost-sensitive* e predição de sequências.

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://hunch.net/~vw/.

5.1.23 WEKA

A plataforma WEKA ou seja *Waikato Environment for Knowledge Analysis*, trata-se de uma ferramenta popular de *machine learning*, escrita em Java que foi desenvolvida na Universidade de Waikato na Nova Zelândia.

A ferramenta dispõem de um conjunto de algoritmos *machine learning* para tarefas de *data mining*, onde os algoritmos podem ser aplicados diretamente a um *dataset* ou invocados pelo próprio código Java. Para além disso, contém ferramentas e algoritmos para pré-processamento de dados, classificação, regressão, *clustering*, associação de regras e visualização para análise de dados e predição de modelos. Tudo isto está disponível através de *interfaces* de utilizador gráficos para um fácil acesso às várias funcionalidades.

As suas técnicas pressupõem que os dados estão disponíveis como um ficheiro *flat* individual ou numa relação, onde cada ponto de dados é descrito por um conjunto de atributos. Proporciona acesso às bases de dados SQL através do Java *Database Connectivity* e pode processar os resultados obtidos pela *query* da base de dados.

Inicialmente desenhada para analisar dados relativos á agricultura, tratava-se de uma versão *front-end* chamada TCL/TK (1993), que fazia modelação de algoritmos implementados noutras linguagens de programação, pré-processamento de dados em C e tinha também um sistema baseado na criação de ficheiros que fazia experiências em *machine learning*. Atualmente com o nome WEKA, encontra-se na sua versão 3.7.2, escrita em Java desde 1997, e é usada em várias áreas de aplicação, principalmente para fins académicos e de investigação. Muitas funcionalidades

existentes na versão anterior a esta foram movidas para *packages* de extensões, o que torna mais fácil que outros contribuam com novas extensões para a WEKA, uma vez que a sua arquitetura modular assim o permite.

Os seus pontos fortes passam pela portabilidade do *software*, uma vez que é escrito em Java e por isso é compatível com quase todas as plataformas modernas, e os seus *interfaces* de utilizador gráficos que permitem que se use a plataforma de forma mais simples e fácil. Os *interfaces* de utilizador que disponibiliza são:

- *Explorer*
- *Knowledge Flow*
- *Experimenter*

A WEKA é um *software open source* desenvolvido sobre a licença *GNU General Public License.*

As informações obtidas relativamente a esta ferramenta encontram-se disponíveis na página: http://www.cs.waikato.ac.nz/ml/weka/index.html.

5.2 Comparação entre as ferramentas suite data mining open source

Após a análise individual de cada uma das ferramentas *open source* de *data mining* que constam na lista do KDnuggets, vamos agora compará-las. Esta comparação vai incidir em aspetos gerais como tipo de licença, linguagem de desenvolvimento, compatibilidade com sistemas operativos, última versão desenvolvida e ano da mesma. O objetivo deste ponto é verificar, de forma de simplificada, algumas das características gerais das ferramentas em estudo.

Na Tabela 5.1 é possível visualizar os aspetos enunciados anteriormente relativamente a cada ferramenta de DM. Toda a informação em estudo foi recolhida e analisada a partir de várias fontes bibliográficas.

Tabela 5.1 - Características das ferramentas open source abordadas

Ferramenta	Licença	Termos de uso	Linguagem	Sistemas operativos	Última versão estável	Ano da última versão	Versão comercial
ADaM	Proprietária	Free Open source	C++/ Python	Windows Mac Linux	4.0.2	2005	Não
Alteryx	Proprietária	Free e Shareware	?	?	?	?	Sim
AlphaMiner	GPLv2	Free Open source	Java	Windows	1.0	2005	Não
CMSR	Proprietária	Free e Shareware	?	?	?	?	Não

Ferramenta	Licença	Termos de uso	Linguagem	Sistemas operativos	Última versão estável	Ano da última versão	Versão comercial
CRAN task view	?	?	R	?	?	2014	Não
Databionic ESOM	GPL	Free Open source	Java	Windows Mac Linux	?	?	Não
ELKI	AGPLv3	Free Open source	Java	Windows Mac Linux	0.6.0	2014	Não
Gnome Data Mining Tools	GPL	Free Open source	C++/ Python	Debian GNU/ Linux	?	?	Não
SCaVis	GPLv3	Free Open source	Java/ scripting lan-guages	Windows Mac Linux Android	2.0	2014	Não
KEEL	GPLv3	Free Open source	Java	?	29-01-2014	2014	Não
KNIME	GPLv3	Free Open source	Java	Windows Mac Linux	2.10.1	2014	Sim
MJL (machine learning in java)	GPL	Free Open source	Java	Windows Mac Linux	1.01 alpha	2002	Não
MiningMart	GPLv2	Free Open source	Java	Windows Mac Linux	V 1.1	2006	Não
ML-Flex	GPLv3	Free Open source	Java	Windows Mac Linux	?	?	Não
MLC++	Proprietária	Free Open source	C++	Windows Linux	1.3X	?	Não
OpenNN	LGPL	Free Open source	C++	Windows Mac Linux	1.0	2014	Não
Orange	GPL	Free Open source	C++/ Pyhton	Windows Mac Linux	2.7 Win	2014	Não
Prediction IO	?	Free Open source	?	Windows Mac Linux	V 0.8.0	?	Não

Ferramenta	Licença	Termos de uso	Linguagem	Sistemas operativos	Última versão estável	Ano da última versão	Versão comercial
RapidMiner	AGPL	Free Open source	Java	Windows Mac Linux	6.0	2013	Sim
RATTLE	GPL v2	Free Open source	R	Windows Mac Linux	3.3.1	2014	Não
TANAGRA	Proprietária	Free Open source	C++	Windows	1.4.50	2013	Não
Vowpal Wabbit	BSD	Free Open source	C++	Windows Mac Linux	7.6	2014	Não
WEKA	GPL	Free Open source	Java	Windows Mac Linux	3.6.11	2014	Não

O parâmetro “termos de uso” foi preenchido com três características possíveis: *free, open source* e *shareware*. Para fins de desambiguação, as características representam as seguintes noções:

- *Free* – significa que não existem custos do uso ou *download* de uma ferramenta de *data mining*;
- *Open Source* - significa que o código fonte do *software* é aberto e disponibilizado aos utilizadores da ferramenta;
- *Shareware* – significa que o programa é facultado gratuitamente, mas com algum tipo de constrangimento e restrição, que geralmente se relaciona com funcionalidades limitadas ou uso gratuito limitado a um período de tempo.

Os campos da tabela preenchidos com “?” referem-se a falta de informação sobre cada aspeto onde estão localizados. A informação disponível sobre as características e funcionalidades é variável de ferramenta para ferramenta, havendo em muitos casos muito pouca documentação. Assim sendo, as próximas conclusões são aplicadas apenas sobre os resultados que se podem fundamentar com a tabela anterior.

Relativamente á informação da Tabela 5.1 podemos tirar as seguintes conclusões:

- A maioria das ferramentas em estudo utiliza uma licença do tipo GPL ou variante;
- Predominam os sistemas *free* e *open source*;
- No que diz respeito às linguagens de desenvolvimento, a linguagem Java é a mais usada;

- Cerca de 15 ferramentas são compatíveis com os três sistemas operativos mais comuns: Windows, Mac e Linux;
- Apenas 12 ferramentas lançaram a sua última versão nos últimos dois anos (2013/2014), sendo que todas as outras disponibilizam versões mais antigas;
- Só o Alteryx, KNIME e o RapidMiner disponibilizam uma versão comercial.

Segundo os gráficos de votação do KDnuggets sobre que *software* de análise de dados/ *data mining* os utilizadores usaram entre os anos de 2010-2014 em projetos reais, obtivemos os resultados da Tabela 5.2.

Tabela 5.2 - Resumo dos resultados da votação "What analytics, Big Data, Data mining, Data Science software you used in the past 12 months for a real project?" do KDnuggets

Ferramentas	1º lugar	2º lugar	3º lugar	4º lugar	5º lugar	6º lugar
2010	RapidMiner	R	KNIME	WEKA	Outras	Orange
2011	RapidMiner	R	KNIME	WEKA	Outras	Orange
2012	R	RapidMiner	KNIME	WEKA	Orange	Outras
2013	RapidMiner	R	WEKA	Python	KNIME	RATTLE
2014	RapidMiner	R	Python	WEKA	KNIME	Outras

Os resultados estão disponíveis nos seguintes endereços eletrónicos:

- 2010 – http://www.kdnuggets.com/polls/2010/data-mining-analytics-tools.html
- 2011 - http://www.kdnuggets.com/polls/2011/tools-analytics-data-mining.html
- 2012 - http://www.kdnuggets.com/polls/2012/analytics-data-mining-big-data-software.html
- 2013 - http://www.kdnuggets.com/polls/2013/analytics-big-data-mining-data-science-software.html
- 2014 - http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-software-used.html

Após estas conclusões, e na impossibilidade de comparar em termos práticos 23 ferramentas *open source* de DM, foram escolhidas as melhores de acordo com os resultados da votação evidenciada na Tabela 5.2, e de acordo com o conjunto abordado em *5.1.Suites data mining open source.* As ferramentas escolhidas para análise são o RapidMiner; Orange; WEKA e o KNIME. Consideramos que estas são as quatro melhores ferramentas *open source* de *data mining*, sendo que, existem

muitas outras que não estão a ser colocadas em estudo, por terem objetivos muito específicos ou serem baseadas em linguagens de programação, como é o caso da ferramenta R e Python que apresentam resultados de utilização muitos bons nas votações anuais do KDnuggets.

Neste sentido, no capítulo seguinte iremos focar-nos na análise e avaliação prática destas quatro ferramentas, uma vez que apresentam o conjunto mais completo de *features* e também por serem reconhecidas no mundo da *data mining* por vários especialistas da área.

6 AVALIAÇÃO PRÁTICA DAS FERRAMENTAS

Após as conclusões e os resultados da análise bibliográfica apresentados capítulo anterior, e na impossibilidade de comparar em termos práticos 23 ferramentas *open source* de DM, foram escolhidas as melhores de acordo com os resultados da votação evidenciada na Tabela 5.2. As quatro ferramentas escolhidas foram o RapidMiner, Orange, WEKA e o KNIME. Estas serão instaladas de forma a avaliar as suas especificidades e pontos fortes.

Uma vez identificadas as melhores ferramentas de *data mining*, vamos agora proceder á instalação de cada uma, de forma a perceber todas as características e requisitos que incluem. Relativamente á instalação, esta ocorreu num sistema operativo Windows 8 de 64 bits. No entanto são também apresentados os requisitos de instalação para outras opções de sistema de operativo.

Posteriormente á instalação das quatro ferramentas são apresentados e avaliados aspetos técnicos sobre cada uma delas, de uma forma mais gráfica, para simplificar a sua interpretação.

6.1 Instalação do RapidMiner

Antes de instalar o RapidMiner é necessário verificar que sistema de operativo se vai usar na máquina onde se vai instalar o *software*. A versão do RapidMiner a instalar é a RapidMiner Studio correspondente á versão 6.1.

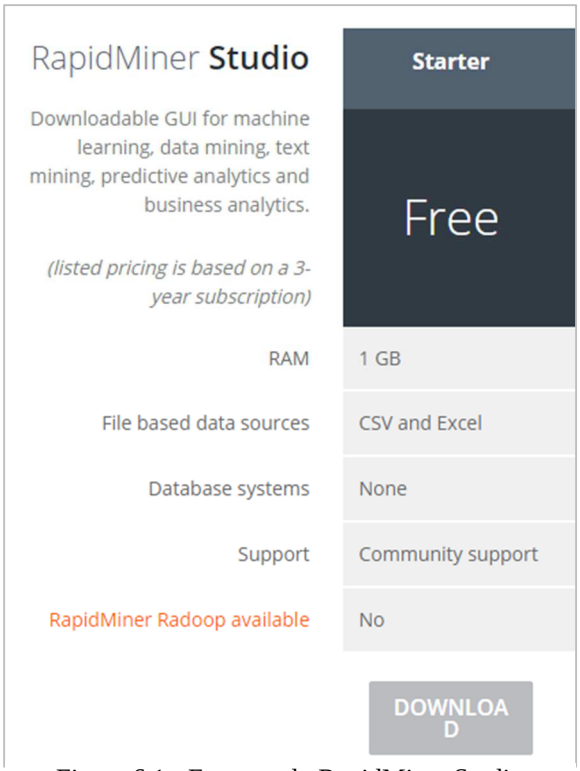

Figura 6.1 - Features do RapidMiner Studio

Relativamente aos requisitos, estes dependem do sistema operativo a usar, no entanto de forma geral, os passos do processo de instalação são parecidos variando só em algumas componentes. As informações e imagens seguintes estão disponíveis na página https://rapidminer.com/.

- **Windows**
    - **Instalar**

Para instalar o RapidMiner no sistema operativo Windows faz-se o *download* do ficheiro executável disponível na página do RapidMiner (https://rapidminer.com/), com a designação **rapidminer-XXX-install.exe.** Para isso, é necessário fazer um breve registo na página da ferramenta, de forma a introduzir dados de utilizador como nome, apelido, telemóvel, instituição e área de trabalho, para criar um perfil. Após a criação do perfil, é possível fazer *download,* gerir as versões dos RapidMiner á escolha e gerar licenças de utilizador. Seguidamente, é necessário escolher que tipo de sistema operativo vamos usar (32 bit ou 64 bit) e descarregar um ficheiro executável. Terminado o *download*, é necessário abrir o executável e correr o *wizard* para seguir todas as instruções do produto. Terminada a instalação é criado um ícon no *desktop* do computador para acesso direto á aplicação. Abrindo esse ícon será necessária a ativação de uma *license key*, facilmente gerada no perfil de utilizador, e o *software* está pronto a ser usado.

    - **Iniciar**

Para iniciar o RapidMiner, basta fazer duplo clique no ícon que se encontra no ambiente de trabalho e a aplicação é iniciada. Ao iniciar, é necessário saber que o RapidMiner usa 90% da memória. No caso de existirem problemas nesse contexto existem outras possibilidades de iniciá-lo, que são apresentadas de seguida.

- **Outras plataformas**

Figura 6.2 - RapidMiner a iniciar

- **Instalar**

Para instalar o RapidMiner noutras plataformas é necessário o Java *Runtime Environment (JRE)* versão 7 ou superior. Depois de seguidas todas as instruções de instalação desta componente é necessário fazer *download* do ficheiro zip da página do RapidMiner com a designação **rapidminer-XXX.zip.** Para extrair o ficheiro de download do RapidMiner é necessário uma ferramenta de extração como são exemplo o WinRAR ou 7Zip. Por fim, o RapidMiner encontra-se instalado e pronto a ser usado.

- **Iniciar**

Para iniciar o RapidMiner existem três opções. Estas opções permitem adaptar o uso de memória pela ferramenta uma vez que a instalação de java usa apenas 64 ou 128 MB de memória.

1 – Definir o máximo de memória que possa ser usada pelo java e pela sua localização através das variáveis MAX_JAVA_MEMORY e JAVA_HOME. Depois pode iniciar-se o *script* RapidMiner.bat no subdiretório de *scripts* do RapidMiner ou invocá-lo pela linha de comandos.

2 – Abrindo o ficheiro lib/rapidminer.jar.

3 – Através da linha de comandos (java –jar rapidminer.jar):

Figura 6.3 - Exemplo da linha de comandos

O máximo de memória pode ser especificado pela opção –Xmx da Java *Virtual Machine*:

Figura 6.4 - Exemplo da linha de comandos

- **Extensões**

O RapidMiner proporciona algumas extensões que podem ser instaladas através do *Marketplace*. O *Marketplace* pode ser acedido pelo menu *Help* (ajuda) em *Updates and Extensions (Marketplace).* As extensões pode ser por exemplo *Text Mining, Web Mining, Image Mining,* etc.

6.2 Instalação da WEKA

Antes de instalar a WEKA é necessário verificar que sistema de operativo se vai usar na máquina onde se vai instalar o *software*. A versão da WEKA a instalar é a 3.6. No entanto existem duas versões que podem ser instaladas: a versão estável que corresponde á última edição do livro de *data mining* que apenas recebe correções (*bug fixes*) e a versão de desenvolvimento que recebe novas *features* e exibe um sistema de gestão de *packages* que permite á comunidade a fácil adição de novas funcionalidades. Vamos instalar a versão estável.

Relativamente aos requisitos, estes dependem do sistema operativo a usar e ainda da versão da WEKA que se vai instalar. As informações e imagens seguintes estão disponíveis na página http://www.cs.waikato.ac.nz/ml/weka/index.html/.

A seguinte imagem apresenta uma tabela que faz a correspondência entre as versões do WEKA e a versão de java necessária para cada uma delas.

		Java			
		1.4	1.5	1.6	1.7
WEKA	<3.4.0	X	X	X	X
	3.4.x	X	X	X	X
	3.5.x	3.5.0-3.5.2	>3.5.2 r2892, 20/02/2006	X	X
	3.6.x		X	X	X
	3.7.x		3.7.0	>3.7.0 r5678, 25/06/2009	X

Figura 6.5 - Correspondência entre versão da WEKA com a versão do java

- **Windows**

Está disponível para o sistema de 32 bits um ficheiro executável que inclui a JVM 1.7 (Java *Virtual Machine*) – "weka-3-6-11jre.exe" - e um ficheiro do mesmo género sem a JVM para instalar se o utilizador já tiver o java 1.6 ou superior no seu sistema - weka-3-6-11.exe. Para o sistema de 64 bits existe o mesmo ficheiro executável com a JVM 1.7- weka-3-6-11jre-x64.exe e sem a JVM 1.7 - weka-3-6-11-x64.exe - que funcionam nas mesmas condições.

- **Iniciar**

Para iniciar o WEKA basta fazer duplo clique no ícon que se encontra no ambiente de trabalho e a aplicação é iniciada.

Figura 6.6 - WEKA a iniciar

- **Mac**

Para este sistema operativo estão disponíveis duas versões. A primeira trata-se de uma *disk image* que inclui uma aplicação com a JVM 1.7 da Oracle - weka-3-6-11-oracle-jvm.dmg - e a segunda trata-se também de uma *disk image* que contém uma aplicação compatível com a JVM 1.6 da Apple - weka-3-6-11-apple-jvm.dmg.

- **Outras plataformas (Linux, etc.)**

Para este sistema operativo existe um ficheiro zip - weka-3-6-11.zip - para *download*. Depois de fazer *unzip*, irá ser criado um diretório com o nome weka-3-6-11. Para o executar é necessário mudá-lo escrevendo: java –Xmx1000m –jar weka.jar. É necessário que o java seja instalado no sistema para que tudo funcione.

6.3 Instalação do KNIME

Antes de instalar o KNIME é necessário verificar que sistema de operativo se vai usar na máquina onde se vai instalar o *software*. A versão do KNIME a instalar é a KNIME *Analytics Platform*. Para além da plataforma, é possível instalar também o KNIME SDK versão 2.10.4 que se encontra disponível para qualquer sistema operativo.

A versão do KNIME *Analytics Platform* está concebida para utilizadores finais proporcionando todas as funcionalidades necessárias a utilizar na ferramenta e extendê-la a outras *packages* desenvolvidas por terceiros. As informações e imagens seguintes estão disponíveis na página http://www.knime.org/knime.

- **Windows**

Estão disponíveis para os sistemas de 32 e 64 bits quatro hipóteses de *download*: instalador; instalador + todas as extensões grátis; *self-extracting archive* e ficheiro Zip.

  - **Iniciar**

Para iniciar o KNIME basta fazer duplo clique no ícon que se encontra no ambiente de trabalho e a aplicação é iniciada.

Figura 6.7 - KNIME a iniciar

Depois de inciado, o KNIME exige a designação de um *workspace*, que tem a função de armazenar todos os projetos num arquivo. Depois de concluído todo o processo o KNIME está pronto a ser utilizado.

- **Linux**

Estão disponíveis para os sistemas de 32 e 64 bits duas hipóteses de *download*: instalador e instalador + todas as extensões grátis.

- **Mac**

Estão disponíveis para os sistemas de 64 bits existem duas hipóteses de *download*: instalador para Mac 10.7 ou superior e instalador + todas as extensões grátis para Mac 10.7 ou superior.

- ***SKD – Software Development Kit***

O seu SDK (*software development kit*) proporciona um JRE (Java *Runtime Environment* - Oracle Java 1.7.0._60), baseado no Eclise Indigo (3.7.2) que facilita o desenvolvimento próprio de *nodes* pelo utilizador de forma simplificada devido às suas extensões. A sua instalação é facultativa, uma vez que esta aplicação tem propósitos muito específicos.

- ***Update site***

Os *plug-ins* adicionais do KNIME podem ser obtidos pelo KNIME *update site*. Este por sua vez pode ser acedido via http://www.knime.org/update/2.10/ ou pelo *download* de um ficheiro Zip disponível na sua página.

- ***Datasets***

O KNIME disponibiliza um conjunto pequeno de *datasets* para *download* na sua página.

- **Extensões**

Para além de *plug-ins* é possível também adicionar extensões ao KNIME. Este tem parceiros nas áreas da ciência como o ChemAxon e Infocom que facultam as extensões intituladas *"Free Marvin Chemistry Extensions"*. Estas incluem *nodes* como *Marvin Sketch*, para desenhar estruturas químicas, questões e reações; *Marvin View*, para visualizar as estruturas químicas individuais ou múltiplas; *Marvin Space*, para visualizar em 3D moléculas, proteínas etc. e o *MolConverter* para conversão de estruturas químicas em outros formatos. Para além destes, o KNIME apresenta compatibilidade de sistema com outros parceiros que facultam as seguintes extensões: *MOE; Jchem/Marvin; Schrodinger Suite; Korilog; Pervasive RushAnalytics and RushAccelerator; Dymatrix e BioSolveIT.*

6.4 Instalação do Orange

Antes de instalar o Orange é necessário verificar que sistema de operativo se vai usar na máquina onde se vai instalar o *software.* A versão do Orange a instalar é a 2.7.

Relativamente aos requisitos, estes dependem do sistema operativo a usar, no entanto de forma geral, os passos do processo de instalação são parecidos variando só em algumas componentes. As informações e imagens seguintes estão disponíveis na página http://orange.biolab.si/.

- **Windows**

Estão disponíveis dois produtos para *download,* para este sistema operativo: *Full package e Pure Orange*. O *Full package* é recomendado para quem vai instalar o Orange pela primeira vez, incluindo todas as bibliotecas necessárias. O *Pure Orange* é uma versão que serve para aqueles que vão atualizar a sua versão relativamente á anterior (2.6), fazendo atualização das bibliotecas necessárias e instalando por cima da versão anterior.

- **Iniciar**

Para iniciar o Orange basta fazer duplo clique no ícon que se encontra no ambiente de trabalho e a aplicação é iniciada.

Figura 6.8 - Orange a iniciar

- **Mac**

Está disponível para *download* um *bundle* universal que contém todas as funcionalidades e *features* para um utilizador avançado. Para instalar o Orange como uma PyPi (*Python Package Index*) package easy_install/pip: easy_install numpy && easy_install orange.

- **Linux**

Está disponível o *download* dos seguintes elementos para construção a partir da fonte *(builing from the source*): *nightly packed sources; archive of selected packed sources* e Orange 2.7 *code repository* através do GitHub. Para instalar e executar o Orange pode usar-se o setup.py que requer GCC, *Python* e *numpy development headers*. Para isso deve extrair-se as componentes da *nightly sources* e executar:

```
python2 setup.py build
sudo python2 setup.py install
```

Assim vai instalar-se também o *script* orange-canvas. Para utilizar o Orange Canvas pela linha de comandos deve executar-se: python2 setup.py install –user.

- **Add-ons**

Estão disponíveis alguns *add-ons* que se podem instalar que são compatíveis a todos os sistemas operativos: Orange-Bioinformatics (version 2.5.37); orangecontrib.earth (version 0.1.3); Orange-ModelMaps (version 0.2.8); Orange-Multitarget (version 0.9.3); Orange-Network (version 0.3.4); Orange-NMF (version 0.1.2); Orange-Reliability (version 0.2.14); Orange-Text (version 1.2a1) e Orange-Textable (version 1.4.2).

6.5 Avaliação técnica das ferramentas

Após a instalação das quatro ferramentas de *data mining* escolhidas para análise e teste prático, vamos agora avaliar os aspetos técnicos de cada uma delas.

Uma vez que já foi feita uma análise relativamente ao estado da arte das ferramentas *open source* de DM, vamos agora comparar as 4 escolhidas. A comparação é fundamentada com duas abordagens: uma resultante da revisão bibliográfica e outra e comprovada pela instalação das ferramentas, resultante do teste prático e uso das mesmas.

A Tabela 6.1 apresenta uma comparação entre as ferramentas de DM ao nível da linguagem de programação utilizada no desenvolvimento das mesmas.

Tabela 6.1 - Linguagem de desenvolvimento

Linguagem de programação	**RapidMiner**	**Orange**	**Weka**	**Knime**
Java	✓	x	✓	✓
C++	x	✓	x	x
Python	x	✓	x	x

É possível avaliar que à exceção da ferramenta Orange todas as ferramentas são desenvolvidas em Java. A ferramenta Orange é desenvolvida em C++ contendo também *Python*. Esta tabela permite-nos concluir que a linguagem Java é claramente a mais escolhida para desenvolvimento de ferramentas. Esta questão deve-se á facilidade que esta proporciona na integração e desenvolvimento de novas funcionalidades.

A Tabela 6.2 apresenta os sistemas operativos compatíveis com as ferramentas em estudo.

Tabela 6.2 - Sistemas operativos

Sistemas operativos	RapidMiner	Orange	Weka	Knime
Windows	✓	✓	✓	✓
Linux	✓	✓	✓	✓
Mac	✓	✓	✓	✓

A Tabela 6.2 permite identificar que todas as ferramentas são compatíveis com os sistemas operativos *Windows, Linux e Mac.*

De seguida, na Tabela 6.3 são exibidos alguns aspetos gerais, importantes relativamente a cada uma das ferramentas em análise, nomeadamente o ano de desenvolvimento das mesmas, o ano de desenvolvimento da última versão estável, o tipo de licença que usam e se têm ou não uma versão comercial disponível.

Tabela 6.3 - Outros aspetos relevantes

Outros aspetos	RapidMiner	Orange	WEKA	KNIME
Licenças	AGPL	GPL	GPL	GPLv3
Ano de desenvolvimento	2007	1997	1997	2006
Ano da última versão	2013	2014	2014	2014
Última versão estável	6.0	2.7	3.6.11	2.10.1
Versão comercial	Sim	Não	Não	Sim

Segundo a Tabela 6.3 todas as ferramentas possuem uma licença do tipo GPL, exceto o RapidMiner que usa a AGPL em específico, cujo propósito é ser uma licença minimamente modificada da GPL, na disponibilização do código. Relativamente aos anos de desenvolvimento podemos concluir que as mais recentes são o RapidMiner e o KNIME. No entanto, o ano de desenvolvimento do RapidMiner remete para a data em que este tomou o seu nome atual, já existindo com o nome YALE desde 2001. Relativamente ao KNIME, este começou a ser desenvolvido na Universidade de Konstanz em 2004, embora só tenha disponibilizado a sua primeira versão em 2006. Tanto o Orange como a WEKA são ferramentas com um nível de maturidade maior, sendo que o Orange foi desenvolvido inicialmente com outro nome – ML* em 1996 e, em 1997 com a integração dos vínculos em *Python* mudou o seu nome para Orange. Quanto á WEKA poderá afirmar-se como a ferramenta mais antiga, datando a sua primeira versão de 1993, também com outro nome na altura - TCL/TK - representada por uma ferramenta *front-end* desenvolvida em C. Em 1997 é totalmente reescrita em Java e adota o seu nome atual. No que diz respeito aos anos das últimas versões,

podemos concluir que todas as ferramentas se encontram atuais, pois estas não tem mais de um ano de existência. Quanto a versões comerciais, apenas o RapidMiner e o KNIME as integram.

A Tabela 6.4 apresenta os resultados para as quatro ferramentas, relativamente aos tipos de bases de dados suportados.

Tabela 6.4 - Bases de dados suportadas

Bases de dados suportadas	RapidMiner	Orange	WEKA	KNIME
HSQLDB	✓	x	✓	x
Ingres	✓	x	x	x
JDBC	✓	x	✓	✓
Microsoft SQL Server	✓	x	✓	✓
MySQL	✓	✓	✓	✓
ODBC	✓	x	✓	x
Oracle	✓	x	✓	✓
PostgreSQL	✓	x	✓	✓
SQLite	✓	x	✓	✓
Sybase	✓	x	x	x
Outras	✓	x	x	✓

A Tabela 6.4 permite identificar que o RapidMiner é a única ferramenta compatível com todas as bases de dados inlcuídas na tabela e o Orange a única que apenas integra uma das bases de dados – MySQL (Chen, Williams, & Xu, 2007). Relativamente ao WEKA e KNIME ambas incluem a maioria das bases de dados apresentadas.

A Tabela 6.5 apresenta os resultados relativamente aos recursos documentais de suporte disponibilizados por cada ferramenta.

Tabela 6.5 - Documentação disponível

Documentação	RapidMiner	Orange	WEKA	KNIME
Documentação disponível com a instalação	x	✓	✓	✓
Documentação disponível na página	✓	✓	✓	✓
Exemplos próprios (*datasets*)	✓	✓	✓	✓
Tutoriais na aplicação	✓	✓	x	x

No que diz respeito á documentação, a Tabela 6.5 permite identificar que todas as ferramentas integram documentação relativa á plataforma e seus constituintes. A localização da documentação encontra-se na maioria dos casos, na página de suporte de cada ferramenta em análise.

A Tabela 6.6 apresenta os resultados relativamente aos formatos dos ficheiros de leitura compatíveis com cada ferramenta.

Tabela 6.6 - Ficheiros compatíveis

Ficheiros compatíveis	RapidMiner	Orange	WEKA	KNIME
.AML	✓	x	x	x
.ARFF	✓	✓	✓	✓
ASCII files	x	x	x	✓
.BASKET	x	✓	x	x
Binary files	✓	x	✓	x
.CSV	✓	✓	✓	✓
.DAT	x	x	✓	x
.DATA	x	x	✓	x
Database (SQL database)	✓	x	x	x
Excel	✓	x	x	x
LibSVM	x	x	✓	x
Microsoft Access Database (.mdb)	✓	x	x	x
.NAMES	x	x	✓	x
PMML	x	x	✓	✓
SAS files	✓	x	x	x
SPSS files (.sav)	✓	x	x	x
.TAB	x	✓	x	x
URL	✓	x	✓	✓
.XML	✓	x	x	✓
.XRFF	x	x	✓	x

Segundo a Tabela 6.6 é possível concluir que apenas os formatos .ARFF e .CSV são compatíveis com todas as ferramentas. A ferramenta com mais ficheiros compatíveis é o RapidMiner.

A Tabela 6.7 apresenta uma comparação entre todas as ferramentas, relativamente a um conjunto de funcionalidades de *data mining*.

Tabela 6.7 - Funcionalidades de data mining

Funcionalidades	RapidMiner	Orange	WEKA	KNIME
Árvores de Decisão	Sim	Sim	Sim	Sim
Regras de Associação	Sim	Sim	Sim	Sim
Avaliação	Sim	Sim	Sim	Sim
Clustering	Sim	Sim	Sim	Sim
Redes Bayesianas	Sim	Sim	Sim	Sim
Redes Neuronais	Sim	Sim	Sim	Sim
SVM	Sim	Sim	Sim	Sim

A Tabela 6.7 permite visualizar que as ferramentas em estudo possibilitam a execução de todas as funcionalidades mencionadas. É possível verificar que as ferramentas escolhidas para análise são muito completas, no que diz respeito a funcionalidades de *data mining*.

A Tabela 6.8 apresenta uma comparação entre todas as ferramentas, relativamente a um conjunto de tarefas de pré-processamento de dados. As tarefas em avaliação foram selecionadas por serem mais utilizadas e mais comuns entre as ferramentas.

Tabela 6.8 - Tarefas de pré-processamento de dados

Tarefas de pré-processamento	RapidMiner	Orange	WEKA	KNIME
Estatística				
Medidas de dispersão de dados	✓	✓	✓	✓
Medidas de tendência central	✓	✓	✓	✓
Limpeza de dados				
Deteção de outliers	✓	✓	✓	✓
Filtragem (Filtering)	✓	✓	✓	✓
Ordenação (Sorting)	✓	✓	✓	✓
Preenchimento (Fill)	✓	✓	✓	x
Substituição (Replace)	✓	✓	✓	✓
Redução e transformação de dados				
Agregação	✓	✓	x	✓
Discretização	✓	✓	✓	✓
Fundir (Merge)	✓	✓	✓	✓
Seleção de atributos	✓	✓	✓	✓

Tarefas de pré-processamento	RapidMiner	Orange	WEKA	KNIME
Visualização de dados				
Gráficos (graph view)	✓	✓	✓	✓
Tabelas (table view)	✓	✓	✓	✓
Texto (text view)	✓	✓	✓	✓

No conjunto de tarefas de pré-processamento em avaliação na Tabela 6.8 podemos concluir que não existe uma discrepância de resultados. Todas as ferramentas integram a maioria das tarefas em avaliação e integram ainda outras tarefas que não são descritas na tabela, específicas em cada ferramenta.

A Tabela 6.9 apresenta uma comparação relativamente às várias formas de visualização gráfica de dados que cada ferramenta oferece.

Tabela 6.9 - Visualização gráfica de dados

Tipos de gráficos	RapidMiner	Orange	WEKA	KNIME
Andrew curves	✓	x	x	x
Block graphs	✓	x	x	x
Conditional box plot	x	x	x	✓
Dendogram	✓	✓	x	✓
Distribuition	✓	✓	x	x
Gráficos 3D	✓	x	x	x
Gráficos circulares (Pie chart)	✓	x	x	✓
Gráficos de barras	✓	x	x	x
Gráficos de bolhas	✓	x	x	x
Gráficos de densidade (density)	✓	x	x	x
Gráficos de dispersão (Scatter plot)	✓	✓	✓	✓
Gráficos de quantis (Box plots)	✓	x	x	✓
Gráficos de sticks	✓	x	x	x
Histogramas	✓	✓	✓	✓
Lift chart	x	x	x	✓
Line plot	x	x	x	✓
Mosaic display	x	✓	x	x

Tipos de gráficos	RapidMiner	Orange	WEKA	KNIME
Paralelas coordenadas (Parallel)	✓	✓	x	✓
Pareto	✓	x	x	x
Quartile	✓	x	x	x
Raddar Plot Appender	x	x	x	✓
Ring	✓	x	x	x
ROC curves	✓	✓	✓	✓
Rule 2D view	x	x	x	✓
Sieve diagram	x	✓	x	x
SOM	✓	✓	x	x
Spark Line Appender	x	x	x	✓
Surface 3D	✓	x	x	x
Survey	✓	✓	x	x
Venn diagram	x	✓	x	x

A Tabela 6.9 permite visualizar que a ferramenta com mais possibilidades de visualização gráfica de dados é o RapidMiner e a com menos possibilidades de gráficos é a WEKA. Os tipos de visualização gráfica de dados partilhados pelas quatro ferramentas são os gráficos de dispersão (*scatter plots*), os histogramas e as curvas ROC.

Nas tabelas seguintes são enumerados alguns algoritmos de *data mining* que se encontram integrados nas quatro ferramentas em estudo. O símbolo "x" indica os algoritmos não suportados pela ferramenta e o símbolo ✓ representa os algoritmos suportados. A cinza estão representados os algoritmos integrados na ferramenta mas com uma variante do algoritmo ou com nome diferente.

A Tabela 6.10 apresenta os resultados relativamente aos algoritmos de associação integrados em cada ferramenta.

Tabela 6.10 - Algoritmos de associação

Algoritmos de Associação	RapidMiner	Orange	WEKA	KNIME
Apriori	x	x	✓	✓
Association Rules	✓	✓	x	✓
DICE	x	x	x	✓
Filtered Associator	x	x	✓	x
FPGrowth	✓	x	✓	✓

Algoritmos de Associação	RapidMiner	Orange	WEKA	KNIME
Fuzzy Rules (RecBF-DDA)	x	x	x	✓
Generalized Sequential Patterns	✓	x	✓	x
Itemsets	x	✓	x	x
JiM	x	x	x	✓
Predictive Apriori	x	x	✓	x
Relim	x	x	x	✓
SaM	x	x	x	✓
TANIMOTO	x	x	x	✓
Tertius	x	x	✓	x

Podemos visualizar que a ferramenta com mais algoritmos de associação incluídos é o KNIME com 9 algoritmos. De seguida a WEKA com 6 algoritmos e o RapidMiner e Orange têm 3 e 2 respetivamente.

Uma vez que os algoritmos de classificação e regressão incluídos nas ferramentas em estudo são numerosos, e para evitar que a leitura da tabela seja exaustiva, os resultados completos encontram-se disponíveis em anexo. De seguida, apresentamos apenas os algoritmos que pelo menos duas ferramentas em estudo incluem.

A Tabela 6.11 apresenta os resultados relativamente aos algoritmos de classificação integrados em pelo menos duas ferramentas.

Tabela 6.11 - Algoritmos de classificação

Algoritmos de classificação	RapidMiner	Orange	WEKA	KNIME
Adaboost	✓	x	✓	✓
Bagging	✓	✓	✓	✓
Decision Stump	✓	x	✓	x
Decision Tree	✓	x	x	✓
Gaussian Processes	✓	x	✓	x
ID3	✓	x	✓	x
KNN	✓	✓	✓	✓
Lib SVM	✓	x	✓	✓
MetaCost	✓	x	✓	x
Multilayer Perceptron	✓	x	✓	✓

Algoritmos de classificação	RapidMiner	Orange	WEKA	KNIME
Naive Bayes	✓	✓	✓	✓
Neural Net	✓	✓	x	x
Random Forest	✓	✓	✓	x
Random Tree	✓	x	✓	x
Stacking	✓	x	✓	x
SVM	✓	✓	x	✓

Na Tabela 6.11 podemos ver que a ferramenta que integra mais algoritmos é o RapidMiner. De seguida, a WEKA integra 13 dos 15 algoritmos, o KNIME integra 7 e o Orange integra apenas 4.

A Tabela 6.12 apresenta os resultados relativamente aos algoritmos de regressão integrados em pelo menos duas ferramentas.

Tabela 6.12 - Algoritmos de regressão

Algoritmos de regressão	RapidMiner	Orange	Weka	Knime
Linear Regression	✓	✓	✓	✓
Logistic Regression	✓	✓	✓	✓
Polynomial Regression	✓	x	x	✓

Podemos visualizar que os principais algoritmos de regressão são suportados por todas as ferramentas.

Nas 2 tabelas anteriores foram exibidos apenas os resultados de algoritmos de classificação e regressão, que estivessem implementados em pelo menos 2 ferramentas em estudo. Os valores completos referentes aos algoritmos de classificação e regressão em cada ferramenta encontram-se na tabela em anexo no capítulo 10. Seguidamente é possível visualizar os resultados totais de cada ferramenta.

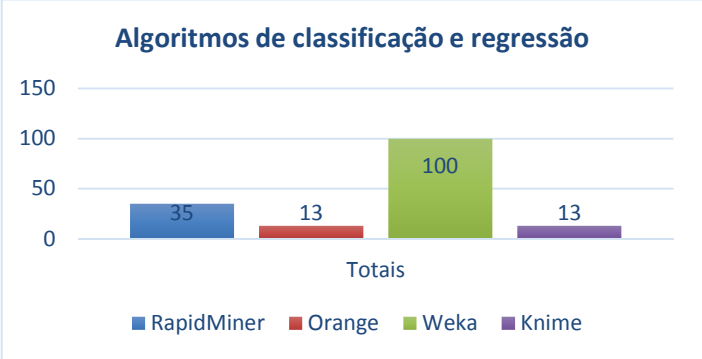

Figura 6.9 - Somatório dos algoritmos de classificação e regressão

É possível concluir que os resultados da tabela 6.11 e 6.12 são ligeiramente diferentes dos apresentados na imagem anterior. A WEKA é a ferramenta que detém maior número de algoritmos de classificação e regressão. Esta questão poderá estar associada com o facto de esta ferramenta ser a mais madura na área da DM e incluir algoritmos implementados há mais tempo. No entanto, considera-se que, apesar do número de algoritmos ser variável nas quatro ferramentas, de uma forma geral, estas implementam os principais algoritmos de classificação e regressão.

A Tabela 6.13 apresenta os resultados relativamente aos algoritmos de *clustering* integrados em cada ferramenta.

Tabela 6.13 - Algoritmos de clustering

Algoritmos de Clustering	RapidMiner	Orange	WEKA	KNIME
Aglomerative Clustering	✓	x	x	x
CLOPE	x	x	✓	x
Cobweb	x	x	✓	x
DBScan	✓	x	✓	✓
EM	✓	x	✓	x
Farthest First	x	x	✓	x
Filtered Clusterer	x	x	✓	x
Flatten Clustering	✓	x	x	x
Fuzzy c-Means	x	x	x	✓
Hierarchical Clusterer	✓	✓	✓	✓
K-Means	✓	✓	x	✓
K-Medoids	✓	x	x	✓
Make Density Bases Clusterer	x	x	✓	x
MDS	x	✓	x	x
OPTICS	x	✓	✓	x
PCA	x	✓	x	✓
Random Clustering	✓	✓	x	x
sIB	x	x	✓	x
Simple K-Means	x	x	✓	x
SOM	x	✓	x	x
SOTA	x	✓	x	✓
Support Vector Clustering (SVC)	✓	x	x	x
Top down Clustering	✓	x	x	x

Algoritmos de Clustering	RapidMiner	Orange	WEKA	KNIME
XMeans	✓	x	✓	x

Na tabela anterior podemos visualizar que a ferramenta WEKA apresenta o maior número de algoritmos de *clustering*. De seguida o RapidMiner e o KNIME com 8 e 7 respetivamente e por fim o Orange com 4. Os únicos algoritmos partilhados pelas quatro ferramentas são o *K-Means* e o *Hierarchical Clustering.*

Após uma avaliação técnica relativamente às características e funcionalidades das quatro ferramentas em estudo, vamos agora a avaliá-las de uma forma mais pessoal e relativamente a característas de usabilidade e suporte.

As *interface*s gráficas de utilizador representam um elemento muito importante, uma vez que fazem a ponte entre o homem e aplicação. Estas devem ser visualmente simples e intuitivas com um aspeto modernizado. Nas imagens seguintes são apresentados vários cenários de processamento, onde é possível visualizar a *interface* gráfica de cada ferramenta em estudo. As imagens são provenientes da página de suporte de cada ferramenta em específico.

Na Figura 6.10 podemos visualizar que o RapidMiner apresenta um *interface* gráfico de utilizador simples e moderno. As análises de dados são configuradas no painel central denominado por *process view.*

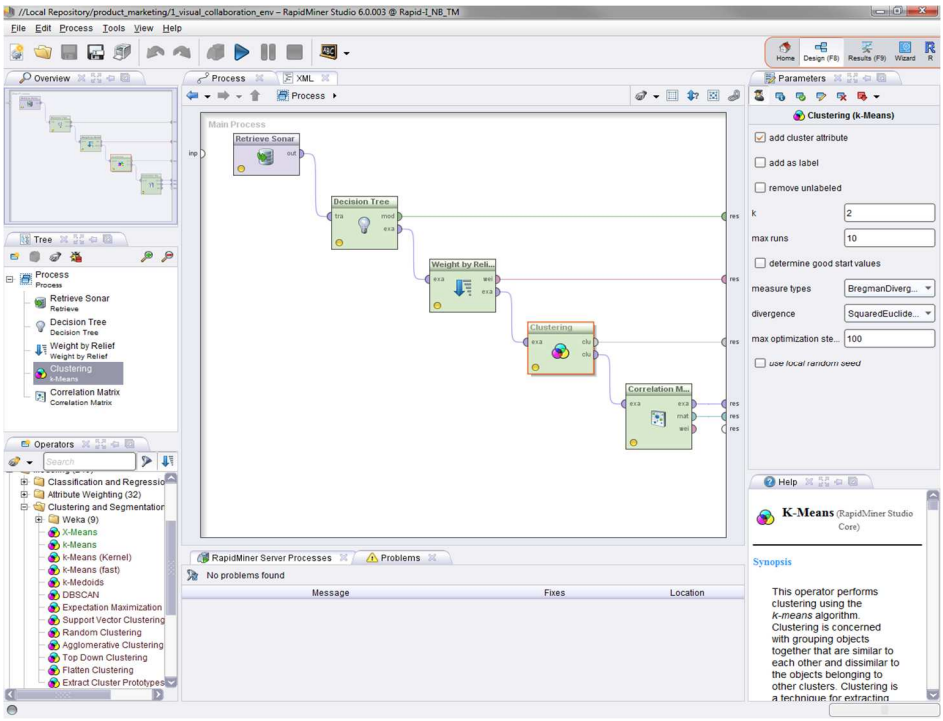

Figura 6.10 - *Interface* gráfica de utilizador do RapidMiner

Cada passo no processo de análise de dados é representado por um operador, que contém um *input* e *output* e que pode comunicar com outros operadores. Os operadores podem receber dados (*input*), e movê-los de forma a gerar modelos nos operadores seguintes. Desta forma é criado um fluxo de dados pelo processo de análise de dados. Para além do *process view* existem outros painéis/ *views* diponiveis:

- *Overview* – representa uma vista minimizada do processo construído no painel *process view.* No caso de processos grandes é possível maximizar a janela para visualizar todos os constituintes.
- Tree – Para utilizadores das versões anteriores do RapidMiner, este tipo de visualização é comum, representando os operadores e os seus "filhos", necessários para o processamento, em forma de árvore.
- *Operators view* – contém os operadores fornecidos pelo RapidMiner. Estes encontram-se agrupados por categorias e para selecionar um deles basta fazer *drag and drop* para o *process view*.
- *Repositories view* – para estruturação e gestão dos processos de análise de dados e usado como fonte de dados.
- *Design perspective* – responsável pela criação, edição e gestão dos processos de análise de dados.
- *Parameters view* – a execução dos operadores disponíveis requer a indicação de alguns parâmetros para funcionamento. A mudança de parâmetros é um fator que influencia os resultados finais.
- *Help and comment view* – responsável por mostrar informações relativas aos operadores e ao processamento. De cada vez que se seleciona um operador é evidenciada neste painel uma descrição breve sobre o mesmo.
- *Problems and log view* – responsável por mostrar o relatar todos os erros de processamento e mensagens de aviso.

Na Figura 6.11 podemos visualizar o carácter moderno do *interface* gráfico de utilizador do Orange, o Orange Canvas. Á semelhança do RapidMiner também possibilita a seleção de funcionalidades através de *drag and drop*. No entanto, em vez de operadores, o Orange utiliza *widgets.*

Os *widgets* representam fluxos de análise de dados que se interligam através de canais. Estes representam o ambiente de programação visual do Orange e podem ser selecionados no painel lateral onde se encontram agrupados por funções. Para alterar os parâmetros ou visualizar resultados de cada *widget* usado basta clicar duas vezes sobre o ícon.

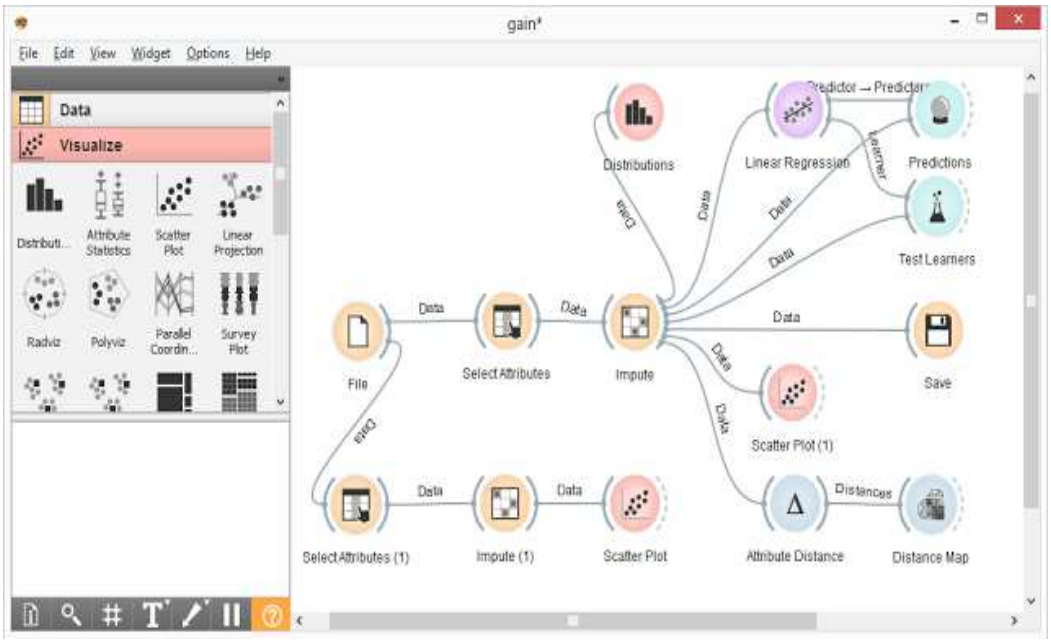

Figura 6.11 - *Interface* gráfico de utilizador do Orange

Na Figura 6.12 podemos visualizar o *interface* gráfico de utilizador da WEKA, o WEKA *Explorer*, que se trata de um ambiente específico para exploração de dados.

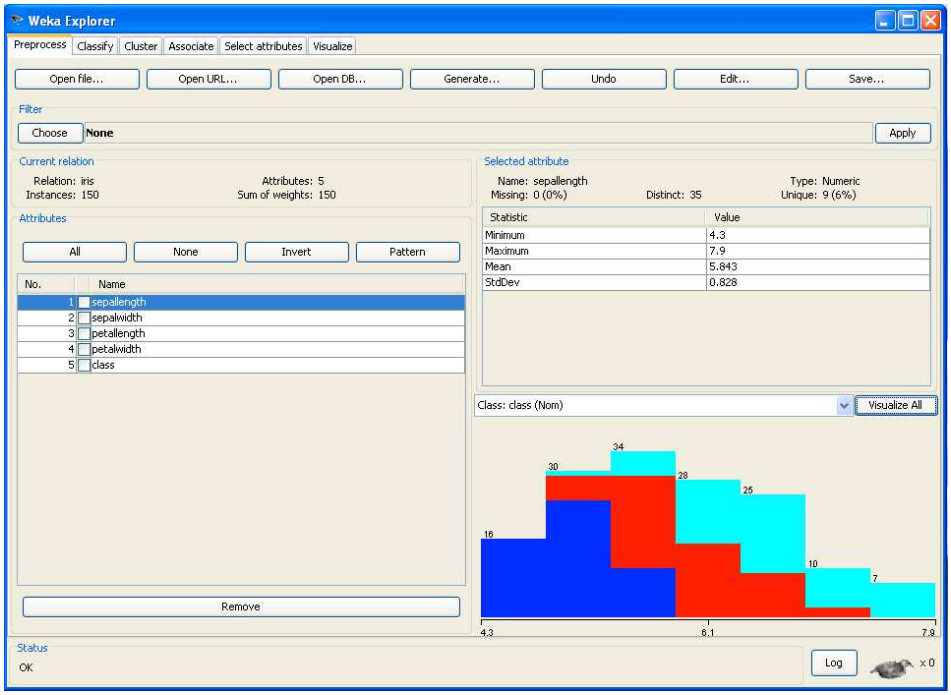

Figura 6.12 - *Interface* gráfico de utilizador da WEKA

Este *interface* gráfico é diferente de todos os outros, pois localiza os seus algoritmos e ferramentas em separadores específicos. Os separadores estão localizados no topo da janela e são os seguintes: *Preprocess; Classify; Cluster; Associate; Select Attributes* e *Visualize.* As análises de dados são configuradas em cada separador específico de acordo com as tarefas que se pretendem implementar.

Na *Figura 6.*13 podemos visualizar o *interface* gráfico de utilizador do KNIME. Este *interface* é visualmente mais parecido com o do RapidMiner e o do Orange, tendo uma apresentação moderna e simples.

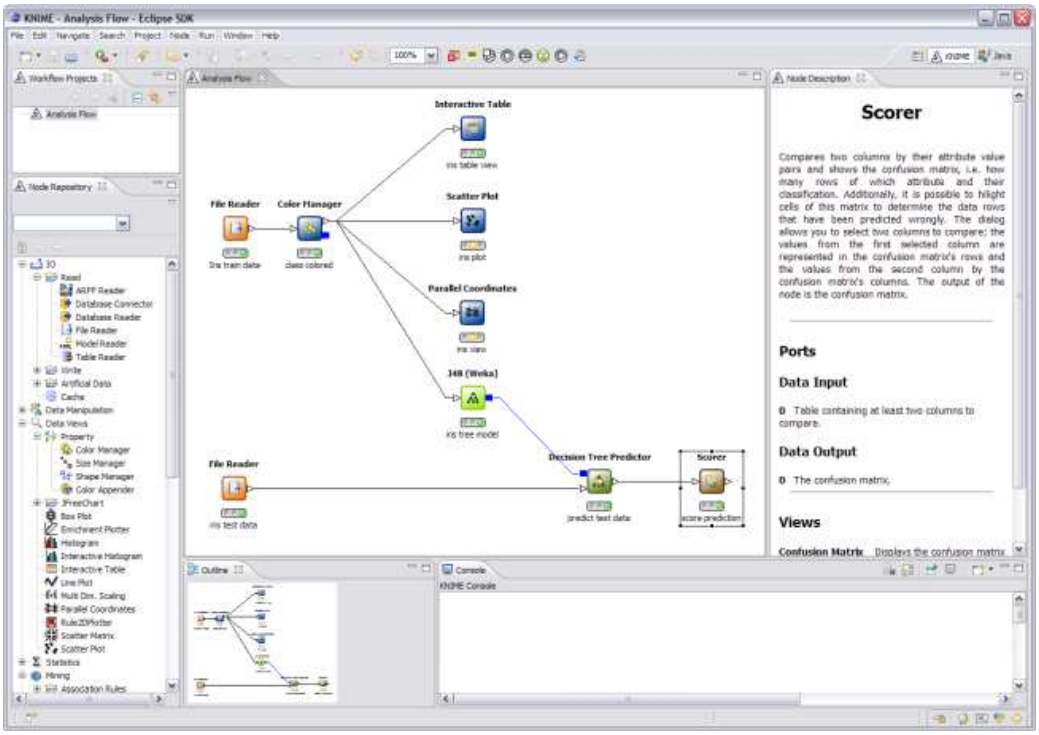

Figura 6.13 - *Interface* gráfico de utlizador do KNIME

As análises de dados são configuradas no painel central denominado por *Workflow editor* e representados por *nodes.* Á semelhança dos operadores do RapidMiner, os *nodes* recebem informação e enviam-na para outros *nodes* dando origem a um fluxo de dados. Para além *Workflow editor* existem outros painéis/ *views* diponiveis:

- *Workflow projects* – Todos os fluxos de dados são visualizáveis neste painel.
- *Favorite Nodes* – Este painel mostra os *nodes* mais usados ou preferidos pelo utilizador.
- *Node Repository* – Responsável por armazenar todos os *nodes* por categorias. Os *nodes* podem ser adicionados do repositório para o *workflow editor* com um simples *drag and drop*.

- *Outline* – Este painel proporciona uma vista geral do *workflow*, mesmo que este não esteja a ser exibido na totalidade pelo *workflow editor*, e por isso pode ser usado para navegação.
- *Node description* – Responsável por mostrar informação relativa ao *node* que se encontra selecionado. Descreve as opções e visualizações disponíveis, informações sobre o *input* e resultados esperados.
- *Console* – As mensagens de erro e aviso são exibidas neste painel, de forma a informar o utilizador do estado do processo de análise de dados.

A Tabela 6.14 apresenta uma avaliação pessoal de alguns parâmetros com uma classificação baseada numa escala de 1 a 5, onde 1 – Muito mau; 2 – Mau; 3 – Razoável; 4 – Bom e 5 Muito bom.

Tabela 6.14 - Classificação das ferramentas (1-5)

Avaliação geral (1-5)	**RapidMiner**	**Orange**	**WEKA**	**KNIME**
Documentação disponível	5	3	4	5
Facilidade de aquisição	5	5	5	5
Facilidade de instalação	5	5	5	5
Facilidade de utilização	4	5	5	4
Informação disponível	5	3	5	5
***Interface* gráfico de utilizador**	5	4	4	5
Página de suporte	5	4	4	5
Velocidade	5	5	5	5
Total	39	34	37	39

A imagem seguinte foi elaborada a partir da tabela anterior e permite simplificar visualmente a atribuição dos valores anteriores através de um histograma.

Na tabela e imagens anteriores podemos concluir que as ferramentas que mais se destacam relativamente a um conjunto de características de usabilidade e suporte são o RapidMiner e KNIME. No entanto, as outras duas ferramentas apresentam também valores equilibrados não existindo grande discrepância de resultados.

6.6 Conclusão da avaliação

Em suma, depois deste conjunto de conclusões práticas sobre as ferramentas de *data mining* em análise, consideramos que estas são muito equilibradas em termos de funcionalidades e

características. De uma forma geral, as ferramentas possuem quase as mesmas componentes implementadas, embora exibidas em cada uma de forma ligeiramente diferente.

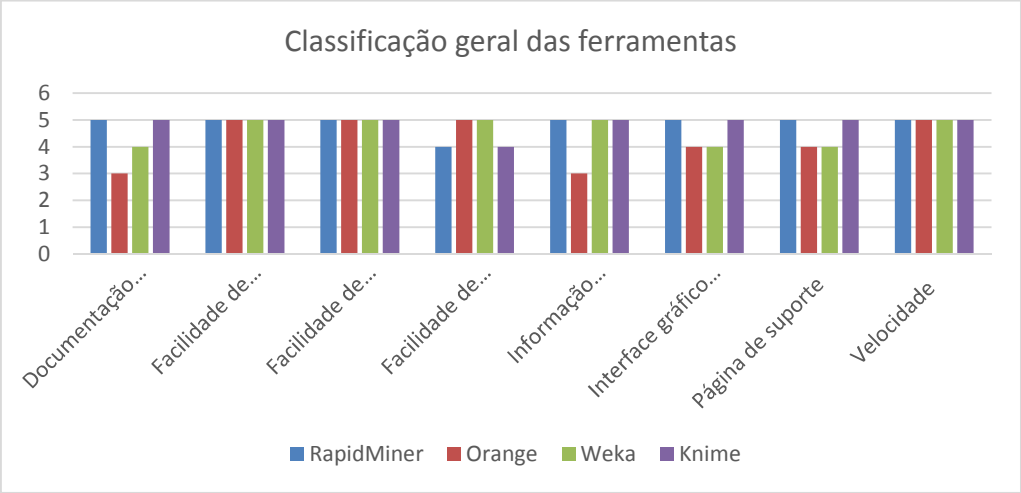

Figura 6.14 - Classificação geral das ferramentas

As principais conclusões retiradas da análise às tabelas e imagens anteriormente apresentadas são as seguintes:

- A WEKA e a Orange são ferramentas completamente *open source*, na medida em que só disponibilizam uma versão para a comunidade;
- Relativamente ao suporte de bases de dados e ficheiros, o RapidMiner é a ferramenta com mais opções integradas;
- Todas as ferramentas possibilitam várias opções de visualização gráfica, no entanto a ferramenta melhor equipada é o RapidMiner e menos equipada é a WEKA;
- As tarefas de pré-processamento são parte integrante de todas as ferramentas em estudo;
- Todas as ferramentas disponibilizam as principais funcionalidades de *data mining*;
- Algumas ferramentas integram mais algoritmos que outras, embora todas incluam os algoritmos principais de *data mining*.

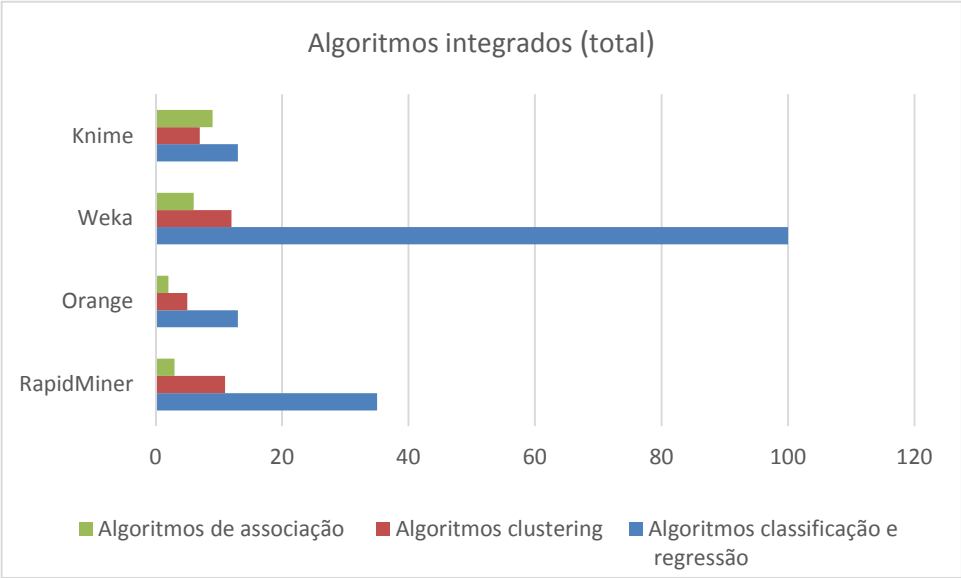

Figura 6.15 – Total de algoritmos integrados

Na Figura 6.15 podemos ver os valores totais obtidos, relativamente aos algoritmos integrados pelas ferramentas em comparação. Apesar de todas incluírem os algoritmos de *data mining* principais, podemos concluir que a WEKA é a ferramenta que integra mais algoritmos de classificação e regressão, com um valor muito superior às outras ferramentas. Relativamente aos algoritmos de *clustering*, os resultados são bastante equilibrados nas quatro ferramentas. E por último, os algoritmos de associação são os que se encontram em menor número nas aplicações em estudo.

No capítulo seguinte vamos usar *datasets* reais para poder testar as ferramentas a um nível mais prático e verificar resultados de *performance*, precisão e velocidade de processamento.

7 IMPLEMENTAÇÃO PRÁTICA

Após os resultados e as conclusões da avaliação técnica obtidos no capítulo anterior, vamos agora avaliar as quatro ferramentas de uma forma prática. A avaliação prática vai incidir no processamento de um conjunto de dados pelas quatro ferramentas de *data mining* em análise, onde as tarefas de classificação, regressão, associação e *clustering* vão ser o alvo de comparação de resultados. O objetivo é usar os mesmos algoritmos nas quatro ferramentas, na avaliação e análise do mesmo conjunto de dados (*datasets)*.

Uma vez obtidos os resultados, vamos compará-los e verificar quais as ferramentas que emitem melhores resultados, ao nível de precisão e de velocidade, quando expostas ao mesmo ambiente de processamento.

Posteriormente são referidas as conclusões do estudo, onde esperamos obter resultados que evidenciem o desempenho superior de uma ou mais ferramentas, passando essas a representar uma solução preferencial a usar num projeto real.

7.1 Conjuntos de dados (datasets)

Os *datasets* a testar são extraídos do conhecido repositório *UCI Machine Learning Repository,* disponível na *web* em http://archive.ics.uci.edu/ml/. Para a tarefa de classificação vão ser testados os seguintes *datasets*:

- Dataset 1 - Renovação de contratos de trabalho (427 instâncias, 12 atributos);
- Dataset 2 – Banco (4621 instâncias, 17 atributos);
- Dataset 3 – Adult (48842 instâncias, 14 atributos).

Para o *clustering* o seguinte *dataset*:

- Dataset 1 – Dresses Attribute Sales (501 instâncias, 13 atributos);

Para a regressão o seguinte *dataset*:

- Dataset 1 - Forest Fires (517 instâncias, 13 atributos);

Para a associação o seguinte *dataset*:

- Dataset 1 – Acute Inflammations (120 instâncias, 6 atributos);

Antes da análise dos dados propriamente dita foi feita uma verificação de todos os *datasets* por forma a introduzir algumas correções pontuais na uniformização dos dados. Relativamente aos *datasets* que apresentavam algumas instâncias com valores nulos ou em branco foi aplicado um critério de preenchimento simples, uma vez que o nosso objetivo é apenas comparar os resultados finais de processamento. O critério foi o seguinte: atributos em falta do tipo nominal foram

preenchidos com o valor mais frequente da classe a que pertence o *record*, p.ex. se estiver em falta o preenchimento do atributo "país nativo" e o "país nativo" mais frequente da "classe: <=50" for "Estados Unidos", preencher como "Estados Unidos."

Relativamente á parametrização dos vários algoritmos a ser testados seguiram-se os valores padrão em cada uma ferramenta, sendo apenas necessário fazer alguns ajustes. Na parametrização da validação foram usados os seguintes critérios: validação: *cross-validation*; *folds*: 10 e *sampling type*: *stratified sampling.*

7.2 Classificação

Como abordamos anteriormente, o objetivo da classificação de atributos é encontrar uma função capaz de associar um conjunto de valores de atributos a uma classe específica. Para avaliar o desempenho dos algoritmos na descoberta dessa função, vamos usar três *datasets* diferentes. O objetivo é criar diferentes cenários de processamento, através de conjuntos de dados onde os atributos e instâncias têm dimensões diferentes. Na tabela seguinte são apresentados os algoritmos em comparação.

Tabela 7.1 - Algoritmos de classificação a comparar

Algoritmos de classificação	RapidMiner	Orange	WEKA	KNIME
Decision Tree	✓	**Classification Tree**	**ADTree**	✓
KNN	✓	✓	**KStar**	✓
Neural Network	**Neural Net** **Auto MLP** **Perceptron**	**Neural Network**	✓	**PNN**
Naïve Bayes	✓	✓	✓	✓
SVM	✓	✓	**libSVM**	✓

Os algoritmos escolhidos para análise são considerados os mais importantes e também os que se encontram incluídos na maioria das ferramentas. Em alguns casos, os algoritmos em análise não se encontram integrados nas ferramentas, mas existem algumas variantes dos mesmos (visíveis na tabela a cinza), que também serão testados.

Depois de preparados os *datasets* e o conjunto de algorimos a testar vamos passar a fase de processamento. Os resultados obtidos são apresentados de seguida.

As tabelas indicam as ferramentas que obtiveram o melhor valor em cada parâmetro, sendo que os valores reais podem ser consultados em anexo, no Capítulo 10.

Na tabela seguinte são apresentados os resultados de processamento do *Dataset* 1 - Renovação de contratos de trabalho (427 instâncias, 12 atributos).

	Parâmetros de análise de performance						
Algoritmos	**Precision**	**Recall**	**F-Measure**	**ROC**	**Accuracy**	**Especificidade**	**Sensibilidade**
Decision Tree	RapidMiner	RapidMiner	WEKA	WEKA	WEKA	RapidMiner	WEKA
KNN	KNIME	KNIME	KNIME	WEKA	KNIME	KNIME	KNIME
Neural Network	Orange	RapidMiner	KNIME	Orange	KNIME	KNIME	RapidMiner
Naïve Bayes	Orange	Orange	Orange	Orange	Orange	RapidMiner	Orange
SVM	WEKA	KNIME	Orange	Orange	Orange	KNIME	KNIME

Tabela 7.2 - Resultados do processamento ao *dataset 1*

Podemos visualizar que não existe uma ferramenta claramente vencedora, estando os resultados bastante dispersos por todas. No entanto, é possível verificar alguns padrões nos resultados ao nível dos algoritmos. Assim conclui-se:

- A ferramenta com melhores resultados na análise do algoritmo *Decision Tree* é a WEKA com 4 dos dos 7 melhores parâmetros (*F-Measure, ROC, Accuracy* e Sensibilidade);
- Nos algoritmos KNN e *Naïve Bayes* as ferramentas KNIME e Orange, respetivamente, obtiveram os melhores resultados, preenchendo 6 dos 7 parâmetros de análise de performance;
- O algoritmo *Neural Network* não apresenta um vencedor claro, havendo resultados dispersos por três ferramentas sem maiorias absolutas;
- Quanto ao algoritmo SVM, os resultados são bastante equilibrados entre as ferramentas KNIME e Orange, preenchendo cada uma 3 dos 7 parâmetros;
- No parâmetro Especificidade podemos visualizar que não existem resultados referentes á ferramenta WEKA, e essa questão deve-se ao facto de esta não incluir a análise deste parâmetro. Pelos mesmos motivos, também para o parâmetro ROC, não existem resultados para a ferramenta KNIME.

Na tabela seguinte são apresentados os resultados de processamento do *Dataset* 2 – Banco (4621 instâncias, 17 atributos).

Tabela 7.3 - Resultados do processamento ao *dataset 2*

	Parâmetros de análise de performance						
Algoritmos	**Precision**	**Recall**	**F-Measure**	**ROC**	**Accuracy**	**Especificidade**	**Sensibilidade**
Decision Tree	Orange	WEKA	WEKA	WEKA	WEKA	RapidMiner	WEKA
KNN	Orange	WEKA	Orange	WEKA	Orange	Orange	Orange
Neural Network	Orange	RapidMiner	Orange	Orange	Orange	RapidMiner	Orange WEKA
Naïve Bayes	KNIME	KNIME	Orange	Orange	Orange	KNIME	KNIME
SVM	RapidMiner	KNIME WEKA	Orange	Orange	RapidMiner	KNIME	KNIME WEKA

Na Tabela 7.3 podemos visualizar que também no *dataset 2* não existe uma ferramenta que claramente se evidencie, estando os resultados bastante dispersos por todas. Assim conclui-se:

- A ferramenta vencedora na análise do algoritmo *Decision Tree* é a WEKA com 5 dos dos 7 melhores parâmetros;
- No algoritmo KNN a ferramenta Orange é que se encontra em maioria com 5 dos 7 parâmetros preenchidos;
- Quanto ao algoritmo *Neural Network* o Orange é também a ferramenta com melhores resultados com 5 dos 7 parâmetros preenchidos;
- Relativamente ao algoritmo Naïve Bayes os resultados são equilibrados entre as ferramentas KNIME e Orange, tendo 4 e 3 parâmentros preenchidos respetivamente cada uma;
- No algoritmo SVM, os resultados são bastante dispersos pelas quatro ferramentas;
- No parâmetro Especificidade podemos visualizar que não existem resultados referentes á ferramenta WEKA, e essa questão deve-se ao facto de esta não incluir a análise deste parâmetro. Pelos mesmos motivos, também para o parâmetro ROC, não existem resultados para a ferramenta KNIME.

Na tabela seguinte são apresentados os resultados de processamento do *Dataset* 3 – Adult (48842 instâncias, 14 atributos).

Tabela 7.4 - Resultados do processamento ao *dataset 3*

	Parâmetros de análise de performance						
Algoritmos	**Precision**	**Recall**	**F-Measure**	**ROC**	**Accuracy**	**Especificidade**	**Sensibilidade**
Decision Tree	KNIME	WEKA	Orange	WEKA	Orange	RapidMiner	WEKA
KNN	Orange	Orange	Orange	WEKA	Orange	Orange	Orange
Neural Network	Orange	RapidMiner	Orange	Orange	Orange	RapidMiner	Orange
Naïve Bayes	KNIME	WEKA	Orange	Orange	Orange	RapidMiner	WEKA
SVM	RapidMiner	KNIME	Orange	RapidMiner	RapidMiner	KNIME	KNIME

Podemos visualizar que no *dataset 3* tambem não existe uma ferramenta claramente vencedora, embora haja alguma predominância da ferramenta Orange. Assim conclui-se:

- No algoritmo *Decision Tree* os resultados são bastante dispersos, embora a ferramenta WEKA seja a melhor com 3 dos 7 pârametros;
- Nos algoritmos KNN, *Neural Network* e *Naïve Bayes* a ferramenta Orange apresenta os melhores resultados, preenchendo 6 dos 7, 5 dos 7 e 3 dos 7 parâmetros de análise de performance respetivamente em cada um;
- Quanto ao algoritmo SVM, os resultados são bastante equilibrados entre as ferramentas KNIME e RapidMiner, preenchendo cada uma 3 dos 7 parâmetros.
- No parâmetro Especificidade podemos visualizar que não existem resultados referentes á ferramenta WEKA, e essa questão deve-se ao facto de esta não incluir a análise deste parâmetro. Pelos mesmos motivos, também para o parâmetro ROC, não existem resultados para a ferramenta KNIME.

Seguidamente são apresentados os resultados do *Dataset* 3 – Adult, relativamente aos tempos de processamento dos algoritmos. Esta questão deve-se ao facto do *dataset* 3 ser de maiores dimensões e, por isso, o tempo de processamento dos algoritmos ser maior. Assim, são apresentados nas tabelas seguintes os 3 piores e 3 melhores resultados de tempo obtidos no processamento nas 4 ferramentas.

Na Tabela 7.5 são apresentados os piores resultados de tempo de processamento obtidos pelas 4 ferramentas. Os resultados são apresentados em horas, minutos e segundos.

Tabela 7.5 - Resultados dos piores tempos do *dataset 3*

	Algoritmo	Tempo (h)	Ferramenta
1º	Neural Net	4h56m	**RapidMiner**
2º	AutoMLP	4h27m	
3º	SVM	1h7m	
1º	KNN	20m	**Orange**
2º	SVM	9m	
3º	Neural Network	21 s	
1º	Multilayer Perceptron	9h30m	**WEKA**
2º	libSVM	4h16m	
3º	Kstar	3h20m	
1º	SVM	9h41m	**KNIME**
2º	PNN	3h40	
3º	Decision Tree	50 s	

Da Tabela 7.5 podemos visualizar que os algoritmos SVM e *Neural Network* (*AutoMLP/ Neural Network/ Multilayer Perceptron/ PNN*) encontram-se presentes em todas ferramentas, como resultado de pior tempo de processamento. Das 4 ferramentas, os piores resultados foram obtidos pelo KNIME, com o algoritmo SVM com 9 horas e 41 minutos. Os resultados de processamento mais baixos foram obtidos pela ferramenta Orange, que no seu conjunto de piores resultados nunca ultrapassou os 20 minutos de processamento.

Na Tabela 7.6 são apresentados os melhores resultados de tempo de processamento obtidos pelas 4 ferramentas. Os resultados são apresentados em horas, minutos e segundos.

Tabela 7.6 - Resultados dos melhores tempos do *dataset 3*

	Algoritmo	Tempo (h)	Ferramenta
1º	Naive Bayes	2s	**RapidMiner**
2º	Decision Tree/ Perceptron	7s	
3º	KNN	6m12s	
1º	Naive Bayes	3s	**Orange**
2º	Classification Tree	8 s	
3º	Neural Network	21 s	
1º	Naive Bayes	2 s	**WEKA**
2º	ADTree	39 s	
3º	KStar	3h20m	
1º	KNN	12 s	**KNIME**
2º	Naive Bayes	14 s	
3º	Decision Tree	50 s	

Na Tabela 7.6 podemos visualizar que os algoritmos *Naïve Bayes* e *Decision Tree* (*ADTree e Classification Tree*) se encontram presentes em todas as ferramentas como resultado de melhores tempos de processamento. O *Naïve Bayes* encontra-se três vezes em 1º lugar e uma em 2º lugar, e o *Decision Tree* três vezes em 2º e uma em 3º lugar. Estes algoritmos são considerados os mais rápidos no processamento de um *dataset* de grandes dimensões (48842 instâncias) nunca ultrapassando os 60 segundos de processamento. Das 4 ferramentas, os melhores resultados foram obtidos pela WEKA e pelo RapidMiner, com o algoritmo *Naïve Bayes* com apenas 2 segundos de processamento.

7.2.1 Conclusões

Em suma, as quatro ferramentas obtiveram resultados de processamento muito equilibrados entre si. Individualmente as ferramentas tiveram *performances* interessantes na análise aos três *datasets*. Quanto ao RapidMiner destacam-se as seguintes:

- No *dataset* 1, os melhores resultados foram obtidos pelo algoritmo *Decision Tree,* onde o RapidMiner atingiu 3 parâmetros de análise de *performance* – *Precision, Recall* e Especificidade;
- No *dataset* 2, os melhores resultados foram obtidos pelos algoritmos *Neural Network* e SVM com os parâmetros *Accuracy* e *Precision*;
- Quanto ao *dataset* 3 foi também o algoritmo SVM que obteve os melhores resultados, com os parâmetros *Precision, ROC* e *Accuracy*;
- Relativamente ao processamento dos algoritmos verificou-se que os piores tempos foram obtidos pelos algoritmos *Neural Net* (4h56m) e *AutoMLP* (4h27).
- O melhor tempo foi obtido pelo algoritmo Naïve Bayes (2s). Na Figura 7.1 é possível visualizar a árvore de operadores construída para obter os resultados de *performance*.

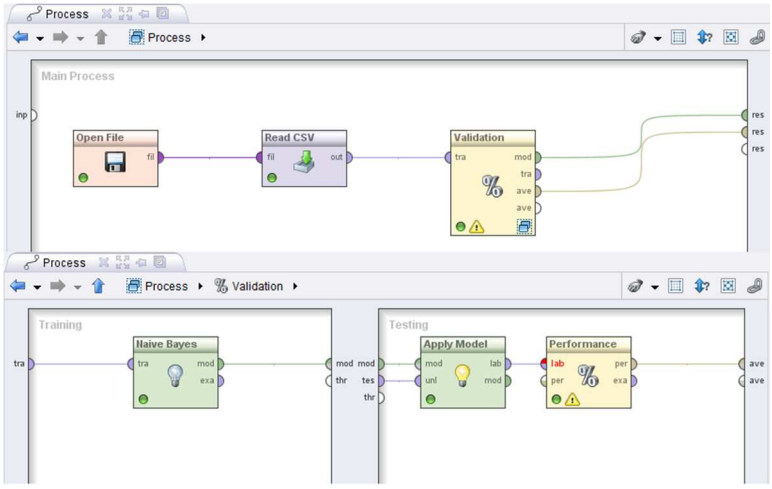

Figura 7.1 - Árvore de operadores para processamento do algoritmo *Naïve Bayes*

Quanto ao Orange destacam-se os seguintes resultados:

- No *dataset* 1, os melhores resultados foram obtidos pelo algoritmo *Naive Bayes* com 6 dos 7 parâmetros de análise de *performance*;
- No *dataset* 2, os melhores resultados foram obtidos pelos algoritmos KNN e *Neural Network*;
- Quanto ao *dataset* 3 os resultados do Orange evidenciaram-se nos algoritmos KNN e *Neural Network e Naive Bayes*;
- Os piores resultados de processamento foram obtidos pelo algoritmo KNN com 20 minutos de processamento e os melhores pelo *Naive Bayes* com 3 segundos. De uma forma geral, conclui-se que esta ferramenta é das mais rápidas, uma vez que o seu pior tempo não ultrapassa os 30 minutos de processamento.

A Figura 7.2 mostra a árvore de *widgets* construída, para análise dos algoritmos de classificação.

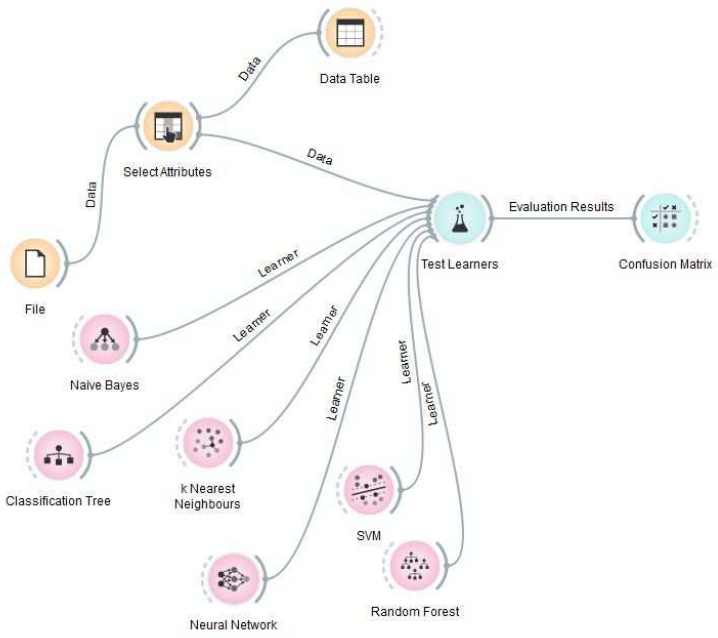

Figura 7.2 - Árvore de *widgets* para algoritmos de classificação

Relativamente á ferramenta WEKA destacam-se os seguintes resultados:

- Nos três *datasets* os melhores resultados foram obtidos pelo algoritmo *Decision Tree*. Esta questão pode relacionar-se com o facto de esta usar uma variante do algoritmo – *ADTree*;
- No parâmetro especificade os resultados não incluem a ferramenta, uma vez que esta não integra a avaliação do mesmo;

- Quanto aos tempos de processamento o algoritmo *Multilayer Perceptron* demorou 9h30m, aproximando-se do pior resultado por 10m (9h40m);
- Á semelhança do RapidMiner o algoritmo com melhor tempo de processamento é o *Naïve Bayes* com 2 segundos. A imagem seguinte mostra o separador de parametrização relativo ao algoritmo em específico.

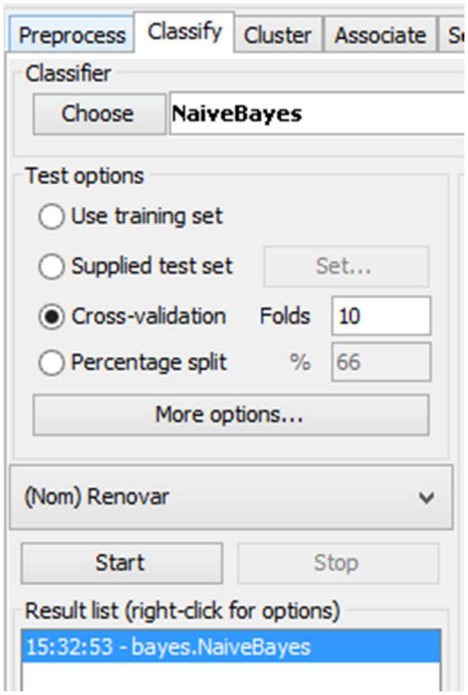

Figura 7.3 - Separador de classificação com algoritmo Naïve Bayes

Quanto ao KNIME destacam-se os seguintes resultados:

- No *dataset* 1, os melhores resultados foram obtidos pelo algoritmo *KNN* com 6 dos 7 parâmetros de análise de *performance*;
- No *dataset* 2, os melhores resultados foram obtidos pelo algoritmo *Naïve Bayes*;
- Quanto ao *dataset* 3 os resultados do SVM foram os melhores com 3 dos 7 parâmetros preenchidos;
- No parâmetro ROC os resultados não incluem a ferramenta, uma vez que esta não integra a avaliação do mesmo para a classe dos atributos;
- O KNIME foi a ferramenta que atingiu o pior resultado de processamento, com o algoritmo SVM a demorar 9h41m. Quanto aos melhores resultados, estes foram atingidos pelos algoritmos *KNN* e *Naïve Bayes*.

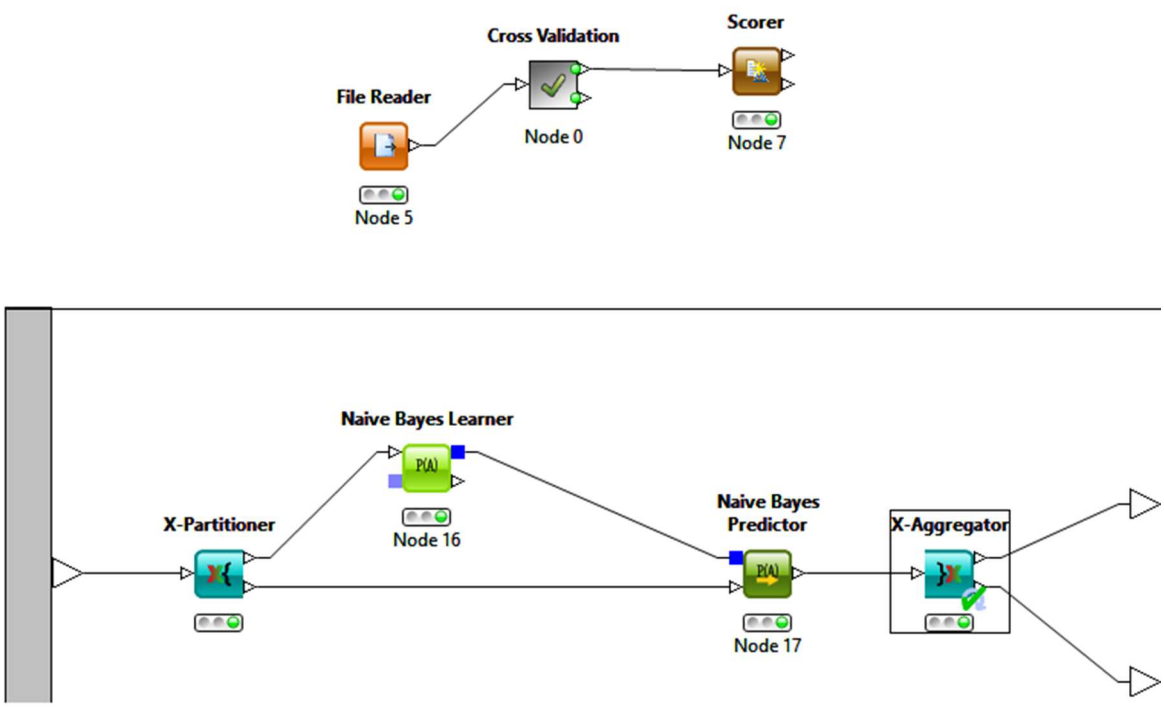

Figura 7.4 - Árvore de widgets para processamento do algoritmo Naïve Bayes

Embora os resultados apontem a ferramenta que obteve melhores valores de *performance* em cada algoritmo, na consulta da tabela em anexo, podemos visualizar que não existem maus resultados. Os valores obtidos são todos muito aproximados e encontram-se equilibrados nas quatro ferramentas.

7.3 Clustering

O objetivo do *clustering* representa a identificação de segmentos, que contêm dados da mesma espécie. Para avaliar o desempenho dos algoritmos na descoberta dessa função, vamos usar um *dataset* com 501 instâncias e 13 atributos. Na tabela seguinte são apresentados os algoritmos em comparação.

Tabela 7.7 - Algoritmos de clustering a comparar

Algoritmos de clustering	**RapidMiner**	**Orange**	**WEKA**	**KNIME**
K-Means	✓	✓	Simple K-Means	✓
DBScan	✓	N.A.	✓	✓
Hierarchical clustering	Agglomerative Clustering	✓	✓	✓

Os algoritmos escolhidos para análise são considerados os mais importantes e também os que se encontram incluídos na maioria das ferramentas. Em alguns casos, os algoritmos em análise não se encontram integrados nas ferramentas, mas existem algumas variantes dos mesmos (visíveis na tabela a cinza), que também serão testados. O vermelho representa o algoritmo que não é suportado pela ferramenta.

Depois de preparados os *datasets* e o conjunto de algorimos a testar vamos passar a fase de processamento. Os valores apresentados de seguida resultam da parametrização dos algoritmos para a geração de dois *clusters*.

Na tabela Tabela 7.8 são apresentados os resultados do processamento do algoritmo K-Means nas quatro ferramentas.

Tabela 7.8 - Processamento do algoritmo K-Means

	Algoritmo K-Means			
	RapidMiner	**Orange**	**WEKA**	**KNIME**
	Coverage			
Cluster 1	264	275	253	256
Cluster 2	236	225	247	244
Accuracy	0,54	0,514	0,394	0,532

Podemos visualizar que a cobertura (*coverage*) obtida para cada *cluster* é relativamente parecida nas quatro ferramentas, sendo que o *cluster* 1 apresenta sempre os valores mais altos relativamente ao *cluster* 2. A ferramenta com maior coverage no *cluster* 1 é o Orange e no *cluster* 2 é a WEKA. Relativamente aos valores da *accuracy* a ferramenta com o melhor resultado é o RapidMiner, seguindo-se o KNIME e o Orange com valores muito aproximados e, com o resultado mais baixo a WEKA. O facto da ferramenta WEKA usar uma variante do algoritmo *K-Means* (*Simple K-Means*) pode estar relacionada com a geração de resultados. Nas imagens seguintes é possível visualizar o separador de *clustering* para parametrização do algoritmo *Simple K-Means* na WEKA e a árvore de *widgets* para o algoritmo K-Means no Orange.

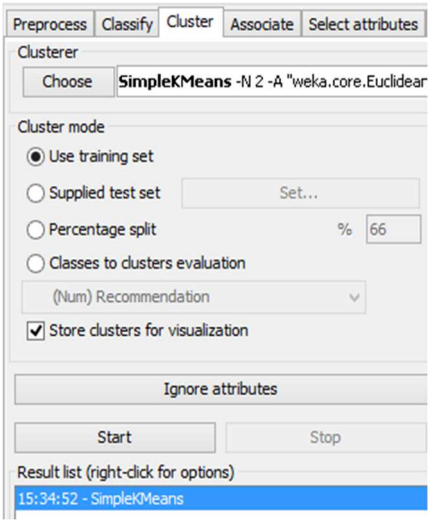

Figura 7.5 - Separador de parametrização para clustering

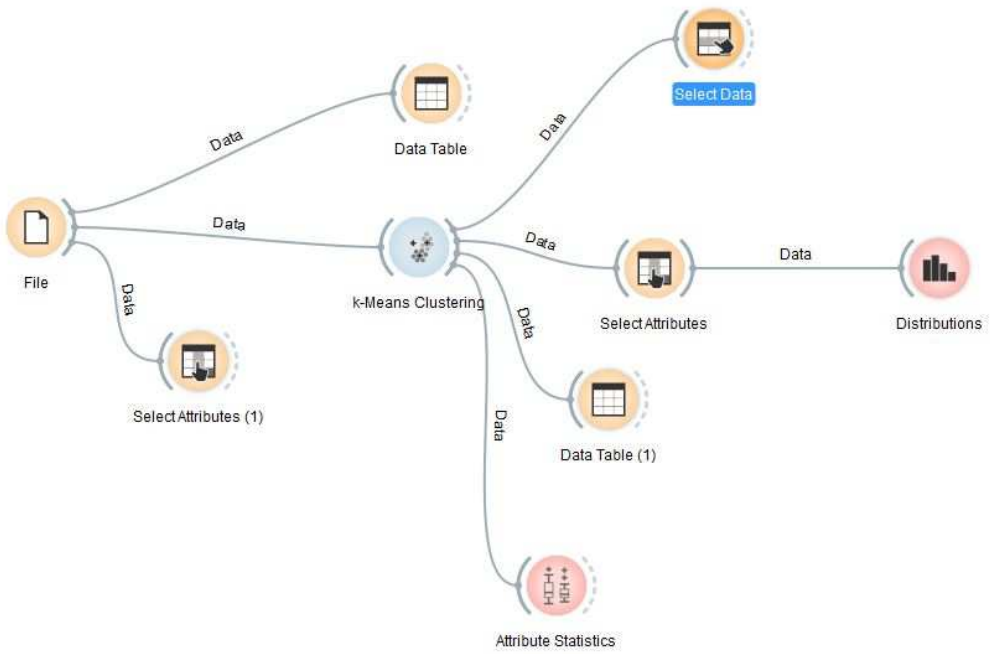

Figura 7.6 - Árvore de widgets para processamento do algoritmo K-Means

Na tabela Tabela 7.9 são apresentados os resultados do processamento do algoritmo *DBScan* nas quatro ferramentas. A parametrização para geração destes resultados foi de *épsilon* = 5,0 e *min points* = 5.

Tabela 7.9 - Resultados do processamento do algoritmo *DBScan*

	Agoritmo DBScan			
	RapidMiner	**Orange**	**WEKA**	**KNIME**
	Coverage			
Cluster 1	500	N.A.	500	500
Cluster 2	--	N.A.	--	--
Accuracy	0,58	N.A.	1	0,58

A vermelho encontra-se representado a inexistência do algoritmo na ferramenta Orange e os "- -" representam a inexistência de resultados nos campos onde se encontram. Nas três ferramentas em comparação apenas foi gerado 1 *cluster* com uma *coverage* de 500. Relativamente aos valores da *accuracy*, o RapidMiner e o KNIME geraram os mesmos valores uma vez que a distribuição dos *clusters* pelas classes era idêntica. A WEKA obteve valor máximo de *accuracy* uma vez que o *cluster* que gerou continha todos os valores da mesma classe (recomendação = 0).

De forma a tentar verificar a geração de mais *clusters* alterou-se a parametrização para *épsilon* = 2,0 e *min points* = 6. Os resultados foram os da tabela seguinte.

Tabela 7.10 - Resultados do processamento do algoritmo *DBScan* 2

	Agoritmo DBScan			
	RapidMiner	**Orange**	**WEKA**	**KNIME**
	Coverage			
Cluster 1	500	N.A.	422	500
Cluster 2	--	N.A.	7	--
Noise	--		71	--
Accuracy	0,58	N.A.	0,98	0,58

Podemos concluir que a WEKA distribui as instâncias por 2 *clusters*, o que não acontece no RapidMiner ou KNIME. No entanto deixa alguns itens de fora dos *clusters* (*noise*), o que indica que com o mesmo número mínimo de pontos as ferramentas funcionam de forma diferente.

Na Figura 7.7 é possível visualizar a árvore de operadores para processamento do algoritmo *DBScan* no RapidMiner.

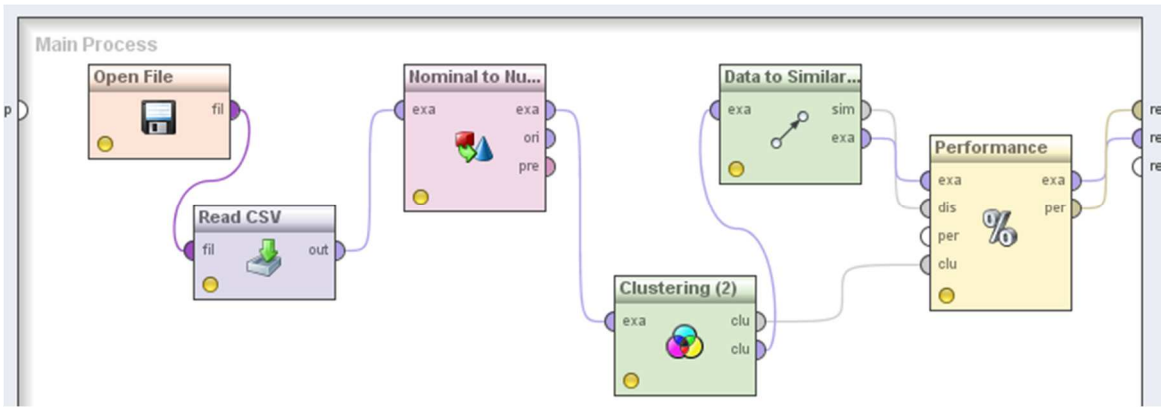

Figura 7.7 - Árvore de operadores para processamento do algoritmo DBScan

Na Tabela 7.11 são apresentados os resultados do processamento do algoritmo *Hierarchical Clustering* nas quatro ferramentas.

Tabela 7.11 - Processamento do algoritmo Hierarchical Clustering

	Algoritmo Hierarchical Clustering			
	RapidMiner	**Orange**	**WEKA**	**KNIME**
	Coverage			
Cluster 1	--	--	499	1
Cluster 2	--	--	1	499
Accuracy	--	--	N.A.	0,418

Na Tabela 7.11 podemos visualizar os resultados de processamento do algoritmo *Hierarchical Clustering*. A vermelho encontra-se representado a falta de resultados do algoritmo na ferramenta WEKA, e os "- -" representam a inexistência de resultados nos campos onde estes se encontram. Também o algoritmo em causa mostrou resultados distintos no RapidMiner, tendo este gerado 999 *clusters*, o que é normal, uma vez que o algoritmo adiciona 500 exemplos aos 499. No Orange foram gerados 12 *clusters*. No KNIME e WEKA os resultados de *coverage* foram semelhantes embora invertidos. Relativamente aos valores da *accuracy* só podem ser medidos nas ferramentas que geraram *clusters*, no entanto não foi possível fazê-lo para a ferramenta WEKA pois este algoritmo não emite os valores de centroides. Assim, o melhor resultado de *accuracy* foi obtido pelo KNIME. Na imagem seguinte é possível visualizar a árvore de operadores para processamento do algoritmo *Hierarchical Clustering* no KNIME.

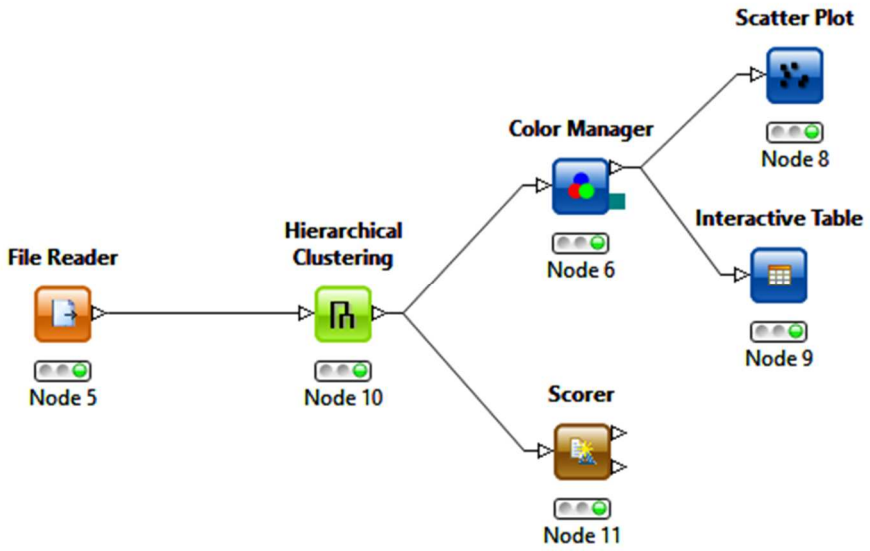

Figura 7.8 - Árvore de nodes para processamento do algoritmo Hierarchical Clustering

7.3.1 Conclusões

Em suma, as ferramentas que geraram resultados obtiveram valores muito próximos. Individualmente as ferramentas tiveram *performances* diferentes á análise do *dataset* com os diferentes algoritmos. Quanto ao RapidMiner destacam-se as seguintes:

- No algoritmo *K-Means* o RapidMiner obteve o melhor valor de *accuracy*;
- Relativamente ao processamento do algoritmo *DBScan,* apesar de só gerar 1 *cluster,* o RapidMiner também obteve o melhor valor de *accuracy;*
- Quanto ao *Hierarchical Clustering*, representado no RapidMiner pelo *Agglomerative Hierarchical*, os resultados foram diferentes, uma vez que este algoritmo gerou 999 *clusters*, não sendo possível calcular o valor de *accuracy.*

Quanto ao Orange destacam-se os seguintes resultados:

- No algoritmo *K-Means* o Orange obteve o 3º melhor valor de *accuracy*, ficando este muito próximo dos dois primeiros;
- O algoritmo *DBScan* não se encontra integrado no Orange;
- Quanto ao *Hierarchical Clustering*, o Orange gerou 12 *clusters.*

Quanto á WEKA destacam-se os seguintes resultados:

- No algoritmo *K-Means* a WEKA foi a ferramenta que obteve o pior valor de *accuracy* com uma diferença de aproximadamente 10 % relativamente ás outras ferramentas;
- Relativamente ao processamento do algoritmo *DBScan,* a WEKA não conseguiu gerar *clusters* para avaliação;
- Quanto ao *Hierarchical Clustering*, apesar do algoritmo gerar os valores de *coverage*, não foi possível calcular o valor de *accuracy*, uma vez que este não emitiu os valores dos centroides.

Quanto ao KNIME destacam-se os seguintes resultados:

- No algoritmo *K-Means* o KNIME obteve o 2º melhor valor de *accuracy*;
- Relativamente ao processamento do algoritmo *DBScan,* em comparação ao RapidMiner só gerou 1 *cluster* e obteve o 2º melhor valor de *accuracy;*
- No algoritmo *Hierarchical Clustering,* o KNIME obteve o melhor valor de *accuracy*.

Embora os resultados apontem para a ferramenta que obteve melhores valores de *accuracy* em cada algoritmo, podemos concluir que dentro dos resultados gerados os valores são muito próximos.

7.4 Regressão

O objetivo da regressão é encontrar uma função para a previsão do comportamento de uma variável o mais aproximado possível. Para avaliar o desempenho dos algoritmos na descoberta dessa função, vamos usar um *dataset* com 517 instâncias e 13 atributos. O algoritmo a comparar entre as quatro ferramentas é a Regressão Linear (*Linear Regression*).

Os parâmetros a comparar referem-se às seguintes fórmulas:

- RMSE/ RMSD – *Root Mean Squared Error/ Root Mean Squared Deviation*

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad \mathrm{RMSD} = \sqrt{\frac{\sum_{t=1}^{n}(\hat{y}_t - y)^2}{n}}. \tag{1}$$

- R2 – *Squared Correlation/ R-squared*

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - f_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \qquad R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST} \tag{2}$$

- MAE – *Mean Absolute Error*

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|\,.$$

(3)

- *Correlation*

$$\rho_{X,Y} = \operatorname{corr}(X,Y) = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

(4)

- MSE – *Mean Squared Error*

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2.$$

(5)

- RRSE – *Root Relative Squared Error*

Root relative squared error

$$\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{(\bar{a} - a_1)^2 + \ldots + (\bar{a} - a_n)^2}$$

(6)

- RAE – *Relative Absolute Error*

Relative absolute error

$$\frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{|\bar{a} - a_1| + \ldots + |\bar{a} - a_n|}$$

(7)

Depois de preparados os *datasets* e o algorimo a testar, vamos passar a fase de processamento. Os resultados obtidos são apresentados de seguida.

Tabela 7.12 - Resultados do processamento do algoritmo Regressão Linear

Parâmetros de análise	RapidMiner	Orange	WEKA	KNIME
RMSE/ RMSD	62,903	64,5102	63,8429	62,121
R2	0,022	(-) 0,0290	N.A.	0,046
MAE	N.A.	21,9378	20,0857	20,299
Correlation	0,147	N.A.	0,0763	N.A.
MSE	N.A.	4 161,5665	N.A.	3859,072
RRSE	98,9 %	101,44 %	100,2281%	N.A.
RAE	N.A.	118,16%	108,035%	N.A.

Na Tabela 7.12 são apresentados os resultados do processamento do algoritmo Regressão Linear. Os parâmetros disponíveis para análise diferem nas quatro ferramentas. A tabela anterior apresenta os parâmetros comuns as todas. A vermelho encontram-se representados os parâmetros que não estão disponíveis para visualização.

Nas figuras seguintes são apresentadas as árvores de operadores/ *widgets* construídas para análise do algoritmo *Linear Regression*.

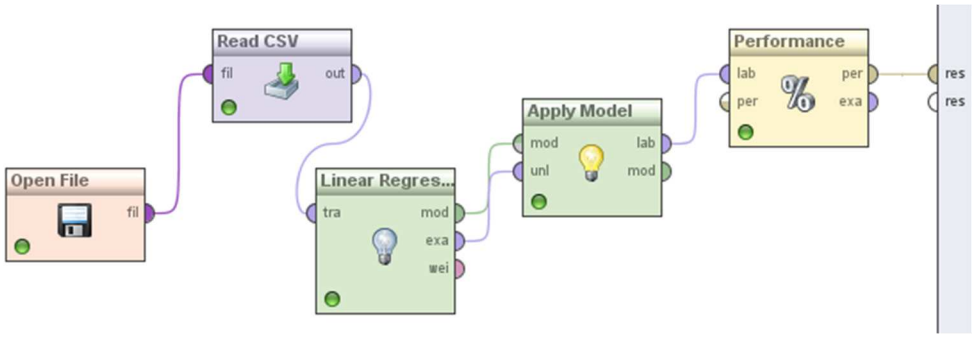

Figura 7.9 - RapidMiner - Árvore de operadores

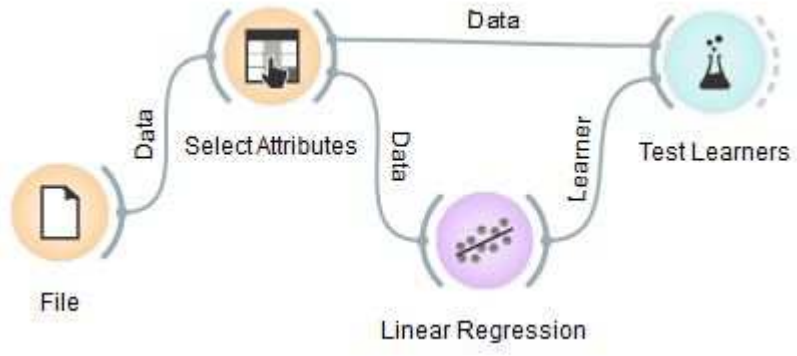

Figura 7.10 – Orange - Árvore de widgets

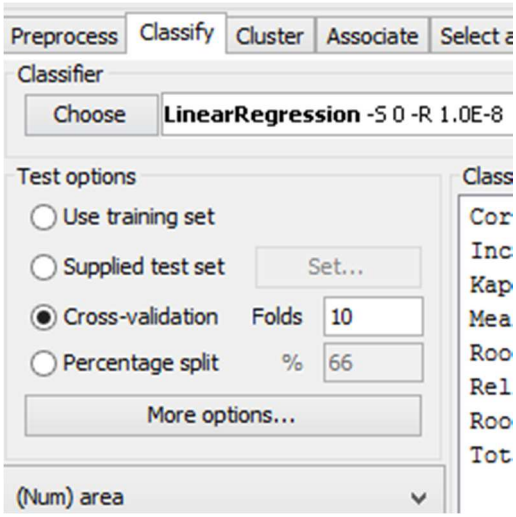

Figura 7.11 – WEKA - Separador de parametrização para o algoritmo Linear Regression

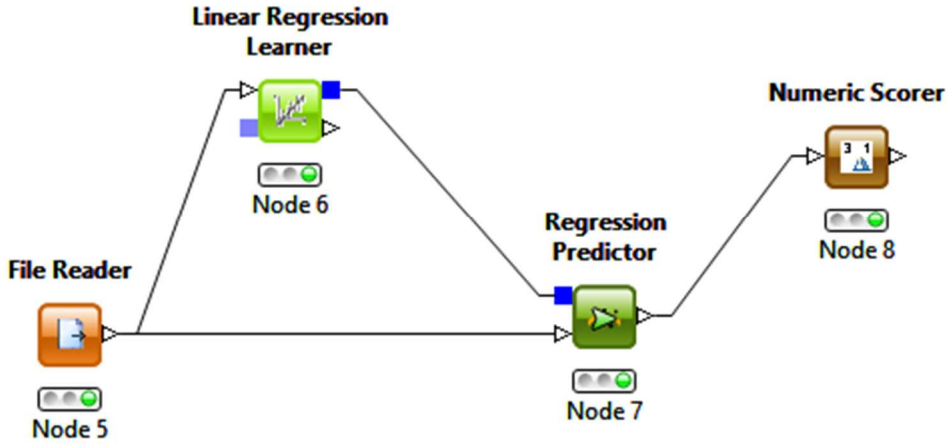

Figura 7.12 – KNIME - Árvore de *nodes*

7.4.1 Conclusões

De acordo com os resultados da Tabela 7.12 é possível concluir que os melhores resultados são obtidos pela ferramenta Orange. No geral, esta apresenta melhores valores e também disponibiliza mais parâmetros de análise de *performance*. Relativamente às restantes ferramentas, os resultados estão muito próximos dos melhores, obtidos pela ferramenta Orange, provando mais uma vez o equilíbrio das quatro ferramentas.

7.5 Associação

O objetivo das regras de associação é identificar dependências entre variáveis de forma a criar um modelo de associações. Para avaliar o desempenho dos algoritmos na descoberta dessa função, vamos usar um *dataset* com 120 instâncias e 6 atributos. Na tabela seguinte são apresentados os algoritmos em comparação.

Tabela 7.13 - Algoritmos de associação a comparar

Algoritmos de associação	**RapidMiner**	**Orange**	**WEKA**	**KNIME**
FP-Growth	✓	N.A	✓	Item Set Finder (Borgelt)
Apriori	N.A.	N.A.	✓	Association Rule Learner (Borgelt) ItemSet Finder (Borgelt)
Association Rules	N.A.	✓	N.A.	✓

Os algoritmos escolhidos para análise são considerados os mais importantes e também os que se encontra incluídos na maioria das ferramentas. Em alguns casos, os algoritmos em análise não se encontram integrados nas ferramentas, mas existem algumas variantes dos mesmos (visíveis na tabela a cinza), que também serão testadas. O vermelho representa os algoritmos que não são suportados pelas ferramentas.

Depois de preparados os *datasets* e o conjunto de algorimos a testar vamos passar a fase de processamento. Os resultados das regras geradas em todos os processamentos podem ser consultados em anexo. Seguidamente são apresentados os resultados da comparação nas quatro ferramentas.

Na coluna esquerda encontram-se representados os parâmetros de comparação onde *Min Sup* representa o valor de Suporte Mínimo (*Minimal Support*). Quanto aos valores mínimos de confiança (*Minimal Confidence)* foram fixados em todos os testes para 0,8.

Na Tabela 7.14 são apresentados os resultados do processamento do algoritmo *Association Rules*.

Tabela 7.14 - Resultados do processamento do algoritmo *Association Rules*

	Association Rules			
	RapidMiner	**Orange**	**WEKA**	**KNIME**
Parâmetros	**Regras geradas (total)**			

Min Sup = 0,8	N.A.	0	N.A.	2
Min Sup = 0,5	N.A.	0	N.A.	9
Min Sup = 0,4	N.A.	4	N.A.	9

Podemos visualizar que entre as duas ferramentas em comparação, o KNIME é a ferramenta que emite mais regras no seu conjunto, mesmo com a variação na parametrização.

Nas duas tabelas seguintes são apresentadas as regras geradas pelas ferramentas e os valores de confiança e *lift* associados a cada uma delas.

Tabela 7.15 - Regras geradas pelo Orange

Regras geradas					
	Antecedente	→ **Consequente**	**Confiança**	**Lift**	**Suporte**
1	Urine pushing = yes, Micturition pains = yes	Decision = yes	1.0	2.034	0.408
2	Occurrence of nausea = no, Micturition pains = no	Decision = no	0.836	1.645	0.425
3	Micturition pains = no	Decision = no	0.836	1.645	0.425
4	Micturition pains = yes	Decision = yes	0.831	1.689	0.408

Tabela 7.16 - Regras geradas pelo KNIME

Regras geradas					
	Antecedente	**→ Consequente**	**Confiança**	**Lift**	**Suporte**
1	yes	no	0.918	1.001	0.835
2	true	no	0.91	0.992	0.752
3	no	yes	0.91	1.001	0.835
4	yes, true	no	0.9	0.981	0.669
5	true	yes	0.9	0.99	0.744
6	no, true	yes	0.89	0.979	0.669
7	no	true	0.82	0.992	0.752
8	yes	true	0.818	0.99	0.744
9	yes, no	true	0.802	0.97	0.669

É possível visualizar que nas regras do Orange existe uma regra (Regra nº 1) que atinge 100 % de confiança. No KNIME o valor de confiança mais alto é gerado pela Regra nº 1 e é de 0,918. Relativamente aos valores de suporte, estes são superiores no KNIME, estando todos acima dos 50 %, ao contrário do Orange que estão todos abaixo do mesmo valor.

Na Tabela 7.17 são apresentados os resultados do processamento do algoritmo *FPGrowth*.

Tabela 7.17 – Resultados do processamento do algoritmo *FPGrowth*

	FPGrowth			
	RapidMiner	**Orange**	**WEKA**	**KNIME**
Parâmetros	**Regras geradas (total)**			
Min Sup = 0,8	0	N.A.	0	4
Min Sup = 0,5	1	N.A.	1	7
Min Sup = 0,4	4	N.A.	16	7

Podemos visualizar que entre as ferramentas em comparação, o KNIME (*Item Set Finder*) é a ferramenta que emite mais regras no seu conjunto, embora a WEKA emita mais com a parametrização *Min Sup = 0,4*. Relativamente ao KNIME as regras geradas por este algoritmo não emitem valores de *lift* ou confiança para comparação, assim as regras geradas pelo RapidMiner e WEKA com *min sup=0,4* são as seguintes:

Tabela 7.18 - Regras geradas pelo RapidMiner

	Regras					
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Burning_of_urethra	Urine_pushing		1.0	1.5	0.416
2	Urine_pushing	Temperature_of_patient		0.875	1.05	0.583
3	Micturition_pains	Temperature_of_patient		0.830	0.996	0.408
4	Micturition_pains	Urine_pushing		0.830	1.245	0.408

Tabela 7.19 - Regras geradas pela WEKA

Regras						
Nº	Antecedente	→	Consequente	Confiança	Lift	Suporte
1	Decision = yes: 59	Urine pushing = yes: 59		1.0	1.5	N.A.
2	Burning of urethra = yes: 50	Urine pushing = yes: 50		1.0	1.5	N.A.
3	Temperature of patient = true, Decision = yes:49	Urine pushing = yes: 49		1.0	1.5	N.A.
4	Urine pushing = yes, Micturition pains = yes: 49	Decision = yes: 49		1.0	2.03	N.A.
5	Micturition pains = yes, Decision = yes: 49	Urine pushing = yes: 49		1.0	1.5	N.A.
6	Urine pushing = yes: 80	Temperature of patient = true: 70		0.88	1.05	N.A.
7	Micturition pains = yes: 59	Temperature of patient = true: 49		0.83	1.0	N.A.

Regras					
Nº	Antecedente	→ Consequente	Confiança	Lift	Suporte
8	Decision = yes: 59	Temperature of patient = true: 49	0.83	1.0	N.A.
9	Micturition pains = yes: 59	Urine pushing = yes: 49	0.83	1.25	N.A.
10	Micturition pais = yes: 59	Decision = yes: 49	0.83	1.69	N.A.
11	Decision = yes: 59	Micturition pains = yes: 49	0.83	1.69	N.A.
12	Decision = yes: 59	Temperature of patient = true, urine pushing = yes: 49	0.83	1.42	N.A.
13	Urine pushing = yes, Decision = yes: 59	Temperature of patient = true: 49	0.83	1.0	N.A.
14	Micturition pais = yes: 59	Urine pushing = yes, Decision = yes: 49	0.83	1.69	N.A.
15	Decision = yes: 59	Urine pushing = yes, Micturition pains = yes: 49	0.83	2.03	N.A.
16	Urine pushing = yes, Decision = yes: 59	Micturition pains = yes: 49	0.83	1.69	N.A.

Podemos verificar que as regras geradas pelo RapidMiner coincidem com algumas regras geradas pela WEKA. As cores representam essa semelhança. Na tabela seguinte vamos comparar os valores dessas regras.

Tabela 7.20 - Regras equivalentes nas ferramentas RapidMiner e WEKA

	RapidMiner			**WEKA**		
	Regras (nº)	**Confiança**	**Lift**	**Regras (nº)**	**Confiança**	**Lift**
Min sup 0,4	1	1.0	1.5	2	1.0	1.5
	2	0.875	1.05	6	0.88	1.05
	3	0.830	0.996	7	0.83	1.0
	4	0.830	1.245	9	0.83	1.25

É possível concluir que a Regra nº 1 do RapidMiner e a Regra 2 da WEKA apresentam exatamente os mesmos valores de confiança e *lift*. Relativamente às restantes regras, os valores de confiança e *lift* são muito próximos embora os mais altos sejam obtidas na WEKA.

Na Tabela 7.21 são apresentados os resultados do processamento do algoritmo *Apriori*.

Tabela 7.21 - Resultados do processamento do algoritmo Apriori

	Apriori				
	RapidMiner	**Orange**	**WEKA**	**KNIME**	
	Regras geradas (total)				
Parâmetros				**ARL (B)**	**ISF (B)**
Min Sup = 0,8	N.A.	N.A.	0	7	1
Min Sup = 0,5	N.A.	N.A.	4	9	4
Min Sup = 0,4	N.A.	N.A.	35	9	4

Podemos visualizar que o KNIME é a ferramenta que emite mais regras entre os dois algoritmos ARL (*Association Rule Learner - Borgelt*) e ISF (*Item Set Finder - Borgelt*), embora a WEKA emita mais com a parametrização *Min Sup = 0,4*. Relativamente ao KNIME as regras geradas pelo algoritmo *Item Set Finder (Borgelt)* não emitem valores de *lift* ou confiança para comparação. De seguida são apresentadas as regras geradas com *min sup=0,5* para as ferramentas KNIME (com o algoritmo ARL (B)) e WEKA uma vez que com *min sup=0,4* os resultados são muito extensos para visualização.

Tabela 7.22 - Regras geradas pela WEKA

Regras geradas				
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**
1	Micturition pains = no 61		Occurrence of nausea = no 61	1.0
2	Urine pushing = yes 80		Temperature of patient = true 70	0.88
3	Lumbar pain = yes 70		Temperature of patient = true 60	0.86
4	Burning of urethra = no 70		Temperature of patient = true 60	0.86

Tabela 7.23 - Regras geradas pelo algoritmo ARL (B) do KNIME

Regras geradas						
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Yes		No	91.8	0.993	101
2	True		No	91	0.984	91
3	No		Yes	91	0.993	101
4	True		Yes	90	0.982	90
5	True, yes		No	90	0.973	81
6	True, no		Yes	89	0.971	81
7	No		True	82	0.984	91
8	Yes		True	81.8	0.982	90
9	Yes, no		True	80.2	0.962	81

É possível verificar que nas regras geradas pela WEKA existe uma regra (Regra nº 1) que atinge 100 % de confiança. No KNIME o valor de confiança mais alto é gerado pela Regra nº 1 e é de 91,8%.

7.5.1 Conclusões

De acordo com os resultados das tabelas anteriores podemos concluir que os resultados obtidos nas regras geradas são muito semelhantes nas quatro ferramentas. A variação de resultados verifica-se de algoritmo para algoritmo, uma vez que nem todas as ferramentas os incluem e representam da mesma forma. Relativamente á ferramenta KNIME, apesar de apresentar resultados interessantes na geração de regras, é impossível compará-los uma vez que a leitura das regras não é clara. Este factor deve-se á representação das regras pela ferramenta, que apenas incluem a referência ao valor do atributo e não incluem a que atributo se refere esse valor. Deste modo, nas tabelas seguintes são apresentadas as regras comuns aos 3 algoritmos em análise e a comparação dos valores de confiança e *lift* nas ferramentas RapidMiner, Orange e WEKA.

Tabela 7.24 - Comparação das regras geradas I

RapidMiner			**Orange**			**WEKA**		
Algoritmo FP-Growth			**Algoritmo Association Rules**			**Algoritmo FP-Growth**		
Regras (nº)	**Confiança**	**Lift**	**Regras (nº)**	**Confiança**	**Lift**	**Regras (nº)**	**Confiança**	**Lift**
1	1.0	1.5				**2**	1.0	1.5
2	0.875	1.05				**6**	0.88	1.05
3	0.830	0.996				**7**	0.83	1.0
4	0.830	1.245				**9**	0.83	1.25
			1	1.0	2.034	**4**	1.0	2.03
			4	0.831	1.689	**10**	0.83	1.69

Como vimos anteriormente, no algoritmo *FPGrowth* existem semelhanças entre as ferramentas RapidMiner e WEKA. Podemos também concluir que as mesmas regras são geradas pelos algoritmos *Association Rules* e *FPGrowth* no Orange e WEKA respetivamente. Relativamente aos valores obtidos por estes dois algoritmos podemos visualizar que não existem diferenças na variação da ferramenta de análise de dados.

Tabela 7.25 - Comparaçao das regras geradas II

RapidMiner			Orange			WEKA		
Algoritmo FP-Growth			Algoritmo Association Rules			Algoritmo Apriori		
Regras (nº)	**Confiança**	**Lift**	**Regras (nº)**	**Confiança**	**Lift**	**Regras (nº)**	**Confiança**	**Lift**
1	1.0	1.5				**7**	1	
2	0.875	1.05				**11**	0.88	
3	0.830	0.996				**14**	0.84	
4	0.830	1.245				**28**	0.83	
			1	1.0	2.034	**10**	1	
			2	0.836	1.645	**23**	0.84	
			3	0.836	1.645	**19**	0.84	
			4	0.831	1.689	**30**	0.83	

Comparando o algoritmo *FPGrowth* do RapidMiner com o algoritmo Apriori na WEKA podemos verificar que também existem semelhanças. Os valores são ligeiramente melhores na WEKA, mas de forma geral, mais uma vez, muito equilibrados. Também no Orange e na WEKA existem relações entre os algoritmos Association Rules e Apriori respetivamente. Relativamente aos valores obtidos por estes dois algoritmos podemos visualizar que não existem diferenças na variação da ferramenta de análise de dados.

Em suma, é possível afirmar que as ferramentas apresentam resultados coerentes entre si. Apesar da mudança de algoritmos, os valores matêm-se nas regras que são iguais. A variação encontra-se no número de regras geradas e na capacidade das ferramentas de as reprentarem. Apesar de ser a ferramenta com mais algoritmos em comparação integrados, o KNIME não se adaptou bem ao *dataset* e não foi possível representar e comparar os resultados das regras geradas.

8 CONCLUSÕES E TRABALHO FUTURO

O objetivo do nosso estudo foi encontrar a melhor ferramenta *open source* de *data mining* para implementar um projeto real. A ferramenta escolhida deveria cumprir as exigências ao nível de *features* e funcionalidades de cada projeto e ainda representar uma oportunidade de modernização para as empresas a custo zero. Este estudo é muito importante, uma vez que, as empresas necessitam de recursos informáticos para executar as suas tarefas e, hoje em dia, é cada vez mais difícil escolher o recurso certo devido á larga oferta dos mesmos. Para além da dificuldade associada á escolha de uma ferramenta de trabalho está também o problema dos custos associados á mesma. No nosso estudo procurámos obter informações e analisar as ferramentas *open source* mais usadas e dentro de um conjunto de 23, avaliámos e comparámos em particular 4 delas.

O nosso estudo iniciou-se com a análise a 23 ferramentas *open source* de *data mining* referenciadas pelo KDnuggets, como *software suites* para *data mining,* análise de dados e *knowledge discovery*. A análise foi feita ao nível do estado da arte de cada ferramenta de DM. Após o levantamento das características das várias ferramentas foram identificadas as quatro melhores soluções, que não exigiam qualquer tipo de conhecimentos de programação, para avaliação técnica e teste prático. As ferramentas RapidMiner, Orange, WEKA e KNIME foram as selecionadas. Seguidamente procedemos a uma análise mais técnica onde foram exploradas caraterísticas próprias de cada ferramenta como compatibilidade e funcionalidades. Por fim foram utilizados *datasets* para medir os valores de *performance* e avaliar o comportamento das ferramentas no processamento de dados reais. O objetivo era não só comparar os resultados obtidos pelas quatro ferramentas, mas também perceber como estas se comportam na emissão e processamento dos mesmos. Os resultados obtidos foram, de forma geral, muito semelhantes nas quatro ferramentas, sendo muito difícil eleger a ferramenta que melhor se comporta num projeto real. No entanto, podemos retirar algumas conclusões para cada ferramenta.

Relativamente ao RapidMiner trata-se de uma solução muito estável e moderna, e por isso uma boa solução para um projeto real. Os resultados obtidos foram positivos, de onde se destacam os seguintes:

- Na tarefa de classificação, o RapidMiner mostrou-se uma ferramenta completa com os algoritmos principais. Os resultados foram bastante equilibrados com os das outras ferramentas. No processamento do *dataset 3,* com maior dimensão, mostrou bons resultados no algoritmo SVM, apesar do longo tempo de processamento. Com o algoritmo *Naïve Bayes* mostrou-se uma das ferramentas mais rápidas no processamento de resultados.
- Relativamente ao *clustering,* o RapidMiner mostrou ser uma ferramenta completa no que diz respeito aos algoritmos integrados. Emitiu os melhores resultados de *accuracy* nos algoritmos *K-Means* e *DBScan.*

- Na regressão o RapidMiner também se mostrou completo na disponibilização de algoritmos. Quanto aos resultados obtidos, estes foram muito equilibrados com as outras ferramentas.
- Por fim na associação, consideramos que o RapidMiner integra poucos algoritmos. As versões anteriores da ferramenta incluíam o algoritmo *Apriori*, mas este foi removido devido a um erro que incorporava, ou seja, na versão em análise apenas foi possível testar o algoritmo *FPGrowth.* Quanto aos resultados emitidos pelo algoritmo podemos concluir que foram equilibrados, mas inferiories aos da ferramenta WEKA na geração de regras.

O RapidMiner é uma solução muito sólida para as empresas, no entanto, tem uma curva de aprendizagem acentuada. Esta curva pode ser facilmente ultrapassada com a documentação e tutoriais que dispõe. Para além disso, disponibiliza uma versão comercial e possui integrações de algoritmos da WEKA e *scripts* em R, que compensam algumas falhas nos algoritmos. O RapidMiner trata-se de uma ferramenta direcionada para utilizadores dispostos a despender algum tempo a aprender e aplicar as técnicas de *data mining*.

Quanto ao Orange, podemos concluir que é uma ferramenta moderna e que apresenta uma margem de progressão grande, no que diz respeito á integração de algoritmos. Os resultados obtidos foram muito positivos de onde se destacam os seguintes:

- Nas tarefas de classificação o Orange emitiu resultados muito equilibrados, estando em particular destaque na análise do *dataset 2.* Relativamente ao tempo de processamento, o Orange revelou-se a ferramenta mais rápida no geral, nunca ultrapassando 1 hora de processamento em todos os algoritmos. Quanto a algoritmos, encontra-se equipado com os principais para as tarefas de classificação.
- Nas tarefas de *Clustering*, os resultados obtidos nos algoritmos que inclui foram equilibrados relativamente às outras ferramentas.
- Quanto às tarefas de regressão, o Orange revelou-se a ferramenta com melhores valores obtidos.
- Na associação, o Orange tem apenas um *widget* integrado, *Association Rules*, mas que inclui vários algoritmos. Apesar de gerar poucas regras, os resultados obtidos são equilibrados.

O Orange encontra-se direcionado para utilizadores com conhecimentos mínimos de *data mining* e de programação. Apresenta uma curva de aprendizagem mínima e tem uma comunidade de apoio relativamente pequena. No entanto, revelou-se muito rápido na emissão de resultados e na construção de análises de dados.

A WEKA trata-se de uma solução mais madura, factor que se revê na quantidade de algoritmos que disponibiliza. Os resultados obtidos foram muito positivos de onde se destacam os seguintes:

- Na tarefa de classificação, a WEKA teve a melhor prestação no processamento do algoritmo *ADTree* nos três *datasets*. No processamento do *dataset 3*, com o algoritmo *Naïve Bayes*, mostrou-se uma das ferramentas mais rápidas no processamento de resultados.
- Nas tarefas de *Clustering*, os resultados obtidos foram equilibrados relativamente às outras ferramentas.
- Na regressão os valores obtidos foram equilibrados em relação às outras ferramentas.
- Por fim, na associação, foi a ferramenta que emitiu mais regras com valores de confiança a 100% nos algoritmos *FPGrowth* e *Apriori*.

A WEKA pode ser uma solução muito interessante para as empresas, uma vez que oferece todas as funcionalidades, é uma ferramenta intuitiva e amigável para profissionais novos na área. Tem a particularidade de estar integrada nas ferramentas RapidMiner e KNIME representando, por isso, uma boa solução para quem deseja iniciar o desenvolvimento de um novo *software*.

Por último, o KNIME representa uma ferramenta moderna e robusta, trata-se de uma solução que inclui os principais algoritmos de *data mining*, e ainda outros que se encontram integrados nas várias extensões que inclui como a WEKA, R, etc. Os resultados obtidos foram muito equilibrados de onde se destacam os seguintes:

- Na tarefa de classificação, o KNIME encontra-se equipado com os algoritmos principais. Os resultados foram bastante equilibrados com os das outras ferramentas, havendo um desempenho superior no processamento do algoritmo SVM. Relativamente ao processamento do *dataset* 3 foi a ferramenta que mais tempo demorou, atingindo 9 horas e 41 minutos.
- Relativamente ao *clustering*, os resultados de *accuracy* foram equilibrados, havendo alguma superioridade no algoritmo *Hierarchical Clustering.*
- Na regressão, o KNIME também se mostrou muito equilibrado na emissão de resultados.
- Por fim, na associação, consideramos que apesar do KNIME gerar várias regras, não se adaptou ao *dataset* em análise. As regras geradas foram numerosas, no entanto não foi possível fazer a sua leitura. Contudo, é uma das ferramentas que mais algoritmos de associação disponibiliza.

Para empresas habituadas ao ambiente de trabalho do *Eclipse*, o KNIME poderá ser a ferramenta ideal, uma vez que disponibiliza uma API modular baseada na plataforma *Eclipse*, facilmente extensível, com sistema de *pluggins*. O KNIME é ideal para utilizadores com necessidades específicas de *data mining*, uma vez que disponibiliza extensões para trabalhar no âmbito da química (Informatics Research and Development Unit of Public Health Informatics & Technology Program Office, 2010).

A ferramenta *open source* de *data mining* que devemos usar, depende dos objetivos do projeto em causa e da pessoa que está á frente dele, bem como da equipa que vai ser responsável por desenvolver as tarefas de *data mining*. Tudo depende do seu conhecimento nas técnicas de *data mining* e programação. No decorrer deste estudo consideramos que o processo de *data mining* é demorado e minucioso, e que requer os conhecimentos mínimos para obtenção de resultados válidos.

Têm havido grandes mudanças nos modelos de *data mining open source* desde há uma década atrás. Os novos modelos oferecem *interfaces* gráficas melhoradas, focam-se na usabilidade e interatividade e suportam grandes extensões de dados e linguagens de programação, proporcionando flexibilidade através da programação visual das linguagens escritas. Para além disso, a documentação melhorou bastante em termos de qualidade e os fóruns e listas de discussão aumentaram (Zupan & Demsar, 2008). Neste sentido, concluímos que as quatro ferramentas em análise, de acordo com os critérios em avaliação que excluem todas aquelas que necessitam de conhecimentos de programação específicos, são as melhores e mais populares ferramentas *open source* de *data mining* do momento.

Em suma, consideramos que as ferramentas *open source de DM* estão atualmente bem equipadas e preparadas para os projetos reais, representando uma solução alternativa às ferramentas comerciais.

No que diz respeito ao trabalho que ainda está por desenvolver consideramos que existem algumas melhorias que podem ser integradas nas ferramentas. Para além disso seria interessante acompanhar o desenvolvimento das ferramentas *open source* e fazer uma comparação de resultados de *performance* entre ferramentas *open source* e ferramentas comerciais de *data mining*. Desta forma, seria possível estabelecer uma comparação direta entre as duas realidades e perceber os custos implicados na obtenção desses resultados.

9 REFERÊNCIAS BIBLIOGRÁFICAS

34 Top Free Data Mining Software. (s.d.). Obtido em 10 de Dezembro de 2014, de Predictive Analytics Today: http://www.predictiveanalyticstoday.com/top-15-free-data-mining-software/

Alteryx, Inc. (2015). *Alteryx*. Obtido em 10 de Dezembro de 2014, de http://www.alteryx.com/products/alteryx-designer

Association Rules Algorithms. (s.d.). Obtido em 10 de Dezembro de 2014, de Data Mining Articles: http://www.dataminingarticles.com/association-analysis/association-rules-algorithms/

Auza, J. (2010). *5 of the best free and open source data mining software*. Obtido em 10 de Dezembro de 2014, de http://www.junauza.com/2010/11/free-data-mining-software.html

Berthold , M., Cebron, N., Dill, F., Gabriel, T., Kotter, T., Meinl, T., . . . Wiswedel, B. (2007). KNIME: The Konstanz Information Miner. Springer.

Berthold, M. R., Cebron, N., Dill, F., Fatta, G., Gabriel, T., Georg, F., . . . Wiswedel, B. (2006). KNIME: The Konstanz Information Miner. *Proc. of the 4th Annual Industrial Simulation Conference, Workshop on Multi-Agent Systems and Simulation.* Palermo.

Berthold, M., Cebron, N., Dill, F., Gabriel, T., Kotter, T., Meinl, T., . . . Wiswedel, B. (2007). *KNIME: open for inovation.* Obtido em 10 de Dezembro de 2014, de http://www.knime.org/knime

Bonaccorsi, A., & Rossi, C. (2003). Why open source software can succeed. *Research policy*.

Britos, P., Merlino, H., Fernández, E., Ochoa, M., Diez, E., & Gárcia-Martinez, R. (2006). Tool Selection Methodology in Data Mining. *Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento (JIISIC).*

Brownlee, J. (25 de Novembro de 2013). *A tour of machine learning algorithms*. Obtido em 10 de Dezembro de 2014, de Machine Learning Mastery: http://en.wikipedia.org/wiki/Self-organizing_map

Chen, X., Williams, G., & Xu, X. (2007). A Survey of Open Source Data Mining Systems. In *Emerging Technologies in Knowledge Discovery and Data Mining.*

Collier, K., Carey, B., Sautter, D., & Marjaniemi, C. (1999). A methodology for evaluating and selecting data mining software. *Proceedings of the 32nd Hawaii International Conference on System Sciences*.

Conklin, M. S. (2006). Beyond Low-Hanging Fruit : seeking the next generation in FLOSs data mining. *IFIP International Federation for Information Processing, 203*.

Creative. (s.d.). *Creative Commons*. Obtido de http://creativecommons.org/licenses/by-nc-sa/3.0/deed.pt_BR

Crowston , K., Wei, K., Howison, J., & Wiggins, A. (2010). Free/libre open source software development: what we know and what we do not know. *ACM Computing Surveys*.

Databionics Research Group. (2006). *Databionic ESOM Tools*. Obtido em 10 de Dezembro de 2014, de http://databionic-esom.sourceforge.net/

Demšar, J., Curk, T., & Erjavec, A. (2013). Orange: Data Mining Toolbox in Python. *Journal of Machine Learning Research 14*, 2349−2353.

Demšar, J., Curk, T., & Erjavec, A. (s.d.). *Orange: Data Mining - Fruitful and Fun*. Obtido em 10 de Dezembro de 2014, de http://orange.biolab.si/

Durst, M., Joehanes, R., Louis, J., Plummer, J., & Schmidt, C. (s.d.). *Machine Learning in Java (MLJ)*. Obtido em 10 de Dezembro de 2014, de http://mldev.sourceforge.net/

E-Business Technology Institute (ETI) of the University of Hong Kong. (2005). *Alphaminer: an open source data mining platform*. Obtido em 10 de Dezembro de 2014, de http://www.eti.hku.hk/alphaminer/

Evers, S. (2000). An introduction to open source software development.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI magazine*, pp. 37-54.

Fitzgerald, B. (2006). The transformation of open source software. *MIS Quarterly*.

Free Software Foundation. (2014). *GNU Operating System*. Obtido em 10 de Dezembro de 2014, de https://www.gnu.org/licenses/quick-guide-gplv3.html

Gama, J., Carvalho, A., Faceli, K., Lorena, A., & Oliveira, M. (2012). *Extração de conhecimento de dados: data mining.* Lisboa: Sílabo.

Godfrey, M. W., & Tu, Q. (2001). Evolution in open source software: a case study.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations, 11*.

Informatics Research and Development Unit of Public Health Informatics & Technology Program Office. (2010). *Open Source Data Mining Software Evaluation*. Obtido em 10 de

Dezembro de 2014, de http://www.phiresearchlab.org/index.php?option=com_content&view=category&layout=blog&id=2&Itemid=4

Information Technology and Systems Center, University of Alabama. (s.d.). *ADaM*. Obtido em 10 de Dezembro de 2014, de http://projects.itsc.uah.edu/datamining/adam/

International Center for Numerc Methods in Engineering. (2014). *Open NN: An Open Source Neural Networks C++ Library*. Obtido em 10 de Dezembro de 2014, de http://opennn.cimne.com/

IST - Information Society Technologies. (s.d.). *Mining Mart*. Obtido em 10 de Dezembro de 2014, de http://www-ai.cs.uni-dortmund.de/MMWEB/index.html

Jailia, M., & Tyagi, A. (07 de 2013). Data Mining: a prediction for performance improvement in online learning systems. *International Journal of Advanced Research in Computer Science and Software Engineering, 3*, pp. 628-635.

Jensen, C., & Scacchi, W. (2004). Data mining for software process discovery in open source software development communities. *26th International Conference on Software Engineering - International Workshop on Mining Software Repositories*.

KEEL . (2014). *KEEL: Knowledge Extraction based on Evolutionary Learning*. Obtido em 10 de Dezembro de 2014, de http://www.keel.es/

Konjevoda, P., & Stambuk, N. (2011). Open-Source Tools for Data Mining in Social Science. *Theoretical and Methodological Approaches to Social Sciences and Knowledge Management*, pp. 163-176.

Kotounin, M. R. (8 de Março de 2013). *The Best Data Mining Tools You Can Use For Free In Your Company*. Obtido em 10 de Dezembro de 2014, de Silicon Africa: http://www.siliconafrica.com/the-best-data-minning-tools-you-can-use-for-free-in-your-company/

Lopez, R. (2014). *Open NN: An Open Source Neural Networks C++ Library*. Obtido em 10 de Dezembro de 2014, de www.cimne.com/flood

Ludwig-Maximilians-Universität München. (s.d.). *ELKI: Environment for Developing KDD-Applications Supported by Index-Structures*. Obtido em 10 de Dezembro de 2014, de http://elki.dbs.ifi.lmu.de/

Machine Learning Group at the University of Waikato. (s.d.). *Weka 3: data mining sofware in java*. Obtido em 10 de Dezembro de 2014, de http://www.cs.waikato.ac.nz/ml/weka/index.html

Microsoft Research . (s.d.). *Vowpal Wabbit (Fast Learning)*. Obtido em 10 de Dezembro de 2014, de http://hunch.net/~vw/

Mikut, R., & Reischl, M. (2011). Data mining tools. *WIREs Data Mining and Knowledge Discovery*, 1-13.

ML-Flex. (s.d.). Obtido em 10 de Dezembro de 2014, de http://mlflex.sourceforge.net/

Open Source Initiative. (s.d.). Obtido em 10 de Dezembro de 2014, de Open Source Initiative: http://opensource.org/osd

Padhy, N., Mishra, P., & Panigrahi, R. (06 de 2012). The Survey of Data Mining Applications and Feature Scope. *International Journal of Computer Science, Engineering and Information Technology*, pp. 43-58.

PAKDD'09. (2009). Open Source in Data Mining workshop (OSDM'09). *Proceedings : 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining*.

PredictionIO. (2015). *PredictionIO*. Obtido em 10 de Dezembro de 2014, de http://prediction.io/

Rakotomalala, R. (2004). *TANAGRA*. Obtido em 10 de Dezembro de 2014, de http://chirouble.univ-lyon2.fr/~ricco/tanagra/en/tanagra.html

Rakotomalala, R. (2005). TANAGRA: a free software for research and academic purposes. *Proceedings of EGC'2005, 2*.

RapidMiner Inc. (2014). *RapidMiner*. Obtido em 10 de Dezembro de 2014, de http://rapidminer.com/

Robbins, J. E. (2005). Adopting Open Source Software Engineering (OSSE) Practices by Adopting OSSE Tools. In *Perspectives on Free and Open Source Software* .

Rosella Software. (2005). *CMSR Data Miner Data Mining & Predictive Modeling Software*. Obtido em 10 de Dezembro de 2014, de http://www.roselladb.com/starprobe.htm

S.Chekanov and jWork.ORG. (2013). *SCaVis: Scientific Computation and Visualization Environment*. Obtido em 10 de Dezembro de 2014, de http://jwork.org/scavis/

Santos, M. F., & Azevedo, C. (2005). *Data mining : descoberta de conhecimento em bases de dados*. Lisboa: FCA - Editora de informática, Lda.

SAS Institute Inc. (1998). *Data Mining and the Case for Sampling: solving business problems using SAS Enterprise Miner Software*. Obtido em 10 de Dezembro de 2014, de http://sceweb.uhcl.edu/boetticher/ML_DataMining/SAS-SEMMA.pdf

Sayad, S. (2010-2015). *Model Evaluation - Regression.* Obtido em 10 de Dezembro de 2014, de Data Mining Map: http://www.saedsayad.com/model_evaluation_r.htm

Seer Consulting. (20 de Outubro de 2014). *Seer*. Obtido em 10 de Dezembro de 2014, de Advances Analytics: http://www.seer-consulting.com/datamining.html

Silicon Graphics International Corp. (2015). *MLC++*. Obtido em 10 de Dezembro de 2014, de http://www.sgi.com/tech/mlc/index.html

Stallman, R. (2009). Why "open source" misses the point of free software. *Communications of the ACM*.

Togaware. (2006). *The Gnome Data Mine*. Obtido em 10 de Dezembro de 2014, de http://www.togaware.com/datamining/gdatamine/

Togaware. (2014). *Rattle: a graphical user interface for data mining using R*. Obtido em 10 de Dezembro de 2014, de http://rattle.togaware.com/

Torsten Hothorn. (s.d.). *CRAN Task View: Machine Learning & Statistical Learning*. Obtido em 10 de Dezembro de 2014, de http://cran.r-project.org/web/views/MachineLearning.html

Weber, S. (2003). *The success of open source.* Forthcoming: Harvard University Press.

Wikipédia. (2013). *ML-Flex*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/ML-Flex

Wikipédia. (2014). *Comparison of free and open source software licenses*. Obtido em 17 de 10 de 2014, de http://en.wikipedia.org/wiki/Comparison_of_free_and_open-source_software_licenses

Wikipédia. (2014). *Comparison of open-source software hosting facilities*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/Comparison_of_open-source_software_hosting_facilities

Wikipédia. (2014). *ELKI*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/ELKI

Wikipédia. (2014). *KNIME*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/KNIME

Wikipédia. (2014). *OpenNN*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/OpenNN

Wikipédia. (2014). *Orange*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/Orange_(software)

Wikipédia. (2014). *RapidMiner*. Obtido em Setembro de 2014, de http://en.wikipedia.org/wiki/RapidMiner

Wikipédia. (2014). *Tanagra*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/Tanagra_(machine_learning)

Wikipédia. (2014). *WEKA*. Obtido em 10 de Dezembro de 2014, de http://en.wikipedia.org/wiki/Weka_(machine_learning)

Wikipédia. (s.d.). *Association Rule Learning*. Obtido em 10 de Dezembro de 2014, de Wikipédia: http://en.wikipedia.org/wiki/Association_rule_learning

Wikipédia. (s.d.). *Self-organizing map*. Obtido em 10 de Dezembro de 2014, de Wikipédia: http://en.wikipedia.org/wiki/Self-organizing_map

Williams, G. (2011). Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery. (Springer, Ed.)

Witten, I. H., Frank, E., & Hall, M. (2011). *Data mining : practical machine learning tools and techniques* (3rd ed.). United States: Morgan Kaufmann.

Wu, X., & Kumar, V. (2009). *The Top Ten Algorithms in Data Mining.* New York: CRC Press.

Zhu, K. X., & Zhou, Z. (2011, June 16). Lock-in strategy in software competition: open-source software vs. proprietary software. *Informations Systems Research*, pp. 1-10.

Zupan, B., & Demsar, J. (2008). Open-Source Tools for Data Mining. *Clinics in Laboratory Medicine, 28*.

Somatório dos algoritmos de classificação e regressão

Tabela com as informações completas relativamente aos algoritmos de classificação e regressão.

Tabela 9.1 - Algoritmos de classificação e regressão (totais)

Algoritmos de classificação e regressão	RapidMiner	Orange	WEKA	KNIME
Adaboost	✓	x	x	x
Adaboost M1	x	x	✓	x
Adaboost.SAMME	x	x	x	✓
ADTree	x	x	✓	x
AODE	x	x	✓	x
AODEsr	x	x	✓	x
AutoMLP	✓	x	x	x
Bagging	✓	✓	✓	✓
Bayes Net	x	x	✓	x
Bayesian Boosting	✓	x	x	x
Bayesian Logistic Regression	x	x	✓	x
BFTree	x	x	✓	x
Citation KNN	x	x	✓	x
CHAID	✓	x	x	x
Classification Tree	x	✓	x	x
CN2 Rule Learner	x	✓	x	x
Complement Naive Bayes	x	x	✓	x
Conjunctive Rule	x	x	✓	x
Cost Sensitive Classifier (CSTC)	x	x	✓	x
Dagging	x	x	✓	x
Decision Stump	✓	x	✓	x
Decision Table	x	x	✓	x
Decision Tree	✓	x	x	✓
Decorate	x	x	✓	x
DTNB	x	x	✓	x
END	x	x	✓	x
Ensemble Selection	x	✓	x	x
Fast Large Margin	✓	x	x	x

Algoritmos de classificação e regressão	RapidMiner	Orange	WEKA	KNIME
FT	x	x	✓	x
Gaussian Processes	✓	x	✓	x
Grading	x	x	✓	x
Grid Search	x	x	✓	x
Hierarchical Classification	✓	x	x	x
HNB	x	x	✓	x
Hyper Hyper	✓	x	x	x
Hyper Pipes	x	x	✓	x
IB1	x	x	✓	x
IBk	x	x	✓	x
ID3	✓	x	✓	x
Isotonic Regression	x	x	✓	x
J48	x	x	✓	x
J48 graft	x	x	✓	x
JRip	x	x	✓	x
KNN	✓	✓	x	✓
K Star	x	x	✓	x
LAD Tree	x	x	✓	x
LBR	x	x	✓	x
Lib LINEAR	x	x	✓	x
Lib SVM	✓	x	✓	✓
Linear Discriminant Analysis (LDA)	✓	x	x	x
Linear Regression	✓	✓	✓	✓
LMT	x	x	✓	x
Logistic Regression	✓	✓	✓	✓
LogitBoost	x	x	✓	x
LWL	x	x	✓	x
M5P	x	x	✓	x
M5Rules	x	x	✓	x
Majority	x	✓	x	x
MDD	x	x	✓	x
Meta Cost	✓	x	✓	x

Algoritmos de classificação e regressão	RapidMiner	Orange	WEKA	KNIME
MIBoost	x	x	✓	x
MIDD	x	x	✓	x
MIEMDD	x	x	✓	x
MILR	x	x	✓	x
MINND	x	x	✓	x
MI Optimal Ball	x	x	✓	x
MISMO	x	x	✓	x
MISVM	x	x	✓	x
MI Wrapper	x	x	✓	x
MSP	x	x	✓	x
MSRules	x	x	✓	x
MultiBoost AB	x	x	✓	x
Multi Class Classifier	x	x	✓	x
Multilayer Perceptron	x	x	✓	✓
MultiScheme	x	x	✓	x
Naive Bayes	✓	✓	✓	✓
Naive Bayes Multinomial	x	x	✓	x
Naive Bayes Multinomial Updateable	x	x	✓	x
Naive Bayes Simple	x	x	✓	x
Naive Bayes Updateable	x	x	✓	x
NB Tree	x	x	✓	x
Neural Net	✓	✓	x	x
NNge	x	x	✓	x
One R	x	x	✓	x
Ordinal Class Classifier	x	x	✓	x
Pace Regression	x	x	✓	x
PART	x	x	✓	x
Perceptron	✓	x	x	x
PLS Classifier	x	x	✓	x
PLS Regression	x	✓	x	x
PNN Learner (DDA)	x	x	x	✓
Polynomial Regression	✓	x	x	✓
Prism	x	x	✓	x

Algoritmos de classificação e regressão	RapidMiner	Orange	WEKA	KNIME
Raced Incremental Logit Boost	x	x	✓	x
Random Committee	x	x	✓	x
Random Forest	✓	✓	✓	x
Random SubSpace	x	x	✓	x
Random Tree	✓	x	✓	x
RBF Network	x	x	✓	x
RBF Regressor	x	x	✓	x
Relevance Vector Machine	✓	x	x	x
REP Tree	x	x	✓	x
Ridor	x	x	✓	x
RIPPER	✓	x	x	x
Rotation Forest	x	x	✓	x
Rprop MLP	x	x	x	✓
Seemingly Unrelated Regression	✓	x	x	x
Serialized Classifier	x	x	✓	x
Simple Cart	x	x	✓	x
Simple Linear Regression	x	x	✓	x
Simple Logistic	x	x	✓	x
SimpleMI	x	x	✓	x
Single Rule Induction	✓	x	x	x
SMO	x	x	✓	x
SMO reg	x	x	✓	x
S Pegasos	x	x	✓	x
Stacking	✓	x	✓	x
StackingC	x	x	✓	x
Subgroup Discovery	✓	x	x	x
SVM	✓	✓	x	✓
SVM (Linear/ Evolutionary/ PSO)	✓	x	x	x
Threshold Selector	x	x	✓	x
Tree to rules	✓	x	x	x
User Classifier	x	x	✓	x
Vector Linear Regression	✓	x	x	x
VFI	x	x	✓	x

Algoritmos de classificação e regressão	RapidMiner	Orange	WEKA	KNIME
Vote	✓	x	x	x
Voted Perceptron	x	x	✓	x
WAODE	x	x	✓	x
Winnow	x	x	✓	x
ZeroR	x	x	✓	x

Resultados de performance nos datasets 1, 2 e 3 referentes aos algoritmos de classificação

Tabela 9.2 - Resultados de classificação do *dataset 1*

Precision								
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,7009	0,8756	Classification Tree= 0.7901	Classification Tree = 0.8171	ADTree = 0,844	ADTree = 0,851	0,786	0,803
KNN	0,5042	0,6387	0,691	0,739	Kstar =0,735	Kstar =0,776	0,798	0,828
Multilayer Perceptron	Perceptron =0,7563 Neural Net = 0,419 AutoMLP = 0	Perceptron = 0,6818 Neural Net = 0,5601 AutoMLP = 0,5597	Neural Network = 0,7938	Neural Network = 0,8541	0,739	0,795	PNN = 0,826	PNN = 0,835
Naïve Bayes	0,6772	0,9075	0,6863	0,9872	0,644	0,768	0,618	0,697
SVM/ libSVM	0,6417	0,7167	0,6685	0,7276	0,875	0,577	?	0,56

Recall								
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,8723	0,7071	Classification Tree = 0,7606	Classification Tree = 0,841	ADTree = 0,803	ADTree = 0,883	0,803	0,828
KNN	0,6330	0,5105	0,6543	0,7699	Kstar =0,707	Kstar =0,799	0,777	0,845
Multilayer Perceptron	Perceptron = 0,4787 Neural Net = 0,2021 AutoMLP = 0	Perceptron = 0,8787 Neural Net = 0,7992 AutoMLP = 1	Neural Network = 0,8191	Neural Network = 0,8326	0,739	0,795	PNN = 0,782	PNN = 0,87
Naïve Bayes	0,9149	0,6569	0,9894	0,6444	0,739	0,678	0,612	0,703
SVM/ libSVM	0,6383	0,7197	0,6436	0,749	0,074	0,992	0	1

F-Measure								
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,7800		Classification Tree =0.7751	Classification Tree = 0.8289	ADTree = 0,823	ADTree = 0,867	0,795	0,835

KNN	0,5639		0,6721	0,7541	Kstar=0,721	Kstar =0,788	0,787	0,836
Multilayer Perceptron	Perceptron = 0,7677 Neural Net = 0,6586 AutoMLP = 0,7176		Neural Network = 0,8063	Neural Network = 0,8432	0,739	0,795	PNN = 0,803	PNN = 0,852
Naïve Bayes	0,7600		0,8105	0,7797	0,688	0,72	0,615	0,7
SVM/ libSVM	0,7164		0,6558	0,7381	0,137	0,729	?	0,718
	ROC (AUC)							
Algoritmos de classificação	**RapidMiner**		**Orange**		**Weka**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,856		Classification Tree = 0.8522		ADTree = 0,936		N.A	N.A
KNN	0,5		0,7117		Kstar =0,867		N.A	N.A
Multilayer Perceptron	Perceptron = 0,855 Neural Net = 0,781 AutoMLP = 0,575		Neural Network = 0,9188		0,867		N.A	N.A
Naïve Bayes	0,856		0,9123		0,819		N.A	N.A
SVM/ libSVM	0,774		0,809		0,533		N.A	N.A

	Accuracy							
Algoritmos de classificação	**RapidMiner**		**Orange**		**Weka**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,7798		Classification Tree = 0,8054		ADTree = 84,77%		0,817	
KNN	0,5641		0,719		Kstar =75,87%		0,815	
Multilayer Perceptron	Perceptron = 0,7025 Neural Net = 0,5364 AutoMLP = 0,5597		Neural Network = 0,8267		77 049		PNN = 0,831	
Naïve Bayes	0,7706		0,796		70,49 %		0,663	
SVM/ libSVM	0,6841		0,7024		58,78%		0,56	

	Especificidade							
Algoritmos de classificação	**RapidMiner**		**Orange**		**Weka**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,8722		Classification Tree = 0,8452	0,7553	N.A	N.A	0,828	0,803
KNN	0,6322		0,7699	0,6543	N.A	N.A	0,845	0,777

Multilayer Perceptron	Perceptron = 0,4781 Neural Net = 0,2000 AutoMLP = 0,728	Neural Network = 0,8326	0,8191	N.A	N.A	PNN = 0,87	PNN = 0,782
Naïve Bayes	0,9149	0,8368	N.A	N.A	N.A	0,711	0,612
SVM/ libSVM	0,6386	0,7490	N.A	N.A	N.A	1	0

Sensibilidade

Algoritmos de classificação	**RapidMiner**		**Orange**		**Weka**		**KNIME**	
Classe	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**	**Não**	**Sim**
Decision Tree	0,7071		Classification Tree = 0,7553	0,8452	ADTree = 0,803	ADTree = 0,883	0,803	0,828
KNN	0,5101		0,6543	0,7699	Kstar =0,707	Kstar =0,799	0,777	0,845
Multilayer Perceptron	Perceptron = 0,8789 Neural Net = 0,800 AutoMLP = 0,8291		Neural Network = 0,8298	0,8285	0,739	0,795	PNN = 0,782	PNN = 0,87
Naïve Bayes	0,6569		0,8032	0,8368	0,739	0,678	0,612	0,711
SVM/ libSVM	0,7198		0,6383	0,7490	0,074	0,992	0	1

Tabela 9.3 - Resultados de classificação do *dataset 2*

	Precision							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**
Decision Tree	0,9792	0,8015	Classification Tree = 0.9804	Classification Tree = 0,8274	ADTree = 0,976	ADTree = 0,881	0,974	0,85
KNN	0,9111	0,3124	0,9577	0,8386	Kstar =0,931	Kstar =0,803	0,913	0,324
Multilayer Perceptron	Perceptron = 0,8919 Neural Net = 0,9737 AutoMLP = 0,9735	Perceptron = 0,4143 Neural Net = 0,8766 AutoMLP = 0,8689	Neural Network = 0,9779	Neural Network = 0,8687	0,971	0,87	PNN = 0,9	PNN = 0,481
Naïve Bayes	0,9826	0,6838	0,9804	0,8274	0,983	0,695	0,987	0,902
SVM/ libSVM	0,9761	0,8960	0,9757	0,8955	0,887	0	0,887	0

	Recall							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**

Decision Tree	0,9737	0,8369	Classification Tree = 0,9776	Classification Tree = 0,8464	ADTree = 0,986	ADTree = 0,806	0,982	0,796
KNN	0,9173	0,2956	0,9839	0,6583	Kstar =0,987	Kstar =0,422	0,917	0,315
Multilayer Perceptron	Perceptron = 0,99 Neural Net = 0,9859 AutoMLP = 0,9849	Perceptron = 0,557 Neural Net = 0,7908 AutoMLP = 0,7889	Neural Network = 0,9841	Neural Network = 0,8253	0,985	0,772	PNN = 0,98	PNN = 0,146
Naïve Bayes	0,9490	0,8676	0,9776	0,8464	0,951	0,869	0,987	0,67
SVM/ libSVM	0,9880	0,810	0,9889	0,8061	1	0	1	0

F-Measure

Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**
Decision Tree	0,8184		Classification Tree = 0.979	Classification Tree = 0.8368	ADTree = 0,981	ADTree = 0,842	0,978	0,82
KNN	0,3050		0,9706	0,7376	Kstar =0,958	Kstar =0,553	0,915	0,319
Multilayer Perceptron	Perceptron = 0,965 Neural Net = 0,8298 AutoMLP = 0,8255		Neural Network = 0,981	Neural Network = 0,8465	0,978	0,818	PNN = 0,938	PNN = 0,224
Naïve Bayes	0,7653		0,979	0,8368	0,967	0,772	0,965	0,769

SVM/ libSVM	0,8497	0,9818	0,8485	0,94	0	0,94	NaN

ROC (AUC)

Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**
Decision Tree	0,933		Classification Tree = 0.9808		ADTree = 0,99		N.A	N.A
KNN	0,500		0,9132		Kstar =0,93		N.A	N.A
Multilayer Perceptron	Perceptron = 0,507 Neural Net = 0,939 AutoMLP = 0,925		Neural Network = 0,9891		0,936		N.A	N.A
Naïve Bayes	0,959		0,9808		0,964		N.A	N.A
SVM/ libSVM	0,936		0,9619		0,5		N.A	N.A

Accuracy

Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**
Decision Tree	0,9582		Classification Tree = 0,9628		ADTree = 96,58%		0,961	
KNN	0,8472		0,9472		Kstar =92,32%		0,849	

Multilayer Perceptron	Perceptron = 0,8847 Neural Net = 0,9639 AutoMLP =0, 9628		Neural Network = 0,9662		96,12%		PNN = 0,886	
Naïve Bayes	0,9398		0,9628		94,22%		0,939	
SVM/ libSVM	0,9680		0,9675		88,72%		0,887	
	Especificidade							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**No**	**Yes**	**No**	**Yes**	**Não**	**Sim**	**No**	**Yes**
Decision Tree	0,9893		Classification Tree = 0,8215	Classification Tree = 0,9815	N.A	N.A	0,796	0,982
KNN	0,9176		0,6641	0,9790	N.A	N.A	0,315	0,917
Multilayer Perceptron	Perceptron = 0,99 Neural Net = 0,9859 AutoMLP = 0,9849		Neural Network = 0,8369	Neural Network = 0,9851	N.A	N.A	PNN = 0,155	PNN = 0,977
Naïve Bayes	0,9334		0,8042	0,9837	N.A	N.A	0,67	0,987
SVM/ libSVM	0,9880		0,8061	0,9880	N.A	N.A	0	1
	Sensibilidade							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	

Classe	Não	Sim	No	Yes	No	Yes	No	Yes
Decision Tree	0,7929		Classification Tree = 0,9815	Classification Tree = 0,8215	ADTree = 0,986	ADTree = 0,806	0,982	0,796
KNN	0,2955		0,9790	0,6641	Kstar =0,987	Kstar =0,422	0,917	0,315
Multilayer Perceptron	Perceptron = 0,556 Neural Net = 0,7909 AutoMLP = 0,7890		Neural Network = 0,9851	Neural Network = 0,8369	0,985	0,772	PNN = 0,977	PNN = 0,155
Naïve Bayes	0,8848		0,9837	0,8042	0,951	0,869	0,987	0,67
SVM/ libSVM	0,8102		0,9880	0,8061	1	0	1	0

Tabela 9.4 - Resultados de classificação do dataset 3

	Precision							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**
Decision Tree	0,8599	0,7369	Classification Tree = 0.8726	0,7811	ADTree = 0,866	ADTree = 0,791	0,885	0,666
KNN	0,8296	0,4516	0,8638	0,5819	Kstar =0,863	Kstar =0,556	0,827	0,458

Multilayer Perceptron	Perceptron = 0,8074 Neural Net = 0,8629 AutoMLP = 0,8731	Perceptron = 0,5992 Neural Net = 0,6764 AutoMLP = 0,6893	Neural Network = 0,8802	Neural Network = 0,7388	0,875	0,666	PNN = 0,838	PNN = 0,307
Naïve Bayes	0,8601	0,7121	0,8786	0,7107	0,864	0,723	0,919	0,595
SVM/ libSVM	0,8976	0,6836	0,866	0,7174	0,766	0,449	0,76	0,5
	Recall							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**
Decision Tree	0,9415	0,5165	Classification Tree = 0,9500	Classification Tree = 0,5629	ADTree = 0,955	ADTree = 0,536	0,9	0,629
KNN	0,8191	0,4697	0,8708	0,5670	Kstar = 0,856	Kstar = 0,567	0,83	0,453
Multilayer Perceptron	Perceptron = 0,9374 Neural Net = 0,9180 AutoMLP = 0,9171	Perceptron = 0,2950 Neural Net = 0,5402 AutoMLP = 0,5798	Neural Network = 0,9328	Neural Network = 0,5997	0,906	0,593	PNN = 0,502	PNN = 0,694
Naïve Bayes	0,9331	0,5215	0,9228	0,5979	0,935	0,535	0,835	0,767

SVM/ libSVM	0,9008	0,6761	0,9318	0,5455	0,977	0,06	0,999	0,004
	F-Measure							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**
Decision Tree	0,6071		Classification Tree = 0.9097	Classification Tree = 0.6543	ADTree = 0,909	ADTree = 0,639	0,892	0,647
KNN	0,4603		0,8673	0,5744	Kstar = 0,859	Kstar =0,561	0,828	0,455
Multilayer Perceptron	Perceptron = 0,3947 Neural Net = 0,5890 AutoMLP = 0,6278		Neural Network = 0,9057	Neural Network = 0,662	0,89	0,627	PNN = 0,628	PNN = 0,425
Naïve Bayes	0,6020		0,9001	0,6494	0,898	0,615	0,875	0,67
SVM/ libSVM	0,6798		0,8977	0,6197	0,859	0,106	0,863	0,009
	ROC (AUC)							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**
Decision Tree	0,880		Classification Tree = 0.8859		ADTree = 0,901		N.A	N.A
KNN	0,500		0,7189		Kstar =0,808		N.A	N.A

Multilayer Perceptron	Perceptron = 0,5 Neural Net = 0,890 AutoMLP = 0,838		Neural Network = 0,9083		0,882		N.A	N.A
Naïve Bayes	0,893		0,9008		0,898		N.A	N.A
SVM/ libSVM	0,901		0,8932		0,518		N.A	N.A
	Accuracy							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**
Decision Tree	0,8393		Classification Tree = 0,8568		ADTree = 85,42%		0,835	
KNN	0,7350		0,7976		Kstar =78,66%		0,739	
Multilayer Perceptron	Perceptron = 0,7827 Neural Net = 0,8270 AutoMLP = 0,8359		Neural Network = 0,8526		83,03%		PNN = 0,548	
Naïve Bayes	0,8340		0,8446		83,85%		0,818	
SVM/ libSVM	0,8467		0,8388		75,59%		0,759	
	Especificidade							
Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**

Decision Tree	0,9978	Classification Tree = 0,6345	0,9035	N.A	N.A	0,629	0,9
KNN	0,8192	0,5671	0,8708	N.A	N.A	0,453	0,83
Multilayer Perceptron	Perceptron = 0,9374 Neural Net = 0, 9180 AutoMLP = 0,9360	Neural Network = 0,5997	0,9328	N.A	N.A	PNN = 0,684	PNN = 0,508
Naïve Bayes	0,9331	0,5979	0,9228	N.A	N.A	0,767	0,835
SVM/ libSVM	0,9008	0,5455	0,9318	N.A	N.A	0,004	0,999

Sensibilidade

Algoritmos de classificação	**RapidMiner**		**Orange**		**WEKA**		**KNIME**	
Classe	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**	**<=50**	**>50**
Decision Tree	0,5165		Classification Tree = 0,9035	0,6345	ADTree = 0,955	ADTree = 0,536	0,9	0,629
KNN	0,4697		0,8708	0,5671	Kstar = 0,856	Kstar =0,567	0,83	0,453
Multilayer Perceptron	Perceptron = 0,2950 Neural Net = 0,5402 AutoMLP = 0,5798		Neural Network = 0,9328	0,5997	0,906	0,593	PNN = 0,508	PNN = 0,684
Naïve Bayes	0,5215		0,9228	0,5979	0,935	0,535	0,835	0,767
SVM/ libSVM	0,6761		0,9318	0,5455	0,977	0,06	0,999	0,004

Resultados da accuracy no dataset 1 referentes aos algoritmos de clustering

Tabela 9.5 – Resultados de accuracy no dataset 1 - clustering

Algoritmo K-Means											
RapidMiner							**Orange**				
		Classe							**Classe**		
		Predicted Class							Predicted Class		
Cluster	Actual Class	0	1	Total			**Cluster**	Actual Class	0	1	Total
	0	162	102	264				0	161	114	275
	1	128	108	236				1	129	96	225
		270	TP+TN						257	TP+TN	
		500	TP+FP+FN+TN						500	TP+FP+FN+TN	
Accuracy		0,54	**TP+TN/ TP+ FP +FN +TN**				**Accuracy**		0,514	**TP+TN/ TP+ FP +FN +TN**	
WEKA							**KNIME**				
		Classe							**Classe**		
		Predicted Class							Predicted Class		
Cluster	Actual Class	0	1	Total			**Cluster**	Actual Class	0	1	Total
	0	120	133	253				0	156	100	256
	1	170	77	247				1	134	110	244
		197	TP+TN						266	TP+TN	

		500	TP+FP+FN+TN					500	TP+FP+FN+TN		
Accuracy		0,394	**TP+TN/ TP+ FP +FN +TN**			**Accuracy**		0,532	**TP+TN/ TP+ FP +FN +TN**		
Algoritmo DBScan (min points = 5; épsilon = 5)											
RapidMiner						**WEKA**					
		Classe						**Classe**			
		Predicted Class						Predicted Class			
Cluster	Actual Class	0	1	Total		**Cluster**	Actual Class	0	1	Total	
	0	290	210	500			0	500	0	500	
	1						1				
		290	TP+TN					500	TP+TN		
		500	TP+FP+FN+TN					500	TP+FP+FN+TN		
Accuracy		0,58	**TP+TN/ TP+ FP +FN +TN**			**Accuracy**		1	**TP+TN/ TP+ FP +FN +TN**		
KNIME											
		Classe									
		Predicted Class									
Cluster	Actual Class	0	1	Total							
	0	290	210	500							
	1										
		290	TP+TN								

	500	TP+FP+FN+TN														
Accuracy	0,58	**TP+TN/ TP+ FP +FN +TN**														
Algoritmo DBScan (min points = 6; épsilon = 2)																
RapidMiner								**WEKA**								
		Classe									**Classe**					
		Predicted Class									Predicted Class					
Cluster	Actual Class	0	1	Total				**Cluster**	Actual class	0	1	Total				
	0	290	210	500					0	422		422				
	1								1	7		7				
												429	**71 noise**			
		290	TP+TN							422	**TP+TN**					
		500	TP+FP+FN+TN							429	TP+FP+FN+TN					
Accuracy		0,58	**TP+TN/ TP+ FP +FN +TN**					**Accuracy**		0,98	**TP+TN/ TP+ FP +FN +TN**					
KNIME																
		Classe														
		Predicted Class														
Cluster	Actual Class	0	1	Total												
	0	290	210	500												
	1															
		290	TP+TN													
	500	TP+FP+FN+TN														

Accuracy		0,58	**TP+TN/ TP+ FP +FN +TN**										
Hierarchical Clustering													
RapidMiner							**Orange**						
		Classe							**Classe**				
		Predicted Class							Predicted Class				
Clusters	Actual Class	0	1	Total			**Clusters**	Actual Class	0	1	Total		
999	0			500			**12**	0					
	1							1					
											500		
		0	TP+TN						0	TP+TN			
		0	TP+FP+FN+TN						0	TP+FP+FN+TN			
Accuracy		0	**TP+TN/ TP+ FP +FN +TN**				**Accuracy**		0	**TP+TN/ TP+ FP +FN +TN**			
KNIME													
		Classe											
		Predicted Class											
Clusters	Actual Class	0	1	Total									
	0	0	1	1									
	1	290	209	499									
		209	TP+TN										
		500	TP+FP+FN+TN										
Accuracy		0,418	**TP+TN/ TP+ FP +FN +TN**										

Resultados das regras geradas no dataset 1 referentes aos algoritmos de associação

Tabela 9.6 - Resultados das regras geradas pelo algoritmo *Association Rules* no *dataset 1*- associação, com *min sup= 0,4 e min conf =0,8*

Algoritmo Association Rules						
Regras geradas pelo Orange						
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Urine pushing = yes, Micturition pains = yes		Decision = yes	1.0		0.408
2	Occurrence of nausea = no, Micturition pains = no		Decision = no	0.836		0.425
3	Micturition pains = no		Decision = no	0.836		0.425
4	Micturition pains = yes		Decision = yes	0.831		0.408
Regras geradas pelo KNIME						
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	yes		no	0.918	1.001	0.835
2	true		no	0.91	0.992	0.752
3	no		yes	0.91	1.001	0.835
4	yes, true		no	0.9	0.981	0.669
5	true		yes	0.9	0.99	0.744
6	no, true		yes	0.89	0.979	0.669
7	no		true	0.82	0.992	0.752
8	yes		true	0.818	0.99	0.744
9	yes, no		true	0.802	0.97	0.669

Tabela 9.7 - Resultados das regras geradas pelo algoritmo *FPGrowth* no dataset 1- associação, com min sup= 0,4 e min conf =0,8

Algoritmo FPGrowth						
Regras geradas pelo RapidMiner						
Nº	**Antecedente**	→	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Burning_of_urethra		Urine_pushing	1.0	1.5	0.416
2	Urine_pushing		Temperature_of_patient	0.875	1.05	0.583
3	Micturition_pains		Temperature_of_patient	0.830	0.996	0.408
4	Micturition_pains		Urine_pushing	0.830	1.245	0.408
Regras geradas pela WEKA						
Nº	**Antecedente**	→	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Decision=yes: 59		Urine_pushing=yes: 59	1.0	1.5	N.A.
2	Burning_of_urethra=yes: 50		Urine_pushing = yes: 50	1.0	1.5	N.A.
3	Temperature of patient=True, Decision=Yes:49		Urine_pushing = yes: 49	1.0	1.5	N.A.
4	Urine pushing=yes, Micturition pains=yes: 49		Decision=yes: 49	1.0	2.03	N.A.
5	Micturition pains=yes, Decision=yes: 49		Urine pushing=yes: 49	1.0	1.5	N.A.
6	Urine pushing=yes: 80		Temperature of patient=true: 70	0.88	1.05	N.A.
7	Micturition pains=yes: 59		Temperature of patient=true: 49	0.83	1.0	N.A.
8	Decision=yes: 59		Temperature of patient=true: 49	0.83	1.0	N.A.
9	Micturition pains=yes: 59		Urine pushing=yes: 49	0.83	1.25	N.A.
10	Micturition pais=yes: 59		Decision=yes: 49	0.83	1.69	N.A.
11	Decision=yes: 59		Micturition pains=yes: 49	0.83	1.69	N.A.

12	Decision=yes: 59		Temperature of patient=true, urine pushing=yes: 49	0.83	1.42	N.A.
13	Urine pushing=yes, Decision=yes: 59		Temperature of patient=true: 49	0.83	1.0	N.A.
14	Micturition pais=yes: 59		Urine_pushing=yes, Decision=yes: 49	0.83	1.69	N.A.
15	Decision=yes: 59		Urine pushing=yes,Micturition pains=yes: 49	0.83	2.03	N.A.
16	Urine pushing=yes, Decision=yes: 59		Micturition pains=yes: 49	0.83	1.69	N.A.
Regras geradas pelo KNIME						
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	yes		no	0.918	1.001	0.835
2	true		no	0.91	0.992	0.752
3	no		yes	0.91	1.001	0.835
4	yes, true		no	0.9	0.981	0.669
5	true		yes	0.9	0.99	0.744
6	no, true		yes	0.89	0.979	0.669
7	no		true	0.82	0.992	0.752
8	yes		true	0.818	0.99	0.744
9	yes, no		true	0.802	0.97	0.669

Tabela 9.8 - Resultados das regras geradas pelo algoritmo *Apriori* no dataset 1- associação, com min sup= 0,4 e min conf =0,8

Algoritmo Apriori						
Regras geradas pela WEKA						
Nº	**Antecedente**	→	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Micturition_pains = no 61		Occurrence of nausea = no 61	1.0	N.A.	N.A.
2	Decision = yes 59		Urine pushing = yes 59	1.0	N.A.	N.A.
3	Temperature of patient = true, Micturition pains = no 51		Occurrence of nausea = no 51	1.0	N.A.	N.A.
4	Micturition_pains = no, Decision = no 51		Occurrence of nausea = no 51	1.0	N.A.	N.A.
5	Occurrence of nausea = no; Decision = no 51		Micturition pains = no 51	1.0	N.A.	N.A.
6	Lumbar pain = no 50		Occurrence of nausea = no 50	1.0	N.A.	N.A.
7	Burning of urethra = yes 50		Urine pushing = yes 50	1.0	N.A.	N.A.
8	Temperature of patient = true, decision = yes 49		Urine pushing = yes 49	1.0	N.A.	N.A.
9	Micturition pains = yes; Decision = yes 49		Urine pushing = yes 49	1.0	N.A.	N.A.
10	Urine pushing = yes; Micturition pains = yes 49		Decision = yes 49	1.0	N.A.	N.A.
11	Urine pushing = yes 80		Temperature of patient = true 70	0.88	N.A.	N.A.
12	Lumbar pain = yes 80		Temperature of patient = true 60	0.86	N.A.	N.A.
13	Burning of urethra = no 70		Temperature of patient = true 60	0.86	N.A.	N.A.
14	Micturition pains = no 61		Temperature of patient = true 51	0.84	N.A.	N.A.
15	Decision = no 61		Temperature of patient = true 51	0.84	N.A.	N.A.

16	Decision = no 61		Occurrence of nausea = no 51	0.84	N.A.	N.A.
17	Decision = no 61		Lumbar pains = yes 51	0.84	N.A.	N.A.
18	Decision = no 61		Micturition pains = no 51	0.84	N.A.	N.A.
19	Micturition pains = no 61		Decision = no 51	0.84	N.A.	N.A.
20	Occurrence of nausea = no; Urine pushing = yes 61		Temperature of patient = true 51	0.84	N.A.	N.A.
21	Occurrence of nausea = no; Micturition pains = no 61		Temperature of patient = true 51	0.84	N.A.	N.A.
22	Micturition pains = no 61		Temperature of patient = true; Occurrence of nausea = no 51	0.84	N.A.	N.A.
23	Occurrence of nausea = no; Micturition pains = no 61		Decision = no 51	0.84	N.A.	N.A.
24	Decision = no 61		Occurrence of nausea = no; Micturition pains = no 51	0.84	N.A.	N.A.
25	Micturition pains = no 61		Occurrence of nausea = no; Decison = no 51	0.84	N.A.	N.A.
26	Micturition pains = yes 59		Temperature of patient = true 49	0.83	N.A.	N.A.
27	Decision = yes 59		Temperature of patient = true 49	0.83	N.A.	N.A.
28	Micturition pains = yes 59		Urine pushing = yes 49	0.83	N.A.	N.A.
29	Decision = yes 59		Micturition pains = yes 49	0.83	N.A.	N.A.
30	Micturition pains = yes 59		Decisin = yes 49	0.83	N.A.	N.A.
31	Urine pushing = yes; Decision = yes 59		Temperature of patient = true 49	0.83	N.A.	N.A.
32	Decision = yes 59		Temperature of patient = true; Urine pushing = yes 49	0.83	N.A	N.A.

33	Urine pushing = yes; Decision = yes 59		Micturition pains = yes 49	0.83	N.A.	N.A.
34	Decision = yes 59		Urine pushing = yes; Micturition pains = yes 49	0.83	N.A.	N.A.
35	Micturition pains = yes 59		Urine pushing = yes; Decision = yes 49	0.83	N.A.	N.A.
Regras geradas pelo KNIME (ARL (B))						
Nº	**Antecedente**	**→**	**Consequente**	**Confiança**	**Lift**	**Suporte**
1	Yes		No	91.8	0.993	101
2	True		No	91	0.984	91
3	No		Yes	91	0.993	101
4	True		Yes	90	0.982	90
5	True, yes		No	90	0.973	81
6	True, no		Yes	89	0.971	81
7	No		True	82	0.984	91
8	Yes		True	81.8	0.982	90
9	Yes, no		True	80.2	0.962	81
Regras geradas pelo KNIME (ISF(B))						
Nº	**Item set**	**Item set size**	**Item set support**	**Confiança**	**Lift**	
1	True, yes	2	90	N.A	N.A	
2	True, yes, no	3	81	N.A	N.A	
3	True, no	2	91	N.A	N.A	
4	Yes, no	2	101	N.A	N.A	