NOVA IMS Information Management School

MGI

Mestrado em Gestão de Informação
Master Program in Information Management

Métodos de Aprendizagem Automática

Um estudo baseado na avaliação e previsão de clientes bancários

Flávia Alexandra Jorge Serras

Trabalho de Projeto apresentado como requisito parcial para obtenção do grau de Mestre em Gestão de Informação

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

MÉTODOS DE APRENDIZAGEM AUTOMÁTICA: UM ESTUDO BASEADO NA AVALIAÇÃO E PREVISÃO DE CLIENTES BANCÁRIOS

por

Flávia Alexandra Jorge Serras

Trabalho de Projeto apresentado como requisito parcial para a obtenção do grau de Mestre em Gestão de Informação, Especialização em Gestão do Conhecimento e Business Intelligence

Orientador: Leonardo Vanneschi

Coorientador: Mauro Castelli

Novembro 2015

Às minhas estrelinhas do céu, em especial a ti Avô.
Ao homem que foste e ao homem que continuarás a ser na minha memória e no meu coração.

AGRADECIMENTOS

"*Feeling gratitude and not expressing it is like wrapping a present and not giving it.*"
William Arthur Ward

Este trabalho resultou de um longo caminho (com algumas pedras...) percorrido com a ajuda de pessoas fabulosas que, felizmente, tive a sorte de encontrar na minha vida. Não faria qualquer sentido começar este trabalho sem mencionar estas pessoas e sem expressar o quão grata eu lhes estou!

Começo por mencionar o Professor Dr. Leonardo Vanneschi, a quem eu devo todo o meu respeito e admiração pelos seus vastos conhecimentos, pela sua competência e pelo seu profissionalismo. Obrigada pelos seus ensinamentos, pelas suas críticas e pela sua assertividade. Obrigada pela sua compreensão, paciência, carinho e apoio. Agradeço ainda por, tão prontamente, ter aceitado orientar-me neste projeto, aumentando a admiração que tenho por si e realizando assim o sonho, que já me acompanhava há algum tempo, de trabalhar consigo.

Ao meu coorientador, o Professor Dr. Mauro Castelli, dirijo-lhe as minhas palavras de agradecimento, pelos seus valiosos conhecimentos, por todo o apoio técnico que me deu e pelo carinho demonstrado. Hoje reconheço e admiro as suas capacidades e qualidades enquanto investigador e orientador. E se antes, por não o conhecer, precisei da indicação do Professor Dr. Leonardo Vanneschi, hoje, sei que não precisaria de qualquer ajuda para escolher o meu coorientador!

Dirijo-me ainda à minha coordenadora de núcleo Marta Veloso que, com as suas meigas palavras e com o seu enorme coração, me conseguiu transmitir a calma e a confiança necessárias para levar este projeto a bom porto. Obrigada pela compreensão, pela flexibilidade e acima de tudo pela amizade.

Não podia não mencionar a mulher mais importante da minha vida. A minha heroína, a minha amiga, a minha mãe. Obrigada mãe pelo amor incondicional, pelas palavras tranquilizadoras, pelas saudades suportadas e pela tolerância à minha rabugice extra.

Obrigada ao meu pai por acreditar em mim e por me motivar a ir sempre mais além.

Obrigada à minha irmã por continuar a ser a minha fonte de inspiração e o modelo de pessoa que continuo a imitar.

Obrigada ao Valter, à Fi e à Inês pela paciência, pelas críticas, pelos conselhos e pelas dicas. Obrigada pela vossa disponibilidade, pelo vosso carinho e pela vossa amizade.

Obrigada aos meus amigos e à minha família por perdoarem a minha ausência e por estarem sempre do meu lado.

Para terminar, dirijo o meu último, enorme e especial obrigada ao Fábio. Obrigada pelo companheirismo, pela amizade, pela confiança, pela paciência e pelo teu amor. Obrigada por seres quem és e por estares sempre ao meu lado. Obrigada por acreditares em mim e por me incentivares a lutar. Por tudo, obrigada!

RESUMO

Data Mining surge, hoje em dia, como uma ferramenta importante e crucial para o sucesso de um negócio. O considerável volume de dados que atualmente se encontra disponível, por si só, não traz valor acrescentado. No entanto, as ferramentas de *Data Mining*, capazes de transformar dados e mais dados em conhecimento, vêm colmatar esta lacuna, constituindo, assim, um trunfo que ninguém quer perder.

O presente trabalho foca-se na utilização das técnicas de *Data Mining* no âmbito da atividade bancária, mais concretamente na sua atividade de *telemarketing.*

Neste trabalho são aplicados catorze algoritmos a uma base de dados proveniente do *call center* de um banco português, resultante de uma campanha para a angariação de clientes para depósitos a prazo com taxas de juro favoráveis. Os catorze algoritmos aplicados no caso prático deste projeto podem ser agrupados em sete grupos: Árvores de Decisão, Redes Neuronais, *Support Vector Machine*, *Voted Perceptron*, métodos *Ensemble,* aprendizagem Bayesiana e Regressões. De forma a beneficiar, ainda mais, do que a área de *Data Mining* tem para oferecer, este trabalho incide ainda sobre o redimensionamento da base de dados em questão, através da aplicação de duas estratégias de seleção de atributos: *Best First* e *Genetic Search*.

Um dos objetivos deste trabalho prende-se com a comparação dos resultados obtidos com os resultados presentes no estudo dos autores Sérgio Moro, Raul Laureano e Paulo Cortez (Sérgio Moro, Laureano, & Cortez, 2011).

Adicionalmente, pretende-se identificar as variáveis mais relevantes aquando da identificação do potencial cliente deste produto financeiro.

Como principais conclusões, depreende-se que os resultados obtidos são comparáveis com os resultados publicados pelos autores mencionados, sendo os mesmos de qualidade e consistentes. O algoritmo *Bagging* é o que apresenta melhores resultados e a variável referente à duração da chamada telefónica é a que mais influencia o sucesso de campanhas similares.

PALAVRAS-CHAVE

Data Mining; *Knowledge Discovery Database*; *Machine Learning*

ABSTRACT

Data Mining becomes more and more an important and crucial tool for the success of a business. The substantial amount of data that is nowadays available, by itself, does not add any value. However, the Data Mining's tools, being able to transform data into knowledge, deal with this gap. Therefore, they are the asset that nobody wants to lose.

The aim of this project is to use Data Mining techniques within the framework of the banking area, more precisely, regarding telemarketing.

In this project, fourteen algorithms are applied to a database that arose from the call center of a Portuguese bank, as a result of a campaign for acquiring clients to deposits with favorable interest rates. The fourteen algorithms applied in this practical case can be aggregated into seven groups: Decision Trees, Artificial Neural Networks, Support Vector Machine, Voted Perceptron, Ensemble methods, Bayesian Learning and Regressions. In order to increase the benefits of the Data Mining techniques, this project also focuses on the resizing of the database, by applying two strategies for attributes' selection: Best First and Genetic Search.

One of the goals of this project is the comparison between its results and the outcomes presented in the working paper of Sérgio Moro, Raul Laureano and Paulo Cortez (Sérgio Moro, Laureano, & Cortez, 2011). Additionally, this project aims to select the most relevant variables to identify potential clients of this financial product.

The main conclusion of this project is that its results are comparable with the published results of the aforementioned authors, regarding quality as well as consistency. The Bagging algorithm is the one that produces better results and the variable related to the duration of the call is the one that most influences the success of similar campaigns.

KEYWORDS

Data Mining; *Knowledge Discovery Database*; *Machine Learning*

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

KDD	Knowledge Discovery Database
DM	Data Mining
ML	Machine Learning
ANN	Artificial Neural Networks
SVM	Support Vector Machine
BN	Bayesian Networks
CFS	*Correlation-based Feature Subset Selection*

1. INTRODUÇÃO

O considerável número de campanhas de marketing com as quais nos deparamos atualmente não para de aumentar. Como consequência, o seu efeito já não é tão eficaz como quando estas eram em menor número (Sérgio Moro et al., 2011). Surge, assim, a necessidade de investir em campanhas de marketing direto.

Segundo Kotler e Armstrong "marketing direto consiste no contacto direto com os seus consumidores-alvo, muitas vezes numa base interativa de um para um" (Kotler & Armstrong, 2012). Ou seja, consiste num processo que seleciona o público-alvo de determinada campanha, através do estudo das suas caraterísticas e necessidades (Ling & Li, 1998). Desta forma a campanha é direcionada para um público específico, produzindo assim melhores resultados. Vendas através de contacto pessoal, vendas por telefone, campanhas por e-mail, campanhas por catálogo e campanhas online são alguns dos exemplos mais comuns de campanhas de marketing direto (Kotler & Armstrong, 2012). O presente projeto foca-se nas vendas por telefone, mais conhecidas por telemarketing.

De uma forma generalizada, telemarketing consiste na oferta de produtos e/ou serviços a clientes através do contacto telefónico. Apesar de ser um método com grande potencial na obtenção de novos clientes, a tolerância das pessoas e a sua recetividade a este tipo de campanhas é cada vez menor, consequência direta do seu excesso de utilização por parte das empresas (Queensland Government, 2014). Os bancos, as seguradoras e as indústrias de retalho utilizam cada vez mais o telemarketing para angariar novos clientes (Ling & Li, 1998). Para que uma campanha deste cariz tenha sucesso, esta precisa de ser muito bem segmentada, de forma a maximizar a coincidência entre os clientes contactados e aqueles que vão de facto adquirir o produto/serviço alvo da campanha (Queensland Government, 2014).

Data Mining surge assim como uma ferramenta que auxilia a atividade de telemarketing, dando potencial aos resultados obtidos.

1.1. OBJETIVOS

Este projeto tem, na sua génese, um conjunto de dados proveniente do *call center* de um banco português. Este conjunto de dados é resultado direto de várias campanhas de marketing direcionadas para a angariação de clientes de depósitos a prazo com taxas de juro favoráveis. Esta base de dados foi previamente utilizada num estudo desenvolvido pelos autores Sérgio Moro, Raul Laureano e Paulo Cortez (Sérgio Moro et al., 2011), onde foram utilizadas técnicas de *Data Mining* para auxiliar a atividade de telemarketing, através da identificação de caraterísticas fulcrais para o sucesso das campanhas.

O objetivo geral deste projeto prende-se com a confirmação de que as técnicas de *Data Mining* constituem uma ferramenta importante e com potencial no âmbito do setor financeiro.

Como objetivos mais específicos realçam-se: (1) a utilização de técnicas de *Data Mining*, para além das utilizadas pelos autores mencionados, com o intuito de alcançar melhores resultados, (2) a pré-seleção de atributos antes da aplicação das várias técnicas de *Data Mining* para perceber o seu impacto nos resultados obtidos e (3) a identificação das variáveis com maior relevância nesta campanha, de forma a potenciar os resultados de próximas campanhas com moldes semelhantes.

1.2. RELEVÂNCIA E MOTIVAÇÃO

Devido à crise financeira que ainda se sente atualmente, existe uma enorme pressão sobre os bancos europeus para melhorarem os seus ativos financeiros (Sérgio Moro et al., 2011). O papel dos depósitos tem vindo a sofrer alterações ao longo do tempo. No entanto, estes constituem um meio para aumentar os fundos dos bancos. Entre 1990 e 2009, este instrumento financeiro constituiu a maior fonte de financiamento das instituições financeiras (Allen & Carletti, 2013).

A relevância deste projeto centra-se na necessidade de tornar a atividade de telemarketing mais eficiente uma vez que:

- Os depósitos têm um papel essencial no financiamento da indústria financeira e a forma mais comum de angariar clientes para este instrumento é através desta atividade;
- O sucesso da atividade de telemarketing é cada vez menor, tal como mencionado anteriormente, pelo facto das empresas não estarem a segmentar os seus clientes, estando por isso a sobrelotar a utilização deste método.

1.3. ESTRUTURA

Este projeto começa com o atual capítulo introdutório que visa apresentar o trabalho desenvolvido, explicitando os principais objetivos e a relevância e motivação que levaram à escolha do tema.

O capítulo 2 apresenta uma introdução ao *Data Mining*, onde se apresenta a sua importância, a sua utilidade e os seus desafios e onde se explora e clarifica a relação e as diferenças entre esta área e a Descoberta de Conhecimento em Bases de Dados e Aprendizagem Automática.

O capítulo 3 apresenta as várias técnicas de Aprendizagem Automática utilizadas neste projeto detalhando a sua metodologia e as suas vantagens e desvantagens.

O capítulo 4 recai sobre a explicação de técnicas de seleção de atributos, a sua importância e as várias decisões relacionadas com a seleção da técnica escolhida.

No capítulo 5 serão apresentados os dados utilizados neste projeto bem como os resultados obtidos.

Por fim, o capítulo 6 resume as principais conclusões deste projeto e expõe alguns pontos que, apesar de não terem sido explorados no presente trabalho, conferem interesse e relevância.

2. INTRODUÇÃO AO *DATA MINING*

A quantidade de dados existente no mundo, não para de aumentar (I. H. Witten, Frank, & Hall, 2011). Estima-se que mais de 90% da totalidade do conhecimento que temos hoje começou a ser adquirido por volta de 1950 (Nisbet, Elder, & Miner, 2009).

Um fator crítico de sucesso das empresas é a sua capacidade de tomar partido de toda a informação disponível. Este desafio torna-se mais difícil com o constante aumento do volume de informação, tanto interno como externo às empresas uma vez que quanto maior for a quantidade de informação disponível, menor será a proporção de dados que o ser humano consegue analisar (Angelis, Polzonetti, & Re, n.d.; I. H. Witten et al., 2011).

A informação dispersa pelo volume de dados disponível poderá ser decisiva no sucesso de um negócio e uma mais-valia aquando da tomada de decisão. Torna-se assim indispensável encontrar a melhor forma de extrair toda a informação que se encontra camuflada numa base de dados. As teorias e ferramentas capazes de auxiliar os humanos na extração de informação útil dos grandes volumes de dados disponíveis são a base da descoberta de conhecimento em bases de dados (Lavalle, Hopkins, Lesser, Shockley, & Kruschwitz, 2010).

2.1. DESCOBERTA DE CONHECIMENTO EM BASES DE DADOS

A Descoberta de Conhecimento em Bases de Dados, doravante designada por KDD (do inglês *Knowledge Discovery Database*), pode ser considerada como um campo interdisciplinar que envolve diferentes conceitos de aprendizagem automática, de estatística, de consultas em bases de dados e de visualização (Wang, 2009). É um processo que extrai dos dados padrões novos, válidos, com potencial e com significado (Fayyad, Piatetsky-Shapiro, & Smyth, 1996).

Os sistemas KDD enfrentam, no entanto, alguns problemas com as bases de dados reais uma vez que estas tendem a ser dinâmicas, incompletas, redundantes, com ruído e de grandes dimensões (Matheus, Chan, & Piatetsky-Shapiro, 1993). Segundo Fayyad, a descoberta de conhecimento em bases de dados traduz-se num processo iterativo e interativo que envolve cinco etapas, tal como se pode ver na Figura 1.

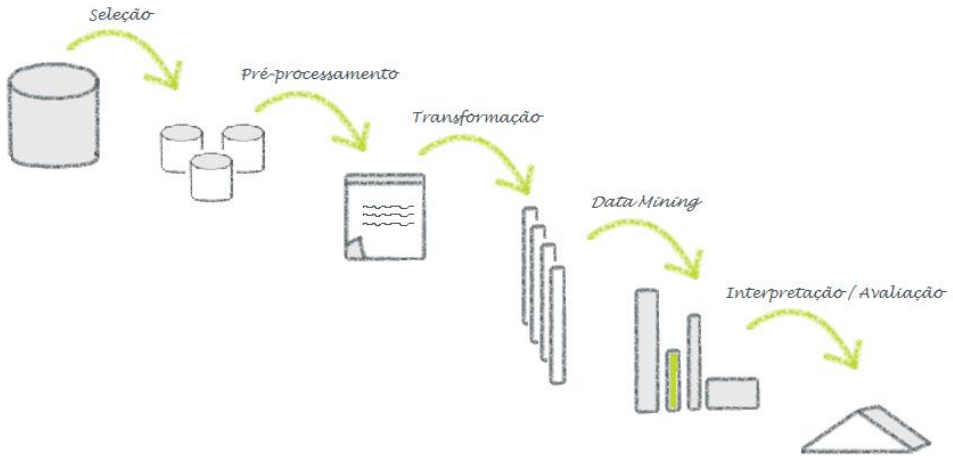

Figura 1 - Processo KDD (adaptado de Fayyad et al., 1996)

Estas etapas, de uma forma resumida, consistem (1) na seleção ou segmentação de um subconjunto de dados relevantes para um objetivo em concreto, (2) na eliminação de informação desnecessária e na consistência do formato dos dados, (3) na transformação dos dados em dados adequados e úteis para a etapa de *Data Mining*, (4) na extração de padrões dos dados e (5) na conversão dos padrões obtidos em conhecimento (Pujari, 2001).

São as três primeiras etapas do processo KDD que garantem a qualidade dos resultados obtidos nas duas últimas (Fayyad et al., 1996).

Os dados subjacentes a este projeto foram previamente e, fora do âmbito deste trabalho, submetidos às três primeiras etapas do processo KDD. No entanto e de forma a tentar melhorar os resultados obtidos, a etapa 2 será novamente aplicada à base de dados disponível, através da redução do número de variáveis existentes (tópico abordado no capítulo 4). O grande foco deste projeto recai, essencialmente, nas duas últimas etapas do processo KDD: *Data Mining* e Interpretação/Avaliação.

2.2. DATA MINING

Data Mining é uma área relativamente recente que começou a ser desenvolvida nos anos 90 e que ganhou identidade própria nos primeiros anos do século XXI (Nisbet et al., 2009). Alguns autores defendem KDD e *Data Mining* como sinónimo (Kononenko & Matjaz, 2007). No entanto, e tal como defende Fayyad na Figura 1, *Data Mining* é uma etapa específica do processo KDD.

O que é, afinal, *Data Mining (DM)*? A definição de DM depende, em grande parte, do *background* e da visão de cada autor (Friedman, 1997). Segundo vários autores da literatura, *Data Mining* é

- *"…*a extração de informação implícita, anteriormente desconhecida e potencialmente útil dos dados" (I. H. Witten et al., 2011);
- *"…*utilizado para descobrir padrões e relações nos dados, com ênfase em grandes bases de dados"(Friedman, 1997);
- *"…*a aplicação de algoritmos específicos para a extração de padrões dos dados" (Fayyad et al., 1996);
- "…um método direcionado para a descoberta de mensagens escondidas, tais como tendências, padrões e relações existentes nos dados" (Hsu & Ho, 2012).

No fundo, o processo de DM consiste na atribuição de significado aos dados e na resultante extração de conhecimento. As ferramentas de DM permitem às organizações tomar decisões fundamentadas e eficientes, uma vez que preveem tendências e acontecimentos através da leitura de padrões encobertos pelas bases de dados (Silltow, 2006).

Data Mining consiste assim na junção de várias áreas de interesse já bastante cimentadas, tais como a análise de dados tradicional, a inteligência artificial e a aprendizagem automática (Nisbet et al., 2009).

2.2.1. Algumas aplicações de *Data Mining*

DM e a Descoberta de Conhecimento em Bases de Dados são duas áreas que trazem consigo a necessidade de resolver problemas práticos (Jadhav & Pawar, 2011). Existem inúmeras aplicações práticas de DM sendo, algumas, elencadas de seguida.

Cross-Selling. O banco Dexia realizou em 2001 um projeto de DM de grande dimensão. Esse projeto tinha como objetivo analisar os clientes de determinados produtos bancários, tentando perceber que outros produtos seriam também do interesse desses clientes (SAS, 2001).

Deteção de Intrusos. Nos dias de hoje, a internet é mais utilizada do que nunca. O número de ameaças à segurança e confidencialidade dos recursos da internet é cada vez maior. A ideia chave por detrás da utilização das técnicas de DM incide sobre a descoberta de padrões úteis e consistentes, relativos ao comportamento do utilizador e às caraterísticas do sistema para que, com o auxilio desta informação, sejam detetadas anomalias e intrusões (Lee & Stolfo, 1998).

***Credit Scoring*.** Os modelos de *credit scoring* foram desenvolvidos com o intuito de, com base nas caraterísticas de um cliente, determinar a sua probabilidade de entrar em *default* no caso da contratação de um empréstimo. Estudos empíricos referentes a *credit scoring* confirmam a utilização de técnicas de *Data Mining* bem como do seu significante contributo (Koh, Tan, & Goh, 2006).

Previsão do abandono de clientes. Esta prática consiste na previsão de clientes que estão em risco de abandonar uma empresa. Uma das grandes preocupações das empresas de telecomunicações prende-se com o facto de os clientes trocarem a empresa atual pela concorrência. Este é um processo muito caro, uma vez que é mais barato manter os atuais clientes do que adquirir novos. Assim, uma aplicação prática de DM permite identificar que clientes pretendem abandonar a empresa e quando (Jadhav & Pawar, 2011).

Segmentação de clientes. Os clientes são o ativo mais importante de um negócio. É impossível perspetivar um negócio se os clientes não estiverem satisfeitos. Assim, é necessário analisá-los de modo a orientar os objetivos das empresas e transmitir a mensagem certa ao cliente certo e no momento certo. Os modelos de DM surgem com o objetivo de compreender os clientes e de prever os seus comportamentos (Tsiptsis & Chorianopoulos, 2009).

2.2.2. Desafios do *Data Mining*

Até agora foi possível absorver vários pontos fortes de DM (descoberta de padrões em bases de dados, capacidade de predição…). Existem, no entanto, limitações que põem à prova as capacidades desta ciência. Seguem-se algumas delas.

Objetivo inicial da informação. As bases de dados, quando criadas, são desenhadas com determinado objetivo. Assim, aplicar técnicas de DM sobre estas bases de dados, para a descoberta de conhecimento sobre um tema diferente do pensado inicialmente, pode constituir um desafio (Pujari, 2001).

Ruído e dados omissos. O ruído tem um impacto negativo na interpretação dos dados e reduz significativamente a precisão destes. Os dados omissos derivam essencialmente da privacidade e da

indisponibilidade da informação, comprometendo assim a qualidade dos dados (Hashemi & Yang, 2009).

Interação do utilizador e conhecimento prévio. Ter conhecimento prévio das ferramentas de DM potencia a sua utilização. Normalmente, os analistas não são especialistas neste tipo de ferramentas, subaproveitando assim os recursos disponíveis (Pujari, 2001).

Volume e atualizações. As bases de dados disponíveis são dinâmicas e de grandes dimensões. Desta forma, à medida que estas são atualizadas, por inserção, atualização ou remoção de dados, torna-se difícil garantir a consistência e a precisão dos dados disponíveis (Pujari, 2001).

2.3. *DATA MINING* E APRENDIZAGEM AUTOMÁTICA

Data Mining é uma disciplina que se encontra relacionada com várias áreas, uma das quais a Aprendizagem Automática (Fayyad et al., 1996). A Aprendizagem Automática (adiante designada por ML, do inglês *Machine Learning*) é "uma área focada no desenvolvimento de teorias computacionais de aprendizagem e na construção de sistemas de aprendizagem" (Michalski, Carbonell, & Mitchell, 1986). Para o âmbito deste projeto, ML pode ser descrita como o conjunto de alguns princípios e algoritmos utilizados em DM (Kononenko & Matjaz, 2007).

A Figura 2 retrata, de uma forma sumária, a relação existente entre *Machine Learning*, *Data Mining* e *Knowledge Discovery Database*.

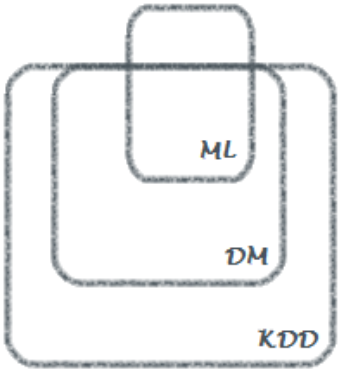

Figura 2 - Relação entre Machine Learning, Data Mining e Knowledge Discovery Database (retirado de Kononenko & Matjaz, 2007)

Como já referido anteriormente, KDD consiste num processo composto por várias etapas, uma das quais DM. DM surge assim como parte integrante do processo KDD. No que diz respeito a ML, esta área não se cinge apenas à área de DM na medida em que engloba outros campos que ultrapassam o âmbito de DM.

Os capítulos seguintes aprofundam a relação existente entre *Data Mining* e Aprendizagem Automática através da apresentação e explicação das várias técnicas de ML utilizadas em DM.

3. TÉCNICAS DE CLASSIFICAÇÃO

Os métodos de aprendizagem automática constituem uma ferramenta poderosa que consegue realizar operações de otimização com a mínima intervenção humana (Cui, Wong, & Lui, 2006). O presente capítulo destina-se a apresentar, numa primeira fase, alguns conceitos introdutórios e, numa segunda e última fase, a enumeração e descrição dos métodos de aprendizagem automática utilizados no âmbito deste projeto de mestrado.

3.1. CONCEITOS INICIAIS

3.1.1. Modelos Preditivos

Existem inúmeros métodos utilizados para criar modelos preditivos e estão constantemente a ser desenvolvidos mais (Finlay, 2014). O grande objetivo da classificação, um caso específico dos modelos preditivos, consiste em criar uma regra que, com base em dados externos, consegue assignar um objeto a uma ou mais classes (Maimon & Rokack, 2005). O caso específico deste trabalho consiste num problema de classificação.

Antes de entrar em maior detalhe sobre os vários modelos subjacentes a este projeto, é necessário relembrar que "os dados são o combustível que conduz o processo analítico" (Finlay, 2014). Existem, assim, dois tipos de dados que são necessários constar da amostra utilizada para desenvolver um modelo:

1. **Dados preditivos**, utilizados para prever;
2. **Dados comportamentais**, que consistem no comportamento que se pretende prever.

Os métodos utilizados para construir modelos preditivos começam por aplicar técnicas matemáticas/estatísticas de forma a encontrar a relação existente entre estes dois tipos de dados. A relação encontrada é capturada e absorvida pelo modelo preditivo. Depois do modelo preditivo ser criado, este pode ser aplicado a novos casos (Finlay, 2014).

3.1.2. Aprendizagem Supervisionada e Não Supervisionada

Falar em algoritmos de aprendizagem automática conduz, normalmente, à referência de dois paradigmas: aprendizagem supervisionada e não supervisionada (Stimpson & Cummings, 2014). Na aprendizagem supervisionada são apresentados dois conjuntos de dados, o conjunto de *input* e o conjunto de *output* esperado (Winandy, Borges Filho, & Bento, 2007). Na aprendizagem não supervisionada apenas é apresentado um conjunto de *input* (Alpaydin, 2004).

A aprendizagem supervisionada está diretamente relacionada com a previsão enquanto a aprendizagem não supervisionada se relaciona mais com a descoberta de padrões num conjunto de dados (Stimpson & Cummings, 2014).

Subjacente a este projeto está uma aprendizagem supervisionada.

3.1.3. O Problema da sobreaprendizagem

Para o sucesso da aplicação das técnicas de ML, uma das abordagens defendidas resume-se nas três etapas que se seguem (Ling & Li, 1998):

1. Dividir a base de dados em dois conjuntos: conjunto de treino e conjunto de teste;
2. Aplicar os métodos de ML sobre o conjunto de treino;
3. Aplicar os mesmos métodos sobre o conjunto de teste e avaliar os resultados decorrentes desta aplicação. Repetir os passos anteriores, se necessário.

Posto isto, torna-se oportuno abordar um tema até aqui não mencionado e que os modelos preditivos devem evitar: a sobreaprendizagem dos dados (também conhecida por *overfitting*). Esta é uma das grandes preocupações que surgem quando se utiliza o conjunto de treino para desenvolver um modelo (Dean, 2014). A sobreaprendizagem consiste no ajustamento excessivo do modelo ao conjunto de treino, dificultando a realização de boas classificações perante novos dados (Hand, Mannila, & Smyth, 2001).

3.1.4. A escolha do melhor modelo

Selecionar o melhor modelo não é fácil. E a dificuldade advém essencialmente do termo "melhor". Existem várias medidas utilizadas para a seleção do melhor modelo, tais como, Lift, Ganho, Critério de Informação de Akaike, Critério de Informação Bayesiana e Kolmogorov-Smirnov (Dean, 2014). As medidas utilizadas neste projeto para avaliar a qualidade dos modelos são: a matriz de confusão, medidas de classificação e a curva ROC.

Tabela 1 - Matriz de Confusão (retirado de Dean, 2014)

Classe real	Classe obtida pelo classificador	
	Negativo	Positivo
Negativo	Verdadeiros Negativos (TN)	Falsos Positivos (FP)
Positivo	Falsos Negativos (FN)	Verdadeiros Positivos (TP)

ROC (Receiver Operating Characteristics) ou Curva ROC, consiste num gráfico que confronta a taxa de verdadeiros positivos (eixo das abscissas) com a taxa de falsos positivos (eixo das ordenadas) (ver

Figura 3 e Tabela 2). A coordenada (0,1) representa uma classificação perfeita, ou seja, 0 falsos positivos e 0 falsos negativos. A área abaixo desta curva é uma medida de precisão (Bradley, 1997).

Tabela 2 - Fórmulas para calcular diferentes medidas de qualidade (retirado de Bradley, 1997)

Medida	Fórmula
Classification rate (accuracy)	$\frac{TN+TP}{Total} \times 100$
Misclassification rate	$\left(1-\frac{TN+TP}{Total}\right) \times 100$
Sensivity (true positive rate)	$\frac{TP}{TP+FN} \times 100$
Specificity (true negative rate)	$\frac{TN}{FP+TN} \times 100$
1-specificity (false positive rate)	$\frac{FP}{FP+TN} \times 100$

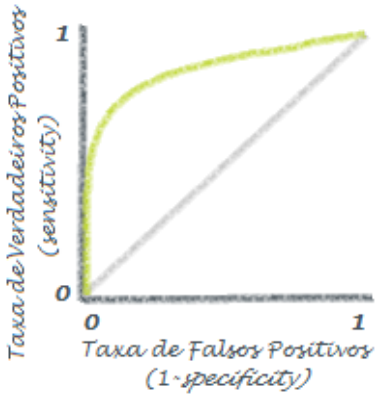

Figura 3 - Curva ROC (adaptado de Bradley, 1997)

3.2. MODELOS DE APRENDIZAGEM SUPERVISIONADA

3.2.1. Árvores de Decisão

Um dos algoritmos de classificação e predição, bastante popular, utilizado em ML tem o nome de Árvores de Decisão (Abdelhalim & Traore, 2009). As Árvores de Decisão são uma forma simples, mas poderosa, para analisar várias variáveis. Permitem prever, explicar, descrever e classificar um conjunto de dados (Ville, 2006). Uma das caraterísticas deste tipo de classificador que mais atrai os seus utilizadores, prende-se com a representação clara de como o conjunto de dados inicial se divide e se associa a uma determinada classe (Neville, 1999). Outros pontos fortes das Árvores de Decisão consistem na aceitação de vários tipos de variáveis (nominais, ordinais e intervalares), no saber lidar com valores omissos, na sua fácil utilização e interpretação e na sua insensibilidade ao fator escala, entre outros (Neville, 1999; Ville, 2006).

As Árvores de Decisão têm por base algoritmos que dividem o conjunto inicial de dados em subconjuntos mais homogéneos que por sua vez se podem dividir em subconjuntos ainda mais homogéneos (Ville, 2006). As árvores são compostas por nós, arcos e folhas (Figura 4). O primeiro nó, a raiz da árvore, apresenta o atributo mais relevante/discriminador da base de dados (Lemos, Steiner, & Nievola, 2005). Cada nó representa um teste específico aplicado a um conjunto de dados com o objetivo de o dividir em subconjuntos mais pequenos e mais homogéneos; cada arco liga um nó ao nó seguinte ou folha; as folhas representam os nós finais, isto é, os nós que contêm o conjunto de dados mais homogéneo que a árvore consegue produzir e que, por isso, não são submetidos a qualquer tipo de teste (Hamilton, Gurak, Findlater, Olive, & Ranson, 2012).

As Árvores de Decisão utilizam, assim, uma estratégia de “dividir para conquistar”, ou seja, o problema apresentado no primeiro nó é decomposto em problemas mais simples nos nós seguintes até sejam alcançados os nós finais, as folhas, que assignam os dados a uma determinada classe (Lemos et al., 2005). Desta forma as Árvores de Decisão podem-se entender como “disjunções de restrições conjuntivas” uma vez que cada arco acrescenta uma conjunção e cada folha acrescenta uma disjunção (Hamilton et al., 2012). Os resultados dos testes inerentes a cada nó são mutuamente exclusivos. Assim, um registo de um conjunto de dados existe apenas num dos nós diretamente a ele associado (Ville, 2006).

A Figura 4 esquematiza uma Árvore de Decisão.

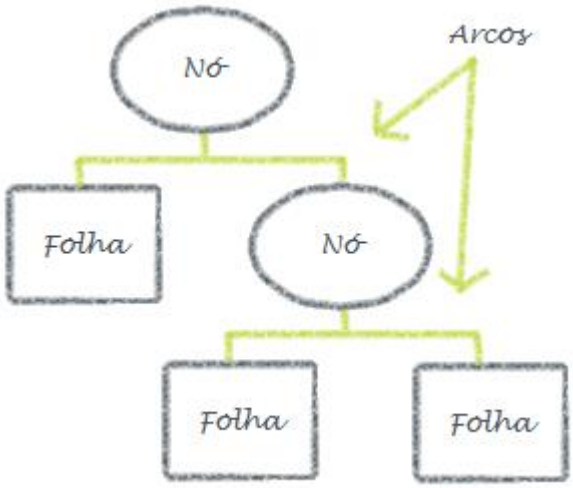

Figura 4 - Árvores de Decisão

Subjacente à construção de uma árvore de decisão encontra-se um algoritmo. Independentemente do algoritmo escolhido, este terá como condições iniciais uma árvore vazia e um conjunto de treino e assemelhar-se-á ao seguinte algoritmo genérico (Murthy, 1998):

1. Todos os exemplos de treino do nó *t* pertencem à categoria *c*?
   1.1. Se sim, diz-se que o nó é puro e este transforma-se numa folha representativa da classe *c* e este ramo da árvore termina;
   1.2. Se não, deve-se avaliar a melhor forma de divisão do conjunto de dados, S, do nó *t*, através de uma medida de qualidade.
      1.2.1. Escolher a partição de *S*, *s**, do nó *t* com melhor qualidade.
      1.2.2. Criar tantos nós quantas as partições efetuadas sobre o conjunto *s**.
      1.2.3. Voltar ao passo 1.

Posto isto, torna-se percetível que existem, essencialmente, duas operações principais durante a construção das árvores: (1) a avaliação e seleção das partições e (2) a conseguinte criação das mesmas (Du & Zhan, 2002).

3.2.1.1. Escolha do algoritmo

Ao longo dos anos foram vários os algoritmos propostos para a indução de árvores de decisão, tais como CLS, ID3, C4.5, CART, SLIQ e SPRINT (Rastogi & Shim, 2000). Dos algoritmos mencionados, os mais conhecidos e utilizados são os algoritmos CART e C4.5, que serão descritos mais à frente (Du & Zhan, 2002).

Em geral, os algoritmos dividem-se em duas fases: (1) a fase da construção da árvore e (2) a fase do desbaste. Na fase da construção da árvore, o conjunto de treino inicial (correspondente à raiz) é dividido em vários subconjuntos (nós) até alcançar um conjunto que contenha apenas registos pertencentes a uma mesma classe (folha). Subjacente à criação dos subconjuntos está a seleção de

uma medida de qualidade que melhor identifique o atributo responsável pela partição de um nó. Esta fase permite a construção de uma árvore perfeita, ou seja, uma árvore que classifica corretamente qualquer objeto.

O facto de ser perfeita para um conjunto de dados conhecido, não significa que tenha o mesmo comportamento num conjunto de dados novo (Quinlan & Rivest, 1989). De forma a evitar a sobreaprendizagem dos dados e de forma a alcançar uma árvore com uma elevada precisão, surge a segunda fase: o desbaste (Rastogi & Shim, 2000). A fase de desbaste permite transformar a árvore de decisão num classificador generalizado, eliminando da árvore as folhas e os nós responsáveis pela partição de um conjunto muito pequeno e muito particular de dados. Ou seja, elimina as partições que não produzem conhecimento generalizável (Du & Zhan, 2002).

Algoritmo CART

As árvores geradas pelo algoritmo CART (**C**lassification **A**nd **R**egression **T**rees) são árvores binárias, ou seja, cada nó reparte-se, exatamente, em dois nós. A seleção da variável responsável pela primeira partição da árvore é realizada com a ajuda do critério do Coeficiente de *Gini* que será explicado com detalhe na secção seguinte.

Uma caraterística importante do algoritmo CART consiste na sua capacidade de gerar árvores de regressões, ou seja, as folhas da árvore, ao invés de preverem uma classe, preveem um número (Maimon & Rokack, 2005).

Algoritmo C4.5

O algoritmo C4.5 difere do algoritmo CART no que diz respeito à repartição binária dos nós da árvore. Ou seja, um nó dividir-se-á em tantos nós, quantos os diferentes valores que uma variável assume. Mais uma vez, existe um critério para selecionar a variável responsável pela partição do primeiro nó. No caso do algoritmo C4.5, esse critério corresponde ao critério da Entropia (Maimon & Rokack, 2005).

3.2.1.2. Escolha da melhor partição

A seleção de um atributo para dividir o conjunto de dados em cada nó é crucial para classificar corretamente os objetos (Sivagama, 2011). Uma das maiores complexidades inerentes à construção de uma árvore de decisão prende-se com a identificação deste atributo, ou seja, a identificação do atributo com maior poder discriminador (Murthy, 1998). Assim, a melhor partição coincide com aquela que cria nós onde uma única classe domina (Sivagama, 2011). Existem várias medidas propostas pelos diversos autores para a definição do critério de partição (Rastogi & Shim, 2000). Duas das mais conhecidas são:

- Entropia; e
- Coeficiente de *Gini*.

A Tabela 3 apresenta o cálculo da entropia e do coeficiente de *Gini* bem como o ganho produzido por cada uma das medidas para um conjunto de dados *S* que contém *m* classes:

Tabela 3 - Cálculo da Entropia e do Coeficiente de Gini (retirado de Du & Zhan, 2002)

Entropia	Coeficiente de Gini
$Entropia\ (S) = -\sum_{j=1}^{m} P_j log P_j$	$Gini\ (S) = 1 - \sum_{j=1}^{m} P_j^2$
$Ganho\ (S,A) = Entropia\ (S) - \sum_{v \in A}(\frac{\lvert S_v \rvert}{\lvert S \rvert} \times Entropia(S_v))$	$Ganho\ (S,A) = Gini\ (S) - \sum_{v \in A}(\frac{\lvert S_v \rvert}{\lvert S \rvert} \times Gini(S_v))$

onde,

P_j corresponde à proporção de indivíduos de *S* na classe *j*;
A, que contém *v* valores distintos, corresponde à variável selecionada para proceder à partição de *S* em S_v subconjuntos;
Ganho(*S,A*), corresponde à redução de impureza no conjunto *S* causada pelo conhecimento da variável *A*.

- **Entropia**

A entropia, geralmente associada ao algoritmo C4.5, varia entre 0 e 1, tal como mostra a Figura 5.

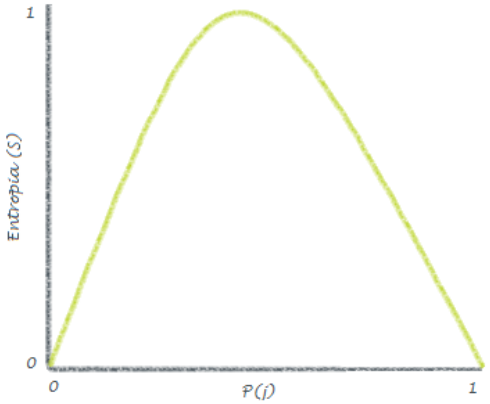

Figura 5 - Valores de entropia consoante a proporção de indivíduos na classe j (retirado de Shannon, 1948)

Tal como referido anteriormente, o objetivo destas medidas consiste na identificação do atributo com maior poder discriminador. Assim, a entropia é máxima (entropia=1) quando a impureza é máxima, i.e., quando a probabilidade de um indivíduo pertencer à classe *j* for igual a 0,5. Por outro lado, a entropia é mínima quando todos os indivíduos pertencem à mesma classe (Figura 6).

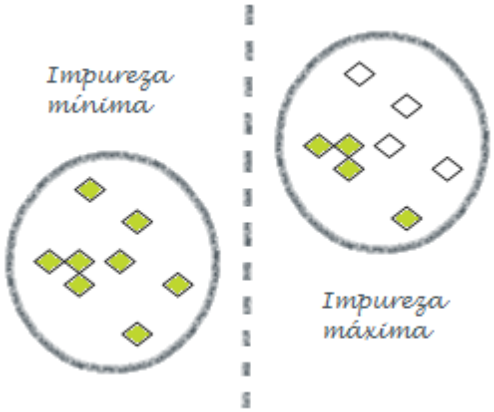

Figura 6 - Impureza máxima e impureza mínima

- **Coeficiente de *Gini***

O coeficiente de *Gini*, bem como a curva de *Lorenz*, foi utilizado inicialmente para avaliar a desigualdade de rendimentos num país (Jianjun, Chaojun, Qianqian, & Ping, 2010). O coeficiente de *Gini* é a base da curva de *Lorenz*, ilustrada na Figura 7.

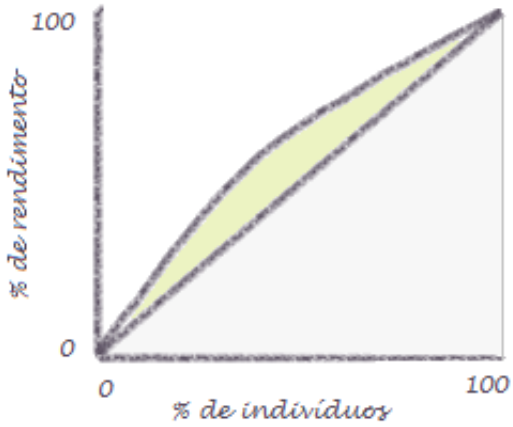

Figura 7 - Curva de Lorenz (retirado de Sivagama, 2011)

A diagonal representa uma distribuição igualitária da riqueza e a curva abaixo representa a distribuição real da economia. O coeficiente de *Gini* é calculado pela área entre a diagonal e a curva, dividida pela área abaixo da diagonal.

Nas Árvores de Decisão o coeficiente de *Gini*, normalmente associado ao algoritmo CART, é utilizado para selecionar atributos responsáveis por determinada partição (Sivagama, 2011). Quando todos os indivíduos de um conjunto de dados pertencem à mesma classe, o coeficiente de *Gini* alcança o seu valor mínimo, zero. Por outro lado, quando os registos de um conjunto de dados estão igualmente distribuídos pelas classes disponíveis, o coeficiente alcança o seu valor máximo.

Independentemente da medida escolhida, a variável responsável pela partição de um nó será aquela que obtiver um maior ganho de pureza em relação ao nó anterior. Assim sendo, as fórmulas do ganho de informação apresentadas na Tabela 3, que apuram a diminuição de diversidade causada pela

partição do conjunto de dados com base numa variável, identificam a variável responsável pela partição de determinado nó.

3.2.1.3. Paragem e desbaste

O tamanho de uma árvore é, talvez, um dos fatores mais determinantes da sua qualidade. Árvores demasiado pequenas poderão não conseguir descrever bem os dados. Árvores demasiado grandes terão folhas com tão poucos registos que não farão nenhuma predição fiável com uma nova amostra (Neville, 1999). O problema inerente a este tópico prende-se com a sobreaprendizagem dos dados. Ou seja, a dada altura a árvore começa a ajustar-se demasiado ao conjunto de treino, fazendo partições com base nas suas idiossincrasias. Quando essa mesma árvore é exposta a um novo conjunto de dados, a sua capacidade de fazer boas classificações fica aquém da capacidade demonstrada com o conjunto de treino. Com o objetivo de prevenir a sobreaprendizagem surge o critério de paragem e o desbaste (Du & Zhan, 2002).

O critério de paragem permite parar o crescimento da árvore antes de esta ter alcançado o nó onde todos os registos estão perfeitamente bem classificados.

Alguns critérios de paragem utilizados são (Neville, 1999):

- O número mínimo de registos numa folha;
- O número mínimo de registos obrigatório para que um nó seja dividido;
- A distância de qualquer folha à raiz.

O desbaste permite a evidência de sobreaprendizagem nos dados e posteriormente recorre ao desbaste da árvore, diminuindo assim o seu comprimento. Ou seja, o desbaste substitui uma parte da árvore por uma folha quando o erro esperado dessa subárvore é superior ao erro dessa folha. O desbaste tem obtido melhores resultados do que o critério de paragem (Gupta, 2008). Existem vários métodos de desbaste. Os mais conhecidos são: *Error-Complexity Pruning* (Breiman, Friedman, Olshen, & Stone, 1984), *Critical Value Pruning* (Mingers, 1987), *Minimum-Error Pruning* (Niblett e Bratko, 1986), *Reduced-Error Pruning* e *Pessimistic Error Pruning* (Quinlan, 1987).

3.2.2. Redes Neuronais Artificiais

A biologia popular defende que o cérebro humano é composto por pequenas células, chamadas neurónios, que transmitem sinais elétricos entre si. De uma forma muito elementar, a informação é transportada de um neurónio para outro quando os estímulos elétricos excedem determinado limiar (Daumé, 2012). As redes neuronais artificiais, importante ferramenta utilizada em DM, surgem com base na imitação do funcionamento do cérebro humano (Cui et al., 2006). As redes neuronais artificiais (ANN, do inglês *Artificial Neural Networks*) assemelham-se ao cérebro humano em dois aspetos: (1) o conhecimento é adquirido pela rede através de um processo de aprendizagem e (2) as forças das conexões existentes entre os neurónios, conhecidas por pesos sinápticos, são utilizadas para guardar o conhecimento adquirido (Haykin, 1999). A Figura 8 apresenta a semelhança existente entre o cérebro humano e ANN.

O objetivo das ANN consiste na compreensão e conseguinte aplicação das bases do sistema biológico, de forma a otimizar a resolução de problemas complexos (Basheer & Hajmeer, 2000). As redes

neuronais oferecem boas soluções para este tipo de problemas, uma vez que conseguem explorar estruturas complexas e encontrar interações lineares e não lineares e, ainda, descobrir padrões (Cui et al., 2006).

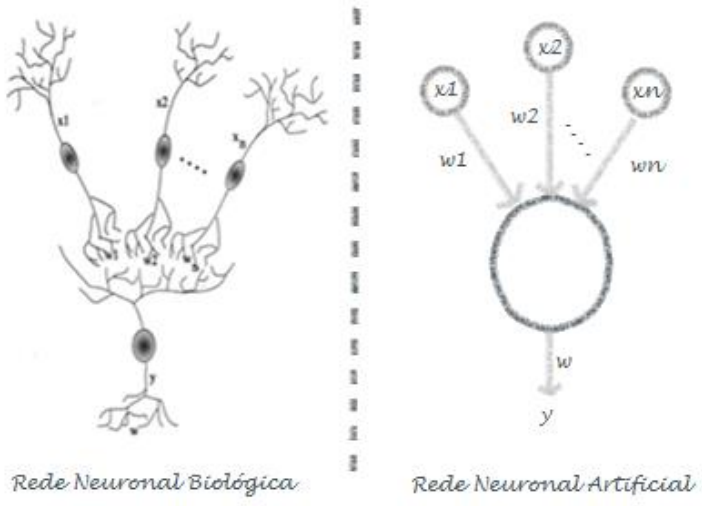

Figura 8 - Redes neuronais: biológica e artificial

Atualmente as ANN são tratadas como ferramentas *standard* de DM e utilizadas em problemas de classificação, análise de séries temporais, predição, *clustering*, entre outros (Maimon & Rokack, 2005).

Algumas das caraterísticas das Redes Neuronais que mais atraem os investigadores são: a não linearidade, a insensibilidade ao ruído, a capacidade de generalização, a robustez e o rápido processamento (Basheer & Hajmeer, 2000).

Os grandes desafios das ANN consistem na determinação do número de neurónios existentes na camada escondida, na determinação dos seus pesos e ainda no facto de geralmente existir um grau de tentativa-erro envolvido (Finlay, 2014).

Na génese das ANN surgem vários investigadores. As primeiras noções de Redes Neuronais surgiram em 1943 por McCulloch e Pitts. Em 1958, Rosenblatt propôs o primeiro modelo de aprendizagem supervisionada: o Percetrão (Haykin, 1999).

3.2.2.1. O Percetrão

O Percetrão é a Rede Neuronal mais simples que existe, utilizada apenas para classificações linearmente separáveis (Haykin, 1999).

De uma forma sumária, o Percetrão recebe um vetor de *input*, calcula a combinação linear desse *input* e, se o resultado for superior a um determinado limiar (*threshold*), o *output* é 1; caso contrário, o *output* é -1 (Mitchell, 1997). Ou seja, dados os *inputs* x_1 até x_n, o *output* $o(x_1, \ldots, x_n)$ gerado pelo Percetrão é dado por:

$$o(x_1, \ldots, x_n) = \begin{cases} 1, & w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_n x_n > 0 \\ -1, & c.c. \end{cases} \qquad (1)$$

Onde,

w_i determina o peso do *input* x_i no *output* do Percetrão;
w_0 corresponde ao *threshold*, um limiar cuja combinação ponderada dos *inputs* deve conseguir ultrapassar para que o *output* do Percetrão seja 1.

Os pesos do Percetrão, incluindo o *threshold*, são ajustados em função da diferença entre o *output* esperado e o *output* obtido, ou seja, em função do erro apurado pelo Percetrão (Basheer & Hajmeer, 2000).

No entanto, o Percetrão resulta apenas perante classificações lineares (Dean, 2014). Ou seja, apenas consegue resolver as operações lógicas AND e OR. Operações lógicas como XOR ultrapassam as capacidades do Percetrão simples (Mitchell, 1997).

3.2.2.2. Percetrão Multicamada

Minsky e Papert apresentaram em 1969 o Percetrão multicamada (Dean, 2014). O Percetrão multicamada surge como um desenvolvimento do Percetrão simples de forma a ultrapassar o obstáculo da não linearidade. A diferença entre a arquitetura do Percetrão multicamada face ao Percetrão simples, reside na existência de camadas (de neurónios) escondidas (Basheer & Hajmeer, 2000). A Figura 9 ilustra a arquitetura do Percetrão multicamada.

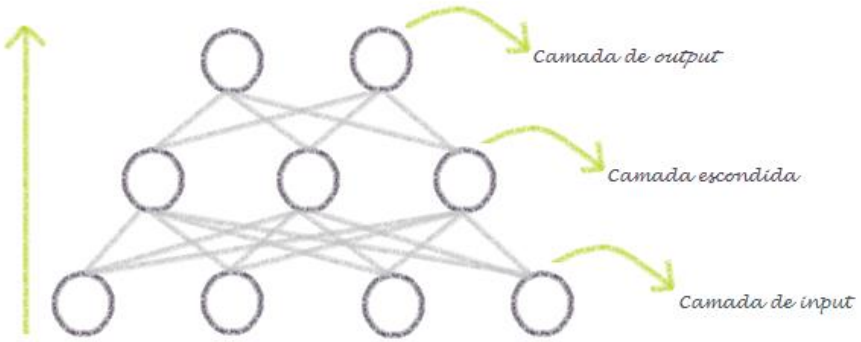

Figura 9 - Arquitetura do Percetrão multicamada

Posto isto, torna-se imprescindível perceber como funcionam as redes neuronais multicamada. Um dos algoritmos de redes neuronais mais populares, e que se encontra subjacente à arquitetura multicamada, tem o nome de retropropagação ou *backpropagation* (Cilimkovic, 2013). Este algoritmo tem como objetivo produzir um *output* com o menor erro possível, recorrendo, para tal, à combinação dos pesos utilizados. Ou seja, tal como explicado anteriormente, os pesos de cada observação que é submetida à rede, são ajustados em função da diferença entre o *output* esperado e o *output* obtido. No entanto, com o algoritmo *backpropagation*, os pesos são ajustados em sentido contrário, ou seja, da camada de *output* para a camada de *input*. Esta metodologia pretende perceber quais os pesos que mais contribuíram para o erro gerado pela rede, para que os mesmos possam ser ajustados e, desta forma, serem obtidos melhores resultados (Rojas, 1996).

O algoritmo de retropropagação pode ser decomposto em quatro principais etapas que se repetem até que o erro seja considerado suficientemente pequeno (Cilimkovic, 2013):

- Propagação para a frente;
- Retropropagação para a camada de output;
- Retropropagação para as camadas escondidas;
- Atualização dos pesos.

A **propagação para a frente** consiste, primeiramente, na apresentação de uma observação à rede, nomeadamente à camada de *input*. Na camada de *input*, que contém pesos iniciais, os valores recebidos por cada nó são transmitidos para os nós da camada escondida. O *output* de cada nó da camada escondida é calculado como uma combinação linear dos seus *inputs* e posteriormente, nos nós da camada seguinte, é aplicada sobre o *output*, uma função de ativação (Basheer & Hajmeer, 2000). Uma das funções de ativação mais populares quando se fala em *backpropagation* é a função sigmoide que é definida pela seguinte expressão (Rojas, 1996):

$$O_j = \frac{1}{1 + e^{-I_j}} \qquad (2)$$

Onde,

O, corresponde à função sigmoide;
j, corresponde ao nó da camada seguinte/camada de *output*; e
I_j, corresponde ao *input* da camada escondida/camada de *output*.

Quando os *outputs* das camadas escondidas são calculados, estes são propagados para a camada seguinte, até chegarem à camada de *output*, onde os *outputs* finais serão calculados (Cilimkovic, 2013).

Determinados os *outputs* da rede, torna-se oportuno calcular o erro inerente a estes. Começam aqui as etapa 2 e 3 introduzidas anteriormente: a **retropropagação para a camada de *output* e para as *camadas escondidas***.

Para cada *output* k é calculado o seu erro, E_k, que é obtido através da expressão seguinte:

$$E_k = O_j\big(1 - O_j\big)(T_j - O_j) \qquad (3)$$

Onde,

T_j corresponde ao valor efetivo que o *output* gerado pela rede deveria assumir.

Para cada nó *j* da camada escondida é calculado o erro, E_j, que resulta da soma ponderada dos erros dos nós *k da* camada seguinte que se encontram ligados a estes, ou seja:

$$E_j = O_j\big(1 - O_j\big)\sum_k w_{kj}E_k \qquad (4)$$

Onde,

w_{kj}, representam o peso existente na ligação entre o nó *j* e o nó *k.*

Após serem calculados todos os erros, cada w_{kj} é atualizado de forma a expurgar o efeito dos erros verificados. A atualização dos pesos é efetuada da seguinte forma:

$$w_{kj} = w_{kj} + \Delta w_{kj} \quad (5)$$

onde,

$\Delta w_{kj} = \alpha E_j O_j$; e

α corresponde à taxa de aprendizagem (Mitchell, 1997).

A taxa de aprendizagem é uma constante que auxilia a atualização dos pesos. Definir uma taxa de aprendizagem correta não é trivial. Se a taxa for demasiado pequena, o algoritmo irá demorar demasiado tempo a produzir um resultado. Por outro lado, se a taxa for demasiado elevada, o algoritmo poderá divergir (Cilimkovic, 2013).

3.2.3. Support Vector Machines

Support Vector Machines (SVM) é um método não-linear de aprendizagem supervisionada. Este classificador tornou-se popular por ter bases bastante sólidas na teoria da aprendizagem estatística (Maimon & Rokack, 2005).

Nas ferramentas de aprendizagem estatística, aprendizagem significa estimar uma função a partir de um conjunto de dados (Dean, 2014). O classificador SVM tem como objetivo encontrar uma função que permita classificar os dados corretamente, ou pelo menos da melhor forma possível, evitando ajustar-se demasiado ao conjunto de treino dos dados (Finlay, 2014).

Os modelos SVM conseguem ser robustos mesmo quando existe enviesamento no conjunto de treino. Além disso, a utilização de funções *Kernel* permite ao modelo ganhar flexibilidade e resolver problemas não lineares. No entanto, a capacidade de execução destes modelos sobre uma base de dados de dimensão considerável é ainda um assunto por resolver, sendo os resultados produzidos, em geral, difíceis de interpretar (Dean, 2014; (Maimon & Rokack, 2005).

3.2.3.1. Problemas Lineares

Para introduzir este método, comecemos por um exemplo de classificação binária, linearmente separável (Figura 10).

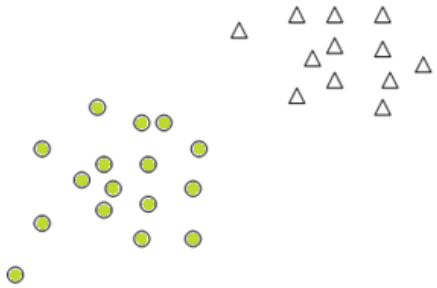

Figura 10 - Classes linearmente separáveis

Um problema é linearmente separável se existir pelo menos um hiperplano que separe as classes existentes (Russel & Norving, 1995). É intuitivo que existem inúmeros hiperplanos capazes de separar as duas classes apresentadas na Figura 10. Torna-se assim necessário decidir qual, de entre as infinitas

possibilidades, "melhor" separa as classes. O melhor hiperplano será aquele que maximiza a distância perpendicular entre as observações das duas classes que se encontram mais próximas (Dean, 2014). Ou seja, pretende-se maximizar a margem. A margem consiste na distância entre os hiperplanos que limitam cada classe (Figura 11).

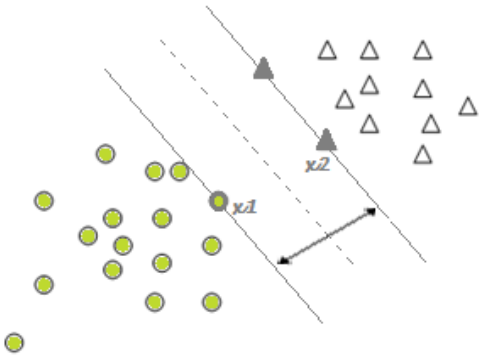

Figura 11 - Classes linearmente separável e representação de um possível hiperplano que separa as classes (adaptado de Dean, 2014)

Os hiperplanos que limitam a margem são canonicamente representadas por:

$$\begin{cases} x_i w + b \leq -1, se\ y_i = -1 \\ x_i w + b \geq 1\ se\ y_i = +1 \\ i = 1, \dots, n \end{cases} \quad (6)$$

Desta forma e utilizando as observações da Figura 11, temos que:

$$\begin{cases} x_1 w + b = -1 \\ x_2 w + b = 1 \end{cases} \rightarrow\ w(x_2 - x_1) = 2 \quad (7)$$

Como w e $x_2 - x_1$ são ortogonais ao hiperplano que se encontra no centro da margem, estes hiperplanos são paralelos entre si e desta forma, tem-se que

$$|w||(x_2 - x_1)| = \|w\|\|(x_2 - x_1)\| \quad (8)$$

Assim, a margem é representada por:

$$\frac{2}{\|w\|} \quad (9)$$

Onde,

$\|w\|$ corresponde à norma do vetor w.

Deduz-se assim um problema de otimização que pode ser reescrito como

$$min\|w\|^2 \quad (10)$$

sujeito a

$$y_i(x_i w + b) - 1 \geq 0, para\ i = 1, \dots, n \quad (11)$$

(Auria & Moro, 2008).

Até aqui assumiu-se que a margem é isenta de observações, isto é, que nenhuma das observações de treino recai sobre essa área. No entanto, são raros os problemas em que todas as observações ficam fora da margem. Assim, para além da maximização da margem, outro critério subjacente ao modelo SVM prende-se com a minimização dos erros de classificação (ξ_i). Introduz-se assim, uma variável de relaxamento, ξ_i, que permite suavizar as restrições impostas na determinação do hiperplano ótimo (Lorena & Carvalho, 2003). Quando não existem erros de classificação, i.e., quando todas as observações são corretamente classificados, tem-se $\xi = 0$; caso contrário tem-se $\xi > 0$. O modelo SVM impõe assim que nenhuma observação recaia sobre a margem excetuando alguns erros de classificação. Desta forma tem-se que:

$$\begin{cases} x_i w + b \leq -1 + \xi_i, se\ y_i = -1 \\ x_i w + b \geq 1 - \xi_i, se\ y_i = 1 \\ i = 1, \dots, n \end{cases} \quad (12)$$

Que pode ser resumido em:

$$y_i(x_i w + b) \geq 1 - \xi_i, \forall\, i = 1, \dots, n \quad (13)$$

Onde,

y_i corresponde a uma observação.

Posto isto, surge a questão "Como otimizar o problema?". A resposta a esta questão deriva da aplicação da seguinte expressão:

$$min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \quad (14)$$

sujeito a

$$y_i(x_i w + b) \geq 1 - \xi_i \quad (15)$$

$$\xi_i \geq 0 \quad (16)$$

Onde,

C corresponde a um parâmetro de "afinação", ou seja, pondera os erros de classificação e controla o comportamento do modelo SVM, evitando ajustamentos excessivos aos dados de treino. Desta forma pretende-se um C baixo, de modo a potenciar a generalização do modelo. É, no entanto, possível provar que existe uma relação direta entre o parâmetro C e a dimensão da margem, que, pelo que vimos até agora se pretende maximizar. É então necessário encontrar o *trade-off* entre estas duas condições.

Este é um problema de otimização convexo com uma função objetivo quadrática e com restrições lineares. Implícito a este problema encontra-se a assunção de que as classes são linearmente separáveis (Smola & Vishwanathan, 2008). Para resolver a questão de otimização apresentada, introduz-se uma função *Lagrangiana*[1].

[1] Mais informações sobre aplicação da função *Lagrangiana* pode ser encontrada em (Auria & Moro, 2008; Lorena & Carvalho, 2003).

3.2.3.2. Problemas Não Lineares

Os problemas lineares apresentados anteriormente podem ser generalizados de forma a resolver problemas não lineares, como o apresentado na Figura 12 (Lorena & Carvalho, 2003).

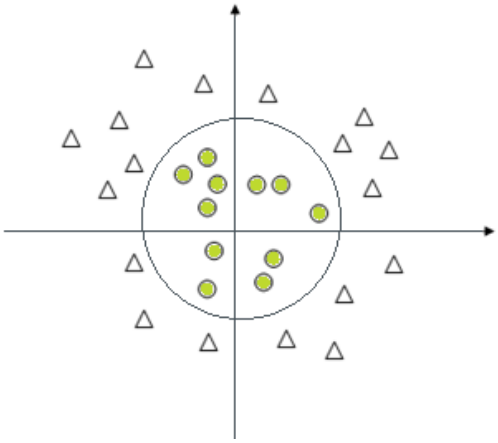

Figura 12 - Problema não linear (adaptado de Dean, 2014)

A ideia base do SVM aplicado a problemas não lineares consiste na transposição dos dados originais (espaço de entrada) para um novo espaço, de maior dimensionalidade, chamado espaço de caraterísticas, com a ajuda de funções reais $(\phi_1, \ldots, \phi_m)$ definidas no espaço dos dados de treino (Figura 13). Através da utilização de funções *Kernel* é possível ajustar a maximização da margem num espaço de maior dimensionalidade. As funções *Kernel* mais utilizadas encontram-se na Tabela 4.[2]

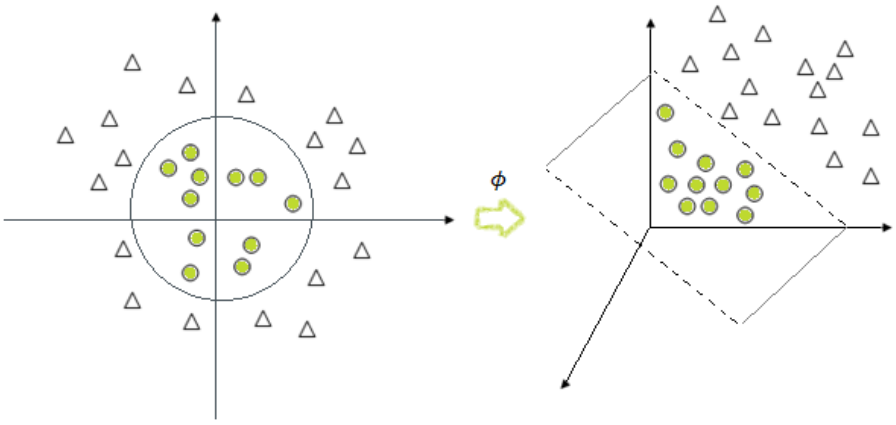

Figura 13 - Transposição dos dados do conjunto de entrada para o conjunto de caraterísticas (adaptado de Dean, 2014)

[2] Mais informação sobre problemas não lineares poderão ser encontradas em (Maimon & Rokack, 2005).

Tabela 4 - Funções Kernel mais comuns (retirado de Dean, 2014)

Kernel	K(xi, xi)
Radial Basis Function	$\exp(-\gamma\|x - x_i\|^2), \gamma > 0$
Inverse multiquadratic	$\frac{1}{\sqrt{\|x - x_i\| + \eta}}$
Polinómio de grau d	$((x^T x_i) + \eta)^d$
Sigmoidal	$\tanh(\gamma(x^T x_i) + \eta), \gamma > 0$
Linear	$x^T x_i$

3.2.4. Voted Perceptron

Freund e Schapire propuseram, em 1999, o algoritmo *Voted Perceptron*. Este é um algoritmo simples e de fácil implementação que combina ANN e SVM, algoritmos já descritos anteriormente. Comparativamente com SVM, este algoritmo apresenta melhores valores de precisão, tempo de aprendizagem e velocidade de previsão.

O *Voted Perceptron* potencia as suas capacidades quando está perante problemas de classificação linear. Não obstante, este algoritmo pode ser aplicado em espaços de grande dimensionalidade com auxílio das funções *Kernel* (Freund & Schapire, 1999).

A Figura 14 apresenta o algoritmo *Voted Perceptron* durante a fase de treino, onde $x \in \mathbb{R}^n$ e a classe y pode assumir os valores $\{-1, 1\}$.

A previsão da classe de um novo registo é dada por $\hat{y} = sign(v_k x_i)$. Se a previsão estiver correta, o vetor v, previamente inicializado, não sofre qualquer alteração. Caso a previsão não seja correta, o vetor v é atualizado para $v = v + yx$. O processo repete-se para todo o conjunto de treino.

Durante a fase de treino, o algoritmo guarda todos os vetores de previsão que foram gerados. Para cada um desses vetores, é contabilizado o número de iterações às quais o vetor sobreviveu, antes de surgir uma classificação errada. Este número irá corresponder à ponderação do vetor, aquando da fase de previsão.

A Figura 15 apresenta a parte do algoritmo *Voted Perceptron* referente à fase de previsão. Todos os vetores apurados na fase de treino dão o seu contributo para o apuramento da classe do novo registo x. No entanto cada um desses vetores dará o seu contributo, i.e., a sua previsão, de forma ponderada, como vimos anteriormente. Desta forma, os vetores de previsão que sobreviveram durante mais tempo na fase de treino, darão um maior contributo para a previsão do novo registo.

Se estivermos perante um problema de classificação linear, este algoritmo, após finitos erros de classificação, consegue alcançar um vetor que classifica corretamente todos os exemplos apresentados.

Treino

Input: Conjunto de treino classificado $\langle (x_1, y_1), \dots, (x_m, y_m) \rangle$

T épocas

Output: Perceptrões ponderados $\langle (v_1, c_1), \dots, (v_k, c_k) \rangle$

1. Inicialização dos parâmetros: $k = 0, v_1 = 0, c_1 = 0$

2. Repetir T vezes:

    2.1. Para $i = 1, \dots, m$:

    * $\hat{y} = sign(v_k x_i)$

    * Se $\hat{y} = y$,

        então $c_k = c_k + 1$

        c.c. $v_{k+1} = v_k + y_i x_i, \quad c_{k+1} = 1, \quad k = k + 1$

Figura 14 - Algoritmo Voted Perceptron – Treino (retirado de Freund & Schapire, 1999)

Previsão

Dados: A lista de percetrões ponderados $\langle (v_1, c_1), \dots, (v_k, c_k) \rangle$

Um exemplo não classificado, x

1. Calcular a classe $\hat{y}$ de x:

$s = \sum_{i=1}^{k} c_i \, sign(v_i x);$

$\hat{y} = sign(s)$

Figura 15 - Algoritmo Voted Perceptron – Previsão (retirado de Freund & Schapire, 1999)

3.2.5. Métodos Ensemble

Se perante um grupo de pessoas, com *expertise* em determinado assunto, lhe for apresentado um determinado problema e posteriormente lhe for pedido para tomar uma decisão, certamente a decisão desse grupo tenderá a ser melhor do que qualquer decisão individual tomada por cada uma dessas pessoas (Finlay, 2014). Esta é a ideia subjacente aos métodos *Ensemble*. As técnicas *Ensemble* procuram as vantagens individuais de cada classificador, combinando-as de forma a obter uma melhor solução (Augusty & Izudheen, 2013). A aprendizagem *Ensemble* é uma técnica que tem vindo a adquirir cada vez maior relevância uma vez que combina vários algoritmos de aprendizagem de forma a melhorar a precisão de uma classificação (Kumar, Kongara, & Ramachandra, 2013).

Um modelo *Ensemble* consiste na combinação de dois ou mais modelos (Dean, 2014). Os *ensembles* (ou conjuntos) podem ser combinados de várias formas. A mais comum consiste numa abordagem democrática, i.e., um objeto é atribuído à classe mais votada pelos classificadores presentes no ensemble (Augusty & Izudheen, 2013).

Alguns dos métodos *Ensemble* mais populares, e os utlizados neste projeto, são:

- *Bagging*
- *Boosting*
- *Random Forest*
- *Random Trees*
- *Alternating Decision Trees*

3.2.5.1. *Bagging*

Bagging é um acrónimo do procedimento **B**ootstrap **Agg**regat**ing** (Breiman, 1996). Este é o método *ensemble* mais simples e com maior sucesso (Dean, 2014).

Este método gera, a partir do conjunto de treino, vários subconjuntos (Kumar et al., 2013). Cada observação é incluída num subconjunto apenas uma vez, ou seja, sem reposição. No entanto, uma observação pode ser incluída em vários subconjuntos. Assim, uma observação pode constar de todos os subconjuntos ou, pelo contrário, não constar em nenhum. Normalmente nenhuma destas situações acontece uma vez que existe um *trade-off* entre o número de subconjuntos e a sua dimensão (Dean, 2014).

Cada subconjunto de treino origina um classificador que será incluído no *ensemble*. Ou seja, os modelos obtidos são acumulados, de forma a criar um modelo final (Kumar et al., 2013). Este modelo é geralmente mais estável que modelos individuais que utilizem o conjunto de dados completo (Dean, 2014).

Posto isto, a maioria ou a média dos votos determina a classe de um registo (Kumar et al., 2013).

As várias demonstrações, tanto práticas como teóricas, convergem no sentido de que, apesar da sua instabilidade, o método *Bagging* direciona-se para a otimização dos processos de classificação. No entanto, este método pode degradar o desempenho de processos estáveis sendo particularmente efetivo quando o modelo subjacente é uma Árvore de Decisão (Breiman, 1996; Dean, 2014).

3.2.5.2. *Boosting*

O método *Boosting*, bastante semelhante ao *Bagging*, consiste numa das ideias mais poderosas dos últimos 10 anos (Kumar et al., 2013)(Dean, 2014).

Este método baseia-se na produção de classificadores em série. O conjunto de treino utilizado por cada membro da série é escolhido com base na performance do classificador anterior (Opitz & Maclin, 1999). Um registo classificado erradamente pelo classificador anterior, tenderá a ser escolhido pelo seguinte classificador, ao contrário dos registos classificados corretamente (Dean, 2014). Assim, o método Boosting tenta ir produzindo novos classificadores com cada vez maior capacidade de classificação, baseando-se no erro do classificador anterior, até que seja atingido um limite: no número de modelos produzidos ou na precisão alcançada (Opitz & Maclin, 1999). De uma forma geral, esta técnica de classificação consiste na criação de um classificador forte através da combinação de vários classificadores mais fracos (Kumar et al., 2013).

3.2.5.3. *Random Forest*

O método *Random Forest* tem por base o método *Bagging* (Kumar et al., 2013). Este método pretende criar um classificador estável e forte através da combinação dos resultados obtidos por várias árvores de decisão (Finlay, 2014). Normalmente o *Random Forest* é utilizado quando existem vários conjuntos de treino e várias variáveis. Normalmente, deste *ensemble* fazem parte dezenas ou centenas de árvores de decisão com o seu tamanho máximo, ou seja, sem serem submetidas ao processo de desbaste (Williams, 2010).

O resultado obtido pelo *Random Forest* tende a ser melhor do que a média obtida pelos resultados das árvores presentes no *ensemble*, comprovando assim que "o todo é melhor do que a soma das várias partes" (Dean, 2014). Ao contrário do que acontece nas Árvores de Decisão, o método *Random Forest* assegura que todas as variáveis são utilizadas, e não apenas as mais relevantes (Finlay, 2014).

Este modelo oferece concorrência aos classificadores não lineares como as Redes Neuronais e o classificador SVM (Williams, 2010).

3.2.5.4. *Random Trees*

O método *ensemble Random Trees* foi introduzido por Leo Breiman e Adele Cutler. Este método consiste na construção de várias árvores de decisão, aleatoriamente. Basicamente, um atributo é selecionado aleatoriamente para fazer a primeira divisão da árvore. De seguida, outro atributo é selecionado para fazer uma nova repartição da árvore. O método *Random Trees* não utiliza critérios de pureza, tal como acontece nas árvores de decisão, uma vez que todas as escolhas são feitas aleatoriamente.

A árvore para de crescer quando:

1. Um nó fica vazio, ou quando todos os registos de um nó pertencem à mesma classe; ou
2. A profundidade da árvore excede determinados limites.

Ao contrário do que acontece no modelo *Random Forest*, o modelo *Random Trees* recorre ao desbaste da árvore, removendo os nós "não necessários". Um nó é considerado "não necessário" se nenhum dos seus descendentes apresenta classes significativamente diferentes das atribuídas por este.

Cada árvore gera um *output*. O *output* do *ensemble* é obtido através da média de todas as árvores (Zhang, Fan, Yuan, Davidson, & Li, 2006).

3.2.5.5. Alternating Decision Trees

O método *Alternating Decision Trees* resulta da combinação entre a simplicidade das árvores de decisão e a eficácia do método *Boosting*. Este método tem a estrutura de uma árvore de decisão. No entanto, os vários nós da árvore têm diferentes objetivos. O primeiro nó, a raiz da árvore, tem o objetivo de prever. Já os nós da camada seguinte são nós de decisão. Assim, o crescimento da árvore vai ser alternado entre nós de previsão e nós de decisão (Williams, 2010).

3.2.6. Aprendizagem Bayesiana

A aprendizagem *Bayesiana* tem por base o conhecimento probabilístico, com especial enfoque na probabilidade condicional, e consegue alcançar resultados impressionantes no processo de classificação (I. Witten & Frank, 2005). No entanto, um obstáculo à utilização dos métodos *Bayesianos* prende-se com a necessidade de conhecimento prévio sobre a probabilidade de determinado acontecimento. Adicionalmente, o elevado custo computacional associado à determinação da hipótese ótima de Bayes poderá constituir um outro obstáculo na aplicação destas técnicas (Mitchell, 1997).

Bayes introduz dois tipos de probabilidade (Dean, 2014):

- *A priori* – que diz respeito ao valor de uma probabilidade antes de se ter conhecimento de qualquer informação adicional; e
- *A posteriori* – que corresponde ao valor de uma probabilidade afetada por informação adicional obtida posteriormente.

Tal como o próprio nome indica, a aprendizagem Bayesiana tem por base o teorema de *Bayes* que, pretendendo obter a melhor hipótese do espaço H dado o conjunto de treino D, é definido por:

$$P(h\backslash D) = \frac{P(D\backslash h)P(h)}{P(D)} \qquad (17)$$

Onde,

$\mathrm{P}(h)$, corresponde à probabilidade *a priori* de *h*;
$\mathrm{P}(\mathrm{D})$, corresponde à probabilidade a *priori* de D;
$P(D\backslash h)$, corresponde à probabilidade *a posteriori* de D;
$P(h\backslash \mathrm{D})$, corresponde à probabilidade *a posteriori* de *h*.

O teorema de Bayes apresenta o cálculo da probabilidade de uma hipótese, com base na sua probabilidade previamente observada (Mitchell, 1997). Este teorema apresenta a relação entre duas probabilidades condicionais que são o inverso uma da outra (Dean, 2014).

3.2.6.1. Redes Bayesianas

As redes *Bayesianas* (BN do inglês *Bayes Networks*) são constituídas por (Jensen, 2001):

1. Um conjunto de variáveis e um conjunto de arestas direcionadas entre as variáveis;
2. Variáveis que no seu todo e juntamente com as arestas direcionadas, formam um grafo acíclico direcionado;
3. Variáveis que, individualmente, têm associado um conjunto de estados finitos e mutuamente exclusivos;
4. Variáveis que, individualmente, têm associada uma tabela de probabilidades condicionais.

As BN são grafos acíclicos e direcionados que permitem a caraterização da distribuição das probabilidades conjuntas de um conjunto de variáveis.

Os nós do grafo representam as variáveis, enquanto as arestas representam dependências probabilísticas entre as variáveis. Adicionalmente, as arestas direcionadas do nó A ao nó B indicam que o nó A influencia diretamente o B. É então percetível que uma BN tem subjacente a premissa da independência condicional das variáveis, ou seja, uma variável é independente das suas variáveis não descendentes, dada a observação da variável à qual deve dependência direta, variável também conhecida por variável "pai". Esta propriedade tem como fundamento potenciar o cálculo das probabilidades *a posteriori* através da redução do número de atributos necessários para caraterizar a distribuição das probabilidades conjuntas das variáveis (Ben-Gal, 2007).

Associada a cada nó está uma tabela de probabilidades condicional que especifica a distribuição condicional da variável dados os seus pais diretos no grafo (Mitchell, 1997).

A probabilidade conjunta dos valores $(y_1, \dots, y_n)$ para as variáveis $Y_1, \dots, Y_n)$ pode ser obtida através da seguinte fórmula (Mitchell, 1997):

$$P(y_1, \dots, y_n) = \prod_{i=1}^{n} P(y_i | Pais(y_i)) \qquad (18)$$

Onde,

$Pais(y_i)$, representa o conjunto imediato de precedentes de y_i na rede; e
$P(y_i|Pais(y_i))$, diz respeito aos valores armazenados na tabela de probabilidades conjuntas associada ao nó y_i.

3.2.6.2. Naive Bayes

Um dos métodos de aprendizagem *Bayesiana* mais utilizado é o classificador *Bayesiano*. Por vezes a performance deste classificador é comparada com a performance das Redes Neuronais e das Árvores de Decisão (Mitchell, 1997).

Dado um conjunto de treino $(y_1, \dots, y_n)$ e uma novo exemplo, a abordagem utilizada pelo método Naive Bayes para classificar a nova observação consiste na adjudicação da classe mais provável, v_{MAP}, da seguinte forma:

$$v_{MAP} = argmax_{v_j \in V} P(v_j | y_1, \dots, y_n) \qquad (19)$$

Utilizando o teorema de Bayes descrito anteriormente, esta expressão pode ser reescrita como:

$$v_{MAP} = argmax_{v_j \in V} \frac{P(y_1, \dots, y_n | v_j) P(v_j)}{P(y_1, \dots, y_n)}$$

$$v_{MAP} = argmax_{v_j \in V} P(y_1, \dots, y_n | v_j) P(v_j) \quad (20)$$

Onde,

$P(v_j)$, consiste na probabilidade associada à classe v_j e pode ser obtida através do cálculo da frequência em que cada classe v_j ocorre;
$P(y_1, \dots, y_n | v_j)$, consiste na probabilidade *a posteriori* do conjunto de treino.

Subjacente ao classificador Naive Bayes está a premissa de que os valores das variáveis são condicionalmente independentes dada a variável *target*, ou seja:

$$P(y_1, \dots, y_n | v_j) = \prod_i P(y_i | v_j) \quad (21)$$

Desta forma, e substituindo este termo na expressão utilizada para calcular a classe mais provável, v_{MAP}, a abordagem utilizada pelo classificador Naive Bayes é a seguinte:

$$v_{NB} = argmax_{v_j \in V} P(v_j) \prod_i P(y_i | v_j) \quad (22)$$

Obtendo v_{NB}, obtém-se a classe da nova observação (Mitchell, 1997).

3.2.7. Regressões

3.2.7.1. Regressões Lineares Simples

"O modelo de regressão linear pode ser utilizado para estudar a relação entre duas variáveis" (Wooldridge, 2013). Sendo x e y duas variáveis representativas de uma população, a regressão linear simples que explica a relação entre estas duas variáveis é dada por:

$$y = \beta_0 + \beta_1 x + u \quad (23)$$

Onde,

y, diz respeito à variável dependente;
x, diz respeito à variável independente;
u, ou erro, representa outros fatores que influenciam y para além de x;
β_0, é o parâmetro de interceção, também conhecido por constante;
β_1, representa o declive na relação entre x e y, mantendo todos os restantes fatores (u) constantes.

Relativamente ao termo u, este poderá ser considerado como o obstáculo à relação funcional entre as variáveis dependente e independente. Assim, para as variações em y derivarem apenas de variações em $\beta_1 x$, é necessário que os outros fatores representados por u tenham uma variação nula, ou seja,

$$\Delta y = \beta_1 \Delta x \ \Rightarrow \Delta u = 0 \qquad (24)$$

Como estimar β_0 e β_1?

Para estimar β_0 e β_1 é necessário recorrer ao método dos mínimos quadrados. O método dos mínimos quadrados (OLS do inglês *Ordinary Least Squares*) tem as seguintes premissas (Lopes, 2009):

1. Modelo linear;
2. Erro aleatório com média = 0;
3. Não colinearidade;
4. Homocedasticidade;
5. Ausência de autocorrelação;
6. Normalidade dos erros.

Respeitadas as premissas, a variável dependente, y, é obtida como resultado da minimização da soma dos quadrados das diferenças entre y_i e $\hat{y}_i$, sendo $\hat{y}_i$ o valor estimado de y_i (Figura 16).

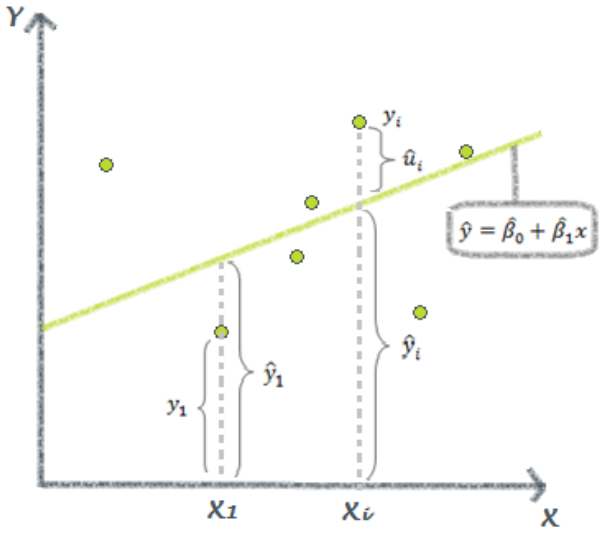

Figura 16 - Valores ajustados e resíduos (adaptado de Wooldridge, 2013

Assim, utilizando a equação da regressão linear simples apresentada anteriormente, temos que:

$$\sum_{i=1}^{n} \hat{u}_i{}^2 = \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)^2 \qquad (25)$$

Perante a amostra de uma população de dimensão n, $\{(x_i, y_i): i = 1, \dots, n\}$, a regressão linear simples, para cada elemento da amostra, pode-se escrever assim:

$$y_i = \beta_0 + \beta_1 x_i + u_i \qquad (26)$$

Os estimadores de β_0 e de β_1, $\hat{\beta}_0$ e $\hat{\beta}_1$ respetivamente, necessários para minimizar a soma dos quadrados dos erros são dados por[3]:

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} \qquad (27)$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad (28)$$

3.2.7.2. Modelo Logit

O modelo *Logit* encaixa nos modelos de probabilidade linear que consistem na aplicação do modelo de múltiplas regressões num problema cuja variável dependente é binária (Wooldridge, 2013).

Perante um problema de classificação binária, pretende-se criar um modelo em função de x com a probabilidade condicional $P(y = 1|x = x_i)$. Para criar este modelo através de uma regressão, surge a regressão *Logit* que é definido pela seguinte função:

$$log\,\frac{p(x)}{1 - p(x)} = \beta_0 + x\beta \Leftrightarrow p(x) = \frac{e^{\beta_0 + x\beta}}{1 + e^{\beta_0 + x\beta}} = \frac{1}{1 + e^{-\beta_0 + x\beta}} \qquad (29)$$

Assim, $y = 1$ quando $p \geq 0{,}5$ e $y = 0$ quando $p < 0{,}5$. Por outras palavras, $y = 1$ sempre que $\beta_0 + x\beta$ for não negativo e $y = 0$ em caso contrário. A regressão *Logit* apresenta assim um classificador linear (Shalizi, 2012).

[3] Os desenvolvimentos necessários para obter as equações dos estimadores $\hat{\beta}_0$ e $\hat{\beta}_1$ podem ser consultados em (Wooldridge, 2013).

4. SELEÇÃO DE VARIÁVEIS

A seleção de atributos é uma técnica utilizada na fase de pré-processamento do processo KDD, cujo objetivo prende-se com a redução do número de variáveis e com a remoção de dados irrelevantes, redundantes ou com ruído. Desta forma, a performance do algoritmo utilizado é potenciado, melhorando a precisão da previsão e a interpretação dos resultados (Kumar et al., 2013).

Os principais objetivos da seleção de atributos são: (1) evitar a sobreaprendizagem do modelo e (2) melhorar a performance do mesmo (Kumar, Kongara, & Ramachandra, 2013). A seleção de variáveis é posta em prática quando se desconfia que as variáveis disponíveis no conjunto de dados são redundantes, ou seja, que algumas delas estão correlacionadas entre si (O'Rourke & Hatcher, 2013).

Por norma, um método de seleção de atributos resume-se às seguintes quatro etapas (Karegowda, Jayaram, & Manjunath, 2011):

1. **Geração de um subconjunto de variáveis candidato**. O conjunto de dados original contém *n* variáveis. Para obter um subconjunto com *m* variáveis (*m* < *n*) é necessário recorrer a procedimentos de pesquisa baseados em determinada estratégia. A estratégia de pesquisa pode ser classificada como (1) completa/exaustiva, (2) heurística e (3) aleatória. No âmbito deste projeto são utilizadas duas estratégias: *best first* (Rich & Knight, 1991) que faz parte da pesquisa completa/exaustiva e *genetic search* (Cherkauer & Shavlik, 1996; Vafaie & De Jong, 1995) pertencente à pesquisa aleatória.
2. **Função de avaliação do subconjunto obtido na primeira etapa**. Após a criação do novo subconjunto de variáveis é necessário avaliá-lo através de uma abordagem *Filter* (Kohavic & John, 1996) ou *Wrapper* (Kohavi, 1995). No âmbito deste projeto, foi utilizada uma abordagem *Filter*, mais especificamente a seleção de atributos com base na correlação das variáveis (CFS do inglês *Correlation-based Feature Subset Selection*).
3. **Condição de paragem**. É necessário definir um critério de paragem para o número de variáveis a incluir no novo subconjunto. Este critério pode derivar de vários aspetos, tais como: o número de variáveis selecionadas, o número de iterações realizadas, a (não) obtenção de melhores resultados com o novo subconjunto, entre outros.
4. **Procedimento para aprovar a validade do novo subconjunto.** Normalmente os resultados obtidos pelo conjunto de atributos original é comparado com os resultados obtidos pelo conjunto de atributos selecionado quando os mesmos são *input* para um algoritmo de indução sobre determinada base de dados. Outra abordagem de validação consiste na utilização de vários algoritmos de seleção de atributos de forma a obter atributos relevantes e posteriormente comparar os resultados através da utilização de classificadores em cada subconjunto de atributos.

A Figura 17 retrata todo o processo de seleção de atributos inerente às quatro etapas acabadas de descrever.

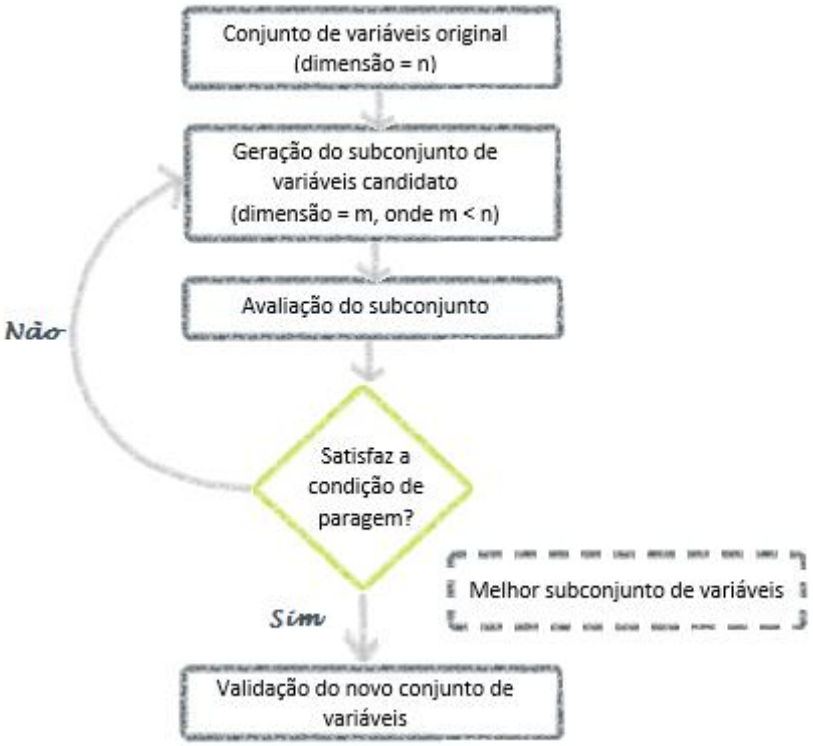

Figura 17 - O processo de seleção de atributos (retirado de Karegowda et al., 2011)

4.1. GENETIC SEARCH

A pesquisa genética como resposta à primeira etapa do processo de seleção de atributos consiste num método de pesquisa estocástico, com capacidade para explorar grandes espaços de pesquisa e com uma performance de pesquisa global.

Subjacente a este método de pesquisa encontra-se a teoria da evolução de Charles Darwin, mais propriamente o princípio da "sobrevivência do mais forte" (Karegowda et al., 2011).

O algoritmo subjacente a este método de pesquisa é iterativo, no qual, as soluções em cada geração, são produzidas através da aplicação de operadores genéticos tais como reprodução, *crossover* e mutação. O primeiro, reprodução, seleciona uma *string* do conjunto inicial de atributos, sendo uma *string* uma sequência binária onde 1 representa a seleção da variável para o subconjunto e 0 a não seleção; o *crossover* combina *strings* com boa avaliação (aos pares) para gerar melhores "filhos"; e a mutação modifica algumas *strings* de forma a melhorar a sua avaliação (M. A. Hall, 1999). Em cada geração a população é avaliada e testada. Se o critério de paragem não for satisfeito, a população será submetida novamente aos operadores genéticos até que a condição de paragem seja alcançada.

4.2. BEST FIRST

O método *Best First* consiste em alterações locais no espaço das variáveis. Este método explora todo o espaço de variáveis, pelo que é necessário um critério de paragem. Como consequência, este critério de paragem poderá envolver a limitação do número de alterações locais, o que prejudica a qualidade do novo subconjunto de variáveis (M. A. Hall, 1999).

4.3. SELEÇÃO DE ATRIBUTOS BASEADA NA CORRELAÇÃO DAS VARIÁVEIS

"As variáveis são relevantes se os seus valores variarem sistematicamente com as diferentes classes" (Gennari, Langley, & Fisher, 1989). Por outras palavras, num modelo devem constar apenas variáveis bastante correlacionadas com a variável classe e pouco correlacionadas entre si.

A função de avaliação utilizada pela seleção de atributos baseada na correlação das variáveis (CFS) é a seguinte:

$$M_s = \frac{k_{\overline{r_{cf}}}}{\sqrt{k + k(k-1)r_{ff}}} \qquad (30)$$

Onde,

M_s, corresponde ao "mérito" heurístico de um subconjunto de variáveis, S;
k, diz respeito ao número de variáveis contidas no subconjunto S;
r_{cf}, corresponde à média das correlações variável-classe ($f \in S$);
r_{ff}, representa a média das correlações variável-variável.

Esta função representa a medida de avaliação de CFS uma vez que:

- O numerador poderá indicar a capacidade de classificação do subconjunto das variáveis;
- O denominador indica a redundância entre as variáveis.

De uma forma simples, o primeiro passo no método CFS consiste no cálculo das correlações variável-classe e variável-variável. Obtidas as correlações, é feita uma pesquisa no espaço das variáveis. O subconjunto com maior "mérito" será o subconjunto escolhido para reduzir o conjunto de dados original.

O algoritmo subjacente ao método CFS pode ser explicado pela Figura 18.

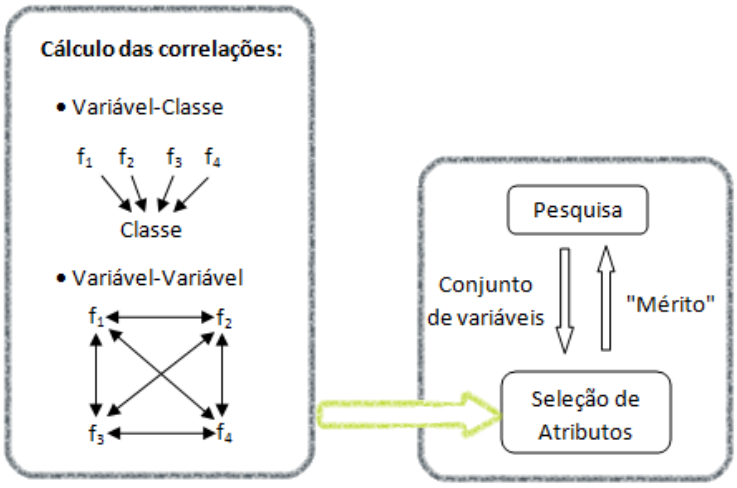

Figura 18 - Método CFS (retirado de Karegowda et al., 2011)

5. METODOLOGIA E RESULTADOS EXPERIMENTAIS

5.1. FERRAMENTAS ANALÍTICAS

Quando confrontados com desafios, os analistas procuram as melhores ferramentas para os resolver e também para extrair valor a partir deles. WEKA, SAS, Linguagens Java, R e Python são algumas das ferramentas mais utilizadas aquando da construção de modelos preditivos (Dean, 2014).

A ferramenta escolhida para o caso de estudo deste projeto foi o WEKA (M. Hall, Frank, & Holmes, 2009). **W**aikato **E**nvironment for **K**nowledge **A**nalysis é uma ferramenta *open source* desenvolvida pela Universidade de *Waikato* na Nova Zelândia. Totalmente implementada em Java, o WEKA é reconhecido pela sua vasta gama de algoritmos. Esta ferramenta permite aplicar diretamente os algoritmos a um conjunto de dados, sem que o utilizador saiba programar em Java (Dean, 2014). Desta forma permite aos utilizadores experimentar e comparar vários métodos de aprendizagem automática de uma forma rápida (M. Hall et al., 2009). Sendo o objetivo deste projeto a análise dos resultados e não a implementação propriamente dita dos algoritmos, o WEKA foi a ferramenta escolhida, satisfazendo, assim, as necessidades existentes.

5.2. APRESENTAÇÃO DOS DADOS

Os dados utilizados no caso de estudo subjacente a este projeto remontam ao estudo mencionado no capítulo 1.1 da autoria de Sérgio Moro, Raul Laureano e Paulo Cortez (Sérgio Moro et al., 2011). O estudo em questão utiliza uma base de dados derivada de uma campanha efetuada através do *call center* de um banco português cujo objetivo consistiu na subscrição de depósitos a prazo. O objetivo do estudo mencionado centrou-se na descoberta de um modelo de DM que conseguisse prever o sucesso de um contacto. A base de dados em questão contém dados referentes a 17 campanhas que ocorreram entre maio de 2008 e novembro de 2010.

Os autores mencionados, apesar de não disponibilizarem o *data set* utilizado nas suas experiências, disponibilizam dois subconjuntos do mesmo designados por "*Bank*" e "*BankAdd*". O primeiro é constituído por 17 variáveis e 45 211 observações. Já o segundo dispõe de 21 variáveis e 41 188 observações. Nenhuma informação extra acerca do *data set* original é disponibilizada.

Relativamente aos dois subconjuntos de dados referidos, ambos têm algumas variáveis em comum, sendo de destacar a variável dependente/*target*, y, que pode assumir os valores "*yes*" ou "*no*" consoante o cliente submeta ou não o depósito a prazo em campanha.

As variáveis que compõem os *data sets* podem ser agrupadas pelos seguintes temas:

- Variáveis relacionadas com o cliente;
- Variáveis relacionadas com o último contacto com o cliente na campanha corrente;
- Outras variáveis;
- Variável *target*.

As tabelas 5 e 6 apresentam os *data sets* "Bank" e "BankAdd", respetivamente.

Tabela 6 - Descrição do Data set "Bank"

Nome	Descrição	Valores
Variáveis relacionadas com o cliente		
age	Idade	[18; 95]
job	Profissão	'admin.', 'blue-collar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', 'unknown'
marital	Estado Civil	'divorced', 'married', 'single'
education	Escolaridade	'primary', 'secondary', 'tertiary', 'unknown'
default	O cliente tem crédito em *default?*	'no', 'yes'
balance		[-8 019; 102 127]
housing	O cliente contraiu um empréstimo à habitação?	'no', 'yes'
loan	O cliente contraiu um empréstimo ao consumo?	'no', 'yes'
Variáveis relacionadas com o último contacto realizado na campanha corrente		
contact	Meio de contacto utilizado	'cellular', 'telephone', 'unknown'
day	Dia referente ao último contacto realizado	[1; 31]
month	Mês referente ao último contacto realizado	'jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'set', 'oct', 'nov', 'dec'
duration	Duração do último contacto, em segundos	[0 ; 4 918]
Outras variáveis		
campaign	Número de contactos realizados durante uma campanha para um cliente	[1 ; 63]
pdays	Número de dias que passaram desde que o cliente foi contactado na campanha anterior	[-1 ; 871]
previous	Número de contactos realizados na campanha anterior para um cliente	[0 ; 275]
poutcome	Resultado da campanha anterior	'success', 'failure', 'unknown', 'other'
Variável target		
y	O cliente subscreveu o depósito a prazo?	'no', 'yes'

Tabela 5 - Descrição do Data set "BankAdd"

Nome	Descrição	Valores
Variáveis relacionadas com o cliente		
age	Idade	[17; 98]
job	Profissão	'admin.', 'blue-collar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', 'unknown'
marital	Estado Civil	'divorced', 'married', 'single', 'unknown'
education	Escolaridade	'basic.4y', 'basic.6y', 'basic.9y', 'high.school', 'iliterate', 'professional.course', 'university.degree', 'unknown'
default	O cliente tem crédito em *default?*	'no', 'yes', 'unknown'
housing	O cliente contraiu um empréstimo à habitação?	'no', 'yes', 'unknown'
loan	O cliente contraiu um empréstimo ao consumo?	'no', 'yes', 'unknown'
Variáveis relacionadas com o último contacto realizado na campanha corrente		
contact	Meio de contacto utilizado	'cellular', 'telephone'
month	Mês referente ao último contacto realizado	'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'set', 'oct', 'nov', 'dec'
day_of_week	Dia do último contacto realizado	mon', 'tue', 'wed', 'thu', 'fri'
duration	Duração do último contacto, em segundos	[0 ; 4 918]
Variáveis socioeconómicas		
emp.var.rate	Taxa de variação do emprego – Indicador trimestral	[-3,4 ; 1,4]
cons.price.idx	Índice de preços no consumidor – Indicador mensal	[92,201 ; 94,767]
cons.conf.idx	Índice de confiança do consumidor – Indicador mensal	[-50,8 ; -26,9]
euribor3m	Taxa Euribor a 3 meses – indicador diário	[0,634 ; 5,045]
nr.employed	Média trimestral do número de cidadãos empregados	[4963,6 ; 5228,1]
Outras variáveis		
campaign	Número de contactos realizados durante uma campanha para um cliente	[1 ; 56]
pdays	Número de dias que passaram desde que o cliente foi contactado na campanha anterior	[0 ; 999]
previous	Número de contactos realizados na campanha anterior para um cliente	[0 ; 7]
poutcome	Resultado da campanha anterior	'failure', 'nonexistent', 'success'
Variável target		
y	O cliente subscreveu o depósito a prazo?	'no', 'yes'

5.3. A ESCOLHA DOS ALGORITMOS

Segundo Jared Dean "dada a complexidade e a dimensão da maioria dos problemas modernos de *Data Mining*, a melhor prática consiste em experimentar várias técnicas e fazer várias tentativas por cada algoritmo, utilizando diferentes definições ou parâmetros" (Dean, 2014).

Desta forma, o caso de estudo apresentado foi pensado com o objetivo de maximizar o número de experiências realizadas e, também, a heterogeneidade entre os algoritmos. Não esquecendo o propósito de comparar e de tentar alcançar resultados melhores do que os obtidos pelos autores mencionados, é, ainda, essencial utilizar nas experiências os algoritmos que os autores utilizaram: Árvores de Decisão, Redes Neuronais, SVM e Regressões *Logit*. Assim, todos os algoritmos apresentados no capítulo 3 são experienciados neste caso prático.

Adicionalmente, de forma a esmiuçar a capacidade dos algoritmos, foi aplicado o método de seleção de atributos já apresentado no capítulo 4. Este método, baseado na correlação das variáveis, foi aplicado com duas estratégias diferentes ao nível da geração do novo subconjunto de variáveis: a estratégia *Best First* e a estratégia *Genetic Search.*

A secção 5.5 tem como objetivo descrever a metodologia utilizada na realização deste caso de estudo.

5.4. PARAMETRIZAÇÃO DOS ALGORITMOS

Construir modelos de classificação precisos e percetíveis implica não só escolher um algoritmo apropriado, mas também despender algum tempo na definição dos parâmetros dos algoritmos. Ajustar e/ou definir os parâmetros de forma manual e com elevada qualidade poderá consumir demasiado tempo. Adicionalmente, para tal, é necessário um excelente conhecimento sobre os algoritmos e sobre as propriedades do domínio de aprendizagem (Koblar, 2012).

Assim, dado o esforço necessário para a parametrização dos algoritmos, neste projeto foram utilizados os algoritmos parametrizados pelo WEKA.

5.5. METODOLOGIA APLICADA

Como já foi referido anteriormente, este caso de estudo conta com dois conjuntos de dados, os quais foram submetidos às mesmas experiências, tendo sido, por isso, utilizada a mesma metodologia. Este capítulo propõe, assim, apresentar e explicar a metodologia praticada. Esta encontra-se esquematizada na Figura 19.

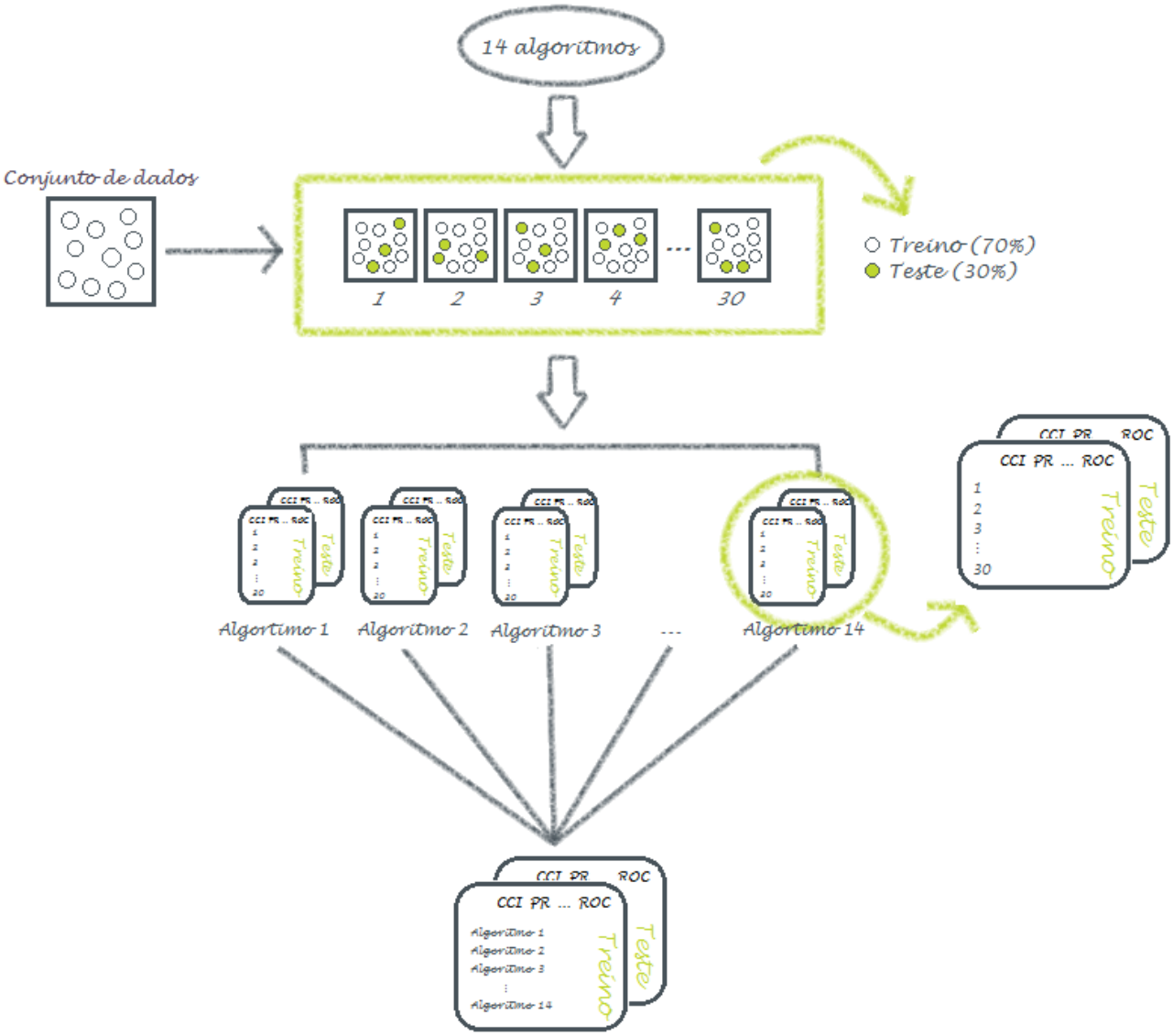

Figura 19 - Metodologia utilizada na execução prática deste trabalho

A primeira etapa deste caso prático consistiu na aplicação direta dos catorze algoritmos, isto é, sem qualquer seleção de atributos.

Sobre o conjunto de dados inicial, foram extraídas 30 partições distintas, sendo as mesmas compostas por um conjunto de treino e um conjunto de teste. Em cada partição, 70% das observações do conjunto de dados inicial (selecionadas de forma independente e identicamente distribuída) foram atribuídas ao conjunto de treino, sendo as restantes atribuídas ao conjunto de teste (Figura 20).

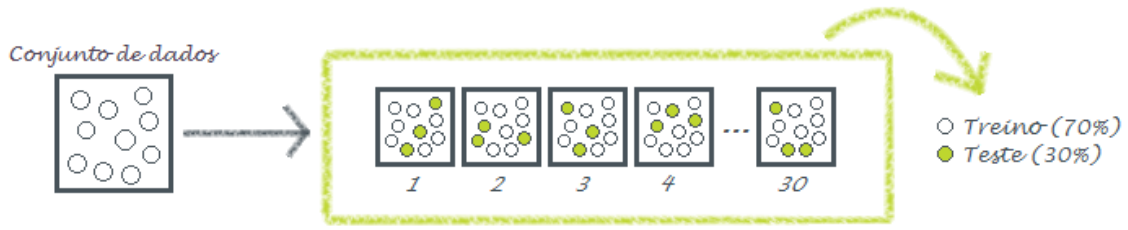

Figura 20 - Extração de 30 partições do data set original, sendo cada uma das partições repartida em dois conjuntos: treino (com 70% das observações) e teste (com as restantes 30%).

Numa primeira fase, os conjuntos de treino foram submetidos aos 14 algoritmos, o que permitiu a criação de catorze modelos de classificação das observações por cada partição). Posteriormente, cada conjunto de teste foi submetido aos 14 modelos gerados a partir do respetivo conjunto de treino. Desta forma, por cada partição foram gerados dois *outputs* (um para o conjunto de treino e outro para o conjunto de teste), contemplando as diferentes medidas de qualidade respeitantes à performance de cada um dos algoritmos (Figura 21).

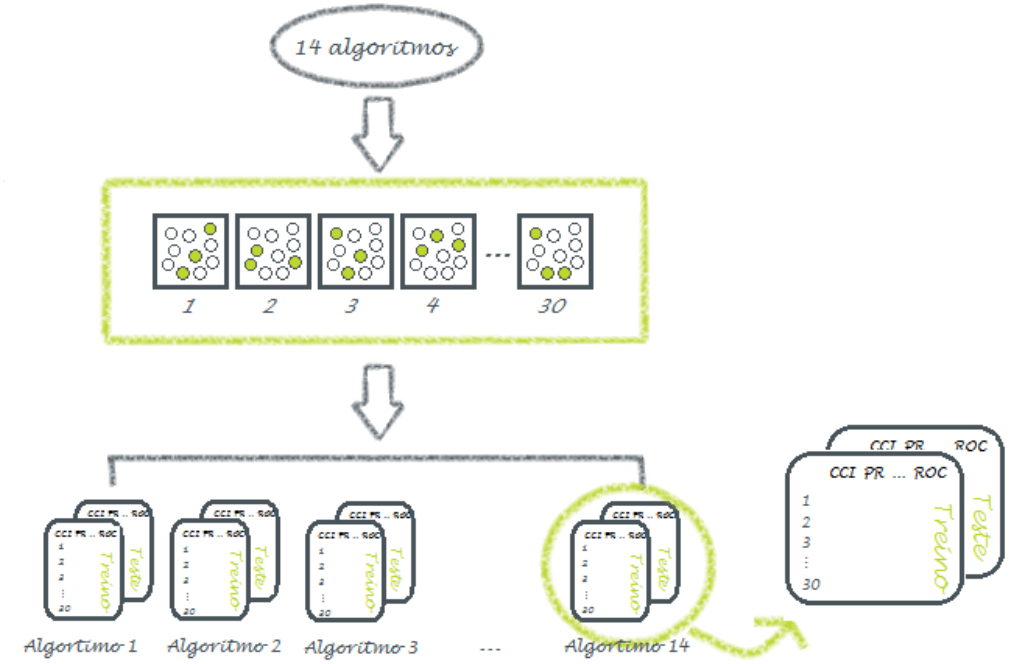

Figura 21 - Outputs de todos os 14 algoritmos, referente ao conjunto de treino e ao conjunto de teste de cada uma das partições

As medidas de qualidade destacadas para analisar a performance dos algoritmos foram:

- CCI - Percentagem de observações corretamente classificadas (*Correctly Classified Instances*);
- PR - Precisão, que resulta da fórmula TP/(TP+FP);
- RE - *Recall* ou Sensivity cuja fórmula é TP/(TP+FN);
- FM - *F-Measure* que tem por base a fórmula 2xPRxRE/(PR+RE);
- ROC - Que corresponde à área abaixo da curva ROC;

Onde,

FN, diz respeito a Falsos Negativos,
FP, representa os Falsos Positivos
TP, refere-se a Verdadeiros Positivos
(conceitos já abordados no Capítulo 3.1.4).

Quanto mais elevadas forem estas medidas (máximo de 100 para a medida CCI e 1 para as restantes), melhor é a qualidade dos algoritmos.

Cada *output* (treino e teste de cada algoritmos) contemplou as diferentes medidas de qualidade para cada uma das partições (rever Figura 21). É de notar que, neste ponto da experiência, cada um dos 14 algoritmos gerou um valor por cada uma das 5 medidas de qualidade, para os conjuntos de treino e de teste de cada uma das 30 partições. Tornou-se necessário simplificar os resultados obtidos, transformando-os num único valor para cada conjugação entre: medida de qualidade, conjunto (treino e teste) e algoritmo.

Para tal, foram calculadas as médias e os desvios-padrões das 30 partições para cada medida de qualidade, conjunto (treino e teste) e algoritmo. Este tratamento dos resultados foi possível tendo em conta o processo estocástico associado à aplicação dos vários algoritmos a cada uma das partições na fase experimental.

O output final compreende assim duas tabelas, uma para cada conjunto (treino e teste), tendo nas colunas cada uma das cinco medidas de qualidade e em linha cada um dos algoritmos (Figura 22).

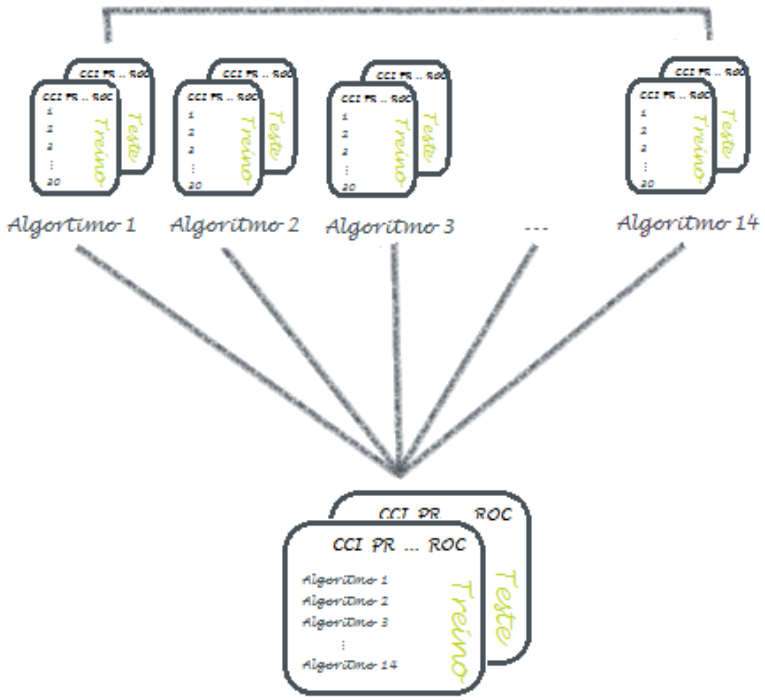

Figura 22 - Output final, resultante da simplificação dos resultados obtidos. O output final consiste em duas tabelas para cada data set, contendo as médias e os desvios-padrões de cada medida de qualidade para cada algoritmo e para cada conjunto (treino e teste)

Por cada linha-coluna é apresentada a média e dentro de parênteses o desvio-padrão das várias partições para cada algoritmo. Na secção seguinte são exibidas as tabelas 7, 8, 9 e 10, que contêm os resultados obtidos para os dois conjuntos (treino e teste) de cada *data set*, "Bank" e "BankAdd".

Numa segunda fase deste projeto, foi aplicada a seleção de atributos de duas formas distintas, como já foi referido anteriormente. Assim, antes da extração das 30 partições do *data set* inicial, este sofreu uma redução ao nível das variáveis, através da seleção de atributos baseada na correlação das

variáveis, sendo utilizada, numa primeira fase a estratégia *Best First* (Figura 23). Após a obtenção do novo *data set*, chamemos-lhe "Bank_BF" e "BankAdd_BF", foi seguida a metodologia já explicada anteriormente.

De igual forma, ao conjunto de dados inicial, foi, numa segunda fase, aplicada a seleção de atributos baseada na correlação das variáveis, desta vez com a estratégia *Genetic Search*. A metodologia já conhecida foi aplicada, novamente, tendo por base, desta vez, os novos *data sets* "Bank_GS" e "BankAdd_GS".

Na secção seguinte, as tabelas 11 a 18, apresentam os *outputs* obtidos para cada conjunto (treino e teste) de cada *data set* para cada uma das estratégias da seleção de atributos baseada na correlação das variáveis.

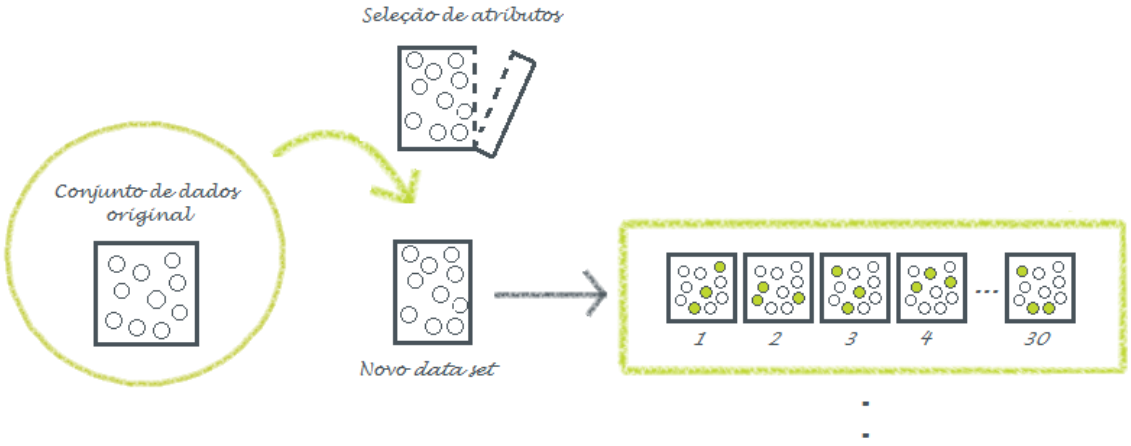

Figura 23 - Metodologia aplicada: Seleção de atributos

5.6. Resultados Obtidos

5.6.1. Resultados obtidos antes da seleção de atributos

Tal como referido anteriormente, a primeira etapa deste projeto recaiu sobre a aplicação dos algoritmos diretamente sobre os dois conjuntos de dados iniciais, "Bank" e "BankAdd", ou seja, sem qualquer seleção de atributos.

As tabelas 7 e 8 apresentam os resultados obtidos para os conjuntos de treino e teste do *data set* "Bank".

Os resultados diferem entre o conjunto de treino e o conjunto de teste. No conjunto de treino, os algoritmos pertencentes ao grupo dos *Ensemble*, apresentam valores bastante elevados, em alguns casos até perfeitos, para as diferentes medidas de qualidade, recaindo o grande foco sobre os algoritmos *Random Trees* e *Random Forest*.

Analisando os resultados obtidos no conjunto de teste, os algoritmos com melhor qualidade para as diferentes medidas já não se centram apenas no grupo dos *Ensemble*, notando-se também valores elevados no grupo das Regressões. Desta vez o destaque recai sobre as Regressões *Logit* e sobre o algoritmo *Bagging*.

*Tabela 7 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **treino** do data set "**Bank**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

Bank Conjunto de Treino		CCI	PR	RE	FM	ROC
Decision Trees	CART	88,1(16,5)	0,9(0,008)	0,912(0,006)	0,9(0,008)	0,762(0,051)
	C4.5	92,8(0,7)	0,922(0,009)	0,928(0,007)	0,92(0,01)	0,862(0,02)
Redes Neuronais	Percetrão Multicamada	98,3(0,3)	0,983(0,003)	0,983(0,003)	0,983(0,003)	0,953(0,008)
SVM	SVM	89,4(0,3)	0,872(0,004)	0,894(0,003)	0,864(0,004)	0,574(0,006)
Voted Perceptron	Voted Perceptron	80,7(15,1)	0,802(0,005)	0,835(0,007)	0,817(0,005)	0,51(0,007)
Ensemble	Bagging	93,5(0,4)	0,933(0,004)	0,936(0,004)	0,928(0,005)	0,963(0,005)
	Boosting	88,6(0,3)	0,791(0,02)	0,886(0,003)	0,833(0,004)	0,798(0,019)
	RandomForest	99,1(0,1)	0,991(0,001)	0,991(0,001)	0,991(0,001)	1(0)
	RandomTrees	100(0)	1(0)	1(0)	1(0)	1(0)
	ADTree	89,9(0,4)	0,886(0,007)	0,899(0,004)	0,882(0,013)	0,889(0,006)
Aprendizagem *Bayesiana*	Redes Bayesianas	88,5(0,4)	0,878(0,004)	0,885(0,004)	0,881(0,004)	0,875(0,005)
	Naive Bayes	87(0,6)	0,879(0,004)	0,87(0,006)	0,874(0,005)	0,851(0,005)
Regressões	Regressão Linear Simples	90,9(0,5)	0,897(0,007)	0,909(0,005)	0,896(0,008)	0,926(0,013)
	Logit	90,6(0,3)	0,893(0,004)	0,906(0,003)	0,892(0,004)	0,904(0,004)

*Tabela 8 - Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **teste** do data set "**Bank**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

Bank Conjunto de Teste		CCI	PR	RE	FM	ROC
Decision Trees	CART	89,5(0,7)	0,879(0,008)	0,895(0,007)	0,881(0,009)	0,744(0,039)
	C4.5	89,4(0,8)	0,879(0,01)	0,894(0,008)	0,882(0,01)	0,771(0,029)
Redes Neuronais	Percetrão Multicamada	88,1(1,1)	0,873(0,009)	0,881(0,011)	0,876(0,01)	0,828(0,015)
SVM	SVM	89,1(0,7)	0,869(0,009)	0,891(0,006)	0,861(0,01)	0,576(0,012)
Voted Perceptron	Voted Perceptron	83,4(1)	0,8(0,014)	0,834(0,009)	0,815(0,011)	0,517(0,014)
Ensemble	Bagging	89,5(0,7)	0,877(0,009)	0,895(0,007)	0,878(0,009)	0,893(0,014)
	Boosting	88,2(0,7)	0,785(0,022)	0,882(0,007)	0,828(0,01)	0,792(0,027)
	RandomForest	89(0,8)	0,868(0,011)	0,89(0,008)	0,87(0,01)	0,863(0,017)
	RandomTrees	86,3(1,1)	0,859(0,012)	0,863(0,011)	0,861(0,011)	0,675(0,023)
	ADTree	89,3(0,7)	0,879(0,011)	0,893(0,007)	0,874(0,016)	0,869(0,013)
Aprendizagem *Bayesiana*	Redes Bayesianas	87,8(0,8)	0,87(0,009)	0,878(0,008)	0,873(0,008)	0,856(0,014)
	Naive Bayes	86,9(1)	0,877(0,007)	0,869(0,009)	0,873(0,008)	0,843(0,016)
Regressões	Regressão Linear Simples	89,6(0,6)	0,88(0,008)	0,896(0,006)	0,881(0,009)	0,882(0,01)
	Logit	90,1(0,6)	0,886(0,008)	0,901(0,006)	0,886(0,008)	0,887(0,011)

À semelhança das tabelas 7 e 8, as tabelas 9 e 10 apresentam os resultados obtidos para os conjuntos de treino e teste do *data set* "BankAdd".

Mais uma vez é possível observar a eficácia dos algoritmos pertencentes ao grupo dos *Ensemble* no conjunto de treino. O destaque recai novamente nos algoritmos *Random Trees* e *Random Forest*.

No que diz respeito ao conjunto de teste, as atenções voltam-se a centrar nos algoritmos *Ensemble* e nas Regressões, mais concretamente nos mesmos algoritmos apurados no conjunto de dados "Bank", ou seja, as Regressões *Logit* e o algoritmo *Bagging*.

Os resultados obtidos pelos autores Moro, Laureano, & Cortez (2011) serão apresentados mais à frente. No entanto, e adiantando desde já a medida de qualidade observada pelos mesmos, é de destacar que o algoritmo *Bagging* em ambos os *data sets* é o que obtém melhores valores para a medida de qualidade ROC.

*Tabela 9 - Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **treino** do data set "**BankAdd**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

BankAdd Conjunto de Treino		CCI	PR	RE	FM	ROC
Decision Trees	CART	92,7(1,2)	0,922(0,013)	0,927(0,012)	0,923(0,013)	0,866(0,031)
	C4.5	94,1(0,6)	0,938(0,007)	0,941(0,006)	0,938(0,008)	0,908(0,041)
Redes Neuronais	Percetrão Multicamada	99,2(0,1)	0,992(0,001)	0,992(0,001)	0,992(0,001)	0,974(0,007)
SVM	SVM	90,7(0,3)	0,892(0,005)	0,907(0,003)	0,889(0,005)	0,631(0,019)
Voted Perceptron	Voted Perceptron	90,1(0,4)	0,883(0,006)	0,901(0,004)	0,872(0,009)	0,592(0,031)
Ensemble	Bagging	95,1(0,3)	0,949(0,003)	0,951(0,003)	0,948(0,003)	0,975(0,003)
	Boosting	89,2(0,5)	0,813(0,039)	0,892(0,005)	0,843(0,01)	0,882(0,019)
	RandomForest	99,3(0,2)	0,993(0,002)	0,993(0,002)	0,993(0,002)	1(0)
	RandomTrees	100(0)	1(0)	1(0)	1(0)	1(0)
	ADTree	91(0,5)	0,904(0,009)	0,911(0,005)	0,902(0,011)	0,937(0,004)
Aprendizagem *Bayesiana*	Redes Bayesianas	87,2(0,4)	0,889(0,004)	0,872(0,003)	0,879(0,004)	0,876(0,007)
	Naive Bayes	87,7(0,3)	0,896(0,003)	0,877(0,003)	0,885(0,003)	0,873(0,004)
Regressões	Regressão Linear Simples	92,4(0,5)	0,916(0,006)	0,924(0,005)	0,917(0,006)	0,957(0,005)
	Logit	91,9(0,2)	0,91(0,003)	0,919(0,002)	0,911(0,003)	0,943(0,003)

*Tabela 10 - Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **teste** do data set "**BankAdd**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

BankAdd Conjunto de Teste		CCI	PR	RE	FM	ROC
Decision Trees	CART	90,9(0,7)	0,902(0,01)	0,909(0,007)	0,904(0,009)	0,856(0,035)
	C4.5	90,3(0,7)	0,897(0,009)	0,903(0,007)	0,898(0,008)	0,816(0,049)
Redes Neuronais	Percetrão Multicamada	89,2(0,6)	0,883(0,009)	0,892(0,006)	0,886(0,007)	0,865(0,013)
SVM	SVM	90,4(0,8)	0,887(0,011)	0,904(0,008)	0,887(0,012)	0,63(0,023)
Voted Perceptron	Voted Perceptron	90,2(1)	0,886(0,012)	0,902(0,01)	0,874(0,017)	0,596(0,03)
Ensemble	Bagging	90,9(0,6)	0,901(0,007)	0,909(0,006)	0,903(0,007)	0,93(0,01)
	Boosting	89,2(0,9)	0,815(0,038)	0,892(0,009)	0,844(0,014)	0,877(0,023)
	RandomForest	90,4(0,7)	0,89(0,009)	0,904(0,007)	0,892(0,009)	0,904(0,009)
	RandomTrees	88,2(0,9)	0,881(0,011)	0,882(0,008)	0,881(0,009)	0,718(0,025)
	ADTree	90,5(0,8)	0,898(0,013)	0,905(0,008)	0,897(0,014)	0,923(0,008)
Aprendizagem *Bayesiana*	Redes Bayesianas	87,5(0,8)	0,894(0,01)	0,875(0,008)	0,883(0,008)	0,877(0,017)
	Naive Bayes	88,1(0,6)	0,902(0,007)	0,881(0,006)	0,889(0,005)	0,883(0,012)
Regressões	Regressão Linear Simples	91(0,6)	0,901(0,008)	0,91(0,005)	0,903(0,007)	0,925(0,011)
	Logit	91,4(0,6)	0,905(0,007)	0,914(0,006)	0,907(0,007)	0,931(0,006)

5.6.2. Resultados obtidos com a seleção de atributos: estratégia Best First

Esta secção apresenta e extrai as principais conclusões relativas aos *outputs* obtidos na segunda fase do projeto, ou seja, quando as experiências foram realizadas após o redimensionamento da base de dados pela estratégia *Best First*. Os novos conjuntos de dados denominam-se por "Bank_BF" e "BankAdd_BF", sendo o primeiro derivado do *data set* "Bank" e o segundo do *data set* "BankAdd".

Começando pela análise dos *outputs* obtidos com a aplicação dos algoritmos sobre o conjunto de dados "Bank_BF", (Tabela 11 e Tabela 12) é percetível que, ao nível do conjunto de treino, o algoritmo *Random Trees* volta a marcar a diferença, apresentando valores quase perfeitos para todas as medidas de qualidade. O grupo dos *Ensemble* continua a ser, à semelhança do que se verificou com o conjunto de treino do *data set* "Bank", o grupo com melhores valores para todas as medidas de qualidade. É, no entanto de destacar que, para o conjunto de treino, a seleção de atributos, não trouxe qualquer melhoria em termos de medidas de qualidade.

*Tabela 11 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **treino** do data set "**Bank_BF**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

Bank_BF Conjunto de Treino		CCI	PR	RE	FM	ROC
Decision Trees	CART	91,2(0,6)	0,9(0,008)	0,912(0,006)	0,9(0,008)	0,762(0,051)
	C4.5	90,2(0,5)	0,891(0,006)	0,902(0,005)	0,887(0,013)	0,788(0,079)
Redes Neuronais	Percetrão Multicamada	90,2(0,3)	0,887(0,004)	0,902(0,002)	0,885(0,006)	0,865(0,009)
SVM	SVM	89,4(0,3)	0,872(0,004)	0,894(0,003)	0,864(0,004)	0,574(0,006)
Voted Perceptron	Voted Perceptron	83,5(0,7)	0,802(0,005)	0,835(0,007)	0,817(0,005)	0,51(0,007)
Ensemble	Bagging	92,1(0,3)	0,914(0,004)	0,921(0,003)	0,911(0,004)	0,922(0,008)
	Boosting	88,5(0)	0,789(0,019)	0,885(0)	0,831(0,001)	0,804(0,021)
	RandomForest	98,2(0,2)	0,982(0,002)	0,982(0,002)	0,982(0,002)	0,998(0)
	RandomTrees	99,6(0,1)	0,996(0,001)	0,996(0,001)	0,996(0,001)	1(0)
	ADTree	89,7(0,3)	0,887(0,007)	0,897(0,003)	0,884(0,012)	0,874(0,006)
Aprendizagem *Bayesiana*	Redes Bayesianas	89,6(0,3)	0,877(0,006)	0,896(0,003)	0,871(0,007)	0,859(0,007)
	Naive Bayes	89,4(0,3)	0,877(0,004)	0,894(0,003)	0,882(0,003)	0,847(0,008)
Regressões	Regressão Linear Simples	89,9(0,4)	0,883(0,007)	0,899(0,004)	0,882(0,009)	0,86(0,009)
	Logit	89,9(0,2)	0,882(0,004)	0,899(0,002)	0,882(0,004)	0,853(0,008)

Analisando os resultados obtidos ao nível do conjunto de teste, mais uma vez os resultados não superam os obtidos antes da seleção de atributos. No que diz respeito aos algoritmos com melhor performance, notam-se valores de qualidade altos em quatro algoritmos (CART, ADTree, *Naive Bayes* e *Logit*), sendo que anteriormente à seleção de atributos, os valores mais altos estavam concentrados em dois algoritmos (Bagging e *Logit*). Relativamente à medida de qualidade ROC, o destaque recai sobre o algoritmo ADTree ao invés do algoritmo Bagging destacado no output do processo sem seleção de atributos.

*Tabela 12 - Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **teste** do data set "**Bank_BF**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

Bank_BF Conjunto de Teste		CCI	PR	RE	FM	ROC
Decision Trees	CART	89,5(0,7)	0,879(0,008)	0,895(0,007)	0,881(0,009)	0,744(0,039)
	C4.5	89,1(0,6)	0,876(0,008)	0,892(0,005)	0,875(0,011)	0,763(0,077)
Redes Neuronais	Percetrão Multicamada	89,3(0,5)	0,874(0,007)	0,893(0,005)	0,876(0,007)	0,854(0,016)
SVM	SVM	89,1(0,7)	0,869(0,009)	0,891(0,006)	0,861(0,01)	0,576(0,012)
Voted Perceptron	Voted Perceptron	83,4(1)	0,8(0,014)	0,834(0,009)	0,815(0,011)	0,517(0,014)
Ensemble	Bagging	89,3(0,6)	0,875(0,007)	0,893(0,005)	0,879(0,006)	0,852(0,016)
	Boosting	88,5(0)	0,791(0,021)	0,885(0)	0,831(0,002)	0,798(0,024)
	RandomForest	87,4(0,7)	0,859(0,008)	0,874(0,007)	0,865(0,007)	0,772(0,015)
	RandomTrees	85,1(0,8)	0,853(0,007)	0,851(0,008)	0,852(0,007)	0,642(0,017)
	ADTree	89,1(0,5)	0,878(0,011)	0,891(0,005)	0,877(0,013)	0,856(0,014)
Aprendizagem *Bayesiana*	Redes Bayesianas	89,4(0,4)	0,872(0,008)	0,894(0,004)	0,869(0,006)	0,851(0,015)
	Naive Bayes	89,3(0,6)	0,876(0,007)	0,893(0,006)	0,881(0,006)	0,841(0,014)
Regressões	Regressão Linear Simples	89,4(0,5)	0,876(0,009)	0,894(0,005)	0,877(0,009)	0,848(0,014)
	Logit	89,7(0,4)	0,879(0,007)	0,897(0,004)	0,88(0,005)	0,849(0,016)

Passando à análise referente à seleção de atributos com a estratégia *Best First* sobre o *data set* "BankAdd" (Figuras 13 e 14) volta a ser destaque, ao nível do conjunto de treino, os valores obtidos pelo algoritmo *Random Trees* referentes a todas as medidas de qualidade. À semelhança do *data set* "Bank", no *data* set "BankAdd" os resultados obtidos no conjunto de treino com seleção de atributos através da abordagem *Best* First" não trouxeram qualquer melhoria em termos de medidas de qualidade.

*Tabela 13 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **treino** do data set "**BankAdd_BF**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

BankAdd_BF Conjunto de Treino		CCI	PR	RE	FM	ROC
Decision Trees	CART	92,2(0,5)	0,916(0,005)	0,922(0,004)	0,916(0,005)	0,868(0,01)
	C4.5	92,1(0,4)	0,916(0,005)	0,921(0,004)	0,916(0,005)	0,893(0,034)
Redes Neuronais	Percetrão Multicamada	91,3(0,2)	0,904(0,004)	0,913(0,002)	0,906(0,004)	0,927(0,004)
SVM	SVM	90(0,2)	0,88(0,004)	0,9(0,002)	0,879(0,003)	0,606(0,008)
Voted Perceptron	Voted Perceptron	90(0,4)	0,884(0,007)	0,9(0,004)	0,87(0,01)	0,586(0,029)
Ensemble	Bagging	93,7(0,3)	0,932(0,003)	0,937(0,003)	0,933(0,003)	0,955(0,005)
	Boosting	89,5(0,5)	0,84(0,047)	0,895(0,005)	0,852(0,014)	0,89(0,022)
	RandomForest	98,1(0,2)	0,981(0,002)	0,981(0,002)	0,98(0,002)	0,997(0,001)
	RandomTrees	98,9(0,1)	0,99(0,001)	0,989(0,001)	0,989(0,001)	0,999(0)
	ADTree	91,3(0,5)	0,91(0,009)	0,913(0,005)	0,908(0,01)	0,935(0,004)
Aprendizagem *Bayesiana*	Redes Bayesianas	91,2(0,3)	0,9(0,005)	0,912(0,003)	0,9(0,006)	0,925(0,007)
	Naive Bayes	90,9(0,3)	0,899(0,003)	0,909(0,003)	0,902(0,003)	0,922(0,004)
Regressões	Regressão Linear Simples	91,8(0,3)	0,909(0,004)	0,918(0,003)	0,909(0,005)	0,938(0,005)
	Logit	91,3(0,2)	0,902(0,003)	0,913(0,002)	0,903(0,003)	0,925(0,004)

Relativamente ao conjunto de teste do *data set* "BankAdd_BF", os resultados, mais uma vez, não apresentam melhorias de qualidade face ao processo sem seleção de atributos, sendo o destaque ao nível da medida de qualidade ROC para as Regressões Lineares Simples (ficando atrás dos resultados obtidos através do algoritmo *Bagging* aquando da não seleção de atributos).

Tabela 14 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de ***teste*** *do data set "**BankAdd_BF**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

BankAdd_BF Conjunto de Teste		CCI	PR	RE	FM	ROC
Decision Trees	CART	91,1(0,6)	0,903(0,006)	0,911(0,006)	0,905(0,006)	0,852(0,017)
	C4.5	90,8(0,6)	0,901(0,006)	0,908(0,006)	0,902(0,006)	0,872(0,03)
Redes Neuronais	Percetrão Multicamada	91(0,5)	0,901(0,006)	0,91(0,005)	0,903(0,006)	0,923(0,008)
SVM	SVM	90,1(0,5)	0,882(0,008)	0,901(0,005)	0,881(0,007)	0,61(0,018)
Voted Perceptron	Voted Perceptron	89,9(0,4)	0,88(0,008)	0,899(0,004)	0,869(0,01)	0,585(0,029)
Ensemble	Bagging	90,7(0,6)	0,899(0,006)	0,908(0,006)	0,902(0,006)	0,919(0,009)
	Boosting	89,4(0,5)	0,838(0,046)	0,894(0,004)	0,851(0,013)	0,88(0,024)
	RandomForest	89(0,6)	0,884(0,006)	0,89(0,006)	0,886(0,006)	0,835(0,015)
	RandomTrees	88,5(0,6)	0,883(0,006)	0,885(0,006)	0,884(0,006)	0,704(0,018)
	ADTree	90,6(0,8)	0,902(0,01)	0,906(0,007)	0,901(0,01)	0,921(0,01)
Aprendizagem *Bayesiana*	Redes Bayesianas	91(0,6)	0,897(0,009)	0,91(0,006)	0,897(0,008)	0,913(0,011)
	Naive Bayes	90,8(0,6)	0,899(0,007)	0,909(0,006)	0,902(0,006)	0,919(0,009)
Regressões	Regressão Linear Simples	91,2(0,5)	0,902(0,006)	0,912(0,005)	0,902(0,005)	0,925(0,008)
	Logit	91,1(0,3)	0,9(0,004)	0,912(0,003)	0,902(0,004)	0,921(0,008)

5.6.3. Resultados obtidos com a seleção de atributos: estratégia *Genetic Search*.

Esta secção apresenta os resultados obtidos com a seleção de atributos com a estratégia *Genetic Search*. Os conjuntos de dados obtidos após o redimensionamento com esta abordagem denominam-se por "Bank_GS" e "BankAdd_GS", os quais foram apurados através dos *data sets* "Bank" e "BankAdd", respetivamente.

A primeira conclusão após a aplicação da estratégia *Genetic Search* na redução dos *data sets*, é a de que os resultados foram piores do que os resultados obtidos na aplicação dos algoritmos sem a seleção de atributos.

Relativamente ao *data set* "Bank_GS" (Tabela 15 e Tabela 16) não existe qualquer evidência de melhoria de resultados face aos obtidos com o *data set* "Bank_BF". Os resultados são idênticos nos conjuntos de treino e teste e os algoritmos com melhor *performance* são exatamente os mesmos que os obtidos na estratégia *Best First*: *Logit, Naive Bayes* e *ADTree*.

*Tabela 15 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **treino** do data set "**Bank_GS**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

Bank_GS Conjunto de Treino		CCI	PR	RE	FM	ROC
Decision Trees	CART	90,6(0,7)	0,894(0,009)	0,906(0,007)	0,894(0,01)	0,757(0,042)
	C4.5	90,3(0,5)	0,891(0,006)	0,903(0,005)	0,888(0,013)	0,79(0,078)
Redes Neuronais	Percetrão Multicamada	90,2(0,3)	0,887(0,004)	0,902(0,002)	0,885(0,006)	0,865(0,009)
SVM	SVM	89,3(0,2)	0,872(0,004)	0,893(0,002)	0,863(0,002)	0,575(0,005)
Voted Perceptron	Voted Perceptron	88,5(0)	0,787(0,017)	0,885(0)	0,831(0,005)	0,511(0,014)
Ensemble	Bagging	92,1(0,3)	0,914(0,004)	0,922(0,003)	0,911(0,004)	0,922(0,008)
	Boosting	88,5(0)	0,788(0,019)	0,885(0)	0,83(0,001)	0,803(0,022)
	RandomForest	98,2(0,2)	0,982(0,002)	0,982(0,002)	0,982(0,002)	0,998(0)
	RandomTrees	99,6(0,1)	0,996(0,001)	0,996(0,001)	0,996(0,001)	1(0)
	ADTree	89,7(0,3)	0,887(0,007)	0,897(0,003)	0,884(0,012)	0,874(0,006)
Aprendizagem *Bayesiana*	Redes Bayesianas	89,6(0,3)	0,877(0,006)	0,896(0,003)	0,871(0,007)	0,859(0,007)
	Naive Bayes	89,4(0,3)	0,877(0,004)	0,894(0,003)	0,881(0,003)	0,847(0,008)
Regressões	Regressão Linear Simples	89,9(0,4)	0,883(0,007)	0,899(0,004)	0,882(0,009)	0,86(0,009)
	Logit	89,9(0,2)	0,882(0,004)	0,899(0,002)	0,882(0,003)	0,853(0,008)

*Tabela 16 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de **teste** do data set "**Bank_GS**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

Bank_GS Conjunto de Teste		CCI	PR	RE	FM	ROC
Decision Trees	CART	89,5(0,5)	0,879(0,007)	0,895(0,005)	0,881(0,008)	0,737(0,033)
	C4.5	89,2(0,6)	0,876(0,008)	0,892(0,005)	0,875(0,01)	0,764(0,076)
Redes Neuronais	Percetrão Multicamada	89,4(0,5)	0,874(0,007)	0,894(0,005)	0,876(0,007)	0,854(0,015)
SVM	SVM	89,2(0,4)	0,869(0,009)	0,893(0,003)	0,863(0,004)	0,573(0,009)
Voted Perceptron	Voted Perceptron	88,5(0,1)	0,786(0,01)	0,885(0,001)	0,831(0,003)	0,509(0,01)
Ensemble	Bagging	89,3(0,6)	0,875(0,007)	0,893(0,005)	0,879(0,006)	0,852(0,015)
	Boosting	88,5(0)	0,791(0,021)	0,885(0)	0,831(0,002)	0,797(0,025)
	RandomForest	87,5(0,7)	0,859(0,008)	0,875(0,007)	0,866(0,007)	0,772(0,015)
	RandomTrees	85,1(0,8)	0,853(0,007)	0,851(0,008)	0,852(0,007)	0,641(0,018)
	ADTree	89,1(0,5)	0,877(0,011)	0,891(0,005)	0,877(0,014)	0,856(0,014)
Aprendizagem *Bayesiana*	Redes Bayesianas	89,4(0,4)	0,872(0,008)	0,894(0,003)	0,869(0,006)	0,851(0,015)
	Naive Bayes	89,3(0,6)	0,877(0,007)	0,893(0,006)	0,881(0,006)	0,841(0,014)
Regressões	Regressão Linear Simples	89,4(0,5)	0,876(0,009)	0,894(0,005)	0,877(0,009)	0,848(0,014)
	Logit	89,7(0,4)	0,879(0,007)	0,897(0,004)	0,88(0,005)	0,848(0,015)

No que diz respeito ao *data set* "BankAdd_GS", existe uma melhoria no conjunto de treino face aos resultados obtidos com o *data set* "BankAdd_BF" (Figura 17). Relativamente ao conjunto de teste (Figura 18) os resultados são semelhantes. No entanto, a estratégia *Best First* obteve resultados ligeiramente superiores aos obtidos com a estratégia *Genetic Search*. Nota-se ainda que na primeira estratégia, os melhores resultados estão concentrados em três algoritmos (CART, Regressões Lineares Simples e *Logit*) enquanto nesta última estratégia estão dispersos por quatro algoritmos (Percetrão Multicamada, *Bagging*, CART e *Logit*). Relativamente à medida de qualidade ROC, esta apresenta os

mesmos valores com ambas as estratégias, sendo, no primeiro caso, obtidos pela Regressão Linear Simples e no segundo caso pelo algoritmo *Bagging*.

Tabela 17 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de ***treino*** *do data set "**BankAdd_GS**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

BankAdd_GS Conjunto de Treino		CCI	PR	RE	FM	ROC
Decision Trees	CART	92,5(0,8)	0,919(0,009)	0,925(0,008)	0,92(0,009)	0,872(0,015)
	C4.5	93(0,7)	0,924(0,007)	0,93(0,007)	0,924(0,009)	0,892(0,044)
Redes Neuronais	Percetrão Multicamada	92,4(0,3)	0,919(0,004)	0,924(0,003)	0,92(0,004)	0,933(0,006)
SVM	SVM	90,1(0,2)	0,882(0,004)	0,901(0,002)	0,879(0,003)	0,601(0,008)
Voted Perceptron	Voted Perceptron	90,1(0,4)	0,884(0,007)	0,901(0,004)	0,871(0,01)	0,589(0,029)
Ensemble	Bagging	94,6(0,4)	0,943(0,004)	0,946(0,004)	0,943(0,004)	0,968(0,004)
	Boosting	89,5(0,5)	0,84(0,047)	0,895(0,005)	0,852(0,014)	0,892(0,022)
	RandomForest	99,1(0,2)	0,991(0,002)	0,991(0,002)	0,991(0,002)	1(0)
	RandomTrees	100(0)	1(0)	1(0)	1(0)	1(0)
	ADTree	91,1(0,4)	0,908(0,011)	0,911(0,004)	0,905(0,012)	0,938(0,004)
Aprendizagem *Bayesiana*	Redes Bayesianas	89,6(0,3)	0,895(0,002)	0,896(0,003)	0,896(0,003)	0,92(0,004)
	Naive Bayes	90,5(0,4)	0,897(0,003)	0,905(0,004)	0,9(0,003)	0,917(0,005)
Regressões	Regressão Linear Simples	91,9(0,3)	0,911(0,004)	0,919(0,003)	0,911(0,005)	0,943(0,006)
	Logit	91,6(0,2)	0,906(0,003)	0,916(0,002)	0,907(0,003)	0,927(0,004)

Tabela 18 – Síntese dos resultados obtidos após a aplicação dos algoritmos sobre o conjunto de ***teste*** *do data set "**BankAdd_GS**". Apresentação da média dos valores obtidos por cada uma das partições para cada medida de qualidade.*

BankAdd_GS Conjunto de Teste		CCI	PR	RE	FM	ROC
Decision Trees	CART	90,9(0,6)	0,901(0,007)	0,909(0,006)	0,903(0,006)	0,849(0,021)
	C4.5	90,6(0,7)	0,896(0,009)	0,906(0,007)	0,898(0,008)	0,841(0,054)
Redes Neuronais	Percetrão Multicamada	91(0,6)	0,903(0,006)	0,91(0,006)	0,904(0,005)	0,915(0,01)
SVM	SVM	90,1(0,4)	0,882(0,008)	0,901(0,004)	0,879(0,007)	0,604(0,018)
Voted Perceptron	Voted Perceptron	89,9(0,4)	0,881(0,007)	0,899(0,004)	0,87(0,009)	0,587(0,029)
Ensemble	Bagging	90,9(0,6)	0,901(0,007)	0,909(0,006)	0,904(0,006)	0,925(0,01)
	Boosting	89,4(0,5)	0,838(0,046)	0,894(0,004)	0,851(0,013)	0,882(0,024)
	RandomForest	90,2(0,6)	0,893(0,007)	0,903(0,006)	0,896(0,006)	0,875(0,014)
	RandomTrees	88,4(0,6)	0,885(0,007)	0,884(0,006)	0,884(0,006)	0,708(0,021)
	ADTree	90,3(0,6)	0,898(0,012)	0,903(0,006)	0,896(0,011)	0,918(0,009)
Aprendizagem *Bayesiana*	Redes Bayesianas	89,5(0,5)	0,894(0,005)	0,895(0,005)	0,894(0,005)	0,912(0,009)
	Naive Bayes	90,5(0,6)	0,897(0,006)	0,905(0,006)	0,9(0,006)	0,914(0,01)
Regressões	Regressão Linear Simples	91,1(0,5)	0,902(0,005)	0,912(0,005)	0,903(0,005)	0,923(0,01)
	Logit	91,3(0,4)	0,903(0,006)	0,913(0,004)	0,904(0,005)	0,922(0,009)

5.6.4. Síntese dos Resultados

Esta secção apresenta os algoritmos que nas mesmas experiências apresentadas anteriormente mostraram uma melhor *performance.*

Adicionalmente, esta secção explora apenas os resultados referentes ao conjunto de teste de ambos os *data sets*, visto que é através das experiências com os conjuntos de teste que se obtém a confirmação da capacidade de generalização de um algoritmo.

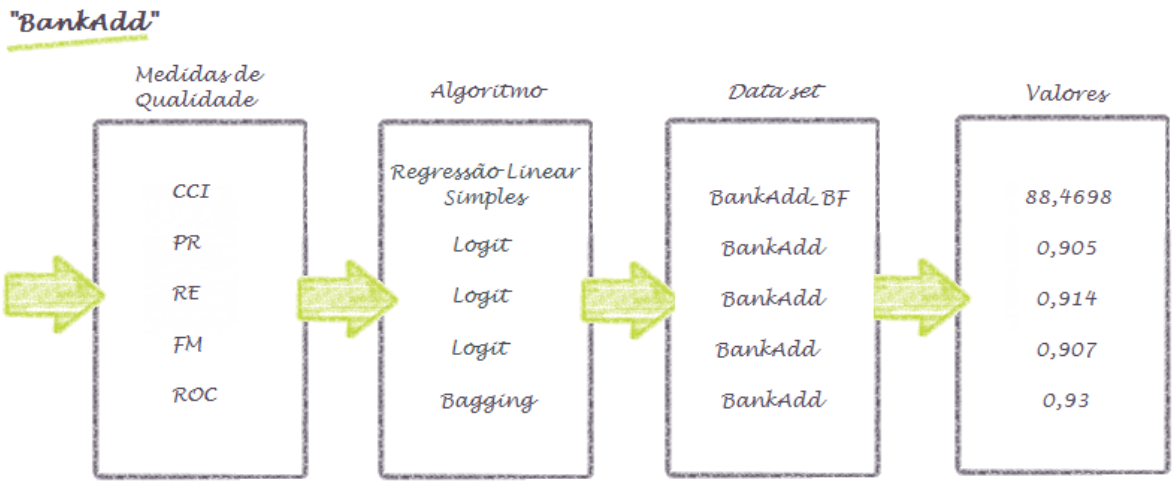

Figura 24 - Identificação dos melhores valores para cada uma das medidas de qualidade disponíveis, bem como a identificação do algoritmo responsável pelos melhores valores e ainda a identificação do data set base, isto é, sem seleção de atributos ("Bank") ou com seleção de atributos ("Bank_BF" ou "Bank_GS").

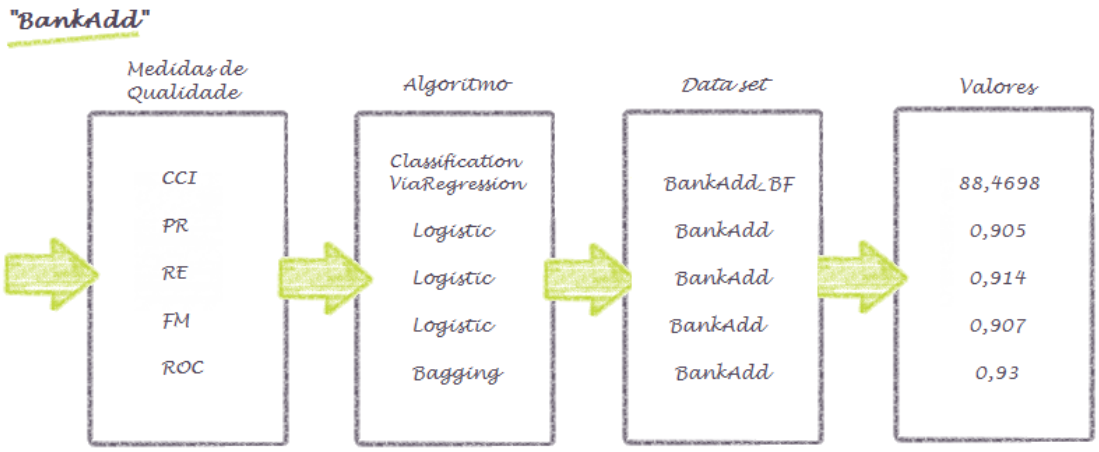

Figura 25 - Identificação dos melhores valores para cada uma das medidas de qualidade disponíveis, bem como a identificação do algoritmo responsável pelos melhores valores e ainda a identificação do data set base, isto é, sem seleção de atributos ("BankAdd") ou com seleção de atributos ("BankAdd_BF" ou "BankAdd_GS").

5.6.5. Análise comparativa de resultados

Esta secção pretende analisar os resultados obtidos neste caso de estudo com os resultados obtidos pelos autores do estudo sobre o qual este projeto se baseia (Sérgio Moro et al., 2011).

Os autores em questão utilizaram três algoritmos nas suas experiências: *Naive Baye, Decision Trees* e *Support Vector Machine*. Podem-se identificar três fases distintas destas experiências:

1. A primeira etapa incidiu sobre a identificação das variáveis a utilizar nas experiências, bem como no cruzamento com bases de dados internas, de forma a obter variáveis que melhor descrevessem os clientes. Posto isto, os três algoritmos foram aplicados à base de dados apurada mas apenas o algoritmo Naive Bayes conseguiu obter resultados, sendo que nos processos associados aos dois restantes algoritmos (DT e SVM) foi gerada falta de memória ou simplesmente não houve um término dos processos ao fim de determinado tempo;
2. A segunda fase resultou do insucesso da primeira. Surgiu a necessidade de simplificar a base de dados reduzindo as possibilidades para os valores da variável *target*. Assim foram selecionadas apenas as observações referentes a contactos conclusivos, ou seja, os contactos que foram possíveis classificar como "sucesso" ou "insucesso". Desta forma, além do algoritmo NB, também as Árvores de Decisão obtiveram resultados. Mais uma vez, o algoritmo SVM não conseguiu apurar qualquer resultado.
3. A terceira e última fase prendeu-se com a redução das variáveis constantes da base de dados. Algumas variáveis, tal como o género, foram consideradas irrelevantes, facto que dificulta a execução dos algoritmos. À semelhança do presente projeto, os autores dividiram a base de dados em 20 partições, sendo as mesmas compostas por um conjunto de treino (2/3 das observações) e um conjunto de teste (1/3 das observações). Desta forma foram executados novamente os três algoritmos, sendo finalmente possível obter resultados através do algoritmo SVM.

Para analisar a *performance* destes algoritmos, foram utilizadas duas medidas de qualidade: AUC (área por baixo da curva ROC) e ALIFT (área por baixo da curva LIFT). A Tabela 19 apresenta os resultados obtidos na terceira etapa das experiências.

Tabela 19 - Resultados obtidos pelos autores

	NB	DT	SVM
AUC (Area Under the ROC Curve)	0,87	0,868	0,938
ALIFT (Area Under the LIFT Curve)	0,827	0,79	0,887

Fazendo a comparação com os resultados obtidos neste caso prático, e analisando apenas a medida de qualidade comum, ROC, chega-se à conclusão de que as experiências efetuadas apresentaram resultados comparáveis com os obtidos pelos autores. O algoritmo SVM no estudo dos autores foi o que apresentou uma melhor *performance*, apresentado o valor de 0,938 para a medida ROC. No caso de estudo deste projeto, os melhores resultados recaíram sobre o algoritmo *Bagging*, sendo que, no *data set* "Bank" e no *data set* "BankAdd" foram obtidos valores de 0,893 e de 0,930, respetivamente.

Os resultados obtidos são comparáveis com os resultados dos autores, não sendo necessariamente piores. Há que ressalvar que os *data sets* utilizados não são exatamente os *data sets* utilizados pelos autores. É ainda necessário ter em conta que os resultados obtidos neste estudo resultam das médias derivadas das 30 partições não sendo possível identificar se os resultados publicados foram obtidos através da média das 20 partições.

Neste estudo, apesar de não se terem melhorado os resultados já publicados, foram testadas várias técnicas para além das testadas pelos autores. Adicionalmente, este caso de estudo analisa a qualidade dos algoritmos através de cinco medidas de qualidade distintas.

Foram ainda testadas duas formas distintas de seleção de atributos sobre os conjuntos de dados disponíveis. Os autores também aplicaram técnicas de redução de variáveis, tendo sido no entanto aplicadas numa fase anterior ao *data set* final que originou os subconjuntos para este caso de estudo. Ou seja, os *data set* utilizados neste projeto já se encontram redimensionados e talvez por isso a seleção de atributos aqui experienciada não tenha trazido melhoria de resultados.

Fazendo uma última comparação com os resultados obtidos pelos autores, e desta vez ao nível da relevância das variáveis, as conclusões, mais uma vez, são semelhantes.

Começando pela análise aos melhores resultados obtidos neste caso de estudo, o algoritmo *Bagging* identificou, no *data set* "Bank", como variáveis mais relevantes para a determinação da adesão do cliente ao depósito as variáveis que se encontram na Figura 26. A variável mais relevante consiste na duração da chamada. Quanto maior for a duração da chamada, maior será a tendência do cliente para aderir ao depósito a prazo. Em segundo lugar, surge o mês, indicando que os meses mais favoráveis à submissão do depósito são março, setembro, outubro e dezembro. Em terceiro lugar encontra-se o resultado da campanha anterior. Esta variável indica que a probabilidade de sucesso de uma campanha está diretamente relacionada com o resultado da campanha anterior. Assim, se um cliente aderiu ao depósito a prazo na campanha anterior, é provável que venha a aderir ao depósito em vigor na campanha corrente.

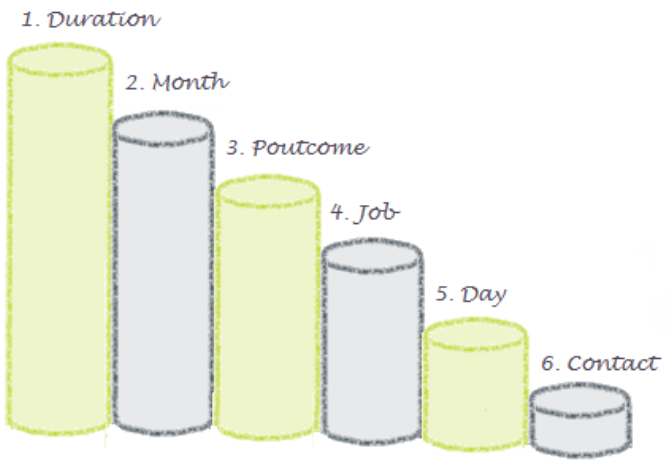

Figura 26 - Ranking das variáveis com maior relevância no data set "Bank" aquando da aplicação do algoritmo que gerou os melhores resultados, o algoritmo Bagging.

Analisando os resultados obtidos pelo algoritmo *Bagging* no *data set* "BankAdd", surgem algumas variáveis em comum, como é o caso da duração da chamada, o emprego e o mês em que a chamada foi realizada. Relativamente à duração da chamada, as conclusões são as mesmas, quanto mais longa for uma chamada mais provável será o cliente aderir ao produto em causa. No que diz respeito à média trimestral do número de cidadãos empregados, quanto menor for o número de indivíduos empregados, maior será a probabilidade de um cliente aderir ao produto. No que diz respeito à

profissão do cliente, clientes desempregados, reformados ou estudantes são mais propensos a aderir ao depósito a prazo.

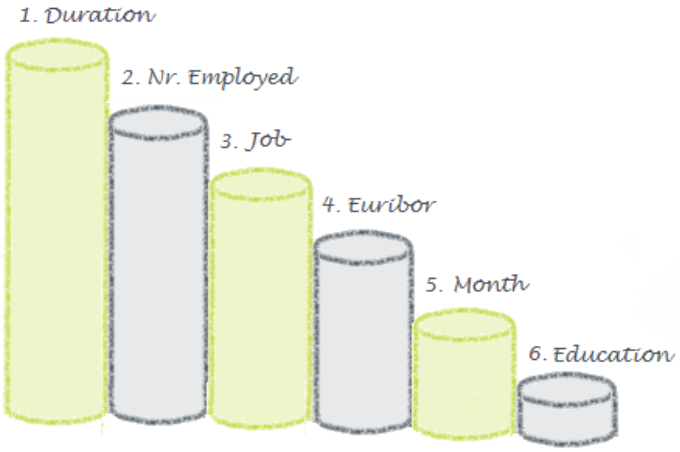

Figura 27 - Ranking das variáveis com maior relevância no data set "BankAdd" aquando da aplicação do algoritmo que gerou os melhores resultados, o algoritmo Bagging.

Os autores em estudo concluem, através dos resultados obtidos com o algoritmo SVM, como variáveis mais relevantes a duração da chamada, o mês em que a chamada é realizada e o número de contactos realizados na campanha anterior para um cliente. Seguem-se outras variáveis também identificada com relevância, sendo elas o número de dias que passaram desde o último contacto, o último contacto realizado e o primeiro contacto realizado.

6. CONCLUSÕES

Inerente a este trabalho encontra-se uma base de dados real, com origem num *call center* associado a um banco português, cuja existência se deve à venda de depósitos a prazo com juros favoráveis.

O caso prático subjacente debruça-se, numa primeira fase, sobre a aplicação de catorze algoritmos de aprendizagem supervisionada sobre a base de dados referida e, numa segunda fase, na aplicação dos mesmos algoritmos sobre uma nova base de dados resultante do redimensionamento da base de dados inicial, através da seleção de atributos.

Pretende-se, com este caso prático, comparar os resultados com os atualmente publicados pelos autores Sérgio Moro, Raul Laureano e Paulo Cortez (Sérgio Moro et al., 2011) e ainda identificar as variáveis que mais discriminam um potencial cliente do produto em questão.

Após a realização das várias experiências propostas e embora os resultados obtidos não sejam superiores aos dos autores Sérgio Moro, Raul Laureano e Paulo Cortez, conclui-se que os algoritmos utilizados estão em linha com os resultados apurados pelos mesmos.

Começando pelos resultados obtidos através dos algoritmos sem qualquer seleção de atributos, existem diferenças ao nível do algoritmo com melhor *performance*. Enquanto no estudo em comparação o algoritmo que apurou resultados superiores foi o SVM, no presente estudo e com ambas as bases de dados, o melhor algoritmo, com resultados consistentes, foi o *Bagging*.

A seleção de atributos, que visa potenciar os resultados obtidos, ficou aquém dos resultados anteriores à mesma, talvez derivado do facto das bases de dados disponíveis terem sido anteriormente sujeitas à remoção das variáveis menos relevantes. É de destacar, no entanto, que o número de observações corretamente classificadas mostrou-se superior nas experiências com seleção de atributos. Ainda tendo em conta a redução da base de dados, destaca-se o apuramento de melhores valores na técnica *Best First*, quando comparada com a técnica *Genetic Search*.

Muito embora as bases de dados disponíveis não sejam as utilizadas pelos autores mencionados, sendo apenas amostras das mesmas, os resultados obtidos apresentam semelhanças ao nível das variáveis mais discriminantes, sendo quase todas elas coincidentes. Torna-se assim oportuno e relevante mencionar que apesar de neste projeto não se ter apostado na parametrização dos algoritmos, a mesma, definida pelo WEKA, não invalidou os resultados obtidos, tal como se pode verificar pelas variáveis classificadas como mais discriminantes. É de acrescentar ainda a eficácia e simplicidade do WEKA aquando da utilização e implementação dos algoritmos e da disponibilização dos resultados.

Importa ainda salientar a semelhança de resultados obtidos ao nível das várias medidas de qualidade dos algoritmos. Em geral, os algoritmos são consistentes na qualidade apresentada nas várias medidas.

Dado que foram experienciados vários tipos de algoritmos, é relevante destacar que os algoritmos pertencentes aos *Ensembles* produziram melhores classificações.

Por fim, é importante destacar que as diferenças ao nível das bases de dados e da parametrização dos algoritmos, não qualifica as experiências como diretamente comparáveis. Assim, apesar dos resultados

deste estudo serem inferiores aos resultados do estudo em comparação, não significa que os primeiros resultados sejam, necessariamente, piores do que os segundos.

6.1. LIMITAÇÕES E RECOMENDAÇÕES PARA TRABALHOS FUTUROS

O presente trabalho respondeu às questões centrais levantando, no entanto, alguns pontos que podem ser aprofundados.

O primeiro ponto incide sobre a parametrização dos algoritmos. Neste projeto, tal como referido, foram utilizadas as definições inerentes aos algoritmos disponibilizados pelo WEKA. No entanto, e apesar de não serem considerados descabidos os resultados obtidos, não se exclui a hipótese de que, com outro tipo de afinamento, os algoritmos apresentados poderiam apresentar melhores resultados.

O segundo ponto recai sobre a aplicação da seleção de atributos. Existem várias formas de redimensionamento da base de dados, sendo que, algumas delas poderão melhorar os resultados obtidos sem qualquer seleção de atributos contrariando assim as conclusões obtidas neste projeto.

7. BIBLIOGRAFIA

Abdelhalim, a., & Traore, I. (2009). A New Method for Learning Decision Trees from Rules. *2009 International Conference on Machine Learning and Applications*, (Mvd). doi:10.1109/ICMLA.2009.25

Allen, F., & Carletti, E. (2013). Deposits and bank capital structure. Retrieved from http://cadmus.eui.eu/handle/1814/26454

Alpaydin, E. (2004). *Introduction to Machine Learning*. Massachusetts: Massachusetts Institute of Technology.

Angelis, F. De, Polzonetti, a, & Re, B. (n.d.). Optimising Performance With Business Intelligence.

Augusty, S. M., & Izudheen, S. (2013). A Survey: Evaluation of Ensemble Classifiers and Data Level Methods to Deal with Imbalanced Data Problem in Protein - Protein Interactions. *Review of Bioinformatics and Biometrics*, 1–9.

Auria, L., & Moro, R. A. . (2008). *Support Vector Machines (SVM) as a Technique for Solvency Analysis*. Berlim.

Basheer, I. a., & Hajmeer, M. (2000). Artificial neural networks: Fundamentals, computing, design, and application. *Journal of Microbiological Methods*, *43*(1), 3–31. doi:10.1016/S0167-7012(00)00201-3

Ben-Gal, I. . (2007). Bayesian Networks. In W. & Sons (Ed.), *Encyclopedia of Statistics in Quality & Reliability*.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145–1159. doi:10.1016/S0031-3203(96)00142-2

Bratko, I., & Konenko, I. (1986). Learning diagnostic rules from incomplete and noisy data. In *Seminar on AI methods in statistics*. London Business School, England: Unicom Seminars Ltd.

Breiman, L. (1996). Bagging Predictor. *Machine Learning*, 123–140.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*. California: Wadsworth International Group.

Cherkauer, K. J. ., & Shavlik, J. W. . (1996). Growing simpler decision trees to facilitate knowledge discovery. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining.*

Cilimkovic, M. (2013). *Neural Networks and Back Propagation Algorithm*. Dublin, Ireland.

Cui, G., Wong, M. L., & Lui, H.-K. (2006). Machine Learning for Direct Marketing Response Models: Bayesian Networks with Evolutionary Programming. *Management Science*, 597–612.

Daumé, H. (2012). A Course in Machine Learning. Retrieved February 3, 2015, from http://www.ciml.info/dl/v0_8/ciml-v0_8-ch03.pdf

Dean, J. (2014). *Big Data, Data Mining and Machine Learning. Value Creation for Business Leaders and Practitioners*. New Jersey: Wiley.

Du, W., & Zhan, Z. (2002). Building decision tree classifier on private data. *Proceedings of the IEEE International Conference on Privacy, Security and Data Mining-Volume 14*, 1–8. Retrieved from http://portal.acm.org/citation.cfm?id=850784

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, *17*(3), 37–53. doi:10.1609/aimag.v17i3.1230

Finlay, S. (2014). *Predictive Analytics, Data Mining, and Big Data. Miths, misconceptions and methods*. Chennai, India: Palgrave Macmillan.

Freund, Y., & Schapire, R. E. (1999). Large Margin Classification Using the Perceptron Algorithm. *Machine Learning*, 277–296.

Friedman, H. J. (1997). Data mining and statistics: What's the connection? *Department of Statistics and Standford Linear Accelerator Center, Standford University*, 1–7.

Gennari, J. H. ., Langley, P. ., & Fisher, D. (1989). Models of incremental concept formation. *Artificial Intelligence*, 11–61.

Gupta, R. (2008). Machine Learning Lecture 3. Retrieved January 27, 2015, from http://www.slideshare.net/cnu/machine-learning-lecture-3

Hall, M. A. (1999). *Correlation-based Feature Selection for Machine Learning*.

Hall, M., Frank, E., & Holmes, G. (2009). *The WEKA Data Mining Sotware: An Update*. SIGKDD Explorations.

Hamilton, H. J., Gurak, E., Findlater, L., Olive, W., & Ranson, J. (2012). Computer Science 831: Knowledge Discovery in Databases. Retrieved from http://www2.cs.uregina.ca/~hamilton/courses/831/notes/ml/dtrees/4_dtrees1.html

Hand, D., Mannila, H., & Smyth, P. (2001). *Principles of Data Mining*. Cambridge, Massachussetts, London: The MIT Press.

Hashemi, S., & Yang, Y. (2009). *Flexible decision tree for data stream classification in the presence of concept change, noise and missing values*. Springer.

Haykin, S. (1999). Neural Networks-A Comprehensive Foundation.

Hsu, M. J., & Ho, C. P. (2012). Creating a knowledge discovery model using MOEX's examination database for in-depth analysis and reporting. In *Proceedings - 2012 IEEE Symposium on Robotics and Applications, ISRA 2012* (pp. 705–707). Kuala Lumpur. doi:10.1109/ISRA.2012.6219288

Jadhav, R., & Pawar, U. (2011). Churn Prediction in Telecommunication Using Data Mining Technology. *International Journal*, *2*(2), 17–19. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.190.5029&rep=rep1&type=pdf#page=30

Jensen, F. V. . (2001). *Bayesian Networks and Decision Graphs*. (Springer, Ed.). New York.

Jianjun, W., Chaojun, R., Qianqian, H., & Ping, L. (2010). Gini coefficient used in the post-evaluation theory of highway construction. *2010 International Conference on Intelligent Computation Technology and Automation, ICICTA 2010*, *2*, 395–398. doi:10.1109/ICICTA.2010.725

Karegowda, A. G., Jayaram, M. A. ., & Manjunath, A. S. . (2011). Feature Subset Selection using Cascaded GA & CFS: A Filter Approach in Supervised Learning. *International Journal of Computer Applications*, 975–8887.

Koblar, V. (2012). *Optimizing Parameters of machine learning algorithms*. Liubliana.

Koh, H. C., Tan, W. C., & Goh, C. P. (2006). A two-step method to construct credit scoring models with data mining techniques. *International Journal of Business and Information*, *1*(1), 96–118. Retrieved from http://www.knowledgetaiwan.org/ojs/files/Vol1No1/Paper_5.pdf

Kohavi, R. (1995). *Wrappers for Performance Enhancement and Oblivious Decision Graphs*.

Kohavic, R., & John, G. . (1996). Wrappers for feature subset selection. In *Artificial Intelligence, special issue on relevance* (pp. 97(1–2):273–324).

Kononenko, I., & Matjaz, K. (2007). *MACHINE LEARNING AND DATA MINING: Introduction to Principles and Algorithms*. *Communications of the ACM*. West Sussex, UK: Horwood Publishing Limited. Retrieved from http://dl.acm.org/citation.cfm?id=319388

Kotler, P., & Armstrong, G. (2012). *Principles of Marketing*. New Jersey: Prentice Hall.

Kumar, G. R., Kongara, V. S., & Ramachandra, G. A. (2013). An efficient ensemble based classification techniques for medical diagnosis. *International Journal of Latest Technology in Engeneering, Management & Applied Science*, 5–9.

Lavalle, S., Hopkins, M. S., Lesser, E., Shockley, R., & Kruschwitz, N. (2010). Analytics : The New Path to Value. *MIT Sloan Management Review*, 1–24. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Analytics+:+The+New+Path+to+Value#0

Lee, W., & Stolfo, S. J. (1998). Data Mining Approaches for Intrusion Detection Data Mining Approaches for Intrusion Detection.

Lemos, E. P., Steiner, M. T. A., & Nievola, J. C. (2005). *Análise de crédito bancário por meio de redes neurais e árvores de decisão: uma aplicação simples de data mining*. São Paulo.

Ling, C., & Li, C. (1998). Data Mining for Direct Marketing: Problems and Solutions. *Kdd*, 73–79. doi:10.1.1.332.1803

Lopes, A. (2009). *Estimação OLS do Modelo de Regressão Linear com Séries Temporais*. Lisboa.

Lorena, A. C., & Carvalho, A. C. P. L. F. de; (2003). *Introdução às Máquinas de Vetores de Suporte (Support Vector Machines)*. São Carlos.

Maimon, O., & Rokack, L. (2005). *The Data Mining and Knowledge Discovery Handbook*. Israel: Springer.

Matheus, C. J., Chan, P. K., & Piatetsky-Shapiro, G. (1993). Systems for knowledge discovery in databases. *IEEE Transactions on Knowledge and Data Engineering*, *5*(6), 903–913. doi:10.1109/69.250073

Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (1986). *Machine Learning. An Artificial Intelligence Approach*. California: Morgan Kaufmann Publishers, Inc.

Mingers, J. (1987). Expert systems - rule induction with statistical data. *Journal of the Operational Research Society*, *38*, 39–47.

Mitchell, T. M. (1997). *Machine Learning*. McGrawHill.

Moro, S., Laureano, R. M. S., & Cortez, P. (2011). Using Data Mining for Bank Direct Marketing: An application of the CRISP-DM methodology. *European Simulation and Modelling Conference*, 117–121.

Murthy, S. K. (1998). Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey. *(A2) Data Mining and Knowledge Discovery*, *2*, 345–389. doi:10.1023/A:1009744630224

Neville, P. G. (1999). *Decision Trees for Predictive Modeling*.

Nisbet, R., Elder, J., & Miner, G. (2009). *Handbook of Statistical Analysis and Data Mining Applications*. California: Elsevier Inc.

O'Rourke, N., & Hatcher, L. (2013). *A Step-by-Step Approach to Using SAS for Factor Analysis and Structural Equation Modeling* (Second Edi.). Cary, North Carolina, USA: SAS Institute Inc.

Opitz, D., & Maclin, R. (1999). Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, 169–198.

Pujari, A. K. (2001). *Data Mining Techniques*. Hyderabad, India: Universities Press (India) Private Limited.

Queensland Government. (2014). Telemarketing. Retrieved May 22, 2014, from https://www.business.qld.gov.au/business/running/marketing/direct-marketing/telemarketing

Quinlan, J. R. (1987). Simplifying decision trees. *International Journal of Man-Machine Studies*, *27*, 221–234.

Quinlan, J. R., & Rivest, R. L. (1989). Inferring Decision Description Trees Using the Minimum Length Principle. *Information and Computation*, *80*(1989), 227–248. doi:http://dx.doi.org/10.1016/0890-5401(89)90010-2

Rastogi, R., & Shim, K. (2000). A decision tree classifier that integrates building and pruning. *Proceedings on Internation Conference on Very Large Databases*, *768*, 1–28. doi:10.1023/A:1009887311454

Rich, E., & Knight, K. (1991). *Artificial Intelligence*. McGrawHill.

Rojas, R. (1996). *Neural Network*. Berlim.

Russel, S. J. ., & Norving, P. (1995). *Artificial Intelligence. A Modern Approach.* New Jersey.

SAS. (2001). Dexia Bank uses data mining to create a model for cross-selling and upselling. Retrieved December 4, 2014, from http://www.sas.com/offices/europe/belux/pdf/success/Dexia_uses_SAS.pdf

Shalizi, C. R. (2012). *Advanced Data Analysis from an Elementary Point of View*. Pittsburgh.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*(July 1928), 379–423. doi:10.1145/584091.584093

Silltow, J. (2006). Data Mining 101: Tools and Techniques. Retrieved August 1, 2006, from https://iaonline.theiia.org/data-mining-101-tools-and-techniques

Sivagama, S. (2011). A Knowledge Discovery Using Decision Tree By Gini Coefficient. *Engineering*, 232–235.

Smola, A., & Vishwanathan, S. V. N. . (2008). *Introduction to Machine Learnin*. Cambridge.

Stimpson, A. J., & Cummings, M. L. (2014). *Assessing Intervention Timing in Computer-Based Education Using Machine Learning Algorithms*.

Tsiptsis, K., & Chorianopoulos, A. (2009). *Data Mining Techniques in CRM: Inside Customer Segmentation*. Atenas: Wiley.

Vafaie, H. ., & De Jong, K. (1995). Genetic algorithms as a tool for restructuring feature space representations. In *Proceedings of the International Conference on Tools with A.I.* IEEE Computer Society Press.

Ville, B. (2006). Decision Trees - What Are They? In *Decision Trees for Business Intelligence and Data Mining: Using SAS Enterprise Miner*. SAS Institute Inc.

Wang, X. W. X. (2009). Intelligent Quality Management Using Knowledge Discovery in Databases. *2009 International Conference on Computational Intelligence and Software Engineering*, 1–4. doi:10.1109/CISE.2009.5364999

Williams, G. (2010). Data Mining - Desktop Survival Guide. Retrieved February 28, 2015, from http://datamining.togaware.com/survivor/Alternating_Decision.html

Winandy, C.-E., Borges Filho, E., & Bento, L. V. (2007). Algoritmos para Aprendizagem Supervisionada.

Witten, I., & Frank, E. (2005). *Data Mining Pratical Machine Learning Tools and Techniques*. (G. W. Inc., Ed.). San Francisco.

Witten, I. H., Frank, E., & Hall, M. a. (2011). *Data Mining: Practical Machine Learning Tools and Techniques, Third Edition*. doi:10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C

Wooldridge, J. M. (2013). *Introductory Econometrics. A modern approach.* (S.-W. C. Learning, Ed.). Michigan.

Zhang, K., Fan, W., Yuan, X., Davidson, I., & Li, X. (2006). Forecasting Skewed Biased Stochastic Ozone Days: analysis and solutions. In IEEE (Ed.), *International Conference on Data Mining* (pp. 753–764). Hong Kong.