POLITÉCNICO DE LEIRIA
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA

RELATÓRIO DA UNIDADE CURRICULAR DE OTIMIZAÇÃO E APRENDIZAGEM AUTOMÁTICA DE MESTRADO EM ENGENHARIA ELETROTÉCNICA

2º Trabalho Prático

Autores: Marco Gameiro Nº: 2213276, Samuel Heleno Nº: 2212417

Docente: João Sousa

Leiria, 8 de fevereiro de 2022

Esta página foi intencionalmente deixada em branco.

Índice

Lista de Figuras

Esta página foi intencionalmente deixada em branco.

Lista de Tabelas

Esta página foi intencionalmente deixada em branco.

1 Introdução

O presente relatório refere-se a um trabalho, no âmbito da unidade curricular de Otimização e Aprendizagem Automática (módulo de Aprendizagem Automática), lecionada no Mestrado em Engenharia Eletrotécnica. Primeiramente considerou-se um problema de aprendizagem não-supervisionada, e de seguida, um de aprendizagem supervisionada. Em ambos procedeu-se inicialmente à análise do *dataset* utilizado, bem como ao respetivo tratamento de dados. Relativamente à aprendizegm não-supervisionada exploraram-se diferentes métodos de *clustering* (hierárquico, K-Mean e Fuzzy C-Mean), tendo-se analisado os resultados obtidos. No problema de aprendizagem supervisionada, recorreram-se a diferentes algoritmos (k-Nearest Neighbours, Artificial Neural Networks e Support Vector Machines), por forma a treinar os vários *subsets* de dados, tendo-se igualmente analisado os resultados obtidos.

Esta página foi intencionalmente deixada em branco.

2 Aprendizagem Não-Supervisionada

Uma das principais metodologias da aprendizagem automática é: a aprendizagem não-supervisionada. Na aprendizagem não-supervisionada são utilizados algoritmos de aprendizagem sem qualquer supervisão que analisam a estrutura interna dos dados em questão, sem qualquer dado de saída/etiqueta. A aplicação mais recorrente denomina-se de método de *clustering* e consiste em detetar grupos de amostras baseados em similaridades, onde amostras com padrões semelhantes são agrupadas num determinado *cluster* e amostras com padrões consideravelmente distintos são esperadas em diferentes *clusters*.

2.1 Análise do *dataset* utilizado

O *dataset* utilizado contém informação acerca da subscrição dos clientes a serviços de empresas de telecomunicações. Este *dataset* é composto por três *features*, sendo estas: *tenure*, *monthly charges* e *total charges*. A *feature tenure* diz respeito ao número de meses de permanência do cliente com o atual operador de telecomunicações. A *feature monthly charges* traduz o valor mensal cobrado ao consumidor, do serviço subscrito. A *feature total charges* representa o valor total já cobrado ao consumidor, pela empresa de telecomunicações.

Foi realizada uma análise ao *dataset* utilizado com recurso ao *software* Jupiter, sendo que este gera um relatório completo do dados disponíveis. Numa visão global, verifica-se que o *dataset* é composto por quatro variáveis numéricas, sendo estas as três *features* e o *consumer ID* (permite distinguir os vários clientes), onde o *consumer ID* e a *feature tenure* são variáveis numéricas discretas e as duas *features* restantes são variáveis numéricas contínuas. O *dataset* é composto por cem observações/amostras, sendo que existe falta de informação em 0,5% das células (2 células), sendo que uma célula corresponde à *feature monthly charges* e a outra corresponde à *feature total charges*. Numa análise individualizada das várias *features* verifica-se que para a *feature tenure* existem 48 valores distintos, sendo o valor médio de 31,21 meses e os valores máximo e mínimo de 72 e de 1,

respetivamente. Afere-se que, relativamente à *feature monthly charges*, existem 92 valores distintos, sendo o valor médio de 60,3€ e os valores máximo e mínimo de 112,25€ e de 19,3€, respetivamente. Para a *feature total charges*, existem 99 valores distintos, sendo o valor médio de 2032,23€ e os valores máximo e mínimo de 8129,3€ e de 19,45€, respetivamente.

Realizou-se uma análise de correlação entre as várias variáveis que compõem o *dataset*, representada na figura 2.1, com recurso ao coeficiente de Spearman's. Verifica-se que existe uma correlação entre as *features tenure* e *total charges* e entre as *features monthly charges* e *total charges*

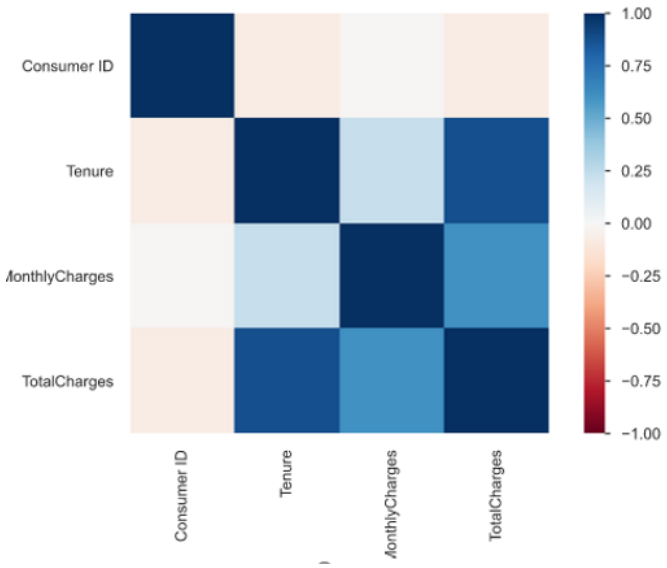


Figura 2.1: Resultado da correlação entre as várias variáveis

2.2 Tratamento de dados

Uma vez que o sistema aprende a partir de um conjunto de dados disponível, torna-se necessário garantir a qualidade desses dados, como tal procedeu-se ao tratamento destes.

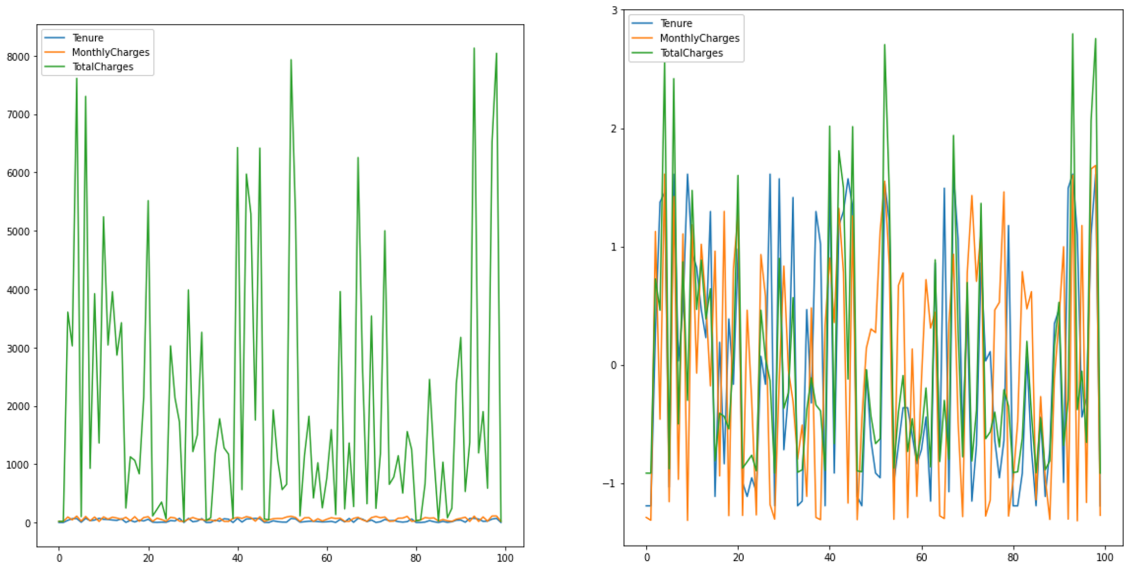
Inicialmente foi descartada a variável *consumer ID*, pois esta não apresenta informação relevante para o sistema de aprendizagem.

De seguida, procedeu-se à deteção da possível existência de campos (células) vazios e ao seu devido preenchimento. De forma a preencher os campos vazios com valores fidedignos e garantir a qualidade dos dados para "alimentar"o sistema, caso exista falta de informação numa célula da *feature ternure* esta será preenchida com o valor resultante da divisão entre o valor da célula da *feature total charges* com o valor da *feature monthly charges*, da respetiva amostra. No caso em que o valor em falta diz respeito à *feature monthly charges*, este campo é preenchido com o valor resultante da divisão entre o valor da célula da *feature total charges* com o valor da *feature tenure*, da respetiva amostra. Para

a falta de informação no campo respetivo da *feature tatal charges*, este será preenchida pelo valor resultante da multiplicação entre o valor da célula da *feature tenure* com o valor da *feature monthly charges*, da respetiva amostra.

Procedeu-se à normalização *standard* dos dados, pois as *features* que "alimentam"o sistema de aprendizagem não-supervisionada apresentam escalas consideravelmente distintas, como se pode verificar no sub-capitulo 2.1 e na figura 2.2(a). Os fatores de normalização estão relacioados com o valor médio e o desvio-padrão dos dados, sendo que a normalização se traduz pela equação 2.1. Na figura 2.2(b) encontram-se representadas as *features* pós-normalização, sendo notório o efeito da normalização.

$$x_{inormalizado} = \frac{x_i - média(X)}{desvio - padrão(X)} \tag{2.1}$$



(a) Pré-normalização (b) Pós-normalização

Figura 2.2: Representação das *features* do *dataset*

Posteriormente, procedeu-se à divisão dos dados do *dataset* em dois *subsets*: *subset* de treino e *subset* de teste. Os dados do *subset* de treino são os dados, efetivamente, utilizados durante o processo de treino do sistema de aprendizagem e correspondem a 70% do dados do *dataset*. Os dados do *subset* de teste são utilizados para avaliar a precisão do sistema, correspondendo aos restantes 30% dos dados do *dataset*. Na figura 2.3 encontra-se a representação gráfica das amostras correspondentes ao *subset* de treino.

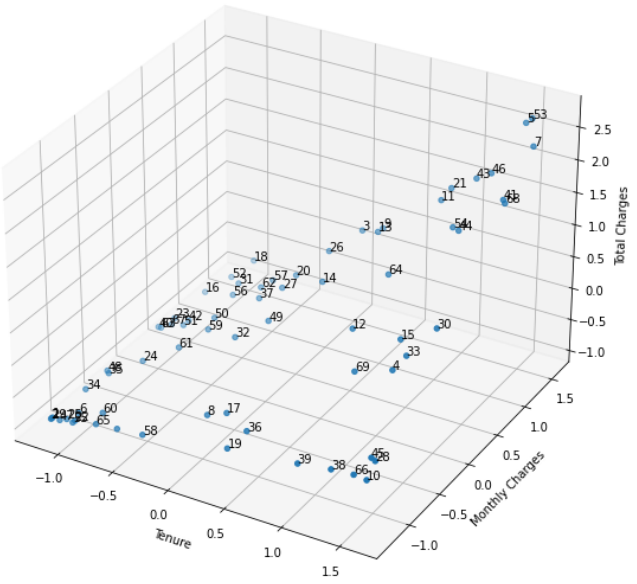


Figura 2.3: Representação gráfica das amostras do *subset* de treino

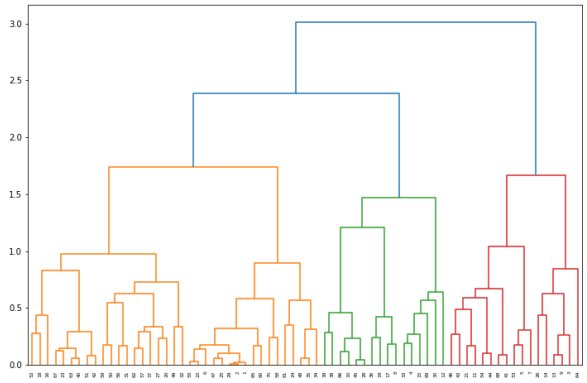
2.3 Clustering Hierárquico

O método de *clustering* hierárquico pode ser realizado através de duas abordagens distintas: aglomerativa e a divisiva. A abordagem implementada foi a aglomerativa e esta consiste em agrupar os objetos em sucessivos *clusters* formados de acordo com as distâncias entre os objetos, usando a abordagem ”de cima para baixo”, com inicio num número de *clusters* igual ao número de objetos e término num único *cluster*.

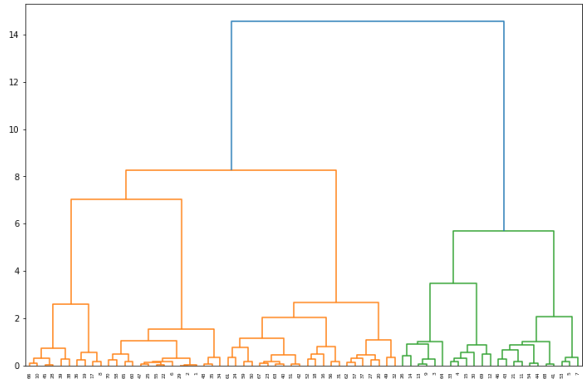
O cálculo das distâncias entre objetos pode ser realizado com recurso a várias métricas de distâncias, sendo que neste caso foram utilizadas duas métricas: a distância Euclidiana e a distância de Manhattan (Cityblock). A distância Euclidiana é obtida pela raiz quadrada dos quadrados das diferenças para cada coordenada e a distância de Manhattan é obtida através da soma das diferenças absolutas para cada coordenada. Tendo as distâncias entre os diferentes objetos calculadas, pode-se proceder ao agrupamento dos vários objetos (*clustering*), sendo que neste processo é necessário fazer uma continua medição das distâncias entre os objetos e *clusters* e entre diferentes *clusters*. Estes cálculos podem ser realizados através de diferentes métodos, sendo que neste caso serão utilizados os métodos: ligação média e ligação Ward. A ligação média é baseada na distância média entre pares de objetos e a ligação Ward é baseada na distância entre centróides ponderada pelos tamanhos dos *clusters*.

De forma a visualizar a forma de como os grupos são formados são utilizadas as representações gráficas denominadas dendrogramas, onde o eixo horizontal identifica os vários objetos do *dataset* e o

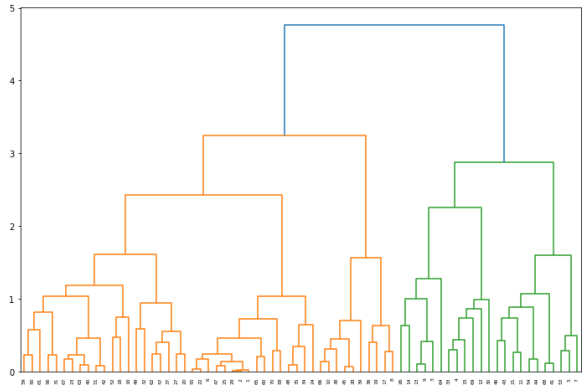
eixo vertical indica a distância entre objetos ou *clusters*. Os resultados obtidos encontram-se na figura 2.4, sendo que se conclui que o *clustering* hierárquico para este *dataset* obteve melhores resultados utilizando a métrica de distância Manhattan e o método ligação média, pois é o que apresenta uma menor distância entre *clusters*.



(a) Distância Euclideana e ligação média



(b) Distância Euclideana e ligação Ward



(c) Distância Manhattan e ligação média

Figura 2.4: Representação dos dendrogramas

De forma a analisar a alocação de objetos pelos *clusters*, definiu-se que existiam quatro *clusters* e procedeu-se ao *clustering*. Na figura 2.5 é possível verificar a distribuição dos objetos do *dataset* pelos vários clusters. De seguida calcularam-se os centróides dos vários *clusters*, sendo que na figura

2.6 encontram-se representados os objetos do *dataset* e os centróides dos quatro *clusters* definidos, para o caso em que o cálculo das distâncias se procedeu através da distância de Manhattan e o método utilizado foi a ligação média. É possível comprovar que os resultados representados nas figuras 2.5 e 2.6 se relacionam, sendo que é notório que o *cluster* alusivo à ”Centroide 4”é o que aloja uma maior quantidade de objetos.

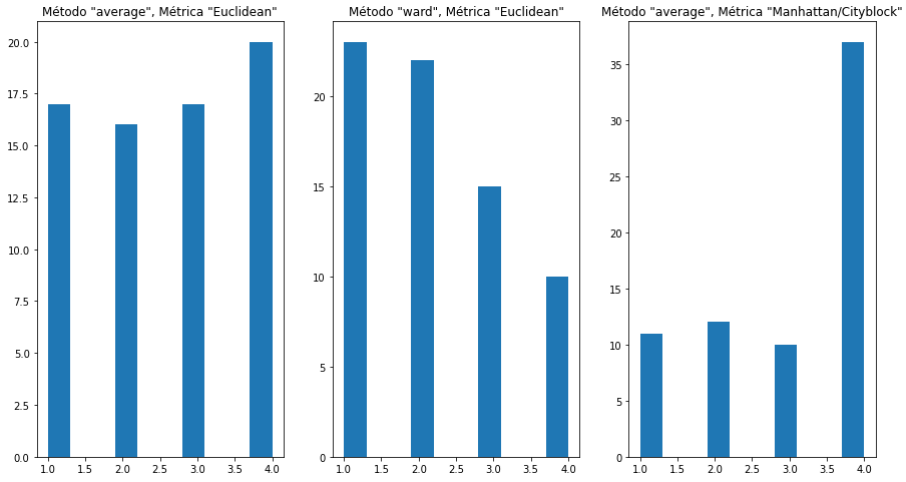


Figura 2.5: Distribuição dos objetos pelos quatro *clusters*

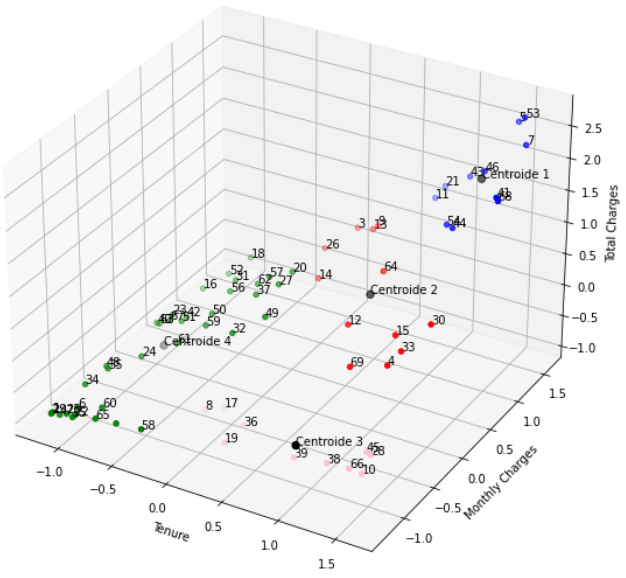


Figura 2.6: Representação dos quatro *clusters* definidos, utilizando a métrica de distância de Manhattan e o método ligação média

Caso novas observações sejam apresentadas, como por exemplo as observações do *subset* de teste, o procedimento a realizar seria o de calcular a distância entre um novo objeto com os centróides dos clusters e atribuir o objeto ao cluster mais perto. Recalcular as novas coordenadas dos centroides e repetir o processo para o resto das novas observações.

2.4 Clustering K-Mean

O método de *clustering* K-Mean consiste num processo de aperfeiçoamento iterativo onde previamente é definido o número de *clusters* desejado. O processo iterativo tem origem numa inicialização aleatória dos centróides dos *clusters* e cada objeto do *dataset* é atribuído ao *cluster* correspondente ao centroide mais próximo, sendo que a distância medida baseia-se na métrica distância Euclidiana. Para cada novo objeto atribuído a um *cluster* os centroides são recalculados. O algoritmo itera de acordo com o procedimento descrito, procurando minimizar a função objetivo: minimizar a distância *intra-cluster*. Como os centroides inicialmente são gerados aleatoriamente e o número de iterações é limitado, este método não garante que o ótimo global seja atingido, por isso é recomendável que se proceda a várias inicializações aleatórias dos centróides. Neste caso foi definido que o algoritmo executa 300 iterações e procede a 5 inicializações aleatórias dos centróides.

Como referido, inicialmente é necessário definir o número de *clusters* e, como tal, foi utilizado o método denominado ”cotovelo”, servindo como ferramenta de apoio na decisão do número de *clusters* a adotar. O ”cotovelo”traduz a relação/compromisso entre a menor variância da distância *intra-cluster* e o número final de *clusters*, onde, tipicamente, o número ótimo de *clusters* se encontra no cotovelo da curva do gráfico. Na figura 2.7 encontra-se representado graficamente o resultado do método ”cotovelo”e verifica-se que o cotovelo da curva corresponde a um intervalo de 3 a 15 *clusters*.

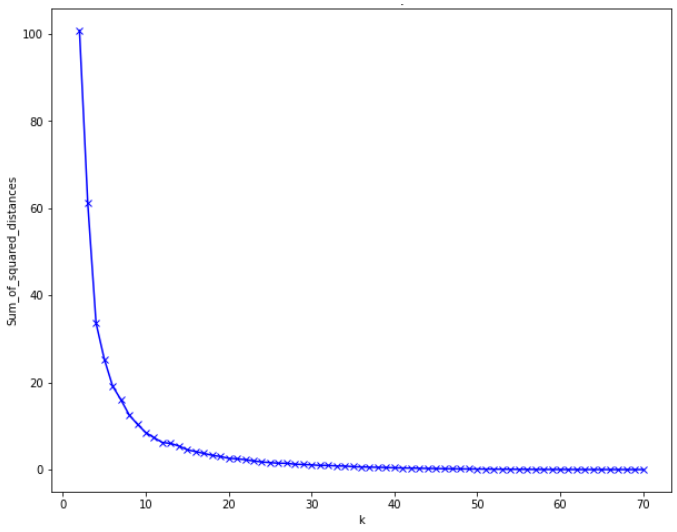


Figura 2.7: Representação gráfica do método ”cotovelo”

Foi utilizada outra ferramenta de apoio na decisão do número de *clusters* a adotar, sendo esta a análise Silhouette. Esta análise permite avaliar se, no processo de clustering, é garantida a consistência na forma como se agrupam os dados, para um determinado número de *clusters*. Procedeu-se à análise de

Silhouette para todas as quantidades possíveis de *clusters* e aferiu-se, através do valor de Silhouette, se cada objeto é similar ao seu próprio *cluster* - coesão -, sendo, também, confrontado com ou outros *clusters* - separação. Verificou-se que os valores médios de Silhouette mais elevados correspondem a 4 e a 7 *clusters*, sendo os valores de 0.525 e 0.521, respetivamente.

Deste modo, procedeu-se ao clustering K-Mean para 4 e 7 clusters, sendo que os clusters resultantes estão representados nas figuras 2.8 e 2.9, respetivamente.

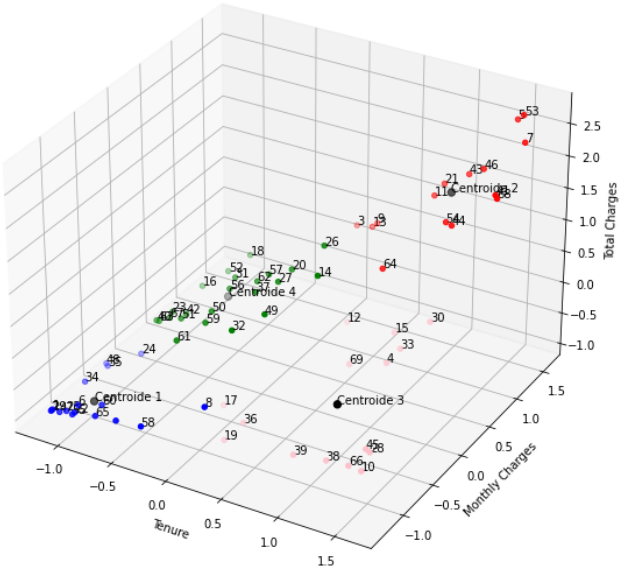


Figura 2.8: *Clustering* K-Mean - 4 *clusters*

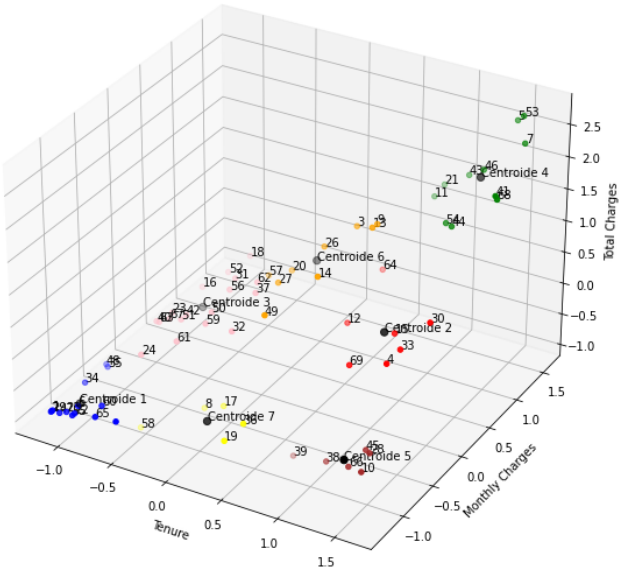


Figura 2.9: *Clustering* K-Mean - 7 *clusters*

2.5 Clustering Fuzzy C-Mean

O método Fuzzy C-Means é similar ao K-Means, sendo que, neste caso, cada objeto pode pertencer a vários *clusters* diferentes, mas em diferentes proporções (grau de pertença). A função utilizada para a implementação deste método, considera que as *features* se encontram dispostas horizontalmente, enquanto que a matriz de dados de treino obtida, contém as *features* dispostas verticalmente. Assim, foi necessário aplicar a transposta à matriz de dados de treino obtida. Na aplicação deste método considerou-se um erro de 0.5 %, e um máximo de 300 iterações. Definiu-se, ainda, o parâmetro *m* como 2 e o parâmetro *init* como *None*, o que significa que o algoritmo é inicializado de forma aleatória.

Na figura 2.10 encontra-se representada a relação existente entre o coeficiente de partição difusa (*FPCS*) e o número de *clusters*, *k*. Tal como era já expectável, há medida que se aumenta o número de *clusters* diminui o grau de pertença a cada um.

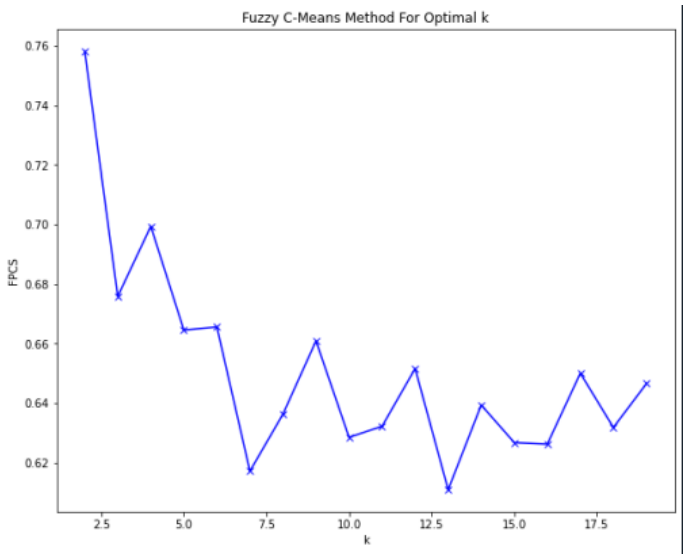


Figura 2.10: Resultado da aplicação do método Fuzzy C-Means

De seguida, procedeu-se ao clustering *Fuzzy* C-Mean para 4 e 7 *clusters*. Os centróides correspondentes aos *clusters* resultantes encontram-se representados nas figuras 2.11 e 2.12[1], sendo que os valores finais das funções objetivo correspondentes são 22,48947 e 9,21565, respetivamente.

[1] Visto que os objetos possuem um grau de pertença distinto a cada *cluster*, e que os graus de pertença são também distintos entre diferentes objetos, não foi possível representar os objetos, com o respetivo grau de pertença a cada *cluster*.

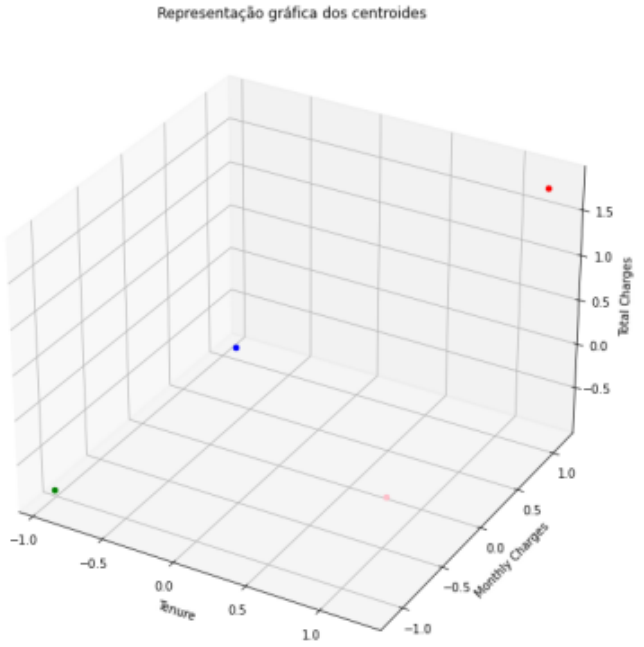


Figura 2.11: *Clustering Fuzzy* C-Mean - 4 *clusters*

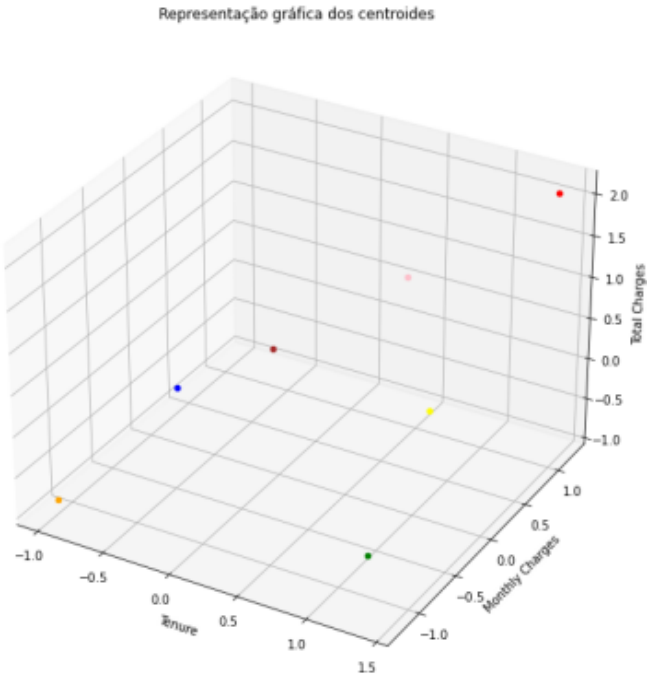


Figura 2.12: *Clustering Fuzzy* C-Mean - 7 *clusters*

2.6 Visão crítica

Como mencionado anteriormente, o *dataset* utilizado diz respeito à subscrição dos clientes a serviços de empresas de telecomunicações. O estudo deste *dataset* pode ser utilizado para efeitos de marketing e para a gestão de serviços oferecidos pelas empresas de telecomunicações. Através do *clustering* os perfis dos clientes são traçados e divididos por 4 ou 7 *clusters*. Deste modo, a empresa de telecomunicações pode criar 4 ou 7 serviços diferentes (com preços de subscrição diferentes), de forma a satisfazer todos os seus clientes.

Esta página foi intencionalmente deixada em branco.

3 Aprendizagem Supervisionada

Uma outra metodologia principal da aprendizagem automática é: a aprendizagem supervisionada. Na aprendizagem supervisionada são utilizados diversos algoritmos de aprendizagem onde os dados presentes no *dataset*, que alimenta o algoritmo, são compostos pelas *features* e pela solução desejada (*label*). Os algoritmos de aprendizagem supervisionada podem ser aplicados em modelos de classificação ou de regressão, onde nos modelos de classificação a saída preditiva é uma *label* binária ou multi-classe e nos modelos de regressão a saída preditiva é uma *label* contínua.

3.1 Análise do *dataset* utilizado

O *dataset* utilizado contém informação sobre clientes de empresas de telecomunicações. Este *dataset* é composto por uma variável *Consumer ID*, permitindo identifcar o cliente, e por oito *features*, sendo estas: *dependents*, *tenure*, *internet services*, *streaming movies*, *contract*, *payment method*, *monthly charges* e *total charges*. A *feature dependents* informa se o cliente tem ou não dependentes. A *feature tenure* diz respeito ao número de meses de permanência do cliente com o atual operador de telecomunicações. A *feature internet services* traduz o serviço de *internet* contratado. A *feature streaming movies* informa se o cliente tem ou não contratado serviço de *streaming* de filmes. A *feature contract* representa a duração do contrato, de cada cliente. A *feature payment method* o método com que o cliente realiza o pagamento do serviço prestado. A *feature monthly charges* traduz o valor mensal cobrado ao consumidor, do serviço subscrito. A *feature total charges* representa o valor total já cobrado ao consumidor, pela empresa de telecomunicações. O *dataset* ainda é composto pela *label churn*, sendo esta representativa do valor alvo a prever pelo algoritmo implementado e corresponde à elevada probabilidade do cliente interromper o atual serviço.

Foi realizada uma análise ao *dataset* utilizado com recurso ao *software* Jupiter. Numa visão global, verifica-se que o *dataset* é composto por três variáveis numéricas, sendo a *feature tenure* numérica discreta e as *features monthly charges* e *total charges* numéricas contínuas, por quatro variáveis

categóricas, sendo a *feature contract* categórica ordinal e as *features internet services*, *streaming movies* e *payment method* variáveis categóricas nominais, e por duas variáveis binárias, sendo a *feature dependents* e a *label churn*. O *dataset* utilizado é composto por 150 observações/amostras, sendo que não existe falta de informação em nenhuma célula que o compõem. Numa análise individualizada das várias *features* verifica-se que para a *feature dependents*, existe uma probabilidade de 67.79 % de esta ter o valor *No*, sendo esta uma variável binária. Já para a *feature tenure*, existem 59 valores distintos, sendo o valor médio de 31,60 meses, o valor máximo de 72 e o valor mínimo de 1. Aferese, relativamente à *feature InternetService*, que a fibra ótica é, de facto, o serviço de *internet* mais contratado, tendo uma incidência de 46,98%. Segue-se o DSL, solicitado 31.54% das vezes. Tem-se ainda que, em 21.92% das vezes, não é contratado qualquer tipo de serviço de *internet*. Para a *feature StreamingMovies*, verifica-se que apenas 36.91% dos clientes contrata o serviço de *streaming* de filmes, sendo que os restantes 41% não contrata este tipo de serviço. Para a *feature Contract*, constata-se que 59.06% dos clientes possui um contrato com duração de apenas 1 mês, 16.78% possui um contrato com duração de 1 ano, e cerca de 24.16% dos clientes possui um contrato de 2 anos. Para a *feature PaymentMethod*, 22.82% dos clientes opta pelo método de pagamento por transferência bancária, 16.78% utiliza o cartão de crédito como método de pagamento, 31.54% dos clientes efetua o pagamento através de um cheque eletrónico, e cerca de 28.86% efetua o pagamento mediante o envio de um cheque através do sistema postal. Para a *feature MonthlyCharges*, existem 92 valores distintos, sendo o valor médio de 60,3€ e os valores máximo e mínimo de 112,25€ e de 19,3€, respetivamente. Para a *feature TotalCharges*, existem 99 valores distintos, sendo o valor médio de 2032,23€ e os valores máximo e mínimo de 8129,3€ e de 19,45€, respetivamente.

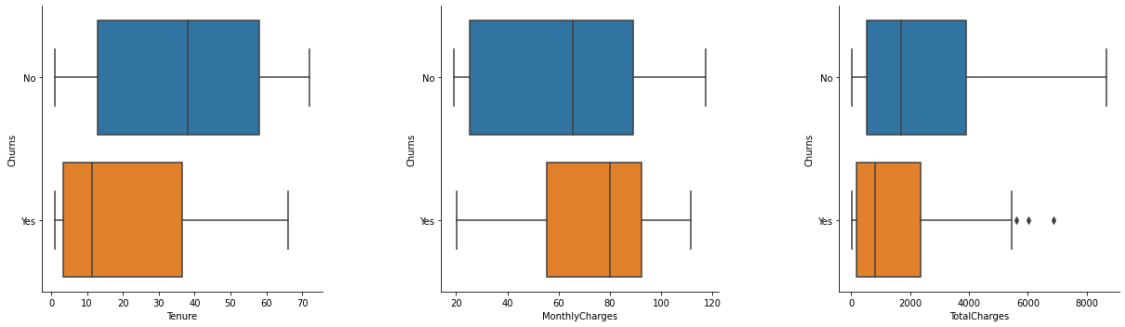
Verifica-se, através da *label Churns*, que a probabilidade do cliente interromper o atual serviço é de 30.87%.

3.2 Tratamento de dados

Uma vez que o sistema aprende a partir de um conjunto de dados disponível, torna-se necessário garantir a qualidade desses dados, como tal procedeu-se ao tratamento destes.

Inicialmente, procedeu-se à deteção da possível existência de campos (células) vazios e ao seu devido preenchimento, sendo que, como referido no sub-capitulo 3.1, não se verificou a falta de informação em nenhuma célula. De seguida, procedeu-se a uma análise aos dados que compõem o *dataset* de forma a identificar amostras não-representativas ou de fraca qualidade. Identificou-se que o cliente com o *consumer ID* 12 é um novo cliente e, como tal, a informação da sua respetiva amostra não apresenta qualidade para alimentar o algoritmo e foi removida. A variável *consumer ID* foi descartada, pois esta não apresenta informação relevante para o sistema de aprendizagem.

Realizou-se uma análise às várias *features*, de modo a verificar se alguma destas seria irrelevante, fazendo com que o sistema de aprendizagem perde-se a capacidade de generalizar um bom comportamento quando sujeito a novos dados. Através da visualização do gráfico boxplot entre a *label* e as várias *features* numéricas, presente na figura 3.1, é possível aferir que as *features* analisadas apresentam uma correlação com a *label*, pois para ambas as hipóteses da *label*, existe uma variação significativa entre os valores das *features*. Como forma de corroborar a correlação mencionada, recorreu-se ao método ANOVA, sendo que este método verifica se existe uma diferença significativa entre os valores médios das variáveis numéricas para cada hipótese da *label*. Os valores obtidos, relativos à métrica designada *p-value*, encontram-se na tabela 3.1. Como se verifica os valores do *p-value* são todos inferiores a 0.05, o que significa que as *features* estão correlacionadas com a *label*. De forma a verificar a correlação entre as *features* categóricas e a *label*, utilizou-se o método Chi-Square test. Este método encontra a probabilidade da hipótese nula (H_0), sendo que se o valor p-value for superior a 0.05 a H_0 é aceite, o que significa que as variáveis não estão correlacionadas. Como se verifica através da tabela 3.2 todos os valores de *p-value* são inferiores a 0.05, logo a H_0 é rejeitado, concluindo que as várias *features* são correlacionadas com a *label*.



(a) Tenure - Churn (b) Montlhy Chrages - Churn (c) Total Charges - Churn

Figura 3.1: Boxplot entre a *label* e as várias *features* numéricas

Tabela 3.1: Correlação - método ANOVA

	Tenure - Churn	***Monthly Charges - Churns***	***Total Charges - Churns***
p-value	8.252e-0.5	0.0119	0.0196

Tabela 3.2: Correlação - método Chi-Square test

	Dependents - Churn	***InternetService - Churns***	***StreamingMovies - Churns***	***Contract - Churns***	***PaymentMethod - Churns***
p-value	0.0435	0.0009	0.0007	5.404e-5	4.525e-6

Posteriormente, houve a necessidade de proceder a uma adaptação nas *features* categóricas, convertendo-as em *features* numéricas. De seguida, procedeu-se à normalização *standard* das *features*. Finalmente, procedeu-se à divisão aleatória das amostras do *dataset* nos *subsets* de treino e de teste, sendo

que o *subset* de treino corresponde a 80% das amostras, onde 16% destas corresponde à validação, e o *subset* de teste corresponde aos restantes 20% das amostras do *dataset*. Na figura 3.2 encontram-se representadas graficamente as *feautures* relativas ao *subset* de treino, pré e pós normalização.

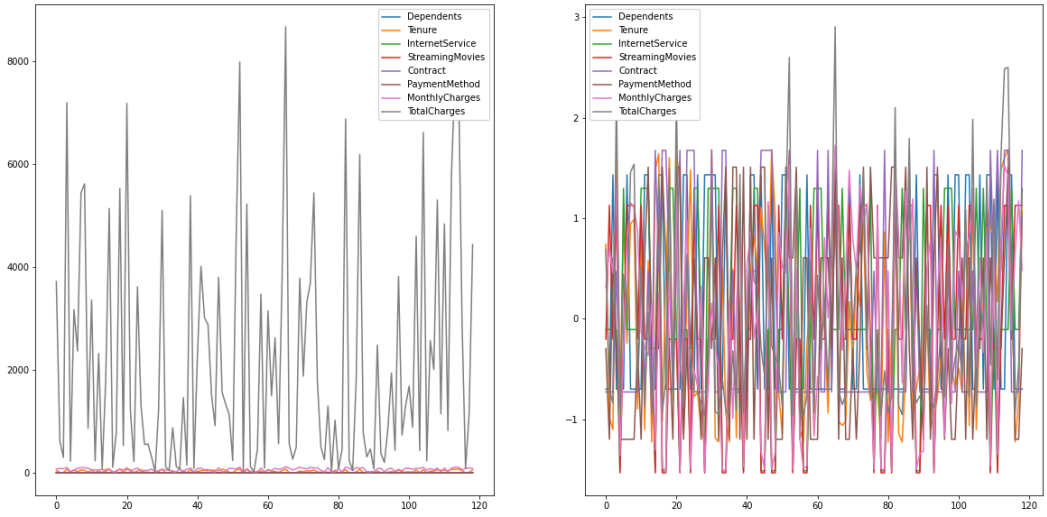


Figura 3.2: Representação das *features* do *subset* treino, pré e pós normalização

3.3 Algoritmo K-Nearest Neighbours

O algoritmo K-Nearest Neighbours (KNN) permite a classificação do conjunto de dados de treino com base no cálculo das distâncias aos vizinhos mais próximos. Por omissão é utilizada a distância de Minkowski, que corresponde a uma generalização das distâncias Euclidiana e de Manhattan, baseada num espaço vetorial normalizado. Neste caso, foi utilizada a distância Euclidiana.

De modo a treinar o algoritmo com o objectivo deste apresentar o melhor desempenho possível, recorreu-se à validação cruzada. A validação cruzada é uma técnica que avalia a capacidade de generalização de um modelo a partir de um conjunto de dados, através da análise do desempenho do sistema para um novo conjunto de dados. Neste caso, foi utilizada a técnica de validação cruzada denominada de K-Fold, sendo que esta apenas é realizada no *subset* de treino. Inicialmente o *subset* de treino é dividido em k partes iguais, onde $k = 5$. A primeira parte serve como *subset* de teste e as restantes $k-1$ partes servem para treinar o modelo. De seguida o modelo é testado com o *subset* de teste. O processo é repetido k vezes, onde em cada iteração é alterada a parte respetiva ao *subset* de teste. Após analisar o desempenho do algoritmo para n vizinhos mais próximos, onde $n \in [1,36]$, obtiveram-se melhores resultados para $n = 2$, sendo que a precisão do modelo com a validação cruzada K-Fold é de 70,58%.

A tabela 3.3 corresponde à matriz de confusão obtida após o treino do algoritmo, sendo a precisão do algoritmo KNN treinado de 83,19%. Na tabela 3.4 encontram-se representados os resultados do

algoritmo, referente ao *subset* de treino.

Tabela 3.3: Matriz de confusão do *subset* de treino - algoritmo KNN

		Valor predito	
		Yes	**No**
Label (Real)	**Yes**	82	0
	No	20	17

Tabela 3.4: Resultados obtidos para o *subset* treino - algoritmo KNN

	precision	recall	f1-score	support
No	0.80	1.00	0.89	82
Yes	1.00	0.46	0.63	37
accuracy			0.83	119
macro avg	0.90	0.73	0.76	119
weighted avg	0.86	0.83	0.81	119

De seguida, introduziu-se no algoritmo o *subset* de teste, de forma a verificar o seu desempenho para novos dados. Na tabela 3.5 encontra-se representada a matriz de confusão referente ao *subset* de teste e na tabela 3.6 encontram-se os resultados obtidos pelo algoritmo para o mesmo *subset*, sendo que a precisão obtida foi de 80%.

Tabela 3.5: Matriz de confusão do *subset* de teste - algoritmo KNN

		Valor predito	
		Yes	**No**
Label (Real)	**Yes**	20	1
	No	5	4

Tabela 3.6: Resultados obtidos para o *subset* teste - algoritmo ANN

	precision	recall	f1-score	support
No	0.80	0.95	0.87	21
Yes	0.80	0.44	0.57	9
accuracy			0.80	30
macro avg	0.80	0.70	0.72	30
weighted avg	0.80	0.80	0.78	30

Analisando os resultados obtidos, verifica-se que o algoritmo KNN treinado apresenta uma boa capacidade de generalização, pois apresenta uma coerência no seu desempenho para os novos dados do *subset* de teste. Os resultados são bastante aceitáveis, sendo a precisão média de 80% e o *recall* (sensibilidade do modelo) de 70%.

3.4 Algoritmo Artificial Neural Networks

O algoritmo Artificial Neural Network (ANN) permite a simulação de uma rede neuronal, sendo que o modelo básico de um neurónio é denominado perceptrão. Numa primeira instância as entradas do perceptrão são ponderadas e somadas, incluindo ainda um termo relativo a *offset*. Assim, o resultado final resulta da aplicação de uma função de ativação à saída intermédia anteriormente calculada.

Testaram-se várias funções de ativação, bem como diferentes valores de *hidden_layer_sizes*[1], tendo-se obtido melhores resultados, para a função de ativação *ReLU (Retangular Linear Unit)*, representada na figura 3.3, para 2 neurónios na camada oculta. Utilizou-se o *solver lbfgs* definido como algoritmo específico de aprendizagem, ativando-se *verbose* como *True* de modo a permitir a visualização da função objetivo em cada iteração. É ainda possível habilitar a interrupção prematura do treino, utilizando a técnica de validação cruzada. Assim, definiu-se o parâmetro *validation_fraction* como 0.16, para que 16% dos dados de treino sejam usados como dados de validação. Os dados de validação vão servir para avaliar a capacidade de generalização do algoritmo na fase de treino, interrompendo o processo de treino quando o erro verificado neste subconjunto de dados tender a aumentar.

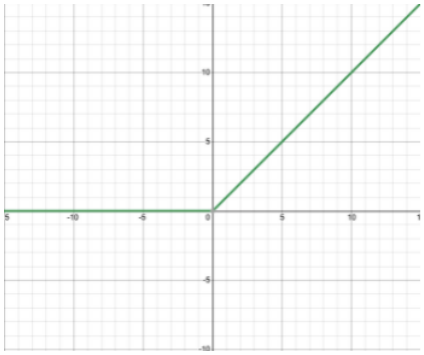


Figura 3.3: Gráfico da função de ativação *ReLU*

Na tabela 3.7 encontram-se representados as matrizes de confusão resultantes do treino do algoritmo e do teste do algoritmo treinado e na tabela 3.8 encontram-se os resultados obtidos pelo algoritmo no processo de treino e no processo de teste do mesmo. Na fase de treino o algoritmo apresentou uma precisão média de 80% e um f1-score médio de 74%, sendo este valores admissíveis. Após o treino, foi realizado um teste de forma a avaliar o seu desempenho, introduzindo novos dados. Obteve-se uma precisão média de 87% e um f1-score médio de 83%.

Na tabela 3.9 encontram-se indicados os valores dos pesos e dos *offsets* obtidos após o treino do algoritmo, sendo que cada linha da matriz dos pesos se refere a uma entrada diferente.

Conclui-se, assim, que neste caso o algoritmo ANN treinado é robusto, não se verificando a ocorrência do fenómeno de *overfitting*, nem de *underfitting*.

[1] Número de neurónios presentes na camada oculta

Tabela 3.7: Comparação das matrizes de confusão obtidas para o classificador ANN.

(a) *Subset* de dados de treino e validação

		Valores reais	
		1	0
Valores Previstos	1	TP - 80	FP - 2
	0	FN - 20	TN - 17

(b) *Subset* de dados de teste

		Valores reais	
		1	0
Valores Previstos	1	TP - 20	FP - 1
	0	FN - 3	TN - 6

Tabela 3.8: Comparação das métricas obtidas para o classificador ANN.

(a) *Subset* de dados de treino e validação

	precision	recall	f1-score	support
0.0	0.80	0.98	0.88	82
1.0	0.89	0.46	0.61	37
accuracy			0.82	119
macro avg	0.85	0.72	0.74	119
weighted avg	0.83	0.82	0.79	119

(b) *Subset* de dados de teste

	precision	recall	f1-score	support
0.0	0.87	0.95	0.91	21
1.0	0.86	0.67	0.75	9
accuracy			0.87	30
macro avg	0.86	0.81	0.83	30
weighted avg	0.87	0.87	0.86	30

(c) Pesos referentes às entradas do modelo

```
-0.8425    1.1296
 0.1064    0.0392
-4.1313   -0.2047
-3.9845   -2.2956
-0.3079    1.5147
-6.5095    0.6755
 0.6095   -0.6049
 3.0487   -0.6777
```

(d) *Offsets* da função de saída intermédia

```
-2.4552   16.1719    9.8959
```

Tabela 3.9: Pesos e *offsets* obtidos para cada entrada

3.5 Algoritmo Support Vector Machines

O algoritmo Support Vector Machines (SVM) é baseado na minimização estrutural do erro, sendo utilizado de modo a determinar a fronteira de decisão ótima para separar diferentes classes e maximizar a margem, oferecendo alguma flexibilidade para classificar novos dados.

Neste caso, foi utilizado um *kernel* linear, tendo-se definido a função custo (C) como 2. Foi ainda utilizado o modelo *Grid Search*, para ajustar os parâmetros associados ao algoritmo. Assim, aplicando este método ao *subset* de dados de treino, foi utilizado o método de validação cruzada *k-Fold*, tendo-se definido k igual a 3. Foi ainda considerado o SVC (classificador de vetores de suporte). A tabela 3.10 representa os valores dos parâmetros fornecidos ao modelo, bem como os melhores valores devolvidos por este.

Tabela 3.10: Aplicação do modelo Grid Search.

(a) Valores dos parâmetros à entrada do modelo *Grid Search*

'kernel'	'rbf'
'gama'	[1e-3, 1e-4]
'C'	[1, 10, 100, 1000]

'kernel'	'linear'
'C'	[1, 10, 100, 1000]

(b) Valores dos parâmetros devolvidos pelo modelo *Grid Search*

'kernel'	'rbf'
'gama'	0.0001
'C'	1000

Na tabela 3.11 encontram-se representados as matrizes de confusão resultantes do treino do algoritmo e do teste do algoritmo treinado e na tabela 3.12 encontram-se os resultados obtidos pelo algoritmo no processo de treino e no processo de teste do mesmo.

Através da análise dos resultados obtidos afere-se que o modelo encontrado é bastante robusto, pois apresenta um bom desempenho para novos dados.

Tabela 3.11: Comparação das matrizes de confusão obtidas para o classificador SVM.

(a) *Subset* de treino

		Valores reais	
		1	0
Valores Previstos	1	TP - 59	FP - 7
	0	FN - 14	TN - 15

(b) *Subset* de validação

		Valores reais	
		1	0
Valores Previstos	1	TP - 16	FP - 0
	0	FN - 1	TN - 6

(c) *Subset* de teste

		Valores reais	
		1	0
Valores Previstos	1	TP - 15	FP - 6
	0	FN - 2	TN - 8

Tabela 3.12: Comparação das métricas obtidas para o classificador SVM.

(a) *Subset* de dados de treino

	precision	recall	f1-score	support
0.0	0.81	0.89	0.85	66
1.0	0.68	0.52	0.59	29
accuracy			0.78	95
macro avg	0.75	0.71	0.72	95
weighted avg	0.77	0.78	0.77	95

(b) *Subset* de dados de validação

	precision	recall	f1-score	support
0.0	0.94	1.00	0.97	16
1.0	1.00	0.86	0.92	7
accuracy			0.96	23
macro avg	0.97	0.93	0.95	23
weighted avg	0.96	0.96	0.96	23

(c) *Subset* de dados de teste

	precision	recall	f1-score	support
0.0	0.88	0.71	0.79	21
1.0	0.57	0.80	0.67	10
accuracy			0.74	31
macro avg	0.73	0.76	0.73	31
weighted avg	0.78	0.74	0.75	31

4 Conclusão

Os objetivos associados à realização deste trabalho foram cumpridos. A realização deste trabalho possibilitou a aquisição de conhecimentos de *machine learning*, de forma a consolidar os conhecimentos anteriormente apreendidos durante as aulas.