Universidade do Minho

Mestrado em Engenharia Informática

Dados e Aprendizagem Automática

Conceção e otimização de modelos de *Machine Learning*

Grupo 12

Ana Murta (pg50184)
Ana Henriques (pg50196)
Diogo Pires (pg50334)
Gonçalo Soares (pg50393)

janeiro, 2023

Conteúdo

Lista de Figuras

1 Introdução

O presente relatório visa acompanhar o desenvolvimento do trabalho prático elaborado ao longo do semestre. Como tal, contempla dois grandes grupos: um grupo exclusivamente dedicado ao *dataset* de competição, tendo sido fornecido um *dataset* de aprendizagem (*training_data.csv*) e outro dataset de teste (*test_data.csv*) pela equipa docente; e outro grupo que confere foco ao *dataset* escolhido pelo grupo de trabalho (*Global_Superstore2.xlsx*). A realização do trabalho prático foi concebida através da linguagem *Python*, que faculta diversas bibliotecas para implementar algoritmos de *data machine*.

As secções do relatório pretendem demonstrar e explicar o trabalho conduzido, apresentando os *datasets*, o trabalho elaborado sobre os mesmos, e os modelos de *Machine Learning* (ML) desenvolvidos, fornecendo um *insight* geral das decisões adotadas pelo grupo.

A metodologia seguida foi a *CRISP-DM* uma vez que permite melhorar os modelos na situação dos resultados não serem satisfatórios. Assim sendo, o primeiro passo é estudar o domínio do *dataset* e as suas característica. De seguida, vem a preparação mais adequada possível dos dados, tendo como objetivo remover *overfitting* e melhorar a performance dos modelos. Entretanto, ambos os conjuntos de dados são divididos em dois, um para treino, e outro para teste, para treinar os modelos. Por fim, é feita uma avaliação da qualidade dos resultados, recomeçando o processo se não forem obtidos os melhores. Sendo considerado o modelo como bom, o processo é dado como concluído.

O conjunto de dados referente às vendas *online*[1] foi obtido através da plataforma *Kaggle*, na qual também se desenrolou a competição do primeiro *dataset*. Após a pesquisa de vários conjuntos de dados para a elaboração do trabalho, o grupo selecionou o *dataset* acima referido devido ao interesse demonstrado pelo tópico, i.e., por se centrar numa tema muito influenciado pela situação atual pós-pandémica, e pela informação disponível acerca das *features* que integram o mesmo.

2 Dataset Competição

2.1 Descrição do *dataset*

O conjunto de dados de competição, descrito mais pormenorizadamente aqui, é constituído por 13 *features* e 5000 entradas (sem contar com o *header*, que contém as informações de cada *feature*).

O objetivo projetado para cada equipa de trabalho passa por desenvolver o melhor modelo possível de modo a prever a quantidade de incidentes rodoviários num dado ponto temporal. O modelo a desenvolver deverá ter, na sua base, *features* como a magnitude do atraso que se verifica numa determinada hora, o tempo de atraso provocado pelos incidentes, a temperatura, pressão atmosférica e a velocidade do vento, entre outras *features* que caracterizam um determinado ponto temporal. A *feature target* neste conjunto de dados é a quantidade de incidentes rodoviários, numa escala qualitativa (`None`, `Low`, `Medium`, `High` e `Very_High`). O modelo a construir deve prever, portanto, para cada registo do *dataset* de teste, o nível de incidentes rodoviários esperados. A métrica utilizada para validar o modelo construído é a *accuracy*, i.e., a percentagem de acerto.

A seguinte listagem apresenta as features presentes no conjunto de dados em questão, enunciando resumidamente o significado de cada um, assim como o tipo de dados usados para a sua representação:

- *city_name* - nome da cidade em causa (atributo qualitativo nominal, representado sob a forma de uma *string*);
- *record_date* - o *timestamp* associado ao registo (atributo qualitativo ordinal, representado sob a forma de um *string*);
- *magnitude_of_delay* - magnitude do atraso provocado pelos incidentes que se verificam no *record_date* correspondente (atributo qualitativo categórico, representado sob a forma de uma *string*);

[1] c.f. https://www.kaggle.com/datasets/apoorvaappz/global-super-store-dataset

- *delay_in_seconds* - atraso, em segundos, provocado pelos incidentes que se verificam no *record_date* correspondente (atributo quantitativo discreto, representado sob a forma de um *int*);
- *affected_roads* - estradas afectadas pelos incidentes que se verificam no *record_date* correspondente (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *luminosity* - o nível de luminosidade que se verificava na cidade de Guimarães (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *avg_temperature* - valor médio da temperatura para o *record_date* na cidade de Guimarães (atributo quantitativo contínuo, representado sob a forma de um *float*);
- *avg_atm_pressure* - valor médio da pressão atmosférica para o *record_date* na cidade de Guimarães (atributo quantitativo contínuo, representado sob a forma de um *float*);
- *avg_humidity* - valor médio de humidade para o *record_date* na cidade de Guimarães (atributo quantitativo contínuo, representado sob a forma de um *float*);
- *avg_wind_speed* - valor médio da velocidade do vento para o *record_date* na cidade de Guimarães (atributo quantitativo contínuo, representado sob a forma de um *float*);
- *avg_precipitation* - valor médio de precipitação para o *record_date* na cidade de Guimarães (atributo quantitativo contínuo, representado sob a forma de um *float*);
- *avg_rain* - avaliação qualitativa do nível de precipitação para o *record_date* na cidade de Guimarães (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *incidents* - indicação acerca do nível de incidentes rodoviários que se verificam no *record_date* correspondente na cidade de Guimarães (atributo qualitativo categórico, representado sob a forma de uma *string*).

A Figura 1 apresenta o cabeçalho do *dataset*, com dados para as *features* anteriormente descritas.

	city_name	magnitude_of_delay	delay_in_seconds	affected_roads	record_date	luminosity	avg_temperature	avg_atm_pressure	avg_humidity	avg_wind_speed	avg_precipitation	avg_rain	incidents
0	Guimaraes	UNDEFINED	0	,	2021-03-15 23:00	DARK	12.0	1013.0	70.0	1.0	0.0	Sem Chuva	None
1	Guimaraes	UNDEFINED	385	N101,	2021-12-25 18:00	DARK	12.0	1007.0	91.0	1.0	0.0	Sem Chuva	None
2	Guimaraes	UNDEFINED	69	,	2021-03-12 15:00	LIGHT	14.0	1026.0	64.0	0.0	0.0	Sem Chuva	Low
3	Guimaraes	MAJOR	2297	N101,R206,N105,N101,N101,N101,N101,N101,N101,N...	2021-09-29 09:00	LIGHT	15.0	1028.0	76.0	1.0	0.0	Sem Chuva	Very_High
4	Guimaraes	UNDEFINED	0	N101,N101,N101,N101,N101,	2021-06-13 11:00	LIGHT	27.0	1020.0	52.0	1.0	0.0	Sem Chuva	High

Figura 1: Excerto do cabeçalho do *dataset* de competição.

2.2 Análise dos dados

O trabalho com este *dataset* segue-se com a elaboração de um estudo geral do estado inicial dos dados no *notebook pre_data_analysis.ipynb*, de modo a determinar o tratamento necessário para a utilização dos dados na criação de modelos de aprendizagem automática.

A exploração estatística efetuada permitiu conhecer, de um modo geral, o conteúdo das *features* que integram o conjunto de dados anteriormente apresentado. A Figura 2 apresenta o *output* da função `describe()` da biblioteca *Pandas*, para os dados numéricos, o que permitiu obter informação estatísticas acerca dos mesmos, desde valores extremos das diversas *features* numéricas, da sua média e desvio padrão. Relativamente aos *missing values*, valores que devem ser tratados aquando da fase de pré-processamento, verificou-se, através da função `info()`, que apenas a *feature affected_roads* apresentava à volta de 1.7% de *missing values*.

	delay_in_seconds	avg_temperature	avg_atm_pressure	avg_humidity	avg_wind_speed	avg_precipitation
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.0
mean	560.567000	14.583000	1018.145000	74.455000	1.253500	0.0
std	1686.859581	4.820514	5.174372	17.204638	1.269847	0.0
min	0.000000	1.000000	997.000000	6.000000	0.000000	0.0
25%	0.000000	11.000000	1015.000000	63.000000	0.000000	0.0
50%	0.000000	14.000000	1019.000000	78.000000	1.000000	0.0
75%	234.000000	18.000000	1022.000000	90.000000	2.000000	0.0
max	31083.000000	35.000000	1032.000000	100.000000	10.000000	0.0

Figura 2: Dados estatísticos relativamente aos valores numéricos.

A Figura 3 oferece uma representação gráfica da distribuição dos dados das *features* numéricas através de *boxplots*. É possível observar que algumas *features* possuem *outliers*, valores atípicos ou anormais em um conjunto de dados, que estão fora de um intervalo de valores esperados ou que são muito diferentes dos demais valores do conjunto de dados. Dependendo da *feature* e do tipo de informação que esta carrega, é preciso analisar se estes valores devem ser removidos e, alternativamente, substituídos pela mediana dessa *feature*, por exemplo, ou se devem ser mantidos. Adicionalmente, recorremos à função `pairplot()` da biblioteca *Seaborn* para elaborar um gráfico de dispersão para visualizar a relação entre as variáveis e para identificar padrões ou tendências nos dados.

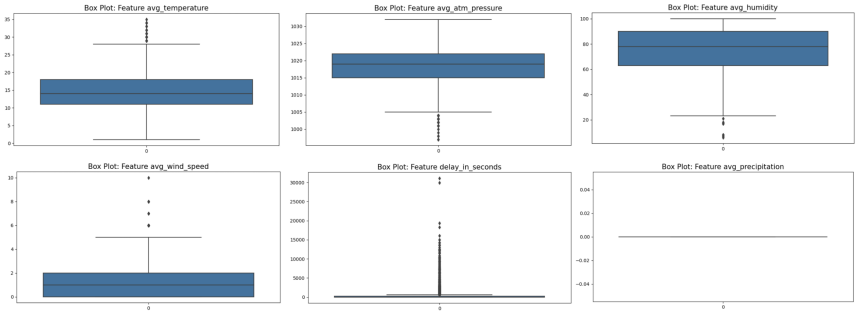


Figura 3: Distribuição dos dados das *features* numéricas.

No que toca ao estudo das *features* categóricas, o grupo optou por se guiar por histogramas para entender a distribuição das mesmas e para identificar padrões ou tendências nos dados. Como podemos ver pela Figura 4, enquanto que, por exemplo, a variável *city_name* apresenta apenas um tipo de valor (*Guimarães*), a variável *affected_roads* apresenta uma enormíssima discrepância entre valores.

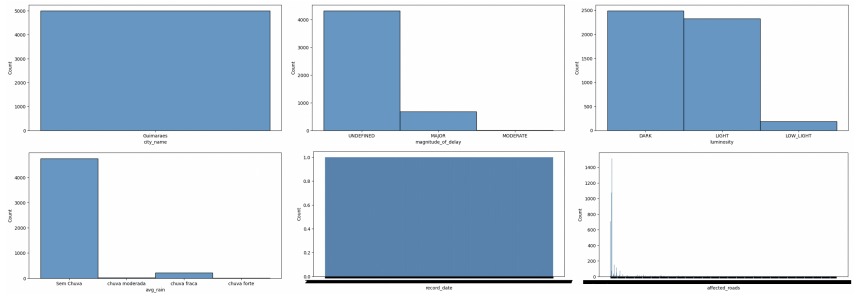


Figura 4: Distribuição dos dados das *features* categóricas.

A análise de correlação apresentada na Figura 5 permite medir a força e direção da associação entre duas variáveis – o que pode fornecer informação útil acerca das *features* que devem incorporar os modelos de aprendizagem automática. De um modo geral, pode observar-se que não existem *features* com uma um grau elevado de correlação. No entanto, a variável *avg_precipity* não possui qualquer espaço na matriz, indicando que só tem um tipo de valores (neste caso, 0.0).

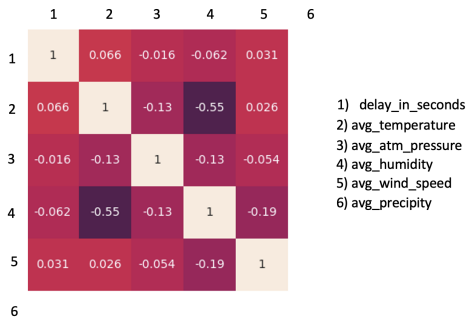


Figura 5: Matriz de correlação.

2.3 Tratamento dos dados

Após analisar o conjunto de dados em mãos, segue-se o trabalho desenvolvido em termos de tratamento e limpeza inicial do *dataset*. A preparação de um *dataset* para a criação de modelos de *Machine Learning* não é um processo com solução única, uma vez que alguns modelos de aprendizagem automática podem impor a implementação de determinados passos de tratamento de dados que podem ser supérfluos para outros modelos. Assim sendo, o foco deste tratamento inicial passa por efetuar um conjunto de operações transversais a vários modelos de *Machine Learning* – sendo, numa fase posterior, efetuada uma preparação adicional dos dados, caso tal seja necessário.

A limpeza dos dados consistiu na seleção das *features* do *dataset* consideradas mais importantes para a previsão – ***Feature Selection***. Este processo levou à remoção das *features city_name*, *avg_precipitation* e *magnitude_of_delay*, seguida também da remoção de entradas (linhas) duplicadas.

Outra técnica aplicada foi a criação de novas *features* a partir da extração de dados da *record_date* – ***Feature Engineering***. Os dados extraídos foram a hora, o dia, o mês e o dia da semana, através da classe *DateTimeIndex* da biblioteca *Pandas*. Adicionalmente, criou-se duas variáveis binárias, uma que indica se determinada data é fim de semana e outra que indica se é um feriado. De modo a facilitar a visualização e compreensão da variável *hour*, agrupou-se as horas do dia em 5 *bins*, visando a uniformidade na proporção de dados entre as mesmas. Atendendo às informações repetitivas e difíceis de ler da *affected_roads*, primeiramente, recolheu-se todas as estradas afetadas presentes no *dataset* e, seguidamente, converteu-se essa variável categórica numa série de variáveis binárias, onde cada linha terá o valor 1 na(s) corrependente(s) estrada(s) afetada(s) e 0 nas demais – ***One-Hot Encoding***.

A partir da análise preliminar dos dados, foi possível verificar que a *feature delay_in_seconds* apresentava um intervalo de valores enorme, o que podia afetar negativamente a performance de alguns modelos de *Machine Learning*. Como tal, essa *feature* foi reduzida, convertendo os segundos para minutos. Todavia, o intervalo de valores permanecia consideravelmente grande e, por consequente, procedeu-se à **normalização** da mesma, apresentando uma escala entre -1 e 1.

O conjunto de dados também apresentava algumas variáveis com *outliers*, o que impulsionou a remoção e substituição dos mesmos pela sua mediana. Isto porque, dentro das medidas de tendências centrais, a mediana é a mais recomendada para este processo, já que é menos suscetível a *outliers*.

2.4 Modelação

O problema em questão consiste num problema de classificação, motivo pelo qual as técnicas de aprendizagem automática aplicadas centram-se em árvores de decisão, regressão (logística), redes neuronais artificiais e modelos de *ensemble learning* – *random forest* e *cat boost*. As **árvores de decisão**, modelos supervisionados de fácil interpretação, utilizam uma estrutura de decisão para classificar as entradas. Por sua vez, as **redes neuronais**, igualmente modelos supervisionados, são capazes de obter excelentes resultados quando lidam com grandes *datasets* e caracterísitcas complexas, daí, no nosso caso, não ter refletido resultados tão frutuosos. Já os modelos de ***ensemble learning*** combinam vários modelos para melhorar a precisão e robustez do modelo final, lidando bem com *overfitting*.

2.4.1 Árvores de Decisão

O modelo de classificação baseado numa árvore de decisão foi construído com recurso à classe *DecisionTreeClassifier* do *scikit-learn*. A função `fit()` é utilizada para treinar o modelo com os dados de treino (que constitui 80% do conjunto de dados).

Posteriormente, o modelo é avaliado usando a técnica de validação cruzada, com 200 *folds*, que avalie a capacidade de generalização do modelo. A métrica de desempenho utilizada é o *accuracy*. A função `cross_val_score()` do *scikit-learn* é usada para calcular as pontuações de validação cruzada e imprimir a média e o desvio padrão dessas pontuações. O valor da ***accuracy*** obtido foi à volta de 0.914, o que significa que o modelo é capaz de prever corretamente 91.4% das vezes a classe *target*. Na Figura 6, temos a representação da matriz de confusão resultante.

2.4.2 *Random Forest*

O presente modelo usa a biblioteca *scikit-learn* para construir um classificador de floresta aleatório – *RandomForestClassifier*. O processo começa por aplicar diferentes transformações de coluna, nomeadamente a criação de passos para remover *features* e outros para tratar de algumas *features* em específico – *record_date* e *affected_roads*. Depois, é construído o *pipeline* que, para tratar dos valores numéricos, inclui o *SimpleImputer*, *MinMaxScaler* e *KBinsDiscretizer* e, para preparar os dados categóricos, inclui o *SimpleImputer* e *OneHotEncoder*.

Assim que terminadas estas transformações, o *pipeline* é usado para selecionar os melhores dados através de *SelectKBest*, *VarianceThreshold* e *f_classif*. A seguir, este classificador é, então, treinado com os dados pré-processados e a especificação de hiperparâmetros – *n_estimators, max_leaf_nodes, min_samples_leaf, max_features, max_depth, n_jobs, bootstrap, random_state*.

Finalmente, o modelo é ajustado usando o *GridSearchCV* para otimizar os hiperparâmetros e o seu desempenho é medido com a métrica *balanced_accuracy*, obtendo um *score* de treino de 0.822.

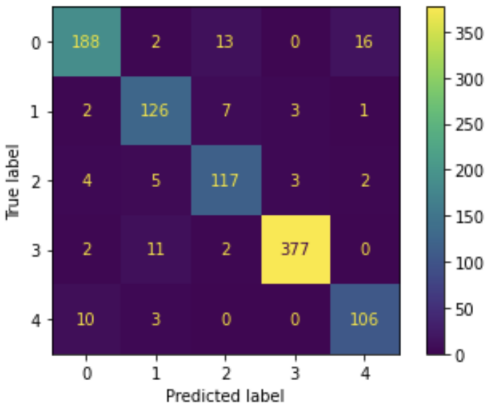


Figura 6: Matriz de confusão de árvore de decisão

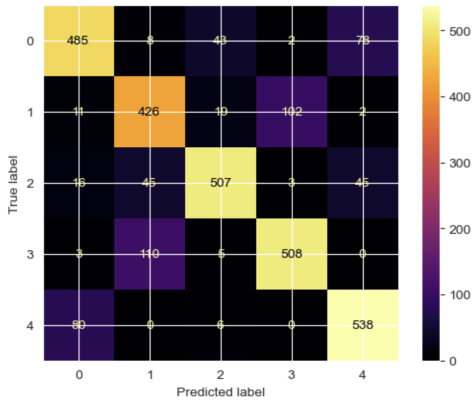


Figura 7: Matriz de confusão de *random forest*

2.4.3 Redes Neuronais

De seguida, foi treinado o modelo de redes neuronais artificiais, sendo, para tal, necessário utilizar a biblioteca *tensorflow*.

Como este modelo precisa que todos os dados sejam numéricos, ao tratamento de dados anteriormente efetuado foi adicionado uma transformação, na qual se transforma todos os dados categóricos em numéricos. É importante denotar que as *features* também foram escaladas usando um *MinMaxScaler*, para garantir que todas as *features* se encontrassem na mesma magnitude de valores.

Depois disto, o modelo foi treinado com 80% dos dados de treino e validado com 20% dos dados

de treino. Numa fase inicial, optou-se por criar um modelo sem *tunning* de *hiperparâmetros*; porém, devido à diferença entre os valores de *accuracy* de treino e de teste e de modo a evitar *overfitting*, decidiu-se fazer *tunning* de um modelo bastante simples com vários parâmetros:

- Número de neurónios na camada de *input*;
- Número de neurónios na *hidden layer*;
- Funções de ativação a serem utilizadas na *hidden layer*;
- *learning rate* utilizado durante o treino do modelo.

As configurações que foram mantidas durante o *tunning* dos parâmetros foram:

- O número de camadas da rede neuronal, sendo que há uma camada de *input*, uma *hidden layer* e uma camada de *output*;
- A função de ativação utilizada na camada de *output*, que é a função *softmax*;
- Um *callback* chamado *EarlyStopping* com o objetivo principal de evitar o *overfitting* do modelo.

A função *softmax* foi escolhida para que o *output* para cada tipo de *target* fosse uma probabilidade. O *callback EarlyStopping* termina o treino do modelo quando uma determinada métrica – neste caso, o *validation loss* – deixou de melhorar.

De um modo geral, este modelo não se comportou tão bem como os outros, não tendo chegado a valores de *accuracy* superiores a 80%.

2.4.4 *CatBoost*

O *CatBoost* é um algoritmo de aprendizagem supervisionada que usa *gradient boosting* com *decision trees*. Neste modelo também foi efetuado o *hypertunning* segundo os seguintes hiperparâmetros:

- *Learning rate*;
- *Depth*;
- *Early stopping rounds*.

Comparativamente com os anteriores, este modelo mostrou uma *accuracy* mais alta e uma resiliência a *overfitting* maior, acabando por se refletir no modelo com maior e melhor *accuracy* nas submissões feitas para a competição.

2.4.5 *XGBoost*

Para o modelo *XGBoost*, foi igualmente aplicado o mesmo tratamento de dados que no anterior, visto que este é baseado em árvores de decisão.

Assim como nos outros modelos, também foi feito o *hypertunning* deste modelo, segundo os seguintes hiperparâmetros:

- Número de *gradient boosted trees* usadas;
- Tamanho máximo de cada árvore;
- *Learning rate*.

De um modo geral, este modelo esteve com valores de *accuracy* bastante parecidos com os do modelo *CatBoost*, não havendo muita comparação a ser feita entre estes dois.

2.5 Conclusões sobre o trabalho realizado

O desenvolvimento dos modelos selecionados para este problema de classificação revelou-se um processo desafiante, no qual foi exigida bastante reflexão por parte do grupo e, por vezes, a reconstrução dos modelos e, até mesmo, do tratamento dos dados. Embora uns melhores tenham oferecido melhores resultados que outros, o grupo acredita ter cumprido os objetivos definidos, priorizando sempre a explicação de cada estratégia abordada. No geral, considera-se que o desempenho dos modelos construídos foram apelativos atendendo ao conjunto de dados trabalhado e às suas características. Para trabalho futuro, o grupo gostaria de aperfeiçoar alguns dos modelos previamente descritos que não demonstraram resultados tão bons e, ainda, adicionar novos dados para melhorar a precisão e a performance dos modelos. Quantos mais os exemplos com que os modelos são treinados, melhor a sua adaptação ao problema, podendo torná-lo capaz de lidar com dados incomuns.

3 *Global Super Store Dataset*

3.1 Descrição do *dataset*

O conjunto de dados selecionado pelo grupo de trabalho consiste no dataset *"Global Super Store Dataset"* com 23 *features*, i.e., colunas, e 51290 entradas.

Este *dataset* é composto por dados relativos às vendas online de vários produtos. Atualmente, como nos enquadramos numa situação pós-pandémica em que se verificou um aumento significativo das compras online, torna-se útil avaliar o seu lucro. Assim sendo, este torna-se vantajoso para empresas que pretendam expandir o seu negócio para este mercado. Portanto, a partir destes dados, o nosso objetivo é criar um modelo capaz de prever o lucro gerado por uma venda.

A seguinte listagem apresenta as *features* presentes no conjunto de dados, descrevendo de um modo sucinto o seu significado, assim como o tipo de dados usados para a sua representação:

- *Row id*: identificador de entrada (atributo qualitativo ordinal, representado sob a forma de um *int*);
- *Order id*: identificador do pedido do cliente (atributo qualitativo nominal, representado sob a forma de uma *string*);
- *customer id*: identificador do cliente (atributo qualitativo nominal, representado sob a forma de uma *string*);
- *order date e ship date*: data em que é realizado e enviado o pedido, respetivamente (atributo qualitativo ordinal, representado sob a forma de um *string*);
- *ship mode*: modo de transporte, podendo este ser *Fisrt Class*, *Second Class*, *Same Day* ou *Standard Class.* (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *customer name*: nome do cliente (atributo qualitativo nominal, representado sob a forma de uma *string*);
- *Segment*: segmento de mercado no qual a compra pertence, como por exemplo: *Consumer*; *Home Office*, entre outros (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *City, state e country* : cidade, estado e país de destino da encomenda (atributo qualitativo nominal, representado sob a forma de uma *string*);
- *Postal code*: Código postal do destino da encomenda (atributo qualitativo nominal, representado sob a forma de uma *int*);

- *Region e Market*: região e mercado do destino da encomenda, respetivamente (atributos qualitativos categóricos, representados sob a forma de uma *string*). Temos que a região pode ser, por exemplo: *East*, *Oceania*, entre outros. Enquanto, o mercado pode ser, por exemplo: *US*, *LATAM - Latin-American*, etc;
- *Product id*: identificador do produto (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *Product name*: nome do produto (atributo qualitativo nominal, representado sob a forma de uma *string*);
- *Category e subcategory*: categoria e subcategoria do produto (atributo qualitativo categórico, representado sob a forma de uma *string*);
- *Sales e Quantity*: custo e quantidade do produto da encomenda (atributo quantitativo contınuo e discreto, respetivamente, representado sob a forma de um *float*);
- *Discount, Shipping Cost e Profit*: desconto, preço de envio da encomenda e lucro, respetivamente(atributo quantitativo contınuo, representado sob a forma de um *double*));
- *Order Priority*: Prioridade da encomenda, podendo esta ser *Critical*, *High*, *Medium* ou *Low*(atributo qualitativo categórico, representado sob a forma de uma *string*);

A Figura 8 apresenta o cabeçalho do *dataset*, com dados para as *features* anteriormente descritas.

Row ID	Order ID	Order Date	Ship Date	Ship Mode	Custom	Custom	Segmen	City	State	Country	Postal Code	Market	Region	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Discount	Profit	Shipping Cost	Order Priority
32298	CA-2012-124891	31-07-2012	31-07-2012	Same Day	RH-1949	Rick Hai	Consum	New Yor	New Yor	United States	10024	US	East	TEC-AC-1000303	Technology	Accessories	Plantronics CS510 - Over-t	2309,65	7	0	762,1845	933,57	Critical
26341	IN-2013-77878	05-02-2013	07-02-2013	Second Class	JR-1621	Justin Ri	Corpora	Vollong	New Sou	Australia		APAC	Oceania	FUR-CH-10003950	Furniture	Chairs	Novimex Executive Leathe	3709,395	9	0,1	-288,765	923,63	Critical
25330	IN-2013-71249	17-10-2013	18-10-2013	First Class	CR-1273	Craig Re	Consum	Brisban	Queensl	Australia		APAC	Oceania	TEC-PH-10004664	Technology	Phones	Nokia Smart Phone, with D	5175,171	9	0,1	919,971	915,49	Medium
13524	ES-2013-157930	28-01-2013	30-01-2013	First Class	KM-1633	Katherin	Home O	Berlin	Berlin	Germany		EU	Central	TEC-PH-10004589	Technology	Phones	Motorola Smart Phone, C	2892,51	5	0,1	-96,54	910,16	Medium
47221	SG-2013-4320	05-11-2013	06-11-2013	Same Day	RH-9495	Rick Hai	Consum	Dakar	Dakar	Senegal		Africa	Africa	TEC-SHA-10000501	Technology	Copiers	Sharp Wireless Fax, High-S	2832,96	8	0	311,52	903,04	Critical
22732	IN-2013-42360	28-06-2013	01-07-2013	Second Class	JM-1565	Jim Mitc	Corpora	Sydney	New Sou	Australia		APAC	Oceania	TEC-PH-10000301	Technology	Phones	Samsung Smart Phone, wi	2862,675	5	0,1	763,275	897,35	Critical

Figura 8: Excerto do cabeçalho do *dataset* do grupo.

3.2 Análise dos dados

Antes de começar o pré-processamento dos dados, foi necessário realizar uma análise geral do *dataset* em mãos – Figura 8, de modo a ser possível definir um esquema eficiente para o tratamento dos dados e, posteriormente, aplicar algumas técnicas de *Machine Learning*, pondo em prática o conhecimento adquirido nas aulas teóricas e trabalhado nas aulas práticas.

Inicialmente foi efetuada uma exploração relativamente às features de *dtype objetc* através do uso da função **head()**. A Figura 9 apresenta o *output* gerado por esta, que permitiu concluir que um único pedido é distribuído em múltiplas linhas, uma vez que o número de valores únicos de *Order ID* é menor que o número total de linhas no dataframe. Para além disto, ainda foi possível inferir que as *features Ship Mode, Segment, Market, Region, Category, Sub-Category* e *Order Priority* são variáveis categóricas, dado que estas tem menos de 20 valores únicos num total de 51 mil valores. Por fim, também se observou que existem mais *Product ID* únicos que *Product Name*, inferindo, assim, que produtos diferentes têm o mesmo nome.

	Order ID	Ship Mode	Customer ID	Customer Name	Segment	City	State	Country	Postal Code	Market	Region	Product ID	Category	Sub-Category	Product Name	Order Priority
count	51290	51290	51290	51290	51290	51290	51290	51290	9994.0	51290	51290	51290	51290	51290	51251	51200
unique	25035	4	1590	795	3	3636	1094	147	631.0	7	13	10292	3	17	3781	4
top	CA-2014-100111	Standard Class	PO-18850	Muhammed Yedwab	Consumer	New York City	California	United States	10035.0	APAC	Central	OFF-AR-10003651	Office Supplies	Binders	Staples	Medium
freq	14	30775	97	108	26518	915	2001	9994	263.0	11002	11117	35	31273	6152	227	29386

Figura 9: Dados sobre as *object features*.

Na Figura 10 podemos observar o *output* da função **describe()** aplicado às *features* númericas, o que possibilitou obter informações estatísticas sobre as mesmas, nomeadamente, a sua média, desvio padrão e os valores extremos das mesmas, entre outras. Através deste *output*, constatou-se que a

feature Discount representa uma percentagem, uma vez que os seus valores se encontram entre 0 e 0.85. Além disto, também se verificou, a partir do valor máximo da *feature Quantity*, que a quantidade máxima de itens em cada linha é quartoze, no entanto, como a *feature Sales* apresenta uma grande variação de valores, podemos concluir que o preço dos produtos individuais é muito diversificado. Por último, foi ainda possível inferir informações relativamente à *feature target*, *Profit*, nomeadamente que também contém uma ampla gama de valores nominais.

	Sales	Quantity	Discount	Profit	Shipping Cost
count	51200.000000	51200.000000	51200.000000	51200.000000	51200.000000
mean	246.865006	3.476426	0.143026	28.639338	26.416522
std	487.908698	2.278923	0.212409	174.491125	57.338685
min	0.444000	1.000000	0.000000	-6599.978000	0.000000
25%	30.880000	2.000000	0.000000	0.000000	2.620000
50%	85.273800	3.000000	0.000000	9.255100	7.810000
75%	251.640000	5.000000	0.200000	36.841500	24.530000
max	22638.480000	14.000000	0.850000	8399.976000	933.570000

Figura 10: Dados sobre as *features* númericas.

A Figura 11 apresenta uma representação gráfica da distribuição das *features* categóricas através de gráficos de barras. A partir destes conseguimos observar e reconhecer tendências nos dados, concluindo que nenhuma das *features* está uniformemente distribuída. Como podemos ver pelo o gráfico da *feature Category*, a maioria das vendas está na categória de material de escritório, o que pode dificultar a aplicação dos modelos de aprendizagem automática.

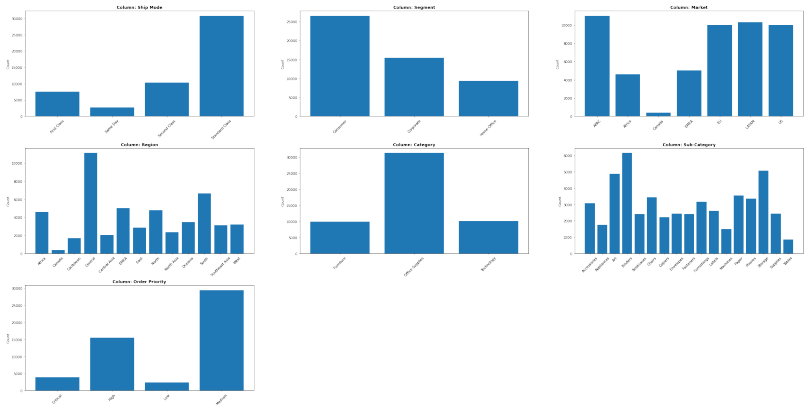


Figura 11: Distribuição dos dados das *features* categóricas.

Em relação à distribuição dos dados das features numéricas, como podemos ver pela Figura 12, as *features Profit, Sales* e *Shipping Cost* apresentam uma faixa de valores ampla. Porém, existe uma concentração de valores em torno de determinados valores. Assim sendo, removemos os outliers com o objetivo de conseguir observar melhor os dados.

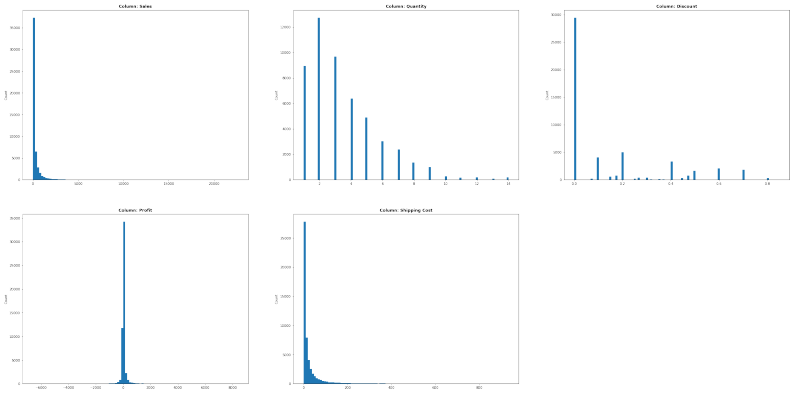


Figura 12: Distribuição dos dados das *features* numéricas.

A Figura 13 oferece uma represenção da dispersão dos dados das *feature Sales, Quantity, Shipping Cost, Discount* e *Profit*, tendo sido aplicado a mesma estratégia que nos gráficos anteriores de remover os outliers para obter uma melhor visualização dos gráficos. É possível observar que algumas *features* contém *outliers*. Perante isto, dependendo da informação que cada uma contém, foram analisadas estratégias a serem aplicadas.

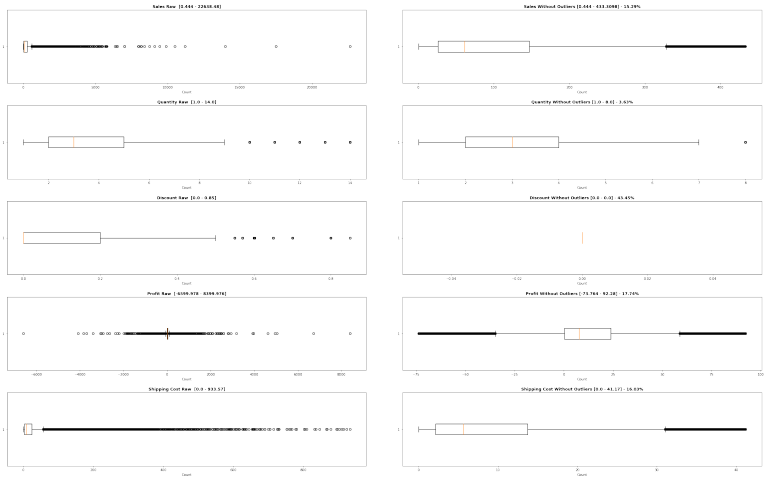


Figura 13: Dispersão dos dados das *features* numéricas.

Em seguida, foi realizada uma análise da correlação entre as variáveis com o objetivo de concluir quais *features* devíamos incorporar nos modelos. Como podemos ver pela figura **??**, temos uma grande correlação entre as *features Shipping Cost* e *Sales*.

Figura 14: Matriz de correlação

Após esta análise, o grupo procurou descobrir mais informação relativamente ao lucro de um venda. Deste modo, foi elaborado um gráfico que mostra a distribuição temporal do dataset, ou seja, o número de encomendas de cada mês, que se encontra representado na Figura 15. A partir deste podemos verificar que existem registos de encomendas desde Jan/2011 até Dez/2014, e ainda algumas *Ship Date* posteriores a Dez/2014 que podem resultar de alguma encomenda nos últimos dias de Dez/2014. Para além disto, é possível verificar que existe uma subida da tendência ao longo do tempo e um ciclo anual que se repete, com, por norma, picos nos meses de novembro e dezembro. Perante isto, de seguida fomos observar as tendências mensais que podemos ver pela Figura 17. Através deste gráfico, podemos concluir que, ao longo dos anos, o número médio de encomendas em cada mês é significativamente diferente. Por exemplo, podemos verificar que, em média, os meses de novembro/dezembro têm quase o dobro de encomendas que janeiro. A Figura 16 apresenta a percentagem de encomendas de cada ano, podendo concluir e confirmar, como visto no gráfico da figura 15, que foi no ano de 2014 que se realizaram mais encomendas online.

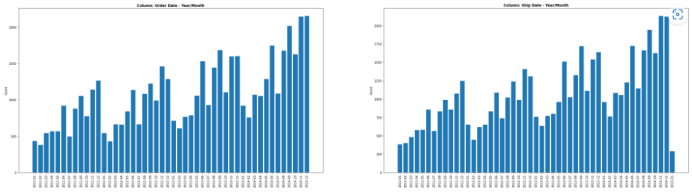


Figura 15: Distribuição Temporal.

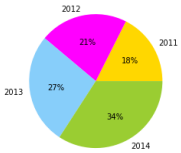


Figura 16: Encomendas por ano

Figura 17: Média de encomendas em cada mês

A Figura 18 apresenta o lucro obtido por categoria, sendo possível observar que a categoria da tecnologia foi a que obteve maior lucro. Por sua vez, a Figura 19 apresenta o lucro obtido por sub-categoria, concluindo que a sub-categoria *Copiers* é que contém um maior lucro e a sub-categoria *Tables* apresenta um lucro negativo.

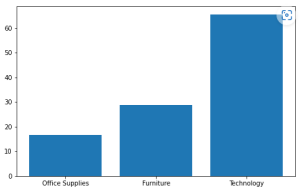


Figura 18: Lucro por categoria

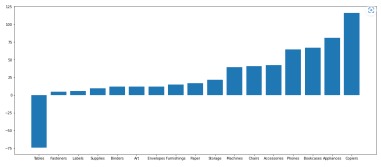


Figura 19: Lucro por Sub-Categoria

Na Figura 20 podemos observar quatro gráficos, um para cada ano, e cada um representa o lucro de cada mês durante esse ano. Por último, na análise dos dados, também foi verificado se existiam *missing values*, cujo tratamento é descrito no capítulo seguinte, linhas duplicadas (inexistentes) e a quantidade de elementos distintos em cada *feature*.

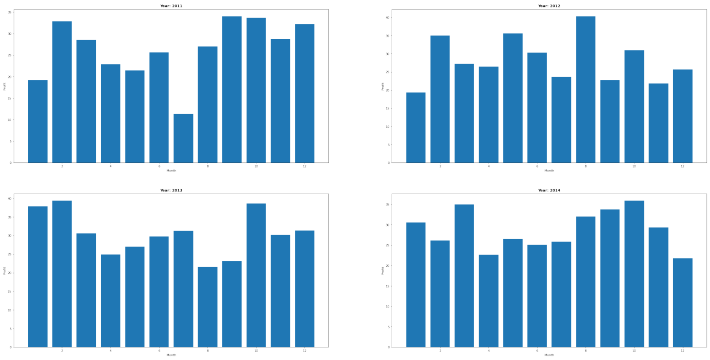


Figura 20: Lucro por mês de cada ano

Também decidimos analisar as entradas do dataset que continham *missing values* na coluna objetivo, e reparámos que todas são da subcategoria "Artes e Envelopes"e têm origem nos USA. Como eram poucas entradas, e não tinham a coluna objetivo, decidimos removê-las.

3.3 Tratamento de dados

Após a análise da informação do *dataset*, segue-se a tarefa de tratamento e limpeza dos dados. Tal como no *dataset* da competição, também para o *dataset* do grupo foram aplicadas as técnicas *Feature Selection*, *Feature Engineering* e *Label Enconding*. Assim sendo, a técnica *Feature Selection* levou à remoção das *features Product Name*, *Product ID*, *Customer Name*, *Customer ID*, permitindo diminuir a probabilidade de *overfitting* dos modelos. A *feature Postal Code* também foi removida devido ao facto de apresentar um elevado número de *missing values*, como visto durante a análise dos dados. Para além disto, o grupo não considerou que perdia informação ao eliminar a mesma, uma vez que o *dataset* contém informação sobre a cidade, o país, o estado e a região. Por fim, também na *feature Profit* foram eliminados os *missing values*, uma vez que estes correspondiam as linhas que dos missing values do *Postal Code*.

Em seguida, foi aplicada a técnica ***Feature Engineering*** relativamente às *features Order Date* e *Ship Date*, extraindo o dia, o mês e o ano de cada uma.

Adicionalmente, com a análise do gráfico da figura 12 relativamente à *Feature Discount,* verificou-se uma grande diversidade nos seus valores, pelo que a estratégia aplicada foi criar quatro intervalos de valores, nomeadamente, entre 0.0 - 0.25, 0.25 - 0.5, 0.5 - 0.75 e 0.75 - 1.0, como podemos observar na figura 22. Para além disto, após o estudo do gráfico da *feature Market* da figura 11, agrupou-se o Canadá aos EUA devido à sua significativa diferença relativamente ao resto dos elementos, como mostra a figura 21. Outra razão é a proximidade destes mercados muito, fazendo sentido associá-los.

Outra estratégica adotada foi o *Label-encoding* das colunas categóricas, para facilitar a aprendizagem dos modelos. É de realçar que as colunas *Ship Mode* e *Order Priority* foram convertidas de forma

a manter as relações entre os valores originais. Assim, uma entrada que tenha uma prioridade mais alta que outra no *dataset* original manterá essa relação no *dataset* transformado.

Seguidamente, também foram criadas duas variáveis binárias para a *feature Ship Mode*, uma vez que, pela a análise do gráfico desta na figura 11, conseguimos realçar um diferença entre o *Standard Class* e o resto dos elementos. Assim sendo, agrupou-se num bin o *Standard Class* e noutro bin o *First Class*, *Second Class* e *Same Day*, procurando obter uma uniformidade na proporção dos dados.

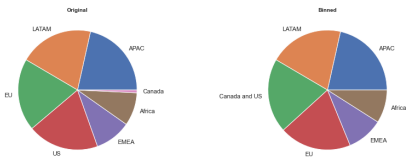


Figura 21: Market

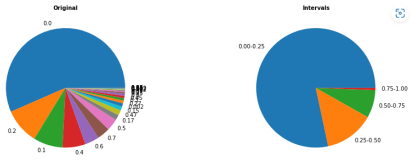
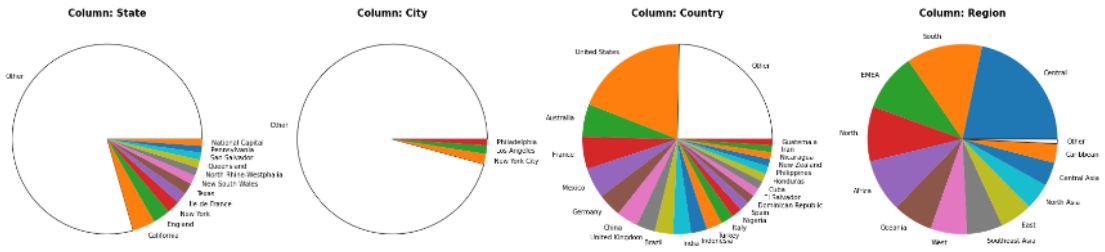


Figura 22: Discount

Após uma análise mais pormenorizada do *dataset*, reparámos que várias entradas estavam associadas a países pequenos, e este facto poderia aumentar o *overfitting* nos modelos. Os modelos poderiam decorar poucos casos associados a um país, e consequentemente não ter a capacidade de aprender casos novos. Por essa razão, os países com pouca representatividade, isto é, com menos de 1%, foram associados. Este tratamento de dados também foi realizado pelas mesmas razões para as colunas: *state*, *city*, *region*, e *sub-category*. A Figura 3.3 apresenta o antes e depois da aplicação desta estratégia.



Por último, considerámos interessante explorar como funcionaria o melhor modelo num *dataset* com tratamento de *outliers*. Deste modo, decidimos fazer esta análise para as colunas *Shipping Cost* e *Sales*, pois apresentam *outliers* mais afastados da maioria dos valores. As *features Quantity* e *Discount* descritas por valores numéricos também têm *outliers*, mas, visto que são valores dispersos, não os consideramos como *outliers*. Assim sendo, o tratamentos dos *outliers* só foi aplicado nas features *Shipping Cost* e *Sales*, deixando as outras colunas inalteradas. Para este tratamento, analisámos como ficaria o *dataset* se alterássemos os *outliers*, substituindo-os pela mediana, visto que esta é a medida menos suscetível a *outliers*, e apercebemo-nos que os dados ficariam demasiado alterados, como os gráficos seguintes o demonstram. Deste modo, a única estratégia aplicada foi manter e remover os *outliers*. Na Figura 23 podemos ver a dispersão dos dados após a susbtituição dos outliers pela mediana. Na Figura 24 podemos ver a dispersão dos dados após a remoção dos outliers.

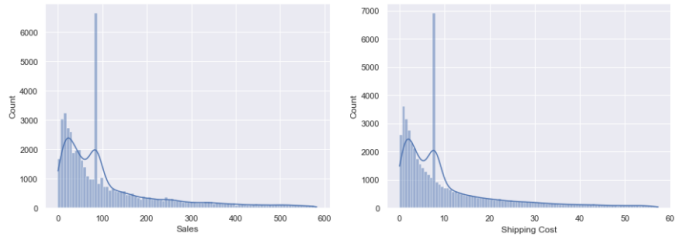


Figura 23: Distribução dos dados após o tratamentos dos outliers pela mediana

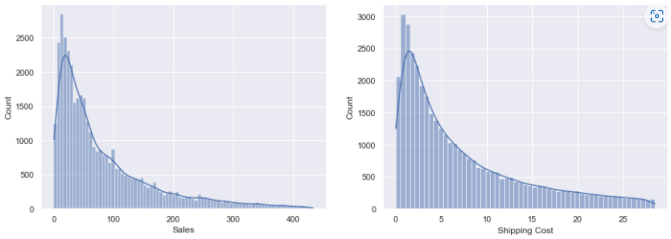


Figura 24: Distribuição dos dados após a remoção dos outliers

Por isso, decidimos remover as entradas que possuíam *outliers*, 30% do total.

Para além disto, nós também achamos que seria importante associar os países a algum dado que pudesse ajudar os modelos a prever. Para isso, decidimos usar um dataset que contêm o PIB de vários países nos anos que estamos a analisar. Assim, associamos a cada entrada o PIB do seu país.

Por fim, verificámos se com estas transformações haveria novas correlações nos dados, e através da análise da matriz de correlação pudemos concluir que não.

3.4 Modelação

3.4.1 Processo de seleção

Nós consideramos importante para a avaliação dos nossos modelos procurar uma métrica ajustada ao nosso objetivo. Por isso, decidimos penalizar mais os erros grandes (maiores que uma unidade), do que tratar todos os erros da mesma forma. Por essa razão, privilegiamos medidas que sejam ao quadrado, pois erros menores que 1 podem ser ignorados. Outro fator importante é ignorarmos se o lucro previsto é superior ou inferior ao correto. Por isso, escolhemos como critério o RMSE, porque os erros maiores são penalizados, e as ordens de grandeza entre esta métrica e os valores do *target* são mais aproximadas.

Para conseguirmos prever da melhor forma o lucro a partir do *dataset*, começamos por procurar diferentes modelos de regressão. Para todos eles, começamos por fazer um tratamento de dados genérico, que facilite a previsão dos modelos. A razão para esta preparação genérica é a seguinte: caso existam dados com escalas muito diferentes (por exemplo as colunas *sales* e *quantity*), ou com variações muito grandes, afeta de forma desigual alguns modelos (por exemplo modelos baseados em árvores versus redes neuronais). Assim, começamos por fazer normalização, escalar os valores, e transformações para ajustar os dados a uma distribuição normal, quando a sua distribuição tem *skew* superior a 0.01.

De seguida, testamos os modelos: LinearRegression, DecisionTreeRegressor, RandomForestRegressor, LinearSVR , XGBRegressor, e uma rede neuronal com 3 camadas. A escolha dos modelos foi baseada na variedade, para podermos explorar vários tipos, como árvores, redes neuronais, e Máquinas de Vectores de Suporte (utilizámos uma versão adequada a *datasets* grandes), e regressão linear.

3.4.2 Modelos escolhidos

Após experimentarmos todos os modelos, concluímos que os modelos baseado em árvores, e o *xgboost* (*ensemble* de árvores), obtiveram os melhores resultados, e por isso decidimos explorá-los mais. O modelo baseado em redes neuronais esteve logo a seguir, e por isso também decidimos melhorar o seu desempenho.

3.4.3 Redes neuronais

Antes de começarmos a implementar, começamos por procurar qual seria a melhor configuração para a nossa rede neuronal, e concluímos que 1 *hidden layer* seria suficiente, dada a complexidade

do problema. De seguida, procuramos alguns parâmetros que consideramos importantes variar para termos um melhor modelo, nomeadamente:

- Learning rate;
- Função de ativação;
- Número de neurónios no nível intermédio;

Esta análise dos modelos também foi feita com *k-fold cross validation*, mas dividimos apenas em 2 folds, dado o tempo que demorava a executar. Por fim, guardámos os hiperparâmetros que minimizavam o erro, e com eles comparamos a evolução do MSE no dataset de treino e de teste, ao longo de várias epochs. Essa evolução é apresentada em baixo, e concluímos que o modelo deve ter apenas 6 *epochs*, porque a partir desse valor não existe variação do erro, e o custo de computação do modelo começa a ser elevado, para não falar da possibilidade de *overfitting*. De seguida apresentamos a evolução do MSE associado ao número de epochs, com um modelo com 12 *hidden neurons* e uma *learning rate* de 0.01.

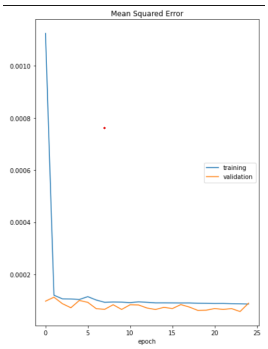


Figura 25: Evolução do MSE em função das epochs.

3.4.4 Árvores e xgboost

O tratamento de dados realizado para a DecisionTreeRegressor, o RandomForestRegressor e para o xgboost é igual, visto que todos utilizam modelos baseado em árvores. Nós procurarmos o tratamento de dados mais indicado para este tipo de modelos, e descobrimos que eles não são afetados por diferentes escalas nos dados de *input*. Logo, para tornar as nossas previsões mais rápidas decidimos não fazer um tratamento de dados específico para este conjunto de modelos. Assim, o tratamento de dados destes modelos é igual ao descrito no capítulo relativo ao tratamento de dados.

Para os três modelos, decidimos realizar as seguintes etapas:

- Realizar uma primeira modelação, e guardar o RMSE obtido;
- Procurar os hiperparâmetros mais adequados ao modelo, e fazer variá-los através da função *GridSearchCV*. Para a árvore de regressão alterámos a profundidade máxima e a função de *split*, para o *random forest* mudamos o número de estimadores e o número máximo de *features*, e para o xgboost altearmos o número de estimadores, a profundidade máxima e o *learning rate*;
- Comparamos com o resultado obtido no primeiro ponto, sendo que diminuía sempre, claro. Também analisámos os gráficos que apresentavam os resultados;

O modelo com melhores resultados foi o xgboost, e os hiperparâmetros que minimizaram o erro e ajustados por nós são os seguintes:

- Profundidade máxima = 2;
- Learning rate = 0.3;

- Número de estimadores = 120;

De seguida, apresentámos um gráfico que apresenta as previsões e os valores reais.

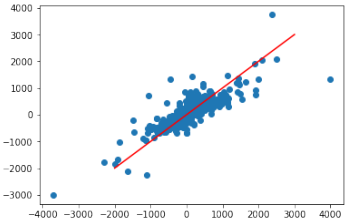


Figura 26: Valores reais vs. Valores previstos.

Por fim, os melhores resultados foram um RMSE de 92 unidades monetárias, demorando 2.5 segundos para treinar e prever o modelo.

Por fim, consideramos interessante analisar duas caraterísticas:

- Quais as categorias mais relevantes neste modelo para a previsão?
- Qual o impacto dos outliers na qualidade do modelo?

Relevância das categorias Para analisar a relevância das categorias, utilizámos *recursive feature elimination*, fazendo variar o número de colunas entre 5 e 23 (total). Para escolher o *input* mais adequado, procurámos minimizar o número de *features* e maximizar a precisão do modelo. Obtivemos esse ponto ideal com 7 colunas, porque mais colunas traziam poucos ganhos à qualidade do modelo. Sendo assim, as colunas mais relevantes na previsão são:

- Segment;
- City;
- State;
- Sub-Category;
- Sales;
- Quantity;
- Discount;
- Order_mounth;
- Ship_day;

Reparamos que estas colunas representam, de forma genérica, as informações que consideramos importantes para o modelo conseguir prever, nomeadamente: tipo de produto (Segment), local de venda (cidade e estado), quantidade de material vendido e data de compra (ignorando o ano).

Qual o impacto dos outliers na qualidade do modelo? Tendo em conta a dispersão dos valores na coluna *profit*, seria de esperar que ao remover as linhas com *outliers* nas colunas *Sales* e *Shipping Cost*, a previsão do modelo fosse muito superior, porque podem ser removidas entradas cujo *profit* seria também um *outlier*, o que acabou por acontecer. Assim, passamos de um RMSE de 93 para 23 unidades monetárias. As diferenças entre os valores previstos e reais encontra-se apresentada a seguir:

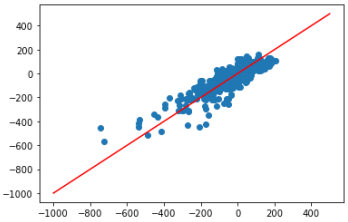


Figura 27: Valores reais vs. Valores previstos.

3.5 Conclusões sobre o trabalho realizado

Por fim, nós consideramos que o nosso modelo apresenta bons resultados, tendo em conta a distribuição dos dados do *dataset*. Consideramos que a distribuição dos valores de profit não é a melhor, e isso prejudica a qualidade das previsões, mas que, ainda assim, o modelo apresenta resultados interessantes. Como trabalho futuro, pensamos que seria interessante treinar os nossos modelos com outros datasets que apresentassem estas informações, e assim obtermos um modelo que tivesse aprendido com dados provenientes de várias fontes, o que poderia dar resultados mais fascinantes.