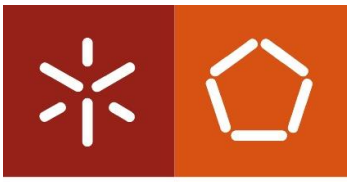
_Amostra



Universidade do Minho

Departamento de Informática

Dados e Aprendizagem Automática
4º ano - MEI

Trabalho Prático

Relatório de Desenvolvimento
Grupo 33

Ana Rita Guimarães (pg46991)
Leonardo Berteotti (pg37156) Rui Armada (pg50737)
Miguel Martins (a89584)

15 de janeiro de 2023

Resumo

O presente documento descreve sucintamente os objetos de avaliação e de análise ao longo de um projeto de análise inserido na unidade curricular Dados e Aprendizagem Automática. Este projeto teve como principais objetivos a análise, tratamento e previsão de dados de dois *datasets* distintos. Ao longo dos vários capítulos e secções presentes no relatório são expostas todas as decisões tomadas por parte da equipa de trabalho relativas aos métodos utilizados para o alcance do objetivo do projeto.

Conteúdo

Capítulo 1

Introdução

No âmbito da Unidade Curricular de Dados e Aprendizagem Automática, foi proposta a realização de um trabalho prático cujo objetivo consistiu no desenvolvimento de um modelo de *Machine Learning* através da utilização de modelos de aprendizagem que foram abordados ao longo do semestre em ambiente *Jupyter Notebook*.

Para a construção de uma solução para o presente trabalho prático foram utilizados dois *datasets*.

O primeiro *dataset* foi fornecido pela equipa docente da unidade curricular e contém dados referentes à quantidade e características dos incidentes rodoviários que ocorreram na cidade de Guimarães em 2021, cobrindo o período entre 01 de Janeiro de 2021 até 31 de Dezembro de 2021. O objetivo neste *dataset* consiste em elaborar modelos de previsão capazes de prever a intensidade de acidentes de trânsito na cidade de Guimarães.

Relativamente ao segundo *dataset*, foi necessário uma consulta e análise de vários *datasets* por parte da equipa de trabalho. Após várias pesquisas, foi escolhido um *dataset* que contém dados referentes ao valor de habitação nos EUA. Este *dataset* contém informação sobre várias habitações e como as suas qualidades influenciam o seu preço de compra. O objetivo para este *dataset* consistiu em desenvolver modelos de previsão do preço (de venda) das habitações.

Capítulo 2

Dataset 1 : Incidentes Rodoviários na Cidade de Guimarães

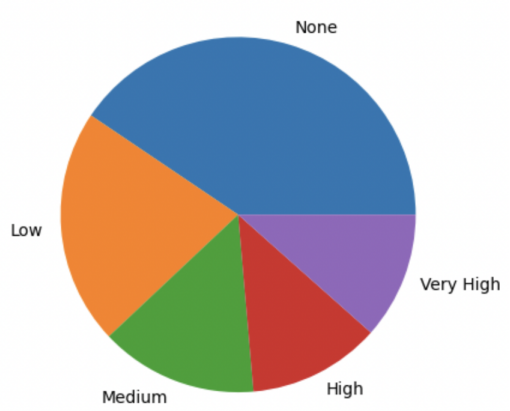
2.1 Análise do dataset

Inicialemente é realizada uma análise cuidada e extensiva do *dataset*. É importante referir que este *dataset* irá conter dados cruciais referentes ao trânsito na cidade de Guimarães durante um determinado período de tempo. Feito isto, verificou-se que o *dataset* inclui os seguintes atributos:

Atributo	**Descrição**
city_name	Nome da cidade em questão.
record_date	*Timestamp* do registo.
magnitude_of_delay	Magnitude do atraso provocado pelos incidentes.
delay_in_seconds	Atraso provocado pelos incidentes em segundos.
affected_roads	Estradas afectadas pelos incidentes.
luminosity	Nível de luminosidade que se verifica na cidade.
avg_temperature	Média da temperatura, para o *record_date*, na cidade.
avg_atm_pressure	Média da pressão atmosférica no *record_date*.
avg_humidity	Média da humidade no *record_date*.
avg_wind_speed	Média da velocidade do vento no *record_date*.
avg_precipitation	Valor médio de precipitação no *record_date*.
avg_rain	Avaliação quantitativa da precipitação no *record_date*.
incidents	Indicação acerca do nível de incidentes rodoviários que se verificam no *record_date*.

Tabela 2.1: Atributos do *dataset*

O objetivo deste *dataset* é descobrir qual é o atributo que será necessário utilizar para se conseguir prever a intensidade de acidentes de trânsito com a maior *accuracy* possível. Como tal, chegou-se à conclusão que o melhor atributo, para atingir este objetivo, é o ***incidents***.




```
Percentual Distribution of incidents
None         40.56
High         21.46
Low          14.36
Very High    12.06
Medium       11.56
```

Figura 2.1: Distribuição do atributo ***incidents***

Como se pode observar no gráfico anterior, o atributo é composto pelos parâmetros ***None***, ***Low***, ***Medium***, ***High*** e ***Very_High*** que definem a intensidade de incidentes. Estes parâmetros constituem a escala que será utilizada nas previsões.

Através do gráfico pode-se elaborar uma ideia dos incidentes que acontecem ao longo do dia. Por exemplo, a maior percentagem corresponde a ***None*** é de cerca de 40%, podendo concluir que existem bastantes períodos do dia em que não existe qualquer tipo de incidente. Já no caso do ***Very_High***, com percentagem de 12%, conclui-se que é referente a períodos do dia como no inicio da manhã e no final de tarde nos quais existe uma intensidade de trânsito mais elevada e uma maior probabilidade de incidentes.

Após a escolha do atributo mais ideal para prever a intensidade do incidentes, foi feito um estudo exaustivo de quais atributos possuem maior correlação com o atributo *target*, o ***incidents***. Dito isto, foram utilizados os métodos **corr()** para gerar a matriz de correlações do *dataset* e **seaborn.heatmap(...)** que permitiu a visualização da mesma.

Figura 2.2: Matriz de Correlação do *Dataset1*

2.2 Tratamento de dados

Como referido previamente, para a construção deste modelo de *Machine Learning*, foi necessário realizar uma análise crítica ao conjunto de dados fornecidos pelo *dataset*, determinando assim quais seriam os dados a utilizar, quais teriam de ser alterados e quais não seriam utilizados.

Para isto, foram analisados múltiplos pairplots para identificar relações entre os diversos atributos.

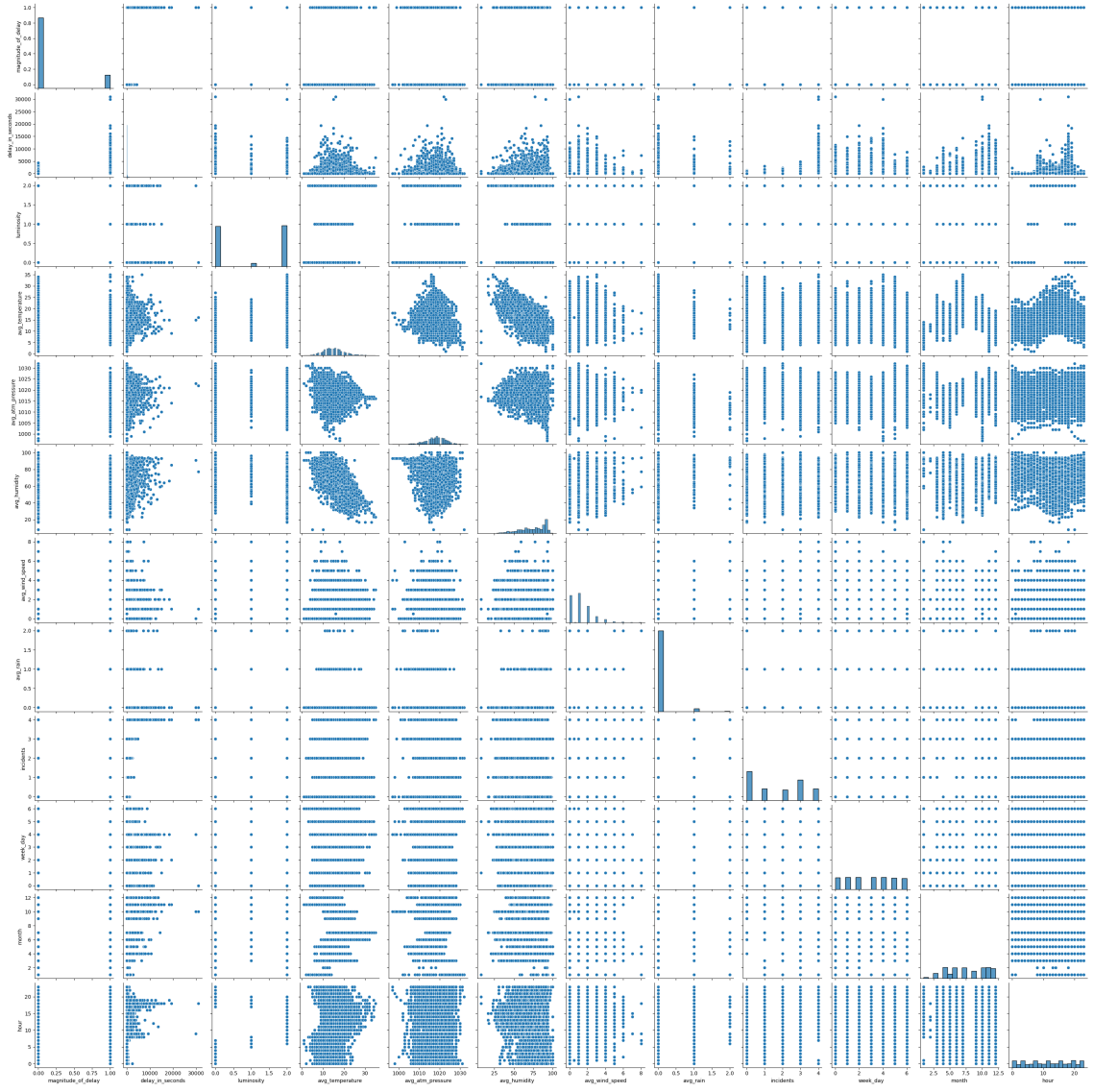


Figura 2.3: Pairplots Dataset 1

Feita esta análise, começou-se pela fase de tratamento do *dataset* que consistiu inicialmente em dar parse à *string* ***"record_date"*** de maneira a extrair o dia da semana, o mês e a hora do dia e separá-las em colunas destintas.

A luminosidade é um fator que afeta a segurança de condução e, por consequência, a velocidade máxima aconselhada. Desta forma foi considerado que este atributo deveria ser incluído no modelo que irá ser posteriormente criado. Para se poder utilizar os valores deste atributo foi aplicado um map que troca os valores {'DARK', 'LOW_LIGHT', 'LIGHT'} por {0, 1, 2}, respetivamente.

String	Inteiro
Dark	0
Light	1
Low_Light	2

Tabela 2.2: Transformação dos valores da coluna ***luminosity***

Também foi necessário tratar os *missing values* nas tabelas ***"avg_precipitation"*** e ***"affected_roads"*** que contém valores como "","considerados também *missing values*. A estratégia adotada para o primeiro foi remover a coluna, enquanto que no segundo foi remover as entradas que continham valores em falta.

De seguida, procedeu-se à transformação das restantes variáveis categóricas em inteiros de forma a conseguir prever a variável alvo.

String	Inteiro
UNDEFINED	0
MAJOR	1

Tabela 2.3: Transformação dos valores da coluna ***magnitude_of_delay***

Para a coluna das ***affected_roads*** utilizou-se o número total de estradas presente na listagem, portanto numa lista com 20 estradas o valor desta entrada passaria para 20. Assim conseguimos quantificar quantas estradas foram afetadas quando foram recolhidos os dados.

```
ar_list = data.affected_roads.str.split(",").map(lambda x:
                                                 len(list(filter(None, x)))
                                                )

data['affected_roads'] = ar_list
```

Figura 2.4: Processamento da coluna *affected_roads*

Para além disto, removeram-se as colunas consideradas desnecessárias para a previsão da intensidade de incidentes ou que continham muitos valores em falta - ***"city_name"***, ***"record_date"***, ***"avg_precipitation"*** e ***"avg_rain"***.

Pare este efeito, foi desenvolvida uma classe de *Python* que centraliza todo o processamento de dados previamente referido e pode ser aplicada aos conjuntos de dados de teste e de treino sem duplicar a quantidade de código desenvolvido permitindo também uma melhor legibilidade na leitura do código.

```python
class IncidentData:
    def process(filename, encoding) -> pandas.DataFrame:
        # Load the csv files
        data = pandas.read_csv('../../Datasets/Incidents/' + filename,
                               encoding=encoding,
                               keep_default_na=False)

        data.columns = data.columns.str.lower()

        data['record_date'] = data.record_date.map(lambda x : parser.parse(x))
        data['week_day'] = data.record_date.map(lambda x : x.weekday())
        data['day'] = data.record_date.map(lambda x : x.day)
        data['month'] = data.record_date.map(lambda x : x.month)
        data['hour'] = data.record_date.map(lambda x : x.hour)
        data.drop('record_date', inplace=True, axis=1)

        data.luminosity = data.luminosity.map({'DARK': 0, 'LOW_LIGHT': 1, 'LIGHT': 2})
        if 'incidents' in data.columns:
            data.incidents = data.incidents.map({'None':0,
                                                 'Low':1,
                                                 'Medium':2,
                                                 'High':3,
                                                 'Very_High':4})

        data['avg_rain'].fillna(0, inplace=True)

        data.avg_rain = data.avg_rain.map({'Sem Chuva':0,
                                           'chuva moderada':1,
                                           'chuva fraca':2,
                                           'chuva forte': 3})

        data['magnitude_of_delay'] = data['magnitude_of_delay'].map({'UNDEFINED': 0,
                                                                     'MAJOR': 1})

        ar_list = data.affected_roads.str.split(",").map(lambda x:
                                                         len(list(filter(None, x)))
                                                        )

        data['affected_roads'] = ar_list

        data.drop('city_name', inplace=True, axis=1)

        data.drop('avg_precipitation', inplace=True, axis=1)

        data.drop('avg_rain', inplace=True, axis=1)

        data.drop('month', inplace=True, axis=1)

        return data
```

✓ 0.3s

Figura 2.5: Classe de processamento de dados

2.3 Modelos desenvolvidos e resultados obtidos

Após uma pesquisa e extensivas considerações de possíveis modelos a serem utilizados para a formulação das previsões da variável alvo concluiu-se que o `Gradient Boosting` e o `XGBoost` eram os candidatos mais adequados para este problema visto que são modelos capazes abordar problemas de natureza regressiva e classificativa.

2.3.1 Versão Inicial

Numa primeira consideração do problema proposto, foi desenvolvido um modelo que realizava as suas previsões com recurso aos modelos `Gradient Boosting` e `XGBoost`. Após o processamento dos dados de teste e de treino utilizaram-se ambos os modelos para efetuar previsões sobre a variável *incidents* e feitas essas previsões, foi efetuada uma estimativa entre ambos os modelos de forma a gerar previsões que seriam colocadas no ficheiro de submissão.

```
predictions = (xg_test + gb_test) / float (2)

pred = numpy.ndarray.round(predictions)
pred[pred==0.] = 0
```

Após estimativa feita, formatou-se a *prediction* da seguinte maneira:

```
teste = predict_test
teste = numpy.where(teste ==0, "None", teste)
teste = numpy.where(teste =="1", "Low", teste)
teste = numpy.where(teste =="2", "Medium", teste)
teste = numpy.where(teste =="3", "High", teste)
teste = numpy.where(teste =="4", "Very_High", teste)
```

Feito isto, foi construido um `CSV` que contém uma coluna `RowID` e a coluna `Incidents` que possuem as previsões geradas por esta primeira versão do modelo de aprendizagem.

Esta versão gerou imensos problemas dado que esta versão gerou valores totalmente anómalos e muito afastados da realidade para conseguir formular uma previsão correta da variável ***incidents***, atingindo assim, nesta primeira versão, uma pontuação de 0.20221%.

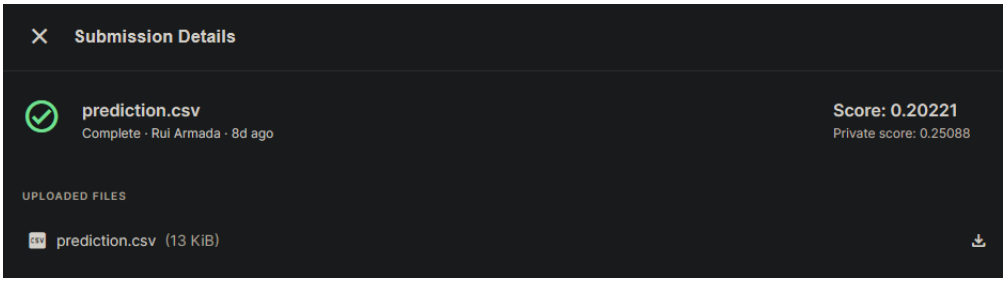


Figura 2.6: Previsões da versão inicial do modelo de aprendizagem

2.3.2 Versão Final - XGBoost

Nesta versão do modelo de aprendizagem foi escolhido o *XGBoost* como modelo principal para a realização das previsões da variável alvo ***Incidents***. Este modelo foi optado por tirar partido do `Gradient Boosting`, que tem em conta o desempenho e a velocidade e constrói uma série de árvores de decisão de forma iterativa utilizando ao mesmo tempo técnicas capases de reduzir o *overfitting* do modelo. Após a realização de várias pesquisas e testes com diferentes combinações de parâmetros chegou-se ao seguinte modelo de previsão:

```
training_data = IncidentData.process('training_data.csv', 'iso-8859-1')
test_data = IncidentData.process('test_data.csv','iso-8859-1')
training_data["id"] = training_data.index + 1
test_data["id"] = test_data.index + 1
y_target = training_data.incidents.values
x = training_data.drop('incidents', axis=1)
y = training_data[['incidents']]
x_train = x.drop(['id'], axis=1)
x_test = test_data.drop(['id'], axis=1)
XGB = xgb.XGBClassifier(
                        n_estimators=2500,
                        max_depth=7,
                        learning_rate=0.01,
                        subsample=0.7,
                        colsample_bytree=1
                        )
XGB.fit(x_train, y_target)
model_xgboost = XGB.predict(x_train)
predict_test = XGB.predict(x_test)
teste = predict_test
teste = numpy.where(teste ==0, "None", teste)
teste = numpy.where(teste =="1", "Low", teste)
teste = numpy.where(teste =="2", "Medium", teste)
teste = numpy.where(teste =="3", "High", teste)
teste = numpy.where(teste =="4", "Very_High", teste)
pandas.DataFrame({"RowId":test_data["id"],"incidents": teste}).to_csv(
f'../../Predictions/Incidents/predict_test_sale_xgboost.csv',index=False)
```

Além disto foi elaborado um gráfico que permitiu uma melhor visualização das previsões geradas pelos diferentes testes efetuados ao longo do desenvolvimento da solução final do problema, que permitiu determinar que uma combinação de parâmetros seria melhor que a anterior e assim sucessivamente. Segue-se então o gráfico que foi gerado pelo modelo de aprendizagem:

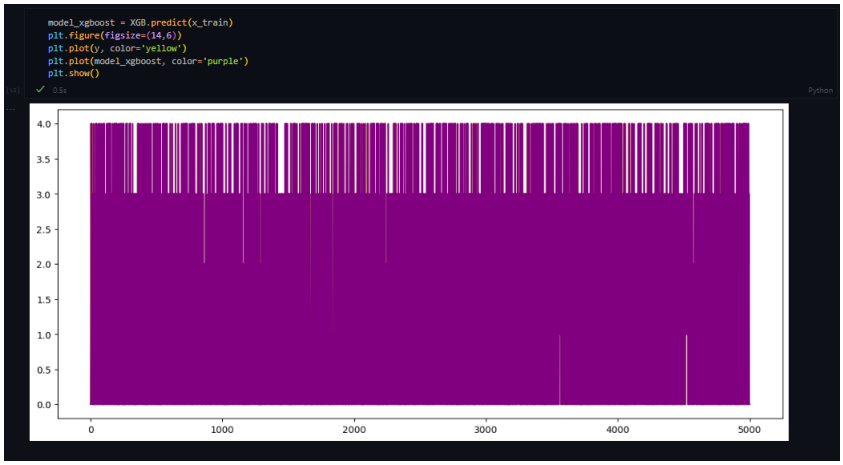


Figura 2.7: Previsões do modelo `XGBoost`

2.3.3 Resultados obtidos

No que toca aos resultados obtidos que mostraram uma melhor *accuracy score* foram os reultados do modelo `XGBoost`. Com a utilização de ambos os modelos, `Gradient Boosting` e `XGBoost`, foram obtidos *scores* muito indesejáveis visto que a utilização de apenas o modelo **XGBoost** mostrou que gerava melhores resultados de *score* quando submetidos na plataforma `Kaggle`. Este último modelo que foi desenvolvido obteve uma *accuracy score* de 0.87534%. Sendo que ainda foi efetuada uma submissão fora de horas do mesmo modelo, com uma ligeira alteração no tratamento da coluna *affected_roads,* que proporcionou um aumento da *accuracy score* para 0.88642%

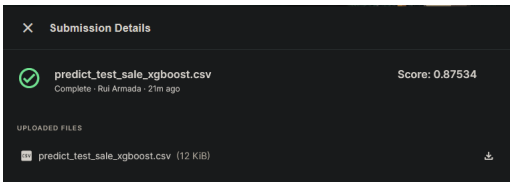


Figura 2.8: Resultados do modelo de aprendizagem

RowId	incidents
1	Very_High
2	None
3	None
4	Low
5	None
6	Very_High
7	Very_High
8	None
9	High
10	Low
11	None
12	Very_High
13	High

Figura 2.9: Amostra dos ficheiros `CSV` gerados

Capítulo 3

Dataset 2 : Valor de habitação nos EUA

3.1 Análise do dataset

Tal como no *dataset* anterior, foi realizada uma análise extensiva do conjunto de dados. É de notar que o *dataset* se refere a dados relativos a habitações nos EUA.

O objetivo deste *dataset* é descobrir qual é o atributo que será necessário utilizar para prever quais as casas são vendidas com a melhor *accuracy* possível. Para este efeito, chegou-se à conclusão que o melhor atributo para atingir este objetivo é o atributo ***SalePrice***. Os atributos existentes neste *dataset* são os seguintes:

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

Figura 3.1: Atributos do *Dataset2*

Após uma análise preliminar ao *dataset* foi determinado que este se trata de um problema de regressão do qual pretende efetuar uma previsão dos valores de habitação nos EUA. Com isto, foi

determinado que o atributo *SalePrice* sería aquele que permetiria uma melhor previsão, e consequentemente, resposta ao problema proposto.

Posteriormente foi realizado um estudo exaustivo dos atributos que possuem maior correlação com o atributo *target* do *dataset*, o ***SalePrice***. Para este efeito construi-se um matriz de correlações do *dataset* presentes na figura 3.2.

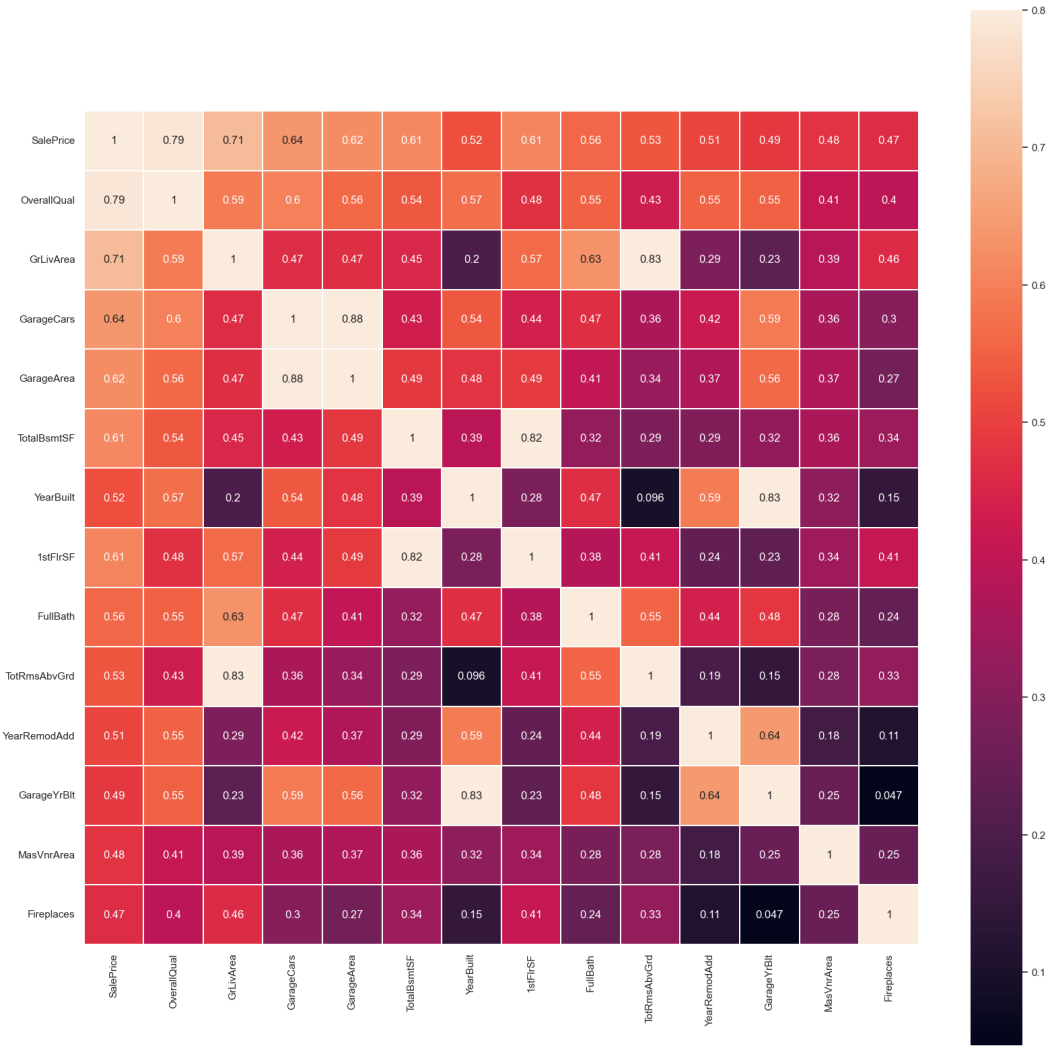


Figura 3.2: Matriz de Correlação do *Dataset2*

Feito isto, procedeu-se à elaboração de *scatter plots* para permitir a visualização dos dados, presentes na figura 3.3, das variáveis com maior correlação com a variável alvo.

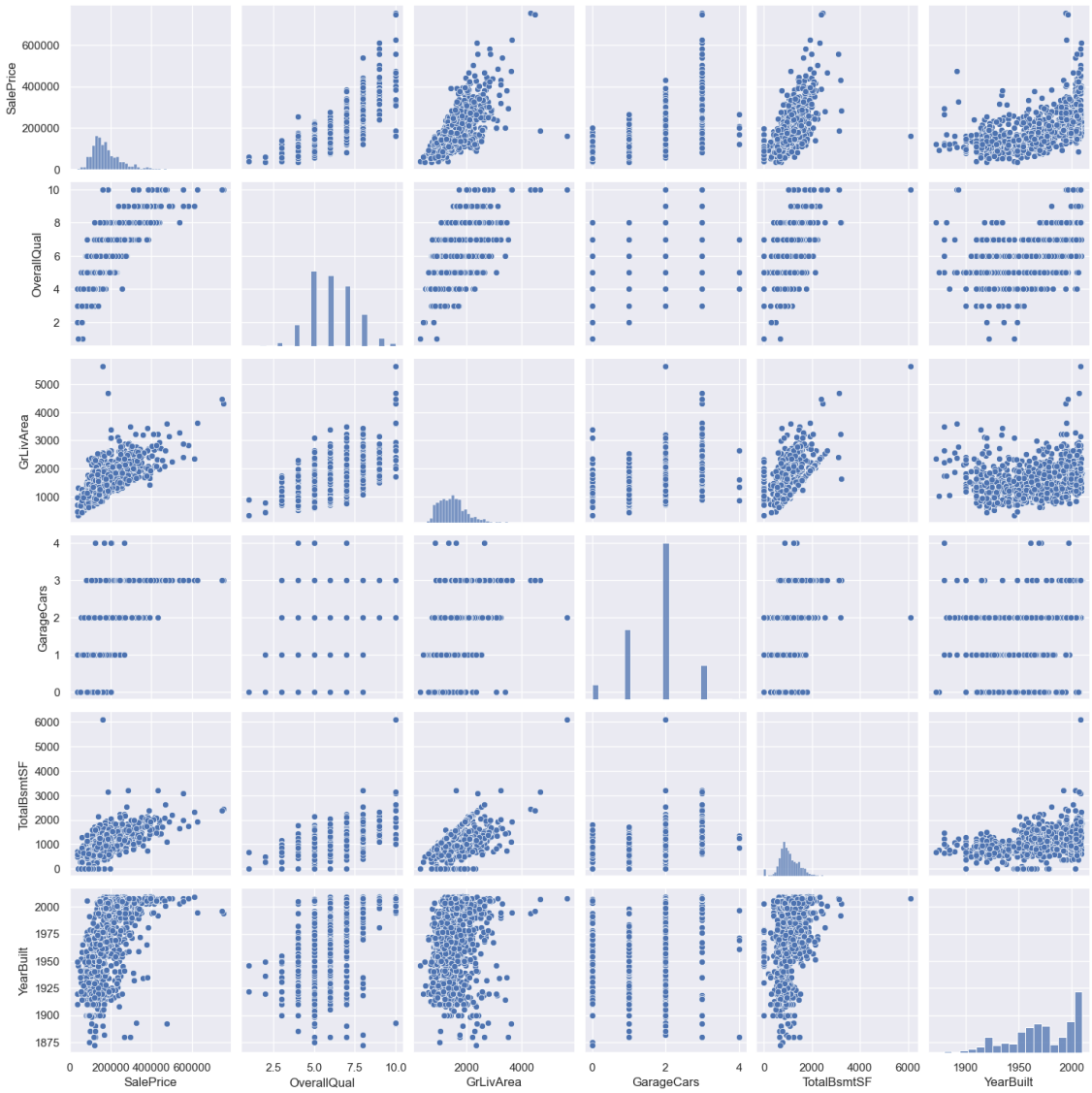


Figura 3.3: Scatter Plots

Concluiu-se que:

- As variáveis ***GrLivArea*** e ***TotalBsmtSF*** estão relacionadas linearmente com a target variable. Quando estas variáveis sofrem um aumento no seu valor, ***SalePrice*** também aumenta;
- As variáveis ***OverallQual*** e ***YearBuilt*** estão relacionadas positivamente com ***SalePrice***.

De seguida elaborou-se um *box plot*, visível na figura 3.4, para observar a relação entre ***OverallQual*** e ***SalePrice***.

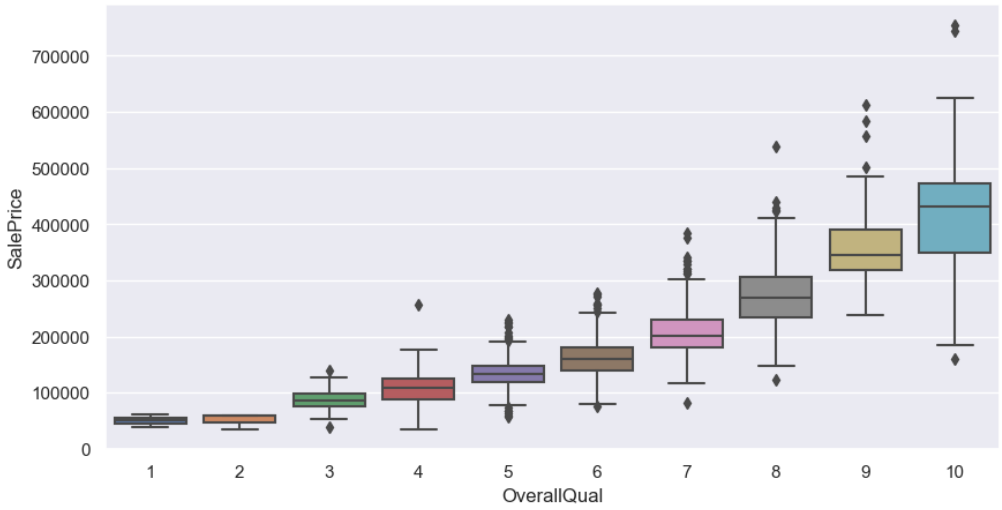


Figura 3.4: Box Plot

3.2 Tratamento de dados

Após uma análise ao conjunto de dados, começou o tratamento do *dataset* para a previsão do *target*. Inicialmente foi necessário tratar todos os *missing values* presentes no *dataset*, elaborando assim uma *bar plot*, figura 3.5, para obter uma representação visual da quantidade em percentagem de valores em falta para cada variável.

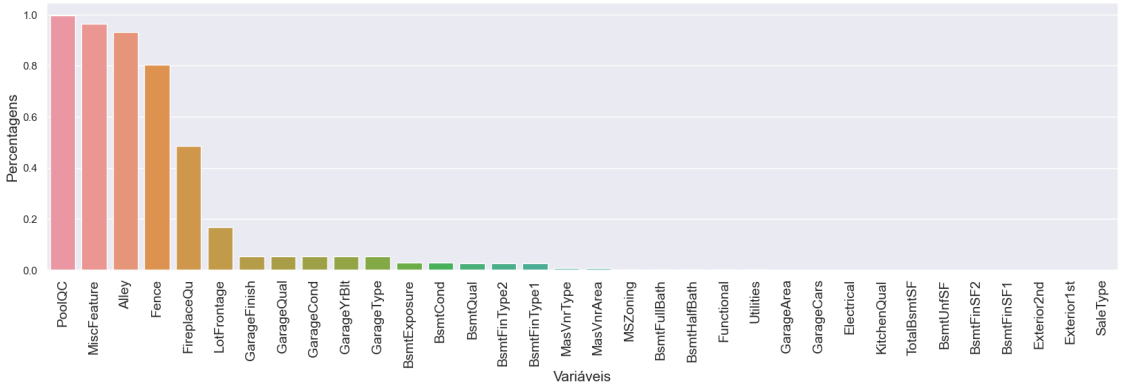


Figura 3.5: Bar Plot de Missing Values em %

- ***PoolQC***

  Uma vez que a maior parte das casas presentes no dataset não têm piscina (+/- 99%) , foi possível substituir os valores 'NULL' por, por exemplo, 'None'.

- ***MiscFeature***

  A maior parte das casas não têm "anotações especiais"(+/- 96%) substituir os 'NULL' values por 'None'.

- ***Alley***

  A maior parte das casas não têm "anotações especiais"(+/- 93%), novamente foi possível substituir os 'NULL' values por 'None'.

- ***Fence***

  A maior parte das casas não têm "anotações especiais"(+/- 80%), novamente foi possível substituir os 'NULL' values por 'None'.

- ***FireplaceQu***

  A maior parte das casas não têm "anotações especiais"(+/- 48%), novamente foi possível substituir os 'NULL' values por 'None'.

- ***LotFrontage***: Medição do comprimento de rua conectada à propriedade.

  Cerca de 17% dos valores são 'NULL'. Pode-se assumir que a distância entre a rua da propriedade em questão será a mesma da propriedade vizinha.

  Foi permitido então substituir os 'NULL' values pela mediana da variável ***LotFrontage*** no bairro onde se encontra inserida.

- ***GarageType***, ***GarageFinish***, ***GarageQual*** e ***GarageCond***

  Todas estas variáveis podem ser tratadas da mesma forma, uma vez que se tratam de variáveis categóricas relacionadas com garagem. Substitui-se novamente os valores 'NULL' por 'None'.

- ***GarageYrBlt***, ***GarageArea*** e ***GarageCars***

  Estas três variáveis podem ser tratadas também da mesma maneira, uma vez que se tratam de variáveis numéricas relacionadas com garagem. Pode-se substituir os valores 'NULL' por '0'.

- ***BsmtQual***, ***BsmtCond***, ***BsmtExposure***, ***BsmtFinType1*** e ***BsmtFinType2***

  Todas estas variáveis podem ser tratadas da mesma forma, uma vez que se tratam de variáveis categóricas relacionadas com Cave. Pode-se novamente substituir os valores 'NULL' por 'None'. 'None' vai indicar que a casa não possui qualquer tipo de Cave.

- ***BsmtFinSF1***, ***BsmtFinSF2***, ***BsmtUnfSF***, ***TotalBsmtSF***, ***BsmtFullBath*** e ***BsmtHalfBath***

  Todas estas variáveis podem ser tratadas da mesma maneira, uma vez que se tratam de variáveis numéricas relacionadas com Cave. Pode-se substituir os valores 'NULL' por '0'.

- ***MasVnrArea*** e ***MasVnrType***

  Estas duas variáveis estão relacionadas com acabamentos exteriores de alvenaria. Um valor 'NULL' nestas variáveis indica que a propriedade em questão não possui qualquer acabamento deste tipo. Uma vez que se tratam de valores numéricos, podem-se substituir estes valores por '0'.

- ***MSZoning*, *Utilities*, *Functional*, *Exterior2nd*, *Exterior1st*, *KitchenQual*, *Electrical* e *SaleType***

  Todas estas variáveis são categóricas. Cada uma possui menos de 5 valores 'NULL', pelo que se podem substituir pelo valor mais comum da variável em questão.

De seguida, procedeu-se à remoção dos *outliers* do *dataset* de forma a normalizar os dados visto que *outliers* teem um impacto muito significativo na performance do modelo porque estes distorcem a distribuição dos dados que acaba por piorar o desempenho do modelo. Este processo que pode ser visualizado nas figuras 3.6 e 3.7.

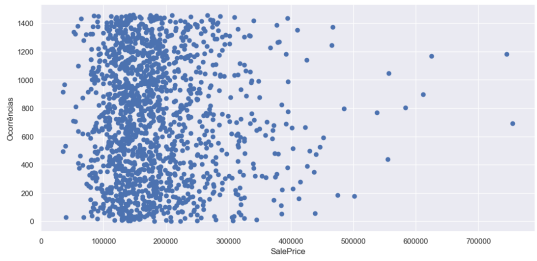


Figura 3.6: Antes do tratamento.

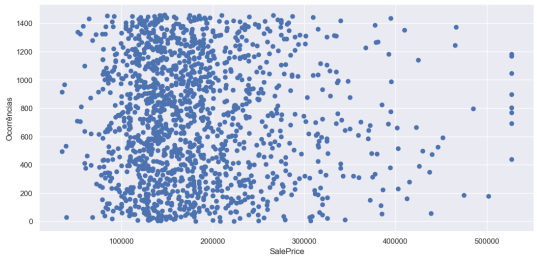


Figura 3.7: Depois do tratamento.

Para finalizar, procedeu-se à transformação de variáveis categóricas em valores numéricos uma vez que todos os modelos requerem que todos os *inputs*/*output* assim o sejam. Desta forma, com o objetivo de tentar obter os melhores resultados possíveis, utilizou-se o *pandas* para converter essas variáveis categóricas em *dummy-variables*.

```
all_data = pandas.get_dummies(all_data)
```

3.3 Modelos desenvolvidos e resultados obtidos

Neste *dataset* os modelos que foram escolhidos para efetuar a previsão foram o `Gradient Boosting` e `XGBoost` pelo que foi efetuada uma análise extensiva de quais parâmetros se iriam adequar de melhor forma ao problema proposto pelo *Dataset*.

No que toca ao modelo desenvolvido para responder ao problema de previsão dos preços de habitação nos EUA, foi implementado o mesmo sistema de treino usado no *Dataset 1*, mas com algumas alterações que passaram a ser explicadas. Neste modelo de aprendizagem utilizam-se ambos os modelos `XGBoost` e `Gradient Boosting` para efetuar previsões da variável alvo de forma a considerar qual dos modelos responderia de melhor forma ao problema. Feito isto são construidos dois gráficos que permitem observar melhor os resultados obtidos a partir dos dois modelos utilizados.

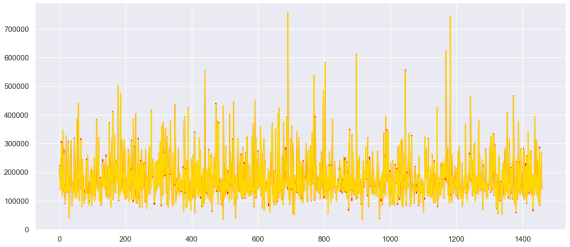


Figura 3.8: Resultados do modelo de aprendizagem *gradient boosting*

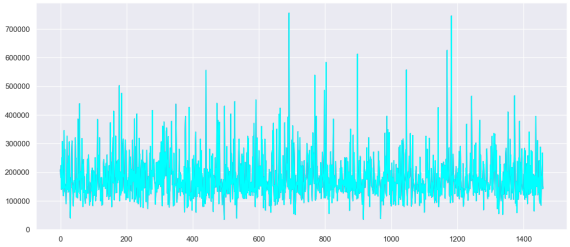


Figura 3.9: Resultados do modelo de aprendizagem *XGBoost*

Após a construção destas representações gráficas dos resultados obtidos, são efetuadas previsões com os dados de teste (utilizando os dois modelos previamente mencionados) de forma a quando se geram os ficheiros `CSV` sejam construidos quatro ficheiros distintos contendo as previsões geradas pelo modelo de aprendizagem.

```
models_train = {"gradient_boosting":boost_train, "xgboost":xgb_train}
models_test = {"gradient_boosting":boost_test, "xgboost":xgb_test}

for key,value in models_train.items():
    pandas.DataFrame({"Id":train["Id"],"SalePrice":boost_train}).to_csv(f'../../Predictions/Housing/predict_train_house_{key}.csv',index=False)

for key,value in models_test.items():
    pandas.DataFrame({"Id":test["Id"],"SalePrice":boost_test}).to_csv(f'../../Predictions/Housing/predict_test_house_{key}.csv',index=False)
```

Figura 3.10: Construção dos ficeiros CSV

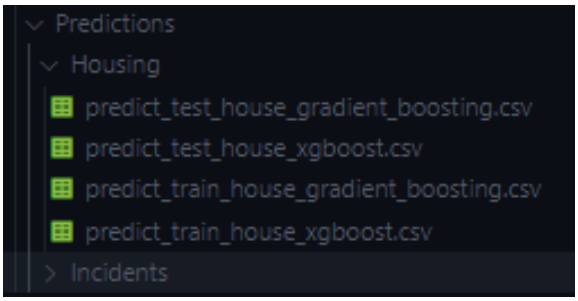


Figura 3.11: Ficheiros Gerados

Id	SalePrice
1	207727.097280484
2	181369.95247885025
3	219585.76822686754
4	141829.76758816882
5	251222.37739705018
6	143354.52467190078
7	302939.8546867039
8	201547.2300006865
9	129312.53080848072
10	118255.03736896101
11	125898.7682487867
12	346561.6249828043
13	141434.7105438862

Figura 3.12: Amostra dos ficheiros `CSV` gerados

Capítulo 4

Conclusão

Com a elaboração deste trabalho prático foi possível ter uma perspetiva perto de um contexto real de como os *datasets* têm de ser tratados de modo a produzirem bons resultados. Também se observou como esse tratamento se consiste num dos passos mais importantes na área de *Machine Learning*.

Foi também possível consolidar várias formas de utilização de modelos, assim como a importância da escolha deste para um determinado *dataset*. Esta escolha juntamente com o tratamento correto do *dataset* pode melhorar consideravelmente a exatidão(*accuracy*) dos dados previstos. Também estamos conscientes que podiamos obter melhores resultados aplicando várias outras técnicas, como a utlização do método `train_test_split` para treinar e testar a accuracy dos diferentes modelos, ou como a utilização do método `get_dummies` no Dataset 1, pos poderíamos ter uma melhor representação dos valores categóricos.

Em suma, este trabalho prático foi crucial para consolidar e aplicar os vários tópicos e conceitos abordados e adquiridos na unidade curricular Dados e Aprendizagem Automática através de uma análise geral da solução implementada, pode-se dizer que a equipa de trabalho foi capaz de atingir todos os objetivos inicialmente estabelecidos.