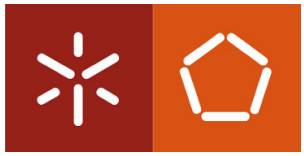
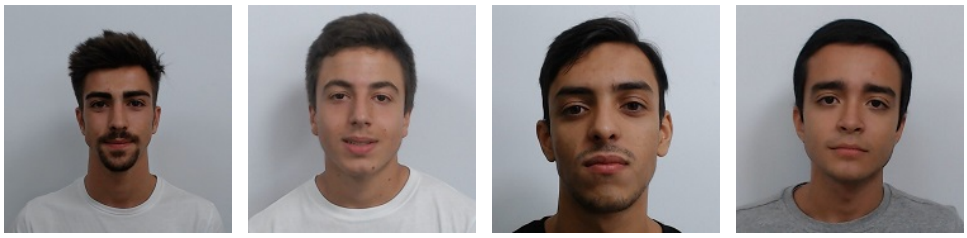


Dados e Aprendizagem Automática

Conceção e otimização de modelos de Machine Learning

Grupo 38

Bruno Filipe de Sousa Dias PG47068
Luís Enes Sousa A89597
Luis Filipe Cruz Sobral A89474
Pedro Miguel de Soveral Pacheco Barbosa PG47577



2 de Janeiro de 2022

Conteúdo

1 Introdução

No presente trabalho prático, foi proposto o desafio de explorar e tratar Datasets, analisando e transformando os seus dados. É ainda proposto que se construam modelos de previsão de resultados baseados em algoritmos de Mahcine Learning estudados ao longo do semestre. Dessa forma o projeto irá incidir em dois Datasets, um deles escolhidos pela equipa docente onde será estudo o trânsito rodoviário numa cidade Portuguesa e o segundo um Dataset escolhido pelo grupo, onde irá ser estudado o desempenho dos alunos dependendo da sua vida pessoal e da sua família.

Dessa forma o seguinte relatório irá descrever todos os passos tomados na concepção deste projeto. Iremos inicialmente perceber quais os domínios e objetivos de cada Dataset, bem como a maneira como achamos que iremos resolver o problema dos mesmos. Seguidamente iremos explicar qual a metodologia de trabalho escolhida pelo grupo. Após ser feita essa explicação, iremos dedicar finalmente um tempo exclusivo aos Datasets, onde iremos apresentar os mesmos, explicar como foi feita a sua exploração e ainda como foram tratados os seus dados. Logo após iremos relatar quais os modelos de Machine Learning construídos ao longo deste projeto e as escolhas que fizemos para cada um deles. Terminamos ainda com uma analise dos resultados obtidos, realizando uma introspecção e acrescentando sugestões para os Datasets estudados.

2 Domínios, Objetivos e Porposta de Resolução

Feita uma breve introdução do projecto, temos de perceber inicialmente quais os domínios que iremos tratar, quais os objetivos do mesmo e ainda qual a proposta que surgiu para a resolução dos mesmos.

2.1 Dataset Trânsito

A modelação do fluxo de tráfego rodoviário é um conhecido problema de características estocásticas, não-lineares. Tem, contudo, aparecido na literatura um conjunto de modelos que demonstram um potencial assinalável neste tipo de previsões. Com isso em consideração, foi construído um dataset que contém dados referentes ao tráfego de veículos na cidade do Porto durante um período superior a 1 ano. O dataset cobre um período que vai desde o dia 24 de Julho de 2018 até ao dia 02 de Outubro de 2019. O objetivo do projeto será então perceber e desenvolver um modelo de Machine Learning capaz de prever o fluxo de trânsito num dado ponto temporal (neste caso na cidade do Porto). Assim, a equipa ira utilizar como metodologia de extração de conhecimento a CRISP-DM.

2.2 Dataset Students Performance

A maneira como uma pessoa percorre a sua vida e vive o seu dia a dia, pode muitas vezes ter impacto na sua carreira e no seu sucesso escolar. Dessa forma, é importante perceber quais os padrões que levam as pessoas a possuírem uma má performance a nível escolar, de forma a poder-se contrariar estes mesmos padrões, com o objetivo de conseguir o melhor aproveitamento escolar possível. Assim, o objetivo deste projeto é desenvolver um modelo capaz de prever a *performance* dos alunos (nos dados recolhidos, alunos dos USA) consoante a sua vida pessoal, dos seus pais, etnia, crenças, entre outros aspectos. O conjunto de passos percorrido para a realização deste projeto, respeitam também eles a metodologia CRISP-DM.

3 Metodologia Seguida

Para começarmos a realização deste projeto "com o pé direito", tivemos inicialmente de decidir qual a metodologia que iria ser utilizada por forma a extrair conhecimento. Esta metodologia consiste num conjunto de etapas e passos que um projeto de extracção de conhecimento tem de ultrapassar para resolver os diferentes problemas. Assim, a metodologia escolhida foi a CRISP-DM (**CR**oss **I**ndustry **S**tandard **P**rocess for **D**ata **M**ining).

Assim, e escolhida uma metodologia, temos de perceber como esta funciona e quais os passos que devemos seguir para o desenvolvimento de uma resposta. Desta forma, esta metodologia é composta por 6 etapas:

- **Business Understanding -** Processo em que verificamos quais são os objetivos do projeto de extração de conhecimento e a que tipos de respostas queremos chegar, de forma a definir com consistência o problema de extração de conhecimento.
- **Data Understanding -** Processo em que obtemos os dados necessários e verificamos se a sua qualidade é do agrado da equipa, bem como perceber com que características é que iremos trabalhar.
- **Data Preparation -** Processo em que preparamos os dados para mais tarde serem explorados, eliminando dados, acrescentando dados em falta, reformulando dados, etc.
- **Modeling -** Processo em que experimentamos diversas ferramentas de extracção de conhecimento, construindo modelos de KE que serão mais tarde testados.
- **Evaluation -** Processo em que comparamos os resultados obtidos através dos modelos construídos com os objetivos inicialmente arquitetados.
- **Deployment -** Processo em que colocamos o modelo construído em produção.

Este processo pode não ser muitas vezes (quase sempre) contínuo. Por vezes são tomados passos atrás de forma a conseguir obter uma melhor resposta par ao problema concebido. Aqui podem-se incluir casos em que o modelo não atinge objetivos propostos e temos de fazer uma nova e/ou diferente preparação dos dados, quando necessitamos de perceber novamente o objetivo do projeto após visualizarmos quais os dados com que a equipa irá trabalhar, entre outros casos.

4 Dataset Trânsito

Neste momento iremos abordar o Dataset fornecido pela equipa docente, que possui informaçãos relativas a um ponto temporal (cidade e data) de forma a conseguirmos prever o fluxo de trânsito.

4.1 Apresentação do Dataset

No presente Dataset, o objetivo será perceber e desenvolver um modelo capaz de prever o fluxo de trânsito num dado ponto temporal. Assim, no presente Dataset várias ruas da cidade do Porto foram agrupadas de maneira a obter uma métrica do estado do trânsito na cidade em si. Temos então um conjunto de dados composto por diversas *features* que nos darão indicação e nos irão permitir inferir informações através de um modelo sobre o fluxo de trânsito pretendido.

Face ao anteriormente descrito, recebemos então dois datasets (um utilizado para atividades de treino e outro para atividades de teste). No dataset de treino possuímos 6812 registos e 14 atributos e no de teste possuímos apenas 1500 registos e 13 atributos. Dado a queremos perver o fluxo de transito, esse é o atributo retirado no dataset de teste. Dessa forma, o dataset de teste não possui o atributo "AVERAGE_SPEED_DIFF".

Desta forma, os atributos presentes nos dois datasets (à exceção do atributo "AVERAGE_SPEED_DIFF"que apenas se encontra no dataset de treino) e a descrição dos mesmos são:

- **city_name -** Nome da cidade do registo.
- **record_date -** Data associada ao registo.
- **AVERAGE_SPEED_DIFF -** Diferença entre a velocidade máxima que os veículos podem atingir em cenários sem trânsito e a velocidade que realmente se verifica naquele ponto temporal. Quanto mais alto o valor, maior a diferença entre a velocidade máxima (sem trânsito) e a velocidade atual e quanto menor o valor, menor a diferença. Assim, se o valor for maior, existe mais trânsito, se o valor for menor existe menos trânsito e se o valor for nulo, não existe qualquer trânsito.
- **AVERAGE_FREE_FLOW_SPEED -** Valor médio da velocidade máxima que os carros podem atingir em cenários sem trânsito.
- **AVERAGE_TIME_DIFF -** Valor médio da diferença do tempo que se demora a percorrer um determinado conjunto de ruas. Quanto mais alto o valor maior é a diferença entre o tempo que demora para se percorrer as ruas e o que se deveria demorar sem trânsito, i.e., valores altos implicam que se está a demorar mais tempo a atravessar o conjunto de ruas.

- **AVERAGE_FREE_FLOW_TIME -** Valor médio do tempo que demora a percorrer um determinado conjunto de ruas quando não há trânsito.
- **LUMINOSITY -** Nível de luminosidade que se verificava no ponto temporal.
- **AVERAGE_TEMPERATURE -** Valor médio da temperatura para a "record_date"e cidade associadas ao registo.
- **AVERAGE_ATMOSP_PRESSURE -** Valor médio da pressão atmosférica para a "record_date"e cidade associadas ao registo.
- **AVERAGE_HUMIDITY -** Valor médio da humidade para a "record_date"e cidade associadas ao registo.
- **AVERAGE_WIND_SPEED -** Valor médio da velocidade do vento para a "record_date"e cidade associadas ao registo.
- **AVERAGE_CLOUDINESS -** Valor médio da percentagem de nuvens para a "record_date"e cidade associadas ao registo.
- **AVERAGE_PRECIPITATION -** Valor médio de precipitação para a "record_date"e cidade associadas ao registo.
- **AVERAGE_RAIN -** Avaliação qualitativa da precipitação para a "record_date"e cidade associadas ao registo.

4.2 Exploração do Dataset

Iniciemos o nosso processo da resolução do problema encontrado, através da exploração do Dataset. Aqui teremos a oportunidade de descobrir factos e extrair algum conhecimento sobre aqueles que serão os dados sobre os quais iremos trabalhar e os atributos e registos que irão ser estudados.

Começamos por verificar qual o tipo de dados e o tipo de atributos com os quais estamos a operar ("Data Types").

```
# Dataset Info
training.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6812 entries, 0 to 6811
Data columns (total 14 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   city_name                6812 non-null   object
 1   record_date              6812 non-null   object
 2   AVERAGE_SPEED_DIFF       6812 non-null   object
 3   AVERAGE_FREE_FLOW_SPEED  6812 non-null   float64
 4   AVERAGE_TIME_DIFF        6812 non-null   float64
 5   AVERAGE_FREE_FLOW_TIME   6812 non-null   float64
 6   LUMINOSITY               6812 non-null   object
 7   AVERAGE_TEMPERATURE      6812 non-null   float64
 8   AVERAGE_ATMOSP_PRESSURE  6812 non-null   float64
 9   AVERAGE_HUMIDITY         6812 non-null   float64
 10  AVERAGE_WIND_SPEED       6812 non-null   float64
 11  AVERAGE_CLOUDINESS       4130 non-null   object
 12  AVERAGE_PRECIPITATION    6812 non-null   float64
 13  AVERAGE_RAIN             563 non-null    object
dtypes: float64(8), object(6)
```

Figura 1: Data Types dos atributos a ser trabalhados

Assim, percebemos que os atributos são do seguinte tipo, ou seja, possuem os seguintes "Data Types":

- ***object*** **-** city_name, record_date, AVERAGE_SPEED_DIFF, LUMINOSITY, AVERAGE_CLOUDINESS e AVERAGE_RAIN.
- ***float64*** **-** AVERAGE_FREE_FLOW_SPEED, AVERAGE_TIME_DIFF, AVERAGE_FREE_FLOW_TIME, AVERAGE_TEMPERATURE, AVERAGE_ATMOSP_PRESSURE, AVERAGE_HUMIDITY, AVERAGE_WIND_SPEED e AVERAGE_PRECIPITATION.

Uma vez que sabemos com que tipo de atributos estamos a trabalhar, iremos então tentar obter informações individuais sobre cada atributo.

Decidimos então verificar inicialmente uma descrição geral do dataset e dos seus atributos. Neste momento iremos obter uma percepção quantitativa de todos os objetos com o "Data Type"*float64*. Aqueles considerados como qualitativos serão estudados mais tarde.

```
# Dataset Description
training.describe()
```

	AVERAGE_FREE_FLOW_SPEED	AVERAGE_TIME_DIFF	AVERAGE_FREE_FLOW_TIME	AVERAGE_TEMPERATURE	AVERAGE_ATMOSP_PRESSURE	AVERAGE_HUMIDITY	AVERAGE_WIND_SPEED	AVERAGE_PRECIPITATION
count	6812.000000	6812.000000	6812.000000	6812.000000	6812.000000	6812.000000	6812.000000	6812.0
mean	40.661010	25.637111	81.143952	16.193482	1017.388139	80.084190	3.058573	0.0
std	4.119023	33.510507	8.294401	5.163492	5.751061	18.238863	2.138421	0.0
min	30.500000	0.000000	46.400000	0.000000	985.000000	14.000000	0.000000	0.0
25%	37.600000	2.275000	75.400000	13.000000	1015.000000	69.750000	1.000000	0.0
50%	40.700000	12.200000	82.400000	16.000000	1017.000000	83.000000	3.000000	0.0
75%	43.500000	36.200000	87.400000	19.000000	1021.000000	93.000000	4.000000	0.0
max	55.900000	296.500000	112.000000	35.000000	1033.000000	100.000000	14.000000	0.0

Figura 2: Descrição geral do Dataset

Visualizando o gráfico encontrado, podemos começar a retirar algumas conclusões:

- No atributo AVERAGE_PRECIPITATION podemos verificar que todos os atributos possuem o valor 0, ou seja, a coluna terá sempre a mesma informação para qualquer registo, diminuindo a eficácia dos modelos.
- No atributo AVERAGE_TIME_DIFF possuímos uma média de aproximadamente 25 valores, um mínimo de 0 valores e um máximo de aproximadamente 296 valores. Isto poderá ser uma indicação de que este atributo possui alguns outliers (serão revistos mais tarde).

Assim, e face ao que foi visto na descrição anterior, comecemos por analisar o número de elementos distintos de cada atributo.

```
# Seeing how many types each atribute has
training.nunique()
```

```
city_name                   1
record_date              6812
AVERAGE_SPEED_DIFF          5
AVERAGE_FREE_FLOW_SPEED   225
AVERAGE_TIME_DIFF        1151
AVERAGE_FREE_FLOW_TIME    442
LUMINOSITY                  3
AVERAGE_TEMPERATURE        38
AVERAGE_ATMOSP_PRESSURE    43
AVERAGE_HUMIDITY           77
AVERAGE_WIND_SPEED         15
AVERAGE_CLOUDINESS          9
AVERAGE_PRECIPITATION       1
AVERAGE_RAIN               13
dtype: int64
```

Figura 3: Número de elementos distintos de cada atributo

Como podemos verificar, os atributos relativos a city_name e a AVERAGE_PRECIPITATION só possuem um tipo de elemento. No atributo respetivo a city_name esta informação era expectável, uma vez que no presente Dataset estamos unicamente a estudar o fluxo de trânsito relativo à cidade do Porto. Quanto ao atributo AVERAGE_PRECIPITATION, percebemos que o facto de só existir um elemento distinto (sendo ,como verificado anteriormente, o valor 0) se poderá dar ao facto de uma imperfeita recolha de dados ou da disposição de informação enganosa no Dataset em estudo. Estes dois atributos irão mais tarde ser removidos, dado a sua inutiliade (devido à sua singularidade), uma vez que poderão diminuir a eficácia dos modelos contruídos.

Uma vez que verificamos também na descrição anterior que podemos possuir outliers no atributo AVERAGE_TIME_DIFF, faremos então uma recolha mais detalhada dos dados deste atributo.

```
# Checking it there is outliers
ax1 = sns.boxplot(x=training['AVERAGE_TIME_DIFF'])
```

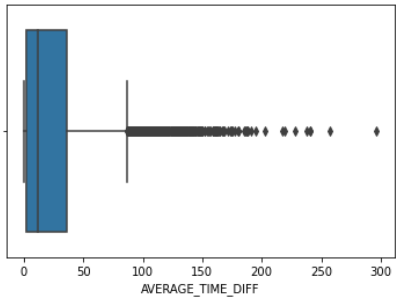


Figura 4: Boxplot do atributo AVERAGE_TIME_DIFF

Como podemos verificar pela boxplot construída, este atributo possui alguns outliers, sende eles considerados todos os valores acima de 200, onde os registos serão mais tarde removidos no tratamento de dados.

Por fim iremos ainda verificar se existem valores em falta nos diferentes atributos, utilizando recuros como somas de valores nulos e heatmaps.

```
# Seeing if there is any missing values
training.isna().any()
```

```
city_name                 False
record_date               False
AVERAGE_SPEED_DIFF        False
AVERAGE_FREE_FLOW_SPEED   False
AVERAGE_TIME_DIFF         False
AVERAGE_FREE_FLOW_TIME    False
LUMINOSITY                False
AVERAGE_TEMPERATURE       False
AVERAGE_ATMOSP_PRESSURE   False
AVERAGE_HUMIDITY          False
AVERAGE_WIND_SPEED        False
AVERAGE_CLOUDINESS         True
AVERAGE_PRECIPITATION     False
AVERAGE_RAIN               True
dtype: bool
```

Figura 5: Verificar se existem valores nulos nos atributos

```
# Seeing how many missing values are in each collumn
print(training.isna().sum())
```

```
city_name                    0
record_date                  0
AVERAGE_SPEED_DIFF           0
AVERAGE_FREE_FLOW_SPEED      0
AVERAGE_TIME_DIFF            0
AVERAGE_FREE_FLOW_TIME       0
LUMINOSITY                   0
AVERAGE_TEMPERATURE          0
AVERAGE_ATMOSP_PRESSURE      0
AVERAGE_HUMIDITY             0
AVERAGE_WIND_SPEED           0
AVERAGE_CLOUDINESS        2682
AVERAGE_PRECIPITATION        0
AVERAGE_RAIN              6249
dtype: int64
```

Figura 6: Soma de valores nulos de cada atributo

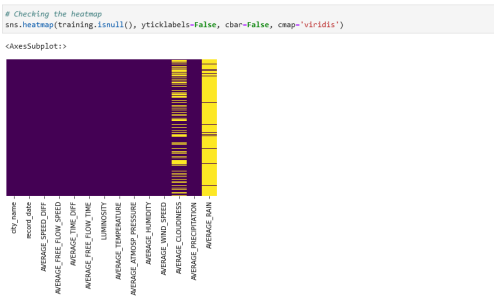


Figura 7: Heatmap de valores nulos

Através desta verificação, percebemos que existem valores em falta em dois atributos, sendo eles AVERAGE_CLOUDINESS e AVERAGE_RAIN. Estes atributos serão então tratados no tratamento de dados, decidindo se devemos preencher os valores em falta ou eliminar o atributo em questão.

4.3 Tratamento de Dados

Uma vez que exploramos os dados do Dataset, teremos de os tratar e/ou remover a informação inútil ou falsa que poderá comprometer o processo de previsão de resultados.

Começamos por realizar "Feature Engineering"no atributo relativo à data do registo, mais espeificamente, ao atributo "record_date". Assim, foram criados mais 4 atributos, sendo eles: YEAR (ano), MONTH (mês), DAY (dia) e ainda HOUR (hora). A criação destes atributos será uma mais valia para a previsão de resultados dos modelos construídos futuramente.

```
# Chaging record-date to DateTime and putting some new collumns
training.record_date = pd.to_datetime(training.record_date)
test.record_date     = pd.to_datetime(test.record_date)

training['YEAR'] = training.record_date.dt.year
test['YEAR']     = test.record_date.dt.year

training['MONTH'] = training.record_date.dt.month
test['MONTH']     = test.record_date.dt.month

training['DAY'] = training.record_date.dt.day
test['DAY']     = test.record_date.dt.day

training['HOUR'] = training.record_date.dt.hour
test['HOUR']     = test.record_date.dt.hour
```

Figura 8: Feature Engineering do atributo record_date

Após ser feito o Feature Engineering relatado anterior, passamos à remoção de outliers. Como podemos verificar anteriormente, o atributo AVERAGE_TIME_DIFF possui alguns outliers que poderão comprometer uma boa previsão de resultados e por isso serão removidos.

```
# Removing the outliers
training = training.loc[training['AVERAGE_TIME_DIFF']<=200]
```

```
# Checking it the outliers
ax1 = sns.boxplot(x=training['AVERAGE_TIME_DIFF'])
```

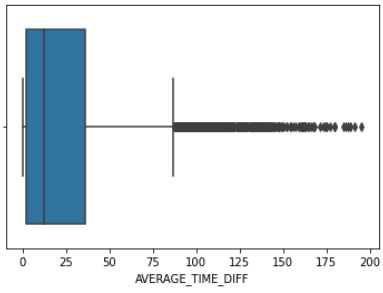


Figura 9: Remoção de outliers do atributo AVERAGE_TIME_DIFF

Neste momento, continuamos a possuir vários atributos com o Data Type object, ou seja, categóricos (ou no caso específico do atributo record_date do tipo datetime64). Uma vez que este tipo de atributos não poderão ser utilizados nos modelos, realizamos uma discretização de valores nominais. Assim, utilizamos especificamente Label Encoding para este processo.

```
# Changing qualitative classifiers to quantitative classifiers

training_luminosity_qt = {'LIGHT': 1, 'LOW_LIGHT': 2, 'DARK': 3}
training_cloudiness_qt = {'céu limpo': 1, 'céu claro': 2, 'nuvens dispersas': 3, 'algumas nuvens': 4, 'céu pouco nublado': 5, 'nuvens quebrados': 6, 'nuvens quebradas': 6, 'tempo nublado': 7, 'nublado': 8}
training_speed_diff_qt = {'None': 1, 'Low': 2, 'Medium': 3, 'High': 4, 'Very_High': 5}

training.replace(training_luminosity_qt, inplace = True)
training.replace(training_cloudiness_qt, inplace = True)
training.replace(training_speed_diff_qt, inplace = True)

test.replace(training_luminosity_qt, inplace = True)
test.replace(training_cloudiness_qt, inplace = True)
if method == 'FILLING_CLOUDINESS':
    test['AVERAGE_CLOUDINESS'] = test['AVERAGE_CLOUDINESS'].fillna(0)
```

Figura 10: Label Encoding dos valores nominais

Neste momento, possuímos ainda o problema de existirem valores em falta na coletânia de dados utilizados. Assim, poderemos optar por duas escolhas: remover o atributo ou preencher os valores em falta.

Como podemos perceber, o atributo AVERAGE_RAIN possui demasiados valores em falta e por isso iremos eliminar este atributo dos dados recolhidos. No entanto, no atributo AVERAGE_CLOUDINESS possuímos uma percentagem de valores em falta que pode ser controlada. Assim, o Utilizador possui a possibilidade de remover este atributo ou de preencher os seus valores em falta (escolha disponibilizada no Notebook). No caso de preencher os valores em falta, será utilizado o método **fillna()** utilizando *bfill*, seguido de uma interpolação com uma limit_direction de *forward*.

Para terminar o processo de tratamento de dados, são removidos todos os atributos que possuem unicamente um elemento distinto (city_name e AVERAGE_PRECIPITATION). Para além disso são também removidos todos os duplicados que possuem a mesma cidade (city_name) e a mesma data (record_date), uma vez que pode ser informação sobre a mesma hora e mesmo local, mas dados diferentes (o que é informação enganosa), ficando apenas o primeiro registo encontrado.

No seguinte Dataset foi ainda efetuado tratamento adicional de dados que foi eventualmente removido devido a *overfitting*. Neste tratamento removido encontra-se: remoção do atributo AVERAGE_ATMOSP_PRESSURE devido à equipa achar um atributo desnecessário; implementação do atributo HOLIDAY (indicação se é feriado ou não); normalização dos dados.

5 Dataset Students Performance

Neste momento iremos abordar o Dataset escolhido pelo grupo, que possui informações relativas a vários alunos e às suas vidas pessoais, por forma a perceber a sua *performance* a nivel escolar.

5.1 Apresentação do Dataset

No presente Dataset, o objetivo será perceber e desenvolver um modelo capaz de prever a *performance* dos alunos consoante a sua vida pessoal, dos seus pais, etnia, crenças, entre outros aspectos. Assim, no presente dataset foram recolhidos dados de diversos alunos com *backgrounds* bastantes distintos. Temos então um conjunto de dados composto por diversas *features* que nos darão indicação e nos irão permitir inferir informações através de um modelo sobre a performance dos alunos.

Face ao anteriormente descrito, recebemos então um dataset que possui 1000 registos e 8 atributos. Assim, este dataset será utilizado tanto como dataset de treino como de teste dos vários modelos construídos. Para isso, o grupo escolheu prever qual a nota obtida de cada aluno respetiva ao "writing score".

Desta forma, os atributos presentes no dataset e a descrição dos mesmos são:

- **gender -** Género do aluno(a).
- **race/ethnicity -** Etnia do aluno.
- **parental level of education -** Nível de escolaridade dos pais.
- **lunch -** Tipo de almoço que aluno come diariamente.
- **test preparation course -** Se aluno já completou ou não o teste de preparação de forma a otimizar a sua performance em testes padronizados.
- **math score -** Nota de Matemática.
- **reading score -** Nota de Leitura.
- **writing score -** Nota de Escrita.

5.2 Exploração do Dataset

Iniciemos o nosso processo da resolução do problema encontrado, através da exploração do Dataset. Aqui teremos a oportunidade de descobrir factos e extrair algum conhecimento sobre aqueles que serão os dados sobre os quais iremos trabalhar e os atributos e registos que irão ser estudados.

Começamos por verificar qual o tipo de dados e o tipo de atributos com os quais estamos a operar ("Data Types").

```
# Dataset Info
training.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64 
 6   reading score                1000 non-null   int64 
 7   writing score                1000 non-null   int64 
dtypes: int64(3), object(5)
```

Figura 11: Data Types dos atributos a ser trabalhados

Assim, percebemos que os atributos são do seguinte tipo, ou seja, possuem os seguintes "Data Types":

- ***object*** **-** gender, race/ethnicity, parental level of education, lunch e test preparation course.
- ***int64*** **-** math score, reading score e writing score.

Uma vez que sabemos com que tipo de atributos estamos a trabalhar, iremos então tentar obter informações individuais sobre cada atributo.

Decidimos então verificar inicialmente uma descrição geral do dataset e dos seus atributos. Neste momento iremos obter uma percepção quantitativa de todos os objetos com o "Data Type" *int64*. Aqueles considerados como qualitativos serão estudados mais tarde.

```
# Dataset Description
training.describe()
```

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

Figura 12: Descrição geral do Dataset

Visualizando o gráfico encontrado, não são detectadas quaisquer anomalias.

Passando agora a analisar o número de elementos distintos para cada atributo, percebemos que não existe nenhum atributo que possua apenas um elemento distinto, o que nos poupará tempo no tratamento de dados.

```
# Seeing how many types each atribute has
training.nunique()
```

```
gender                         2
race/ethnicity                 5
parental level of education    6
lunch                          2
test preparation course        2
math score                    81
reading score                 72
writing score                 77
dtype: int64
```

Figura 13: Número de elementos distintos de cada atributo

Por fim verificamos ainda se existem valores em falta nos diferentes atributos, o que se não acontecerá.

```
# Checking if there is any missing values
training.isna().any()
```

```
gender                         False
race/ethnicity                 False
parental level of education    False
lunch                          False
test preparation course        False
math score                     False
reading score                  False
writing score                  False
dtype: bool
```

Figura 14: Verificar se existem valores nulos nos atributos

5.3 Tratamento de Dados

Uma vez que exploramos os dados do Dataset, teremos de os tratar e/ou remover a informação inútil ou falsa que poderá comprometer o processo de previsão de resultados. Neste Dataset, a qualidade dos dados revelou-se superior e por isso, o tratamento de dados será mais simples do que no Dataset de trânsito.

Assim, começamos por fazer uma transformação dos valores categóricos. Uma vez que este tipo de atributos não poderão ser utilizados nos modelos, realizamos uma discretização de valores nominais. Assim, utilizamos especificamente Label Encoding para este processo.

```
# Changing qualitative qualifiers to quantitative qualifiers
training_gender_qt = {'male': 1, 'female': 2}
training_race_qt = {'group A': 1, 'group B': 2, 'group C': 3, 'group D': 4, 'group E': 5}
training_parentalEducation_qt = {'some high school': 1, 'some college': 2, 'high school': 3, "associate's degree": 4, "master's degree": 5, "bachelor's degree": 6}
training_lunch_qt = {'free/reduced': 1, 'standard' : 2}
training_testPreparation_qt = {'none': 1, 'completed': 2}

training.replace(training_gender_qt, inplace = True)
training.replace(training_race_qt, inplace = True)
training.replace(training_parentalEducation_qt, inplace = True)
training.replace(training_lunch_qt, inplace = True)
training.replace(training_testPreparation_qt, inplace = True)

test.replace(training_gender_qt, inplace = True)
test.replace(training_race_qt, inplace = True)
test.replace(training_parentalEducation_qt, inplace = True)
test.replace(training_lunch_qt, inplace = True)
test.replace(training_testPreparation_qt, inplace = True)
```

Figura 15: Label Encoding dos valores nominais

Além desta discretização dos valores nominais, o grupo deicidiu ainda fazer um binning dos dados relativos ao writing score (valor que queremos prever). Assim a equipa decidiu prever se a nota do aluno seria entre 1 e 5 (em vez de entre 0 e 100). Estes valores ordinais (de 1 a 5) correspondem à escala de notas americanas:

- **1 -** Corresponde a **F**, ou seja, valores 00 e 59.
- **2 -** Corresponde a **D**, ou seja, valores 60 e 69.
- **3 -** Corresponde a **C**, ou seja, valores 70 e 79.
- **4 -** Corresponde a **B**, ou seja, valores 80 e 89.
- **5 -** Corresponde a **A**, ou seja, valores 90 e 100.

Assim, o binning foi feito da seuinte forma:

```
# Changing the 0-100 notes to 1-5
training["writing score"] = pd.cut(training["writing score"], bins=[-1, 59, 69, 79, 89, 100], labels=[1, 2, 3, 4, 5])
```

Figura 16: Binning dos valores de writing score

Para terminar o processo de tratamento de dados, são removidos todos os atributos que possuem unicamente um elemento distinto (não existindo nenhum neste caso). Para além disso são também removidos todos os duplicados, uma vez ser informação redundante, ficando apenas o primeiro registo encontrado.

No seguinte Dataset não foi efetuado qualquer outro tratamento de dados eventualmente removido.

6 Modelos construídos

Uma vez explorados os Datasets, bem como tratados os dados respectivos a cada um deles, e seguindo a metodologia escolhida pela equipa (CRISP-DM), chegamos então à etapa de modelação, ou seja, de construção de modelos de previsão de resultados. Dessa forma, a equipa construiu os mesmos seis algoritmos de Machine Learning para ambos os Datasets e problemas em estudo. Dessa forma, necessitamos de apresentar quais as características, como e sobre que parâmetros foi realizado o tuning do modelo e ainda outros detalhes que sejam oportunos de fornecer.

6.1 Decision Tree Classifier

Uma Árvore de Decisão é um grafo hierarquizado, ou seja, uma árvore, em que cada nodo interno testa um atributo do dataset, onde cada ramo identifica um valor (ou conjunto de valores) do nodo testado e cada folha representa uma decisão.

Assim, os parâmetros utilizados foram os seguintes:

- **criterion -** gini
- **splitter -** best
- **max_depth -** None
- **min_samples_split -** 2
- **min_samples_leaf -** 1
- **min_weight_fraction_leaf-**0.0
- **max_features -** None
- **random_state -** 2021
- **max_leaf_nodes -** None
- **min_impurity_decrease -** 0.0
- **class_weight -** None
- **ccp_alpha -** 0.0

6.2 Linear Regression

A Regressão Linear tenta prever os valroes, colocando uma linha reta (função linear) pelos dados de modo a diminuir o erro total entre o valor real e o valor obtido para cada registo.

Assim, utilizamos os seguintes parâmetros:

- **fit_intercept -** True
- **normalize -** False
- **copy_X -** True
- **n_jobs -** None
- **positive -** False

6.3 Logistic Regression

A Regressão Logística permite-nos resolver problemas de Classificação, onde os valores em estudo são discretos. Dessa forma, utilizaremos os seguintes parâmetros:

- **penalty -** l2
- **dual -** False
- **tol -** 1e-4
- **C -** 1.0
- **fit_intercept -** True
- **intercept_scaling-** 1
- **class_weight -** None
- **random_state -** 2021
- **solver -** lbfgs
- **max_iter -** 10000
- **multi_class -** auto
- **verbose -** 0
- **warm_start -** False
- **n_jobs -** None
- **l1_ratio -** None

6.4 SVC

O algoritmo usa como recurso SVM (Support Vector Machines), um algoritmo de Machine Learning supervisionado que pode ser usado tanto para problemas de classificação como de regressão. A princiaal função é encontrar um hiperplano capaz de diferenciar as diferentes classes de um dado atributo. Assim, os parâmetros utilizados neste modelo foram:

- **C -** 1.0
- **kernel -** rbf
- **degree -** 3
- **gamma -** scale
- **coef0 -** 0.0
- **shrinking -** True
- **probability -** False
- **tol -** 1e-3
- **cache_size -** 200
- **class_weight -** None
- **verbose -** False
- **max_iter -** -1
- **decision_function_shape -**ovr
- **break_ties -** False
- **random_state -** 2021

6.5 GridSearchCV

O GridSearchCV utiliza como recurso o algoritmo SVC no nosso caso e criamos uma grid onde serão encontrados os melhores valores que nos fornecerão a melhor previsão de resultados possível. Assim, os parâmetros utilizados foram:

- **estimator -** SVC(random_state=2021)
- **param_grid -** {'C':[200], 'gamma':[0.0001], 'kernel':['rbf']}
- **scoring -** None
- **n_jobs -** None
- **refit -** True
- **cv -** None
- **verbose -** 3
- **pre_dispatch -** 2*n_jobs
- **error_score -** np.nan
- **return_train_score -** False

6.6 Artificial Neural Networks

Uma ANN ou Artificial Neural Network, é um sistema computacional baseado na conexão de neurónios para resolução de problemas. Este neurónios são unidades computacionais com capacidade de aprender e que se encontram conectadas entre si de forma a criar uma estrutura interconectada (ANN). Assim, para as diferentes camadas interiores (hidden layers) de neurónios possuímos no seguinte modelo Keras constrído com:

- **Units (Dataset Transito) -** 36, 20, 1
- **Units (Dataset Students) -** 16, 8, 1
- **Activation (Both Datasets) -** 'relu', 'relu', 'relu'

Tendo em conta que os datasets possuem um número de atributos diferentes, temos então para a primeira camada uma **input_dim** de **13** para o Dataset Trânsito e de **7** para o Dataset Students. Para compilação do modelo utilizamos inda os seguintes parâmetros:

- **loss -** 'mae'
- **optimizer -** tf.optimizers.Adam(learning_rate)
- **metrics -** ['mae', 'mse', 'accuracy']

Para acabar a explicação deste algoritmo, é importante realçar que o fit do modelo foi feito com:

- **epochs -** 20
- **batch_size -** 32

7 Resultados obtidos, apreciação critica, Sugestões e recomendações dos Datasets e dos seus resultados obtidos

Criados os modelos, chegamos à altura de os testar e portanto, os resultados obtidos foram os seguintes:

MODELO / DATASET	Dataset Trânsito	Dataset Students Performance
Decision Tree Classifier	0.72	0.65
Linear Regression	0.75	0.82
Logistic Regression	0.795	0.75
SVC	0.34	0.74
GridSearchCV	0.795	0.74
Artificial Neural Networks	-0.12 (average of r2 scores)	-0.14 (average of r2 scores)

Uma vez que apresentamos os valores relativos aos resultados obtidos para cada modelo e cada Dataset, vamos agora fazer uma apreciação crítica dos mesmos.

7.1 Dataset Trânsito

Este Dataset foi o mais complicado dois dois uma vez que a qualidade dos dados iniciais e dos seus registos não era a melhor. Dessa forma a limpeza e tratamento dos dados foram postos mais em prova. Assim e após todo o tratamento de dados feito, achamos que fizemos as escolhas certas quanto à remoção de atributos escolhidos, preenchimento de valores em falta através da interpolação, do tratamento de valores duplicados e ainda da remoção de alguns outliers encontrados. A este processo acrescenta-se ainda a criação de novos atributos obtidos através de Feature Engineering do atributo record_date. Todos estes foram aspetos fulcrais para uma melhor accuracy final nos resultados obtidos.

Assim e analisando os resultados, podemos perceber que o algoritmo que melhor se adequou a este Dataset foi o GridSearchCV. Esta ferramenta de aprendizagem foi otimizada pela equipa de forma a obter o melhor resultado possível. Apesar de uma *accuracy* de aproximadamente 80% não ser tão alto quanto aquela que pretendíamos achamos que pode ser uma solução tanto que viável para a utilização no mundo real, ainda que possa ultrapassar por processos de melhoramento para se conseguir obter resultados mais corretos.

Assim, e tendo em conta que estamos a analisar o transito rodoviário, achamos que a maior qualidade do Dataset poderia influenciar bastante nos resultados obtidos. A remoção forçada do atributo relativo à chuva (AVERAGE_RAIN) pode ter diminuido a eficâcia e qualidade deste Dataset. Assim, a equipa acredita que a maior recomendação deste Dataset seria obter valores verdadeiros quanto à quantidade média de chuva (AVERAGE_RAIN).

7.2 Dataset Students Performance

Este Dataset foi, comparando com o Dataset fornecido pela equipa docente, um Dataset muito mais limpo e por isso mais fácil de tratar os seus dados. Neste Dataset não possuíamos qualquer tipo de outlier ou até valores em falta. Além disso, todos os atributos possuíam mais do que um elemento distinto e existiam apenas alguns registos duplicados que são facilmente tratáveis.

Analisando mais uma vez os resultados podemos perceber que neste caso, o algoritmo que melhor se adequa será o de Linear Regression. Assim, possuímos uma *accuracy* de aproximadamente 82%, devido a uma menor percentagem de valores contínuos, o que torna mais fácil a previsão de resultados. A *accuracy* poderia ser muito melhor do que a obtida e a equipa acha que o maior obstáculo para o mesmo não acontecer é a escassez de registos, uma vez que 1000 registos são poucos para poderem ser realizados testes com elevada eficácia. Assim, os resultados não são completamente fiáveis, mas poderiam ser utilizados para fins de previsão estatística no mundo real (ainda que possuíssem uma grande margem de erro, uma vez que o ser humano é imprevisível e impossível de prever, mesmo sabendo qual o seu *background*).

Finalizando, precisamos de perceber que ao contrário do que acontece no transito rodoviário, a atividade humana pode variar muito mais constantemente. Com isto entende-se que, daus pessoas com o mesmo *background* podem ter um sucesso escolar muito diferente. Além disso, casos em que possuímos um "bom *background*"podem acabar num mau sucesso e performance escolar e casos com um "mau *background*"podem acabar com um elevado sucesso e performance escolares. Assim, apesar de ser bastante vcariável, acreditamos que o Dataset poderia ser melhorado se a quantidade de registo fosse maior, uma vez que a equipa acha que 100 registos são uma baixa quantidade.

8 Conclusão

Dado por concluído o projeto prático, o grupo considera ter realizado um bom trabalho. Além de poder pôr em prática aquilo que foi aprendido nas aulas, e dessa forma fazendo uma maior consolidação da matéria dada, o grupo acha que o trabalho foi importante para perceber que, por vezes, termos de olhar com mais atenção a pormenores que outrora não olharia-mos. Ainda assim, percebemos que por vezes achamos que estamos a acrescentar algo útil e a retirar algo desnecessário, e na verdade estamos unicamente a realizar *overfitting* do modelo. Contudo, e mais importante ainda, fica também a ideia que é sempre bom experimentar coisas novas e a prática, a tentativa e insistência fazem a perfeição.

Assim, e de forma a colocar um ponto final neste projeto, o grupo pensa ter consolidado notavelmente o conteúdo leccionado ao longo do semestre na UC de Dados e Aprendizagem Automática, tendo ainda a possibilidade de crescer um pouco mais e ganho mais *soft skills* a cada obstáculo ou adversidade encontrada.