Universidade do Minho

Escola de Ciências da Universidade do Minho
Departamento de Informática

Mestrado em Matemática e Computação
Mestrado Integrado em Engenharia Informática

Redes Neuronais Recorrentes para previsão do fluxo de tráfego rodoviário

Alunos:
Andreia Costa (PG37013)
Henrique Faria (A82200)
Paulo Barbosa (PG40160)
Rui Teixeira (PG37021)

Docentes:
Bruno Fernandes
Victor Alves

Unidade Curricular: Classificadores e Sistemas Conexionistas

Maio
2020

Conteúdo

1 Introdução **1**

2 *Dataset* **2**
2.1 *Traffic Flow Braga* 2
2.2 *Traffic Incidents Braga* 3
2.3 *Weather Braga Descriptions* 3
2.4 *Weather Braga* 4
2.5 Preparação dos dados 4

3 Problema **14**
3.1 Resolução do Problema 14

4 Modelo **16**
4.1 Avaliação do comportamento do modelo 17
4.1.1 Rua 1 18
4.1.2 Rua 2 21
4.1.3 Rua 3 24
4.1.4 Rua 4 27

5 Extra **30**
5.1 *Dataset* sem incidentes 30
5.2 Usar modelo da rua 2 para fazer previsões para a rua 1 33
5.3 Usar modelo da rua 1 para fazer previsões para a rua 3 34

1 Introdução

2 *Dataset*

Aquando da apresentação do presente trabalho foram disponibilizados dados referentes a duas cidades: Braga e Porto, sendo que o grupo escolheu os dados relativos à cidade de Braga para trabalhar.

Os dados encontram-se distribuídos em 4 *datasets*:

- *Traffic Flow Braga Until 20191231*;
- *Traffic Incidents Braga Until 20191231*;
- *Weather Braga Descriptions Until 20191231*;
- *Weather Braga Until 20191231.*

Todos os *datasets* contêm dados relativos ao período entre 15 Janeiro 2019 e 31 Dezembro 2019.

2.1 *Traffic Flow Braga*

O *dataset "Traffic Flow Braga"* é constituído pelos seguintes atributos:

- $city\_name$;
- $road\_num$;
- $road\_name$;
- $functional\_road\_class\_desc$;
- $current\_speed$;
- $free\_flow\_speed$;
- $speed\_diff$;
- $current\_travel\_time$;
- $free_flow\_travel\_time$;
- $time\_diff$;
- $creation\_date$.

2.2 *Traffic Incidents Braga*

- $city\_name$;
- $description$;
- $cause\_of\_incident$;
- $from\_road$;
- to_road;
- $affected\_roads$;
- $incident\_category\_desc$;
- $magnitude\_of\_delay\_desc$;
- $length\_in\_meters$;
- $delay\_in\_seconds$;
- $incident\_date$;
- $latitude$;
- $longitude$.

2.3 *Weather Braga Descriptions*

- $city\_name$;
- $cloudiness$;
- $atmosphere$;
- $snow$;
- $thunderstorm$;
- $rain$;
- $sunrise$;
- $sunset$;
- $creation\_date$.

2.4 *Weather Braga*

- *city_name*;
- *temperature*;
- *atmospheric_pressure*;
- *humidity*;
- *wind_speed*;
- *clouds*;
- *precipitation*;
- *current_luminosity*;
- *sunrise*;
- *sunset*;
- *creation_date*.

2.5 Preparação dos dados

Após análise dos quatro *datasets* concluiu-se que, antes de se desenvolver o modelo para a previsão da *feature speed_diff*, era necessário fazer uma prévia preparação dos dados.

Começou-se por fazer um tratamento inicial do *dataset Traffic_Incidents*. Para isso, a cada incidente atribuiu-se os vários valores da coluna *road_num*, para que posteriormente fosse possível avaliar a distância entre os incidentes e as ruas em estudo e verificar de que forma estes incidentes afetam uma determinada rua.

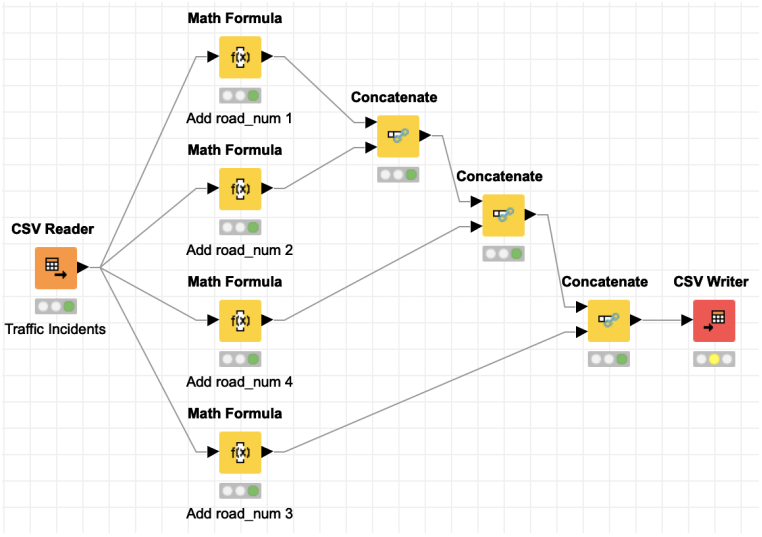

Figura 1: Preparação do *dataset Traffic_Incidentes*.

De seguida, recorrendo à latitude e longitude dos diferentes acontecimentos, calculou-se a distância dos incidentes a cada uma das ruas, para perceber o raio de influência dos incidentes para as ruas em estudo, de modo, a ser possível, posteriormente, remover incidentes que se encontrem muito afastados das ruas em estudo. Para isso, recorreu-se à fórmula:

$$dist(A,B) = R * \arccos(sin(lat_A) * \sin(lat_B) + \cos(lat_A) * \cos(lat_B) * \cos(lon_A - lon_B)).$$

onde,

- lat_A: latitude do ponto A;
- lat_B: latitude do ponto B;
- lon_A: longitude do ponto A;
- lon_B: longitude do ponto B;
- R: raio da Terra.

tendo-se implementado o seguinte código.

```
import pandas as pd
from math import radians, sin, cos, atan2, sqrt

df = pd.read_csv('Traffic_Incidents.csv', delimiter = ',',
    error_bad_lines = False, encoding = 'ISO-8859-1')
```

```
5
6  def distance(p1, n):
7    R = 6371.0
8    if n == 1:
9    lat2 = radians(41.548331)
10   lon2 = radians(-8.421298)
11   elif n == 2:
12   lat2 = radians(41.551356)
13   lon2 = radians(-8.420001)
14   elif n == 3:
15   lat2 = radians(41.546639)
16   lon2 = radians(-8.433517)
17   else:
18   lat2 = radians(41.508849)
19   lon2 = radians(-8.462299)
20   lat1, lon1 = radians(p1[0]), radians(p1[1])
21   dlon = lon2 - lon1
22   dlat = lat2 - lat1
23   a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon /
      2)**2
24   c = 2 * atan2(sqrt(a), sqrt(1 - a))
25   distance = R * c
26 return distance
27
28 df['Distance'] = df.apply(lambda row: distance((row['latitude
     '],row['longitude']), row['road_num']), axis=1)
```

Após calculadas todas as distâncias fez-se um tratamento estatístico, tendo-se obtido os seguintes resultados:

- $max = 6313,251$;
- $min = 0,0228$;
- $mean = 4,507$;
- $standard\ deviation = 81,789$.

Através dos resultados obtidos é possível verificar que existem dados mal classificados, uma vez que, sendo os dados recolhidos referentes apenas à cidade de Braga era impossível que a distância máxima dos incidentes às ruas fosse de cerca de 6313 km. Fez-se um estudo desta informação e verificou-se que estes dados dizem respeito a uma cidade que não pertence a Braga.

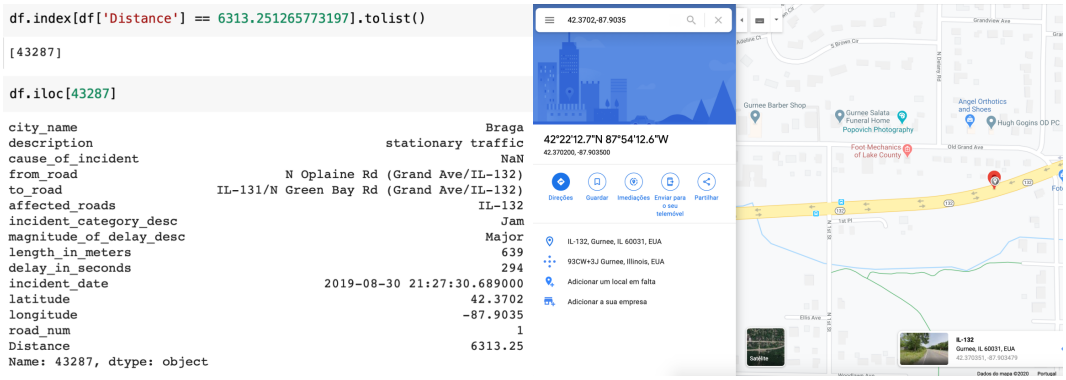

Figura 2: Dado mal classificado.

Devido a este facto, optou-se por remover alguns dados do *dataset*. Uma vez que a distância é medida em linha reta utilizou-se como *threshold*, para remover dados, vários valores, nomeadamente, 0.5, 1 e 1.5.

Após feito este tratamento procedeu-se à preparação dos dados referentes aos restantes *datasets*, com o intuito de se obter, no final, um único *dataset*.

Começou-se por fazer o tratamento do *dataset Weather_Descriptions_Braga*, tendo-se removido as colunas: *city_name*, *snow* e *cloudiness*. A coluna *snow* apresentava apenas *missing values*, daí se ter optado pela sua remoção. Relativamente à coluna *cloudiness*, optou-se por fazer a remoção da mesma, uma vez que existe uma coluna que está diretamente relacionada com esta, a coluna *cloud*, pertencente ao *dataset Weather_Braga*, e que não apresenta *missing values*.

De seguida, procedeu-se à remoção das colunas *city_name* e *precipitation* do *dataset Weather_Braga*. A remoção da coluna *precipitation* deveu-se ao facto desta apenas apresentar um único valor, o 0.

De modo a unir o resultado da preparação dos dados feita para os *datasets* anteriores, recorreu-se ao nodo *Joiner*, e uniram-se os *datasets* por *creation_date*, tendo-se efetuado, de seguida, a extração da data e do tempo, tendo-se extraído: o mês como uma variável numérica, a hora, o dia do mês e o dia da semana.

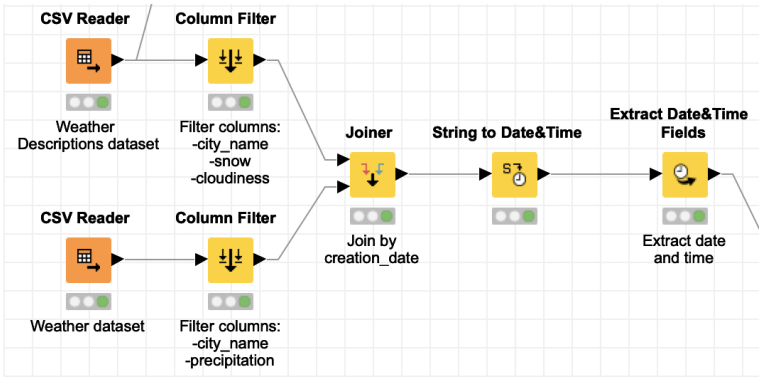

Figura 3: Preparação dos *datasets* *Weather_Descriptions_Braga* e *Weather_Braga*.

De seguida, procedeu-se à preparação do *dataset* *Traffic_Flow_Braga*, procedendo-se à remoção das colunas *city_name* e *road_name*, seguida da extração da data e hora, à semelhança do que foi feito para o *dataset* anterior.

O *dataset* *Traffic_Flow_Braga* tinha registos de 20 em 20 minutos e o *dataset* obtido anteriormente tinha registos de hora em hora, assim, de modo a unir o *dataset* com os dados relativos ao *Weather*, optou-se por agrupar os registos do *dataset* *Traffic_Flow_Braga* por hora, mês, dia e rua, recorrendo-se ao nodo *GroupBy*, tendo-se feito a média de todos os valores numéricos para as restantes colunas.

Assim, de modo a juntar este *dataset* ao obtido anteriormente, recorreu-se ao nodo *Joiner*, unindo-se os *datasets* por hora, dia do mês e mês, fazendo-se um *Left Outer Join*. Optou-se por fazer um *Left Outer Join*, uma vez que não se queriam as condições atmosféricas de registos em que não havia dados de tráfego.

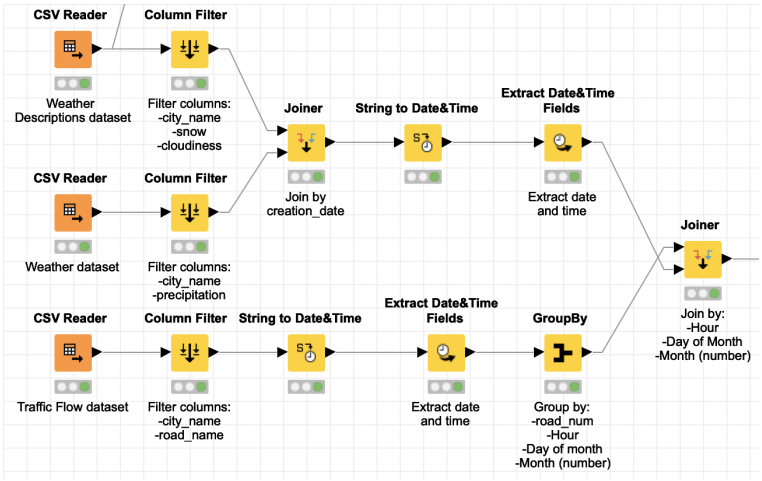

Figura 4: Preparação do *dataset Traffic_Flow_Braga*.

Após a junção dos *datasets*, eliminou-se a coluna *creation_date* e transformaram-se os valores "N/A", das colunas *rain*, *thunderstorm* e *atmosphere*, em *missing values*, recorrendo ao nodo *String Manipulation*. De seguida, quando não existiam valores na coluna *rain* atribuia-se o valor da coluna *thunderstorm*, uma vez que as *labels* da coluna *thunderstorm* faziam referência ao estado da chuva. De seguida, alteraram-se alguns dos valores ("$trovoada\ com\ chuva\ fraca$" $\rightarrow$ "$chuva\ fraca$", "$trovoada\ com\ chuva\ forte$" $\rightarrow$ "$chuva\ forte$" e "$trovoada$" $\rightarrow$ "$chuva$"), tendo-se removido, no final, a coluna *thunderstorm*. Por fim, eliminaram-se as colunas *sunrise* e *sunset*, uma vez que não se achou que estas colunas eram relevantes para prever o *speed_diff*.

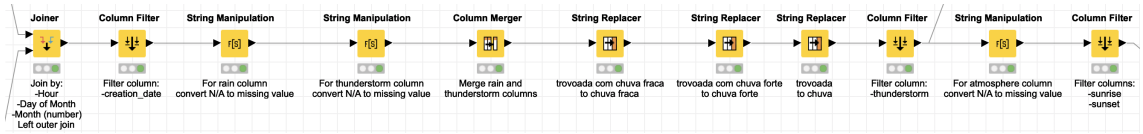

Figura 5: Preparação dos dados.

Por fim, tratou-se o *dataset Traffic_Incidents_Braga* cuja *feature Distance* tinha apenas valores inferiores a 0.5 km. Recorrendo à coluna *incident_date*, procedeu-se à extração do dia, da hora e do mês e removeram-se colunas irrelevantes, nomeadamente, as colunas *city_name*, *incident_date* e *cause_of_incident*, uma vez que esta última apresentava maioritariamente *missing values*.

Uma vez que a coluna *Distance* já inclui informação que permite relacionar a influência de um dado inicidente com as ruas em estudo, optou-se por remover as colunas: *from_road*, *to_road*, *affected_roads*, *latitute* e *longitude*.

Após tratado este *dataset*, e recorrendo ao nodo *Joiner*, uniu-se este *dataset* com o obtido anteriormente por hora, dia do mês, mês e *road_num*. Deste modo, uniram-se os 4 *datasets* iniciais num único.

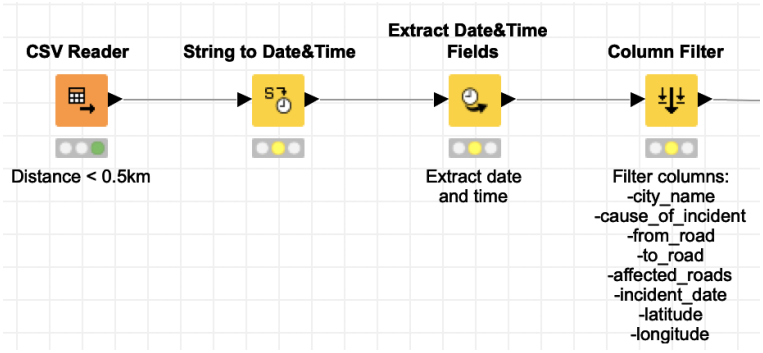

Figura 6: Preparação do *dataset* resultante do tratamento do *dataset Traffic_Incidents_Braga*.

Após se ter apenas um *dataset* verificou-se que este continha 26 colunas, o que se achou serem demasiadas. Assim, de modo a tornar o *dataset* mais pequeno, recorreu-se ao nodo *Rank Correlation* e avaliou-se a correlação que existia entre as diferentes colunas, tendo-se removido as seguintes: *free_flow_speed*, *current_travel_time*, *free_flow_travel_time* , *atmospheric_pressure*, *humidity*, *current_luminosity* e *magnitude_of_delay_desc*. Deste modo, o *dataset* ficou apenas com 18 colunas.

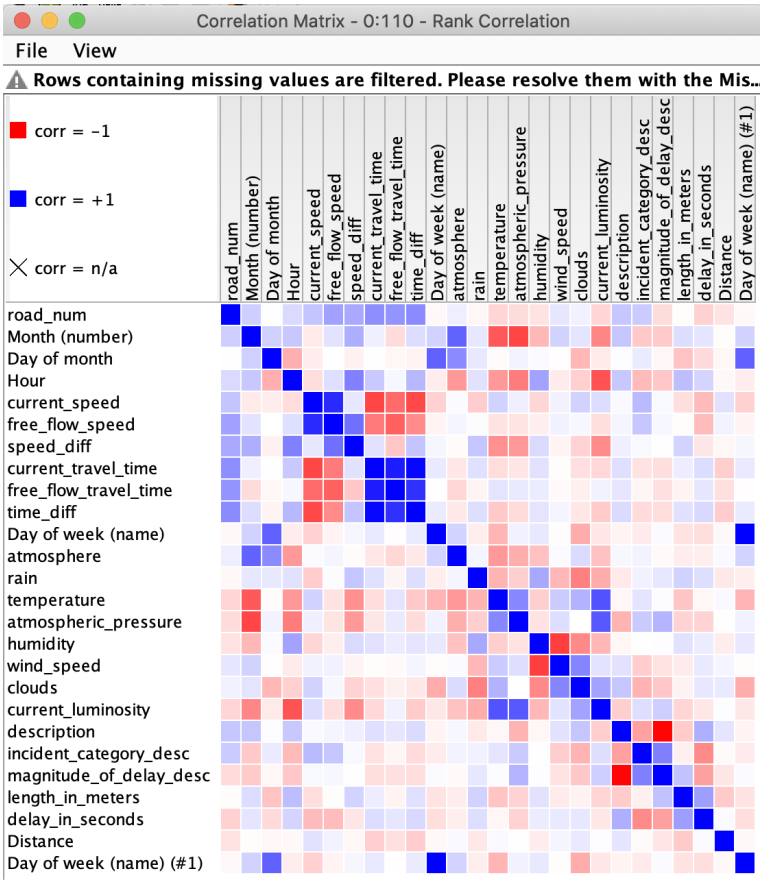

Figura 7: Análise da correlação entre as diferentes *features*.

Aos valores *Undefined* da *feature descriptions* atribui-se o valor *Unknown Delay*, uma vez que estes têm significado semelhante.

De seguida, e tendo em conta que as colunas *atmosphere* e *rain* apresentam muitos *missing values*, procedeu-se ao tratamento dos mesmos.

Começou-se, então, por tratar os *missing values* da coluna *atmosphere*, uma vez que esta era a que apresentava menos *missing values*, tendo-se separado o *dataset* em dois, recorrendo ao nodo *Rule-based Row Splitter*. Um *dataset* apresenta a coluna *atmosphere* apenas com *missing values* e o outro apresenta a coluna *atmosphere* com os vários valores. De seguida, utilizaram-se *Random Forest* para fazer a previsão dos *missing values*.

Com o intuito de perceber quais os melhores parâmetros a utilizar efetuou-se o *tunning* do modelo.

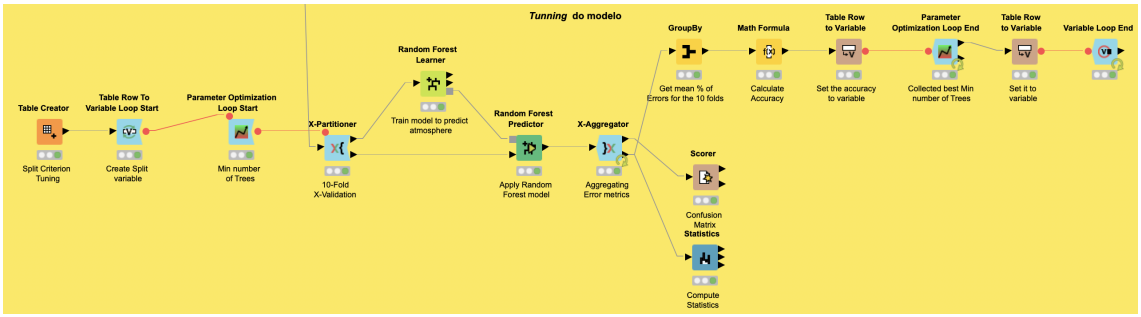

Figura 8: *Tunning* do modelo.

Após efetuado o *tunning* do modelo, concluiu-se que este apresentava melhores valores se fosse treinado com 60 árvores e usando como critério de *split* o *Information Gain*, tendo-se uma *accuracy* de cerca $99,5\%$

Data table of flow variables - 0:129 - Va...

File Hilite Navigation View

Table "default" – Rows: 3

Row ID	Trees	Split
Row0	60	InformationGain
Row1	160	InformationGainRatio
Row2	160	Gini

Output data - 0:133 - Math Form...

File Hilite Navigation View

Table "default" – Rows: 7

Row ID	Error in %	Accuracy
Row0	0	99.446
Row1	0.298	99.446
Row2	0.299	99.446
Row3	0.595	99.446
Row4	0.597	99.446
Row5	0.896	99.446
Row6	1.194	99.446

Figura 9: Melhores parâmetros para construir o modelo.

Por fim, sabendo quais os melhores parâmetros, contruiu-se um novo modelo usando 100% dos dados para o treinar.

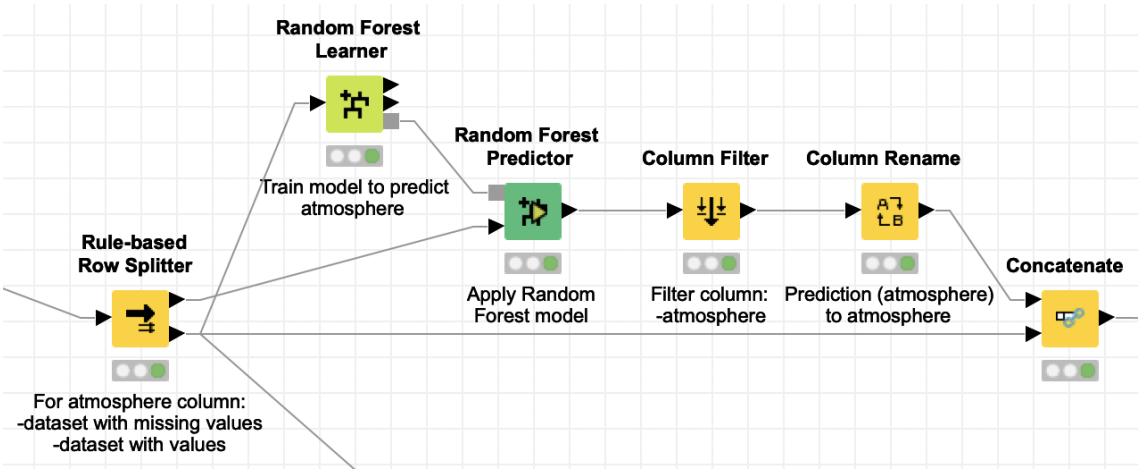

Figura 10: Previsão dos *missing values* da *feature atmosphere*.

Após feita a previsão dos *missing values* para a *feature atmosphere*, procedeu-se à previsão dos *missing values* do atributo *rain*, tendo-se utilizado o mesmo esquema para obter os melhores parâmetros, tendo-se obtido uma *accuraccy* de cerca de $97,8\%$.

Para finalizar o tratamento de dados, no *Knime*, recorrendo ao nodo *Duplicate Row Filter*, eliminaram-se linhas repetidas e efetuou-se o *Label Encoding* dos valores correspondentes às *features*: *Day of week (name)*, *description*, *incident_category_desc*, *atmosphere* e *rain*, uma vez que o objetivo é utilizar redes neuronais para prever o *speed_diff*, e estas apenas aceitam valores numéricos.

É de notar que, após feito todo este tratamento, existem colunas que apresentam *missing values*. No entanto, estes *missing values* ocorrem nas colunas correspondentes aos incidentes, porque não houve incidentes naquela hora, numa distância inferior a 0.5 km. Estes *missing values* foram substituídos por um valor *default*, -1.

Por fim, observou-se que existiam dias com horas repetidas, devido ao facto de para uma mesma hora existir mais do que um incidente. Assim, para que isto não acontecesse, recorreu-se ao nodo *GroupBy* para agrupar os incidentes, optando-se por ficar com o incidente que estava mais próximo da rua em estudo, ou seja, o incidente que apresentava na coluna *Distance* o valor mais baixo.

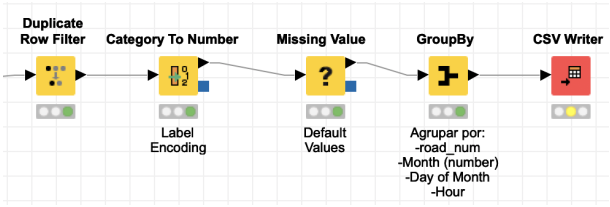

Figura 11: Tratamento final.

Após feito este tratamento, recorrendo ao nodo *Pie chart (local)* percebeu-se que o *dataset* apresentava dias e horas em falta como, por exemplo, o mês de Março.

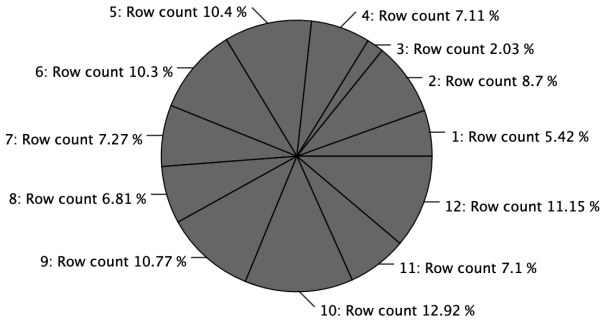

Figura 12: Dias em falta no *dataset*.

3 Problema

O objetivo do trabalho consiste em utilizar Redes Neuronais para fazer previsão do fluxo de tráfego rodoviário. Tendo em conta os dados disponibilizados, o grupo optou por fazer a previsão da *feature speed_diff* de uma dada rua, baseando-se nos 3 dias anteriores para prever o dia seguinte. Optou-se por fazer previsões para as ruas em separado, uma vez que se acredita que o fluxo de tráfego das ruas é distinto entre elas. Para isso, dividiu-se o *dataset* original em 4, sendo que cada *dataset* correspondia a uma determinada rua.

Uma vez que se trata de um problema de séries temporais, optou-se por implementar um modelo *multistep* e *multivariate* que utiliza *LSTM*'s.

3.1 Resolução do Problema

Para resolver este problema era então necessário perceber quais os dias que estavam incompletos, ou seja, quais os dias que não tinham as 24 horas

preenchidas, procedendo-se à eliminação destes, com o intuito de se ter um *dataset* sem "buracos". Para isso, implementou-se o seguinte algoritmo:

```
i=0
for i in range(1,13):
  for j in range(1,32):
  L=df[(df['Month (number)']==i)&(df['Day of month']==j)].
   dropna()
  L1=L[['Month (number)','Day of month','Hour','road_num']]
  L1 = L1.drop_duplicates()
  indexNames = df[(df['Month (number)']==i)&(df['Day of month
   ']==j)].index
  if len(L1)<24:
  try:
    df.drop(indexNames, inplace=True)
  except:
    pass
```

Note-se que se verificou que nos dias em que havia horas em falta, normalmente o número de horas em falta era elevado, não se justificando a reconstrução dessas horas, recorrendo a métodos matemáticos, pelo que se optou por eliminar esses dias.

Visto que no final do capítulo 2.5 verificámos que existiam dias em falta e como se pretende que sejam dados ao modelo, como *input*, 3 dias para prever o próximo, quer garantir-se que, de facto, esses 4 dias são seguidos, ou seja, que os 3 dias dados como *input* mais o dia a prever sejam seguidos, evitando que ocorram situações em que o primeiro dia seja, por exemplo, 16 de Janeiro e o dia a prever seja 30 de Janeiro.

Assim, para ter a certeza que se treina o modelo com dias consecutivos, percorreu-se o *dataset* construindo blocos de 4 dias, para verificar se estes 4 dias são seguidos, calcula-se a diferença entre o último dia do bloco e o primeiro e verifica-se se é 4. Note-se, no entanto, que se para um dado bloco tivermos dias de meses distintos, esta diferença é negativa, sendo este problema corrigido dependendo do mês em causa.

Antes de se aplicar o seguinte algoritmo, ordenou-se o *dataset* por mês, dia e hora.

```
n_future = 24 # next 24 hours speed diff forecast
n_past = 24*3 # Past 3 days

x_train = []
y_train = []
label = df_1['speed_diff']

for i in range(0,len(df_1)-n_past-n_future+1):
  dias = df_1.iloc[i : i + n_past+24]
  mes = dias.iloc[0]['Month (number)']
```

```
11  dia_1 = dias.iloc[0]['Day of month']
12  dia_4 = dias.iloc[24*3+1]['Day of month']
13  if (mes == 4 or mes == 6 or mes == 9 or mes == 11) and (
     dia_4 - dia_1 == 3 or dia_4 - dia_1 == -29):
14     x_train.append(df_1.iloc[i : i + n_past])
15     y_train.append(label.iloc[i + n_past : i + n_past +
     n_future ])
16  elif (mes == 1 or mes == 3 or mes == 5 or mes == 7 or mes
     == 8 or mes == 10 or mes == 12) and (dia_4 - dia_1 == 3 or
      dia_4 - dia_1 == -28):
17     x_train.append(df_1.iloc[i : i + n_past])
18     y_train.append(label.iloc[i + n_past : i + n_past +
     n_future ])
19  elif mes == 2 and (dia_4 - dia_1 == 3 or dia_4 - dia_1 ==
     -26):
20     x_train.append(df_1.iloc[i : i + n_past])
21     y_train.append(label.iloc[i + n_past : i + n_past +
     n_future ])
```

Sabendo-se que cada *input* é constituído por 3 dias seguidos, com as 24 horas completas e, portanto, as colunas *Month (number)*, *Day of month* e *Hour* já não são relevantes, tendo-se feito a remoção das mesmas. Além disso, removeram-se as colunas *Day of week (name)*, *incident_category_desc* e *Distance*. Esta última foi removida, uma vez que apenas serviu para saber quais os incidentes que deviam permanecer no *dataset*.

Após feito todo o tratamento acima mencionado, o *dataset* está pronto para ser aplicado a uma rede que permita prever a *feature speed_diff*.

4 Modelo

O problema que se pretende resolver é um problema de séries temporais, como já foi anteriormente referido. Deste modo, para prever a *feature speed_diff* optou-se por construir um modelo *multistep* e *multivariate* que utiliza *LSTM*'s.

Tendo em conta que o objetivo é utilizar 3 dias para prever as 24 horas seguintes, utilizando 11 *features*, definiu-se como $input_shape = (24 * 3, 11)$.

Antes de se começar a treinar o modelo procedeu-se à normalização dos dados:

```
1 # Features normalization
2 scalers=[]
3 for i in range(11):
4 sc = MinMaxScaler(feature_range=(0,1))
5 x_train[:,i] = sc.fit_transform(x_train[:,i])
6 x_test[:,i] = sc.fit_transform(x_test[:,i])
```

```
scalers.append(sc)

# Labels normalization
sc1 = MinMaxScaler(feature_range=(0,1))
y_train = sc1.fit_transform(y_train)
y_test_n = sc1.fit_transform(y_test)
```

Uma vez que os dados estavam normalizados no intervalo $[0, 1]$ recorreu-se à função de ativação *sigmoid*.

Relativamente às métricas utilizadas, utilizou-se como *loss* o *mean square error* e como métrica o *root mean square error*. Estas foram as métricas escolhidas, uma vez que o objetivo era penalizar erros grandes, e estas são as melhores métricas para o fazer.

Assim, recorrendo a técnicas de intuição e experimentação construiu-se o seguinte modelo:

```
model = Sequential()
model.add(CuDNNLSTM(units=24*3, return_sequences=True,
    input_shape = (24*3,11) ) )
model.add(Dropout(0.2))
model.add(CuDNNLSTM(24*3 , return_sequences=True))
model.add(Dropout(0.2))
model.add(CuDNNLSTM(24*3, return_sequences=True))
model.add(Dropout(0.2))
model.add(CuDNNLSTM(24*2))
model.add(Dropout(0.2))
model.add(Dense(24,activation='sigmoid'))
model.compile(optimizer='adam', loss='mean_squared_error',
    metrics=rmse)
checkpointer = ModelCheckpoint(filepath="best_weights.hdf5",
    monitor = 'val_loss', verbose=1, save_best_only=True)
callback = tf.keras.callbacks.EarlyStopping(monitor='loss',
    patience=30)
history=model.fit(x_train, y_train, validation_data=(x_test,
    y_test_n), epochs=1000, callbacks=[callback, checkpointer
    ])
```

Note-se que se recorreram a dois *callbacks*: um para garantir que se guardava o melhor modelo, que seria porteriormente usado para fazer previsões, e outro que permitia que o treino parasse, quando não houvesse uma melhor do valor de *loss* durante 30 épocas seguidas.

4.1 Avaliação do comportamento do modelo

Após construído o modelo, procedeu-se a um conjunto de testes, utilizando 200 dados de teste, com o intuito de ser possível avaliar o comportamento do mesmo.

Primeiro determinou-se, para os 200 dados de teste, o número de ocorrências de cada dia da semana. Para se perceber se o comportamento do modelo era o esperado procedeu-se à utilização de duas métricas de erro, a média dos erros e a diferença absoluta, com a seguinte expressão, respetivamente:

$$average\_diff = \frac{\sum_i |\hat{y}_i - y_i|}{24}, \text{ onde } i = 0, .., 23$$

$$max\_diff = \max |\hat{y}_i - y_i|, \text{ onde } i = 0, .., 23$$

Ambas as métricas de erro parecem ser úteis para perceber o comportamento do modelo. No entanto, a métrica *max_diff* apresenta melhores resultados para exemplos específicos. Considere-se um exemplo em que temos 23 valores a 0 e outro apresenta um speed_diff de 15 e, na previsão, o modelo acerta os 23 casos em que é o valor era 0, mas no caso em que o valor real era 15 o valor previsto foi 9. Então, neste caso, tem-se que average_diff = 0.25, o que nos dá um valor de erro baixo, induzindo-nos em erro ao acharmos que os valores previstos eram bons, o que não acontece, uma vez que o único valor que era difícil de prever o modelo erra por uma diferença de 6 km. Por outro lado, a métrica *max_diff* diz-nos que o erro é 6, o que permite concluir que o modelo não fez uma boa previsão para este exemplo.

4.1.1 Rua 1

Começou-se por treinar o modelo com 5045 dados da rua 1, e fazer a previsão para essa mesma rua.

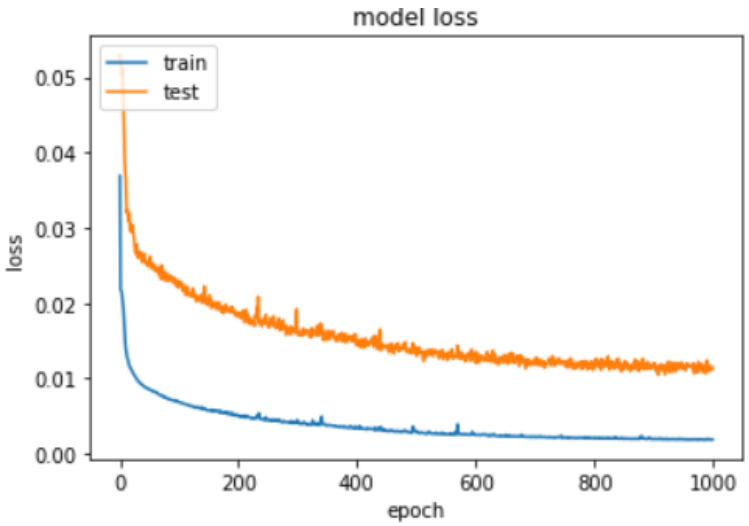

Figura 13: Curvas de aprendizagem.

Observando a Figura 13, conclui-se que o modelo criado apresenta *underfitting*, uma vez que as curvas de aprendizagem são distintas uma da outra. Pode ainda referir-se que, apesar de existir *underfitting*, é provável que com o aumento do número de épocas não se observe uma convergência das curvas, uma vez que se verifica que ambas as curvas parecem ter estagnado, o que pode evidenciar que o modelo não está a aprender.

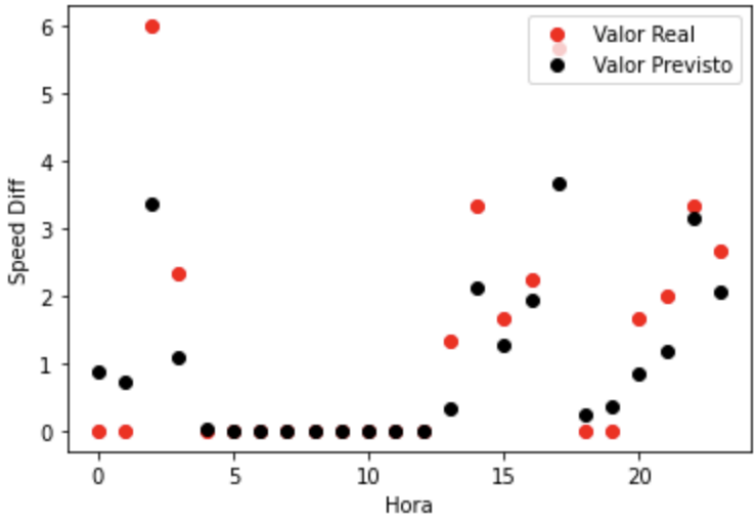

Figura 14: Valores reais vs previstos, durante 24 horas, para um dia aleatório.

Hora	Real	Previsto
0	0	0.88
1	0	0.73
2	6	3.37
3	2.33	1.10
4	0	0.023
5	0	0.00017
6	0	0.00014
7	0	0.012
8	0	0.00011
9	0	0.00021
10	0	7.70e-05
11	0	1.90e-06
12	0	0.00088
13	1.33	0.32
14	3.33	2.13
15	1.67	1.26
16	2.25	1.95
17	5.67	3.66
18	0	0.25
19	0	0.37
20	1.67	0.86
21	2	1.19
22	3.33	3.15
23	2.67	2.05

A Figura 14 e a tabela permitem comparar o valor previsto com o valor real, durante um período de 24 horas. Ora, através da análise do gráfico, observa-se que não existe uma grande discrepância entre o valor real e o valor previsto. Pelo que se pode considerar que, para este caso, o modelo construído faz boas previsões. É de notar que, neste caso, os valores previstos são mais distintos dos valores reais às 3 e 4 da manhã, sendo normal que o modelo erre nestas horas, uma vez que não é comum que o valor do *speed_diff* seja mais elevado nestas, ou seja, pode inferir-se que este dia não representa os dias comuns.

Por fim, faça-se uma análise dos erros obtidos. Após calculados os diferentes valores do erro, obtiveram-se os seguinte resultados:

	Número de Ocurrências	*Average Diff*	*Max Diff*
Segunda-Feira	29	0.45	2.69
Terça-Feira	32	0.67	3.34
Quarta-Feira	21	0.73	3.54
Quinta-Feira	28	0.87	3.57
Sexta-Feira	26	0.65	3.07
Sábado	40	0.83	3.82
Domingo	24	0.36	2.91

Tabela 1: Cálculo dos erros.

Tendo em conta os gráficos e os resultados apresentados nas tabelas, pode concluir-se que o modelo prevê a *feature speed_diff* para esta rua com sucesso.

4.1.2 Rua 2

Considerando os dados de treino da rua 2, treinou-se o modelo construído, obtendo-se os seguintes resultados.

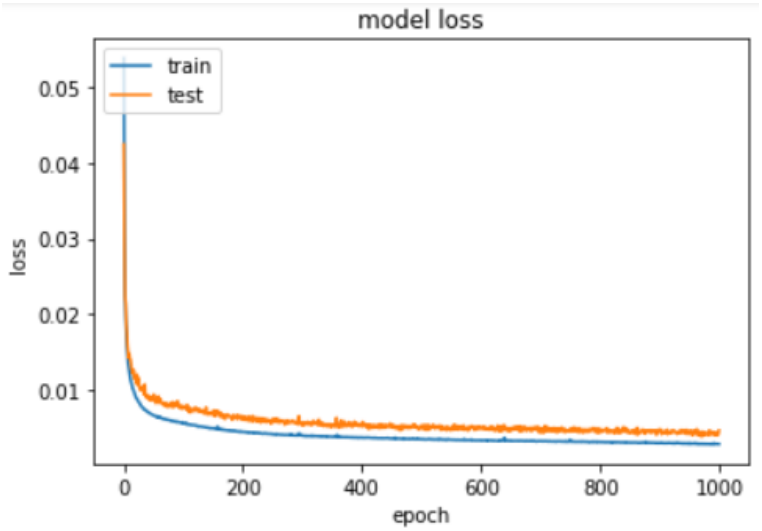

Figura 15: Curvas de aprendizagem.

Analisando o gráfico, observa-se que, em relação ao apresentado para a rua 1, agora as curvas de aprendizagem estão mais próximas, sendo ainda possível observar-se que o modelo chegou ao seu limite de aprendizagem, uma vez que não se verifica um decréscimo da reta.

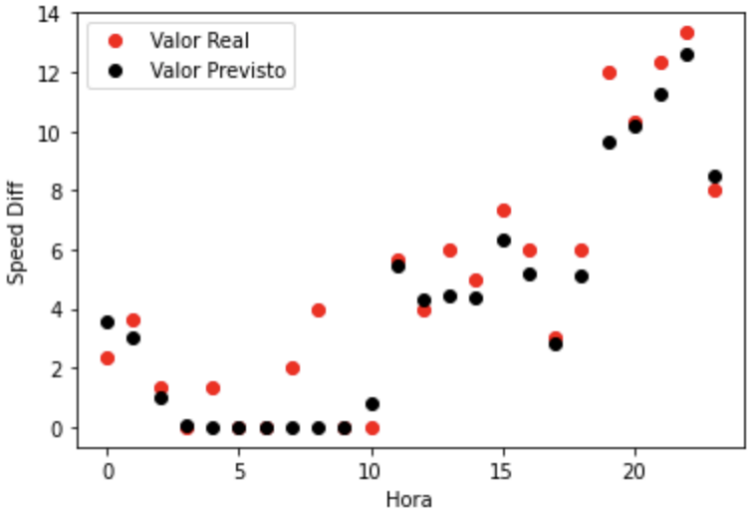

Figura 16: Valores reais vs previstos, em 24 horas.

Hora	Real	Previsto
0	2.33	3.6011205
1	3.67	3.05
2	1.33	0.98
3	0	0.056
4	1.33	0.00011
5	0	5.72e-07
6	0	1.04e-07
7	2	3.18e-08
8	4	0.00016
9	0	0.028
10	0	0.82
11	5.67	5.46
12	4	4.28
13	5.99	4.42
14	5	4.38
15	7.33	6.31
16	5.99	5.21
17	3.0	2.84
18	6.0	5.12
19	12.0	9.61
20	10.33	10.15
21	12.33	11.24
22	13.33	12.63
23	7.99	8.48

Analisando os resultados acima apresentados pode concluir-se, que para este exemplo, o modelo fez boas previsões. Estas conclusões são reforçadas pelos valores obtidos para as métricas de erro:

- average_diff
- max_diff

Por fim, faça-se uma análise dos erros obtidos. Após calculados os diferentes valores do erro, obtiveram-se os seguinte resultados:

	Número de Ocurrências	*Average Diff*	*Max Diff*
Segunda-Feira	29	0.29	1.38
Terça-Feira	32	0.51	2.27
Quarta-Feira	21	0.42	1.70
Quinta-Feira	28	0.44	1.78
Sexta-Feira	26	0.48	2.04
Sábado	40	0.43	1.79
Domingo	24	0.36	1.80

Tabela 2: Cálculo dos erros.

Analisando os resultados dos erros observa-se que os valores obtidos são uniformes e semelhantes, ou seja, para esta rua, parece que o modelo prevê todos os dias com a mesma dificuldade, o que não se verificava no modelo da rua 1. Para o modelo da rua 1, verificou-se que este tinha mais dificuldade em prever Quintas-Feiras, do que Domingos.

4.1.3 Rua 3

Considerem-se agora os resultados obtidos pelo modelo treinado para a rua 3. Sabe-se que esta rua diz respeito à Rua do Caires, que corresponde a uma rua com bastante trânsito sendo, por isso, expectável grandes oscilações no valor da variável *speed_diff*, de uma hora para a outra.

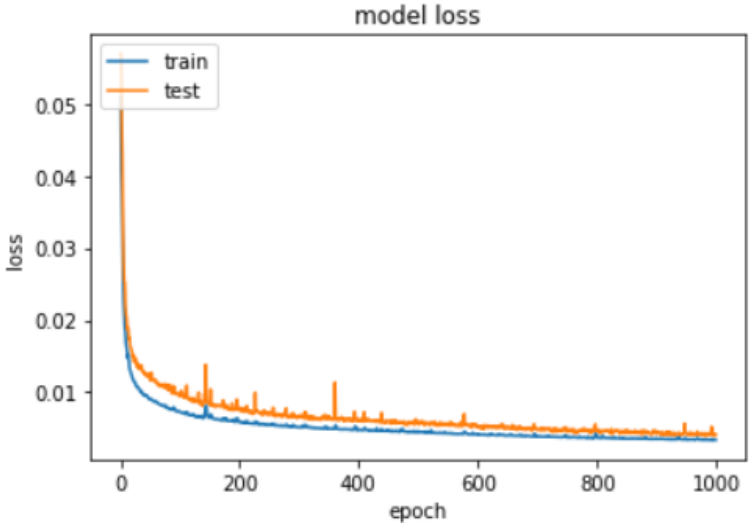

Figura 17: Curvas de aprendizagem.

À semelhança do que foi observado no exemplo anterior também aqui se verifica a convergência das curvas de aprendizagem.

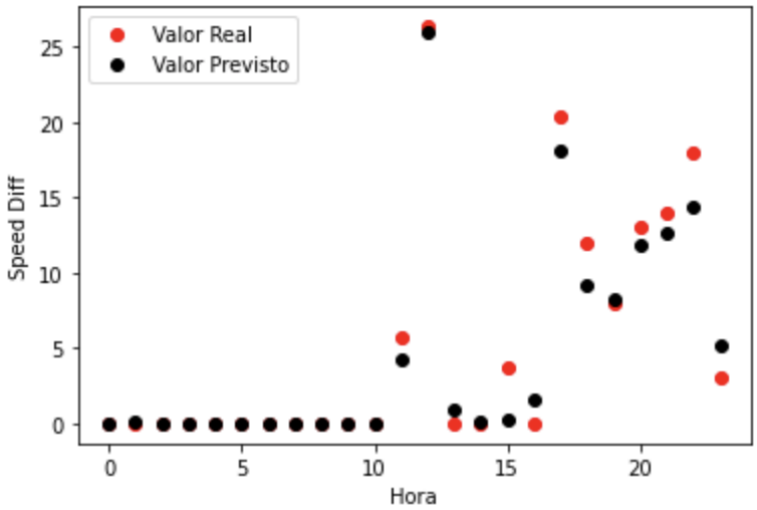

Figura 18: Valores reais vs previstos, em 24 horas.

Hora	Real	Previsto
0	0	0.032280684
1	0	0.06226761
2	0	0.0001824899
3	0	0.002977331
4	0	0.038694322
5	0	8.884608e-05
6	0	7.37009e-05
7	0	0.00014307268
8	0	1.6951117e-05
9	0	0.0026792889
10	0	0.009467335
11	5.67	4.2219253
12	26.33	26.00911
13	0	0.9039111
14	0	0.08721095
15	3.67	0.22982505
16	0	1.524076
17	20.33	18.167295
18	12	9.12594
19	7.99	8.301455
20	13	11.784495
21	14	12.6543255
22	18	14.313829
23	3	5.1567326

Note-se que à hora 12 o *speed_diff* real era de 26.33 e o modelo fez uma previsão de 26.00, à semelhança do que aconteceu para a hora 17, realçando o excelente desempenho do modelo ao prever este dia.

Por fim, faça-se uma análise dos erros obtidos. Após calculados os diferentes valores do erro, obtiveram-se os seguinte resultados:

	Average Diff	*Max Diff*
29 Segundas	1.1104795417657065	5.096033663632976
32 Terças	1.264305124666961	5.205189819066919
21 Quartas	1.1223343328243598	4.69481890565819
28 Quintas	1.340759207220143	5.520887696665271
26 Sextas	1.0976789264484141	4.579785451503434
40 Sábados	1.2548639571139115	4.768039370036238
24 Domingos	0.5671839880583601	3.284548810598506

Tabela 3: Cálculo dos erros.

Apesar de se verificar, através do gráfico, que o modelo fez boas previsões, observa-se que os valores obtidos para os erros são mais elevados neste caso do que no caso dos modelos apresentados para as ruas 1 e 2. No entanto, basta olhar para o exemplo apresentado para verificar que esta rua apresenta valores para o *speed_diff* bastante elevados. De facto, para o exemplo apresentado o valor máximo para o *speed_diff* é de 26.33km/h e, por exemplo, para o caso da rua 1, o valor máximo do *speed_diff* é 5.67 km/k. Assim, é importante notar que o modelo da rua 3 tem que prever valores muito mais elevados do que o da rua 1. Ou seja, dados estes valores pode dizer-se que errar 1 km/h na rua 3 pode ser considerado um erro menor do que errar 0.5 km na rua 1, concluindo-se, assim que, embora os erros apresentados nesta tabela sejam mais elevados, o nosso espírito crítico, leva-nos a concluir que o modelo da rua 3 até pode ser considerado melhor que o da rua 1.

4.1.4 Rua 4

A presente secção pretende analisar os resultados obtidos com o modelo treinado para a rua 4. Note-se que, à semelhança da rua 3, esta é uma rua com bastante trânsito, sendo expectável obterem-se resultados semelhantes aos apresentados para essa rua.

Apresentam-se, de seguida, os valores obtidos pelo modelo treinado com os dados da rua 4.

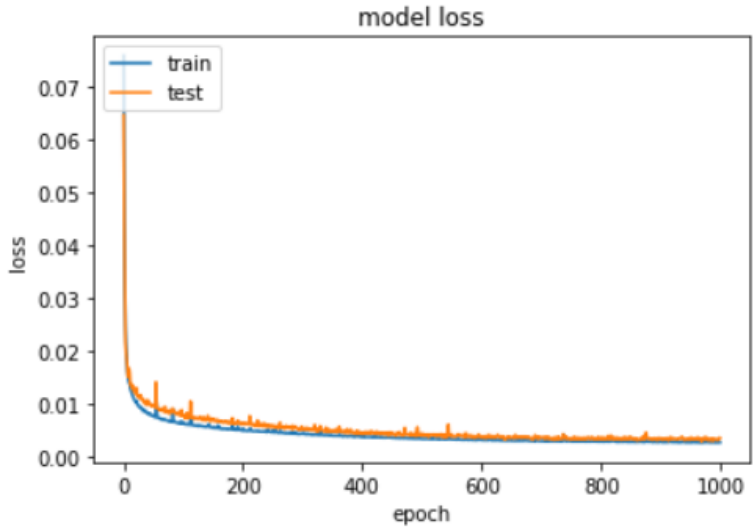

Figura 19: Curvas de aprendizagem.

Tal como observado para os gráficos das curvas de aprendizagem dos modelos das ruas 2 e 3, também para o modelo obtido para a rua 4 se verifica que as curvas de aprendizagem estão a convergir.

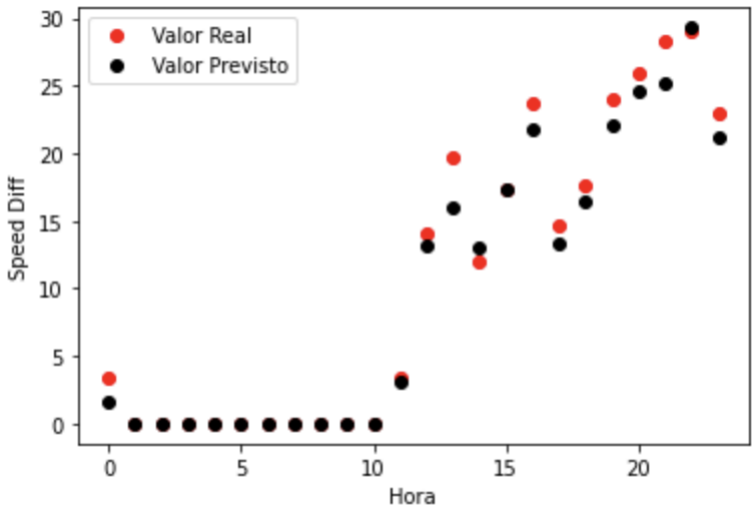

Figura 20: Valores reais vs previstos, em 24 horas.

Hora	Real	Previsto
0	3.33	1.6831238
1	0	0.0036463984
2	0	3.9301664e-09
3	0	6.119647e-05
4	0	6.241123e-10
5	0	2.987104e-06
6	0	2.3474257e-05
7	0	0.00030335836
8	0	7.2902294e-05
9	0	2.5659345e-05
10	0	0.022530494
11	3.33	3.0792594
12	14	13.116628
13	19.67	15.995314
14	12.0	13.002526
15	17.33	17.276192
16	23.67	21.827679
17	14.67	13.331219
18	17.674	16.39099
19	24	22.018396
20	26	24.598763
21	28.33	25.148285
22	29	29.302673
23	23	21.183737

Observando os resultados apresentados, tiram-se conclusões semelhantes às obtidas para o exemplo apresentado para a rua 3.

Por fim, faça-se uma análise dos erros obtidos. Após calculados os diferentes valores do erro, obtiveram-se os seguinte resultados:

	Average Diff	*Max Diff*
29 Segundas	0.6434349814782563	4.357169496163079
32 Terças	1.0732532610132068	5.173105200712162
21 Quartas	0.9179087441886314	5.071644801119003
28 Quintas	1.1919938745260334	5.7381584813503155
26 Sextas	0.91356394178488	5.536007864537613
40 Sábados	0.9611405798139803	4.2988463422039525
24 Domingos	0.4436569034521404	4.482812049263807

Tabela 4: Cálculo dos erros.

Apesar destas ruas serem ruas com muito trânsito é de notar que, a rua 4 apresenta valores de erro mais baixos do que os da rua anterior, isto pode ter em conta o facto de que a rua 3 em certos meses, por exemplo, Junho, Julho, Agosto, quando não há período escolar, esta apresenta um valor de trânsito muita mais baixo, o que faz com que o nosso dataset apresente valores de speed diff menores neste período. Ou seja, ao treinar o modelo com estes dados, conjetura-se que o modelo criado possa não ser tão preciso.Isto para concluir que a rua 4, sendo uma rua que apresenta o mesmo fluxo de trânsito durante um ano inteiro, uma das razões possíveis para os erros darem mais baixo para esta rua é o facto supramencionado.

5 Extra

O presente capítulo pretende analisar o comportamento dos modelos obtidos para as diferentes ruas, testando-os para outras ruas, e analisar o comportamento do modelo treinado com o *dataset* no qual não foram incluído os incidentes.

5.1 *Dataset* sem incidentes

Este subcapítulo pretende analisar o comportamento do modelo treinado com o *dataset* sem incidentes, para a rua 2, utilizando 8 *features*.

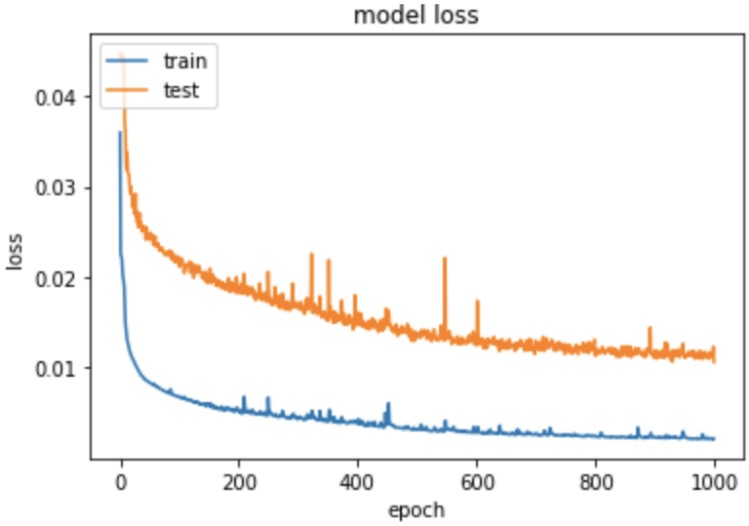

Figura 21: Curvas de aprendizagem.

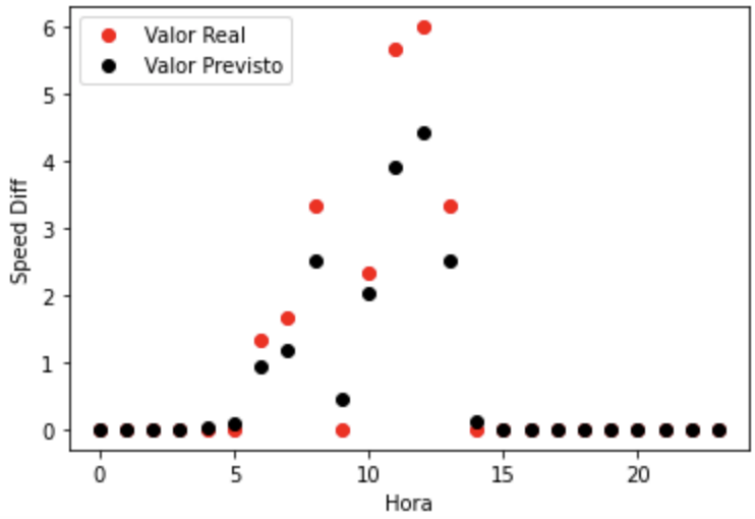

Figura 22: Valores reais vs previstos, em 24 horas.

Hora	Real	Previsto
0	0	6.9066814e-08
1	0	2.5607434e-07
2	0	2.7351797e-07
3	0	2.0879885e-05
4	0	0.01707314
5	0	0.09949607
6	1.33	0.9240082
7	1.67	1.1667032
8	3.33	2.5204258
9	0	0.45500085
10	2.33	2.0409627
11	5.67	3.920147
12	6	4.414104
13	3.33	2.5295854
14	0	0.12146372
15	0	0.0020382823
16	0	3.961704e-09
17	0	6.731966e-08
18	0	1.2392364e-06
19	0	1.6075727e-07
20	0	1.254804e-06
21	0	1.0915311e-09
22	0	5.4359017e-12
23	0	4.7200618e-05

Por fim, faça-se uma análise dos erros obtidos. Após calculados os diferentes valores do erro, obtiveram-se os seguinte resultados:

	Average Diff	*Max Diff*
29 Segundas	0.8580420333193335	3.0794863584689107
32 Terças	0.8183510621136405	3.096227680302678
21 Quartas	0.872193862194359	2.8067537848154704
28 Quintas	0.8651441438734818	2.96524466102997
26 Sextas	1.04050743303996	3.0716054604671785
40 Sábados	0.5317055662987664	2.3876666585233886
24 Domingos	0.5553113784184134	2.172586509579598

Tabela 5: Cálculo dos erros.

Analisando os resultados apresentados na tabela e comparando-os com os obtidos no subcapítulo 4.1.2, observa-se que estes são muito mais elevados, o que nos permite concluir que os dados relativos aos incidentes têm grande importância na previsão do *speed_diff*.

5.2 Usar modelo da rua 2 para fazer previsões para a rua 1

Decidiu-se testar o modelo da rua 2 na rua 1 uma vez que estas ruas são semelhantes. Note-se ainda que se optou por testar o comportamento do modelo treinado com a rua 2, uma vez que este era o que apresentava melhores resultados.

Veja-se o comportamento do modelo treinado com dados da rua 2, para fazer previsões para a rua 1.

De seguida, apresenta-se, graficamente, os resultados obtidos, para um dia de teste:

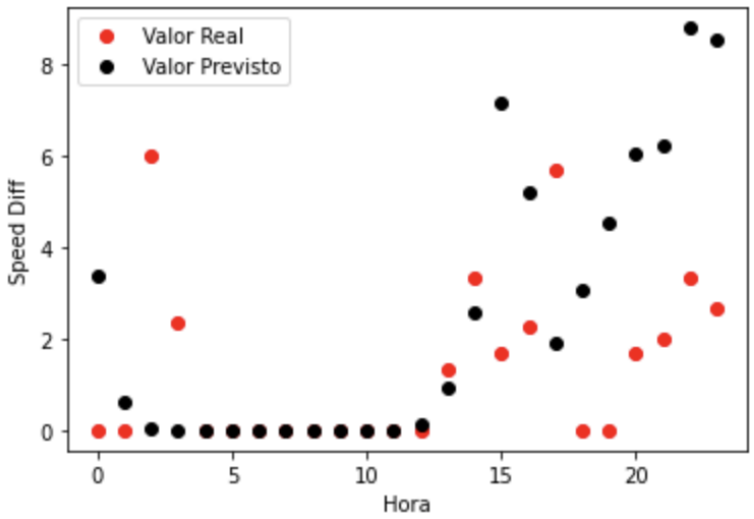

Figura 23: Valores reais vs previstos, em 24 horas.

Analisando o gráfico, conclui-se que o modelo não se comporta de modo ótimo. Note-se que, nas horas 22 e 23 existe uma grande discrepância nos valores.

Relativamente aos erros obtidos, foram os seguintes:

	Average Diff	*Max Diff*
Segundas	1.4970504526843698	6.902452780405327
Terças	1.5817615829476248	7.0771182393603675
Quartas	1.564599320078336	7.752798380280793
Quintas	1.7489723562496609	6.787156304928169
Sextas	1.8400726700992243	7.6605935197642
Sábados	1.7762016329303	7.453315566827695
Domingos	1.766229661484953	7.906649859769938

Tabela 6: Cálculo dos erros.

Por fim, analisando o valor obtido para os erros observa-se que estes são superiores aos obtidos quando se usou o modelo treinado com a rua 2 para fazer a previsão da variável *speed_diff* para essa mesma rua.

5.3 Usar modelo da rua 1 para fazer previsões para a rua 3

O presente subcapítulo pretende

De seguida, apresenta-se, graficamente, os resultados obtidos, para o primeiro dia de teste:

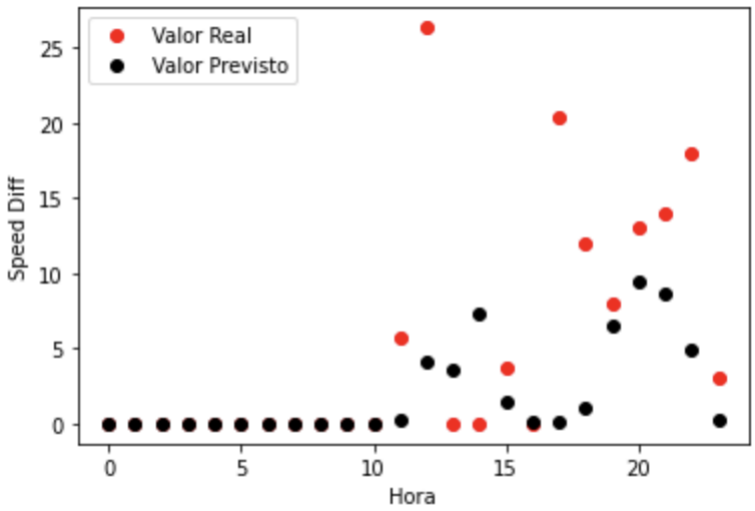

Figura 24: Valores reais vs previstos, em 24 horas.

Relativamente aos erros obtidos, foram os seguintes:

	Número de ocurrências	*Average Diff*	*Max Diff*
Segundas	29	4.97	21.61
Terças	32	4.51	20.73
Quartas	21	4.85	18.83
Quintas	28	4.64	17.88
Sextas	26	4.00	15.85
Sábados	40	5.07	21.99
Domingos	24	3.64	13.07

Tabela 7: Cálculo dos erros.

Verificando os resultados, observam-se erros muito elevados. Isto deve-se ao facto do modelo prever valores de *speed_diff* baixos, uma vez que a rua 1 apresenta um baixo fluxo de trânsito valores baixos para a *speed_diff*, e a rua 3 ser uma rua com bastante trânsito e apresentar valores de *speed_diff* bastantes elevados.