Pré-Processamento de Dados

Prof. Gustavo Willam Pereira

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Introdução

- Uma das grandes vantagens em se utilizar o Python é ter acesso a uma infinidade de Pacotes (também denominadas bibliotecas ou módulos).
- Temos pacotes de plotagem gráfica (matplotlib), operação com matrizes (numpy), processamento de dados (pandas), e nosso foco principal, aprendizagem de máquina (Scikit Learn).
- Se você souber o básico do Python, consegue fazer bom uso desses pacotes.
- À medida que for utilizando o Python, a sintaxe se tornará, naturalmente, mais fácil.

Introdução

- A primeira coisa que temos que fazer, antes de iniciar a execução dos algoritmos de aprendizagem de máquina é o pré-processamento dos dados.
- O pré-processamento é a preparação dos dados para serem utilizados nos algoritmos de aprendizagem de máquinas.
- Apesar de muitos não darem a devida atenção ao pré-processamento, podemos dizer que, ele é uma etapa muito importante na elaboração do seu modelo.
- Afinal, se entrar lixo no seu modelo, irá, com certeza, sair lixo.
- Temos que visualizar a distribuição dos dados, verificar e preencher dados ausentes, verificar e remover dados anormais (outliers) ou redundantes, padronizar ou normalizar os dados, tratar dados categóricos, reamostrar dados, dentre outros.
- Dessa forma, os pacotes Numpy, Pandas e Matplotlib irá nos ajudar nessa etapa.

Pré-Processamento de dados

- O pré-processamento dos dados é a etapa de preparação dos dados para serem utilizados nos algoritmos de machine learning.
- Dependendo do algoritmo e dos dados, mais ou menos etapas de pré-processamento serão necessárias.
- A maior parte do pré-processamento ou visualização será realizado utilizam as bibliotecas Pandas e SciKit Learn.
- Pré-processamento envolve a remoção de registros (linhas do banco de dados) sem dados (NAN). Para eliminar dados NAN poderíamos simplesmente remover toda a linha em questão, no entanto, nesse caso informações importantes de outras variáveis poderiam ser perdidas.
- Dessa forma, uma abordagem é utilizar dados “vizinhos” para estimar os dados NAN (faltantes).

Pré-Processamento de dados

- Outro pré-processamento que deveremos realizar é a substituição de dados categóricos, por exemplo: alto, médio e baixo, por dados numéricos.
- Também temos que dividir o banco de dados em dados de treinamento (dados parar modelagem) e dados de teste (validar o modelo).
- Além disso, alguns algoritmos são influenciados pela magnitude numérica dos dados.
- Assim temos que colocar todos os dados em um mesmo patamar numérico (escala). Podemos fazer utilizando a normalização ou padronizar os dados.

Pré-Processamento de dados

- Ler arquivo usando Pandas.

```
 9 # Importando o pandas
10 import pandas as pd
11
12 ############## Importando o banco de dados Dados_compra.csv
13
14 dataset = pd.read_csv('Dados_compra.csv')
```

- Na linha 10 o Pandas é importado, na linha 14 utilizamos o Pandas (pd) para abrir um banco de dados chamado “Dados_compra.csv”. O formato do arquivo é um arquivo texto do tipo *csv*.
- Agora vamos criar uma variável X e y. A variável X será um Dataframe com os dados de treinamento (variável independente) e a variável y será os dados objetivo (variável dependente). Veja o código abaixo.

Pré-Processamento de dados

```
16 ############### Cria as variáveis independentes e independente
17
18 X = dataset.copy()
19 X  = X.drop(['Compra'], axis=1) #remove a variável dependente do X
20 y = dataset.iloc[:, 3]
```

- Na linha 18 é criado uma cópia do DataFrame *dataset*. Na linha 19 a coluna "Compra", que é a variável dependente (y), será removida do DataFrame X utilizando a função *drop*. A variável *axis = 1* é para indicar que iremos remover uma coluna.
- Na linha 20 é criado a variável dependente y fazendo um recorte no DataFrame *dataset* usando o método *iloc*[:,3]. Os dois pontos indica que iremos usar todas as linhas a coluna número 3. Lembre-se que o Python inicia a numeração do zero.
- Tratar dados ausentes.
    - Agora vamos tratar dados faltantes (NAN). O pandas oference várias alternativas para remoção ou substituição de dados faltantes. A melhor forma de entender é praticando.

Pré-Processamento de dados

- Tratar dados categóricos.
  - No arquivo que abrimos no Python, temos a coluna País.
  - Nela temos três diferentes países: Brasil, Espanha e Alemanha.
  - Vamos substituir essas informações por dados numéricos. Para isso utilizamos três opções. Veja no código abaixo. Na linha 54 dicionário chamado *subs*. Nele contém a chave: país e o valor: número inteiro. Na linha 56 fazemos a substituição (replace) na coluna País usando o dicionário *subs*.
  - Na linha 58 a 63 tem a opção de substituir por número inteiros, no entanto, de forma automática usando o método *cat.codes*.
  - Na opção 3, linha 71, utilizamos a função *get_dummies*. Essa função transforma cada categoria em uma nova coluna e a linhas receberão valores 1 ou 0 (verdadeiro ou falso).

Pré-Processamento de dados

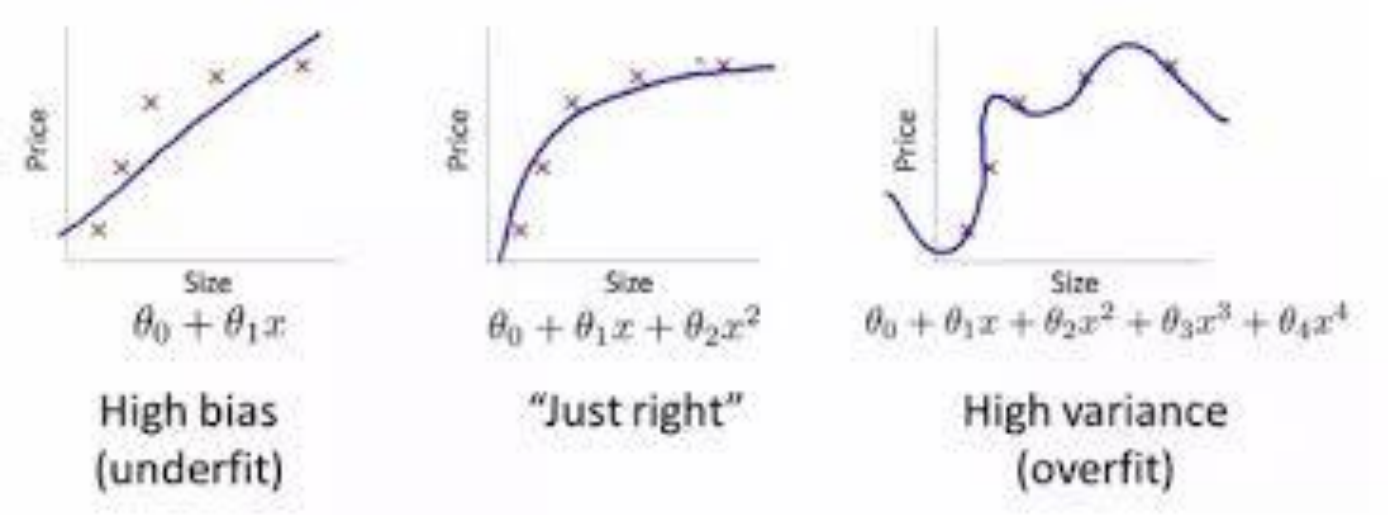
```
50 ############## Tratar dados categóricos com Pandas
51
52 ### OPÇÃO 1 - substituir o valor por um valor numérico a sua escolha
53
54 subs = {'Brasil':1, 'Espanha':2, 'Alemanha':3 }
55 df = X.copy()
56 df['Pais'] = df['Pais'].replace(subs)
57
58 ### OPÇÃO 2 - substituir por valor de forma automática
59 df = X.copy()
60 #transformar o dados em categórico
61 df["Pais"] = df["Pais"].astype('category')
62 #converter cada pais em um número
63 df['Pais'] = df['Pais'].cat.codes
64 #OBS: o algorítmos poderá interpretar valores pequenos como ruins.
65 #Então a opção 3 poderá ser melhor (hot encoding), veja:
66
67 ###OPÇÃO 3 - hot enconding - converte cada categoria em uma nova coluna
68 #então cada linha assume valor 0 e 1, tipo verdadeiro e falso.
69 #issa opção deve ser a preferida pois não irá gerar pesos diferentes
70 df = X.copy()
71 df = pd.get_dummies(df, columns=["Pais"])
72
73 #usando a opção 2 para a variável y
74 dfy = y.copy()
75 dfy = dfy.astype('category')
76 dfy = dfy.cat.codes
```

Pré-Processamento de dados

- Dividir o conjunto de dados em dados de treinamento e dados de teste.
  - Quando trabalhamos com modelos de machine learning precisamos dividir o conjunto de dados em dados de treinamento e teste.
  - O conjunto de dados de treinamento é para o “ajuste” do modelo.
  - Existe vários algoritmos a serem testados e diferentes características do fenômeno a ser aprendido (variáveis dependentes) que podemos utilizar.

Pré-Processamento de dados

- Dividir o conjunto de dados em dados de treinamento e dados de teste.
    - Além disso, os modelos podem ser excessivamente treinados (*overfitting*) e acabar por “decorar” os objetivos. Quando o modelo “decora” os resultados irá gerar um modelo aparentemente muito bom, no entanto, ele é muito ruim.



Pré-Processamento de dados

- Dividir o conjunto de dados em dados de treinamento e dados de teste.
  - Isso significa que ele foi excessivamente treinado.
  - Precisamos gerar bons modelos, para isso, os modelos deverão ter capacidade de generalização, pelo menos dentro da faixa de variação dos dados.
  - Para termos como testar nossos modelos precisamos separar o conjunto de dados em dados de treinamento e dados de teste.
  - É comum utilizarmos 20, 25 ou 30% dos dados para teste.

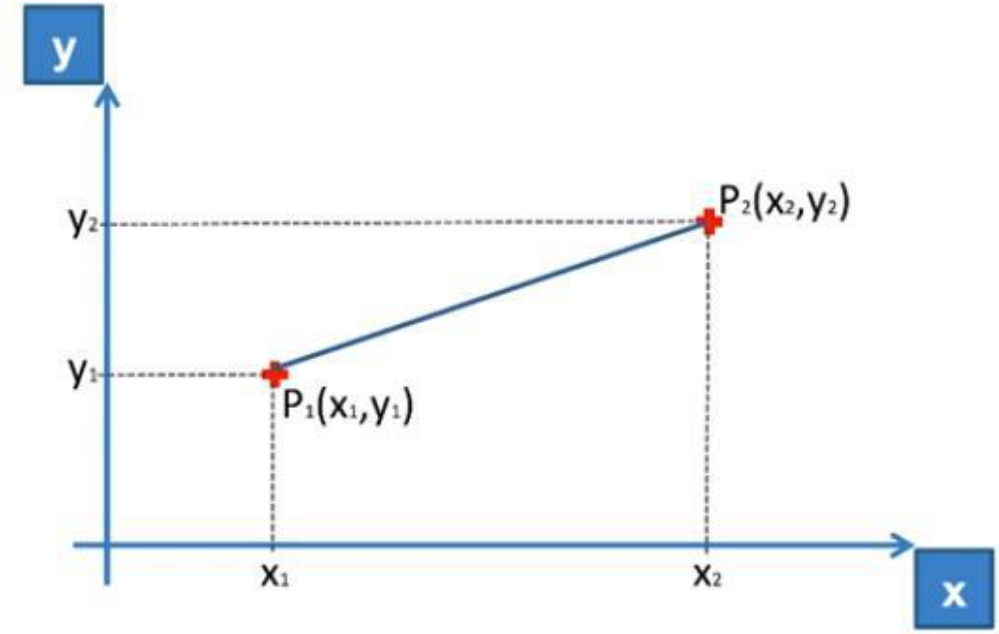
Pré-Processamento de dados

- O pandas oferece uma função que faz divisão do conjunto de dados em dados de treinamento e dados de teste.
- Veja o código abaixo. Na linha 88 é importado a função *train_test_split* do pacote *Scikit Learn*. Essa função que divide o conjunto de dados.
- Na linha 90 são criadas 4 (quatro) variáveis diferentes para treinamento e teste, das variáveis X e y.
- Na função *train_test_split* são passados os conjuntos dados X e y (dfx e dfy), e o tamanho do conjunto de teste (*test_size* de 20%).
- A variável *random_state* é para sempre gerar o mesmo conjunto de dados, apenas para todos obterem os mesmos resultados.

```
88 from sklearn.model_selection import train_test_split
89
90 X_train, X_test, y_train, y_test = train_test_split(dfx, dfy, test_size = 0.2, random_state = 0)
```

Normalizar e Padronizar as variáveis

- Muitos algoritmos utilizaram a distância euclidiana.



- Para cálculo da distância euclidiana é utilizado a seguinte equação:

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Normalizar e Padronizar as variáveis

- Pela equação é possível observar que a distância euclidiana será influenciada pela escala das variáveis.
- Se x tiver uma maior escala numérica em relação a y, a variação de x irá impactar mais na distância.
- Por isso é importante realizar a normalização ou padronização das variáveis.
- Recomenda-se realizar a normalização inclusive nas variáveis binárias (0 e 1).
- Abaixo é apresentado um código para normalizar e padronizar as variáveis.
- Para isso é utilizado o pacote Scikit Learn.

Normalizar e Padronizar as variáveis

```
92 ############ Normalizar ou padronizar as variáveis
93
94 ### OPÇÃO 1 -  padronizar
95 from sklearn.preprocessing import StandardScaler #padronizar
96
97 scaleX = StandardScaler()           #cria um objeto
98 scaleX = scaleX.fit(X_train)        #ajusta aos dados
99 X_train = scaleX.transform(X_train) #transforma os dados
100 X_test = scaleX.transform(X_test)   #utransforma os dados
101 #OBS: a variável X_test tem duas dimensões (dataframe). Se for transformar
102 # y_test deverá transformar em um dataframe (y_test.to_frame()). Veja doc
103
104 ### OPÇÃO 2 -  normalizar
105 from sklearn.preprocessing import MinMaxScaler   #normalizar
106
107 minmaxX = MinMaxScaler()                  #cria o objeto
108 X_train = minmaxX.fit_transform(X_train) #ajusta e transforma
109 X_test = minmaxX.transform(X_test)       #tranfoma
```

Normalizar e Padronizar as variáveis

- Na linha 95 é importado a classe *StandardScaler* para padronizar os dados.
- Na linha 97 é criado o objeto com nome *scaleX* com a classe.
- Na linha 98 é ajudados os dados de treinamento.
- Na linha 99 as variáveis são padronizadas com o objeto *scaleX*.
- Na linha 100 o mesmo objeto *scaleX* é utilizado para padronizar também as variáveis de teste.
- A padronização é realizada com base na média e desvio padrão populacional dos dados. Veja abaixo a equação de padronização.

$$X_{novo} = \frac{x - média(x)}{desvio\ padrão\ (x)}$$

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Normalizar e Padronizar as variáveis

- Na linha 105 é importado a classe *MinMaxScaler* para normalização dos dados.
- Da mesma forma é criado o objeto, ajudado e então o objeto é utilizado para normalizar os dados.
- A equação de normalização é realizada com base nos valores máximos e mínimos dos dados, veja equação abaixo.
- Então os dados serão normalizados entre 0 e 1.

$$x_{novo} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

INSTITUTO FEDERAL
Sudeste de Minas Gerais

INSTITUTO FEDERAL
Sudeste de Minas Gerais