Trabalho de Modelagem

July 1, 2019

Trabalho de Modelagem e Simulação de Sistemas Dinâmicos *(01/07/2019)*
Profŕ. Guilherme
Nome: Marcelo Augusto de Carvalho - **164450031**
Introdução: O presente trabalho tem como objetivo fazer a análise de um banco de dados, através dos recursos oferecidos pelo Python, e aplicar as funções para estimação de parâmetros vistas em sala.

`Obs: O trabalho foi desenvolvido dentro da plataforma Anaconda que possui uma estrutura de apresentação das informações um pouco diferente das ferramentas mais conhecidas como Word e LibreOffice, o que pode gerar certa limitação visual, pois expõe também todos os códigos utilizados.`

Tópicos trabalhados:
1) Escolha do banco de dados
2) Plotagem do gráfico
3) Escola do modelo
4) Separação dos dados
5) Estimação dos parâmetros
6) Validação
7) Conclusão
Primeiramente, vamos importar as bibliotecas necessárias

```
In [4]: import matplotlib.pyplot as plt
        import numpy as np
        from sklearn import datasets, linear_model
        import pandas as pd
        from sklearn import linear_model
        from datetime import datetime
```

1) A seguir, é importado o banco de dados que usaremos para análise, que no caso será sobre registros imobiliários nos EUA, em formato .csv

```
In [5]: df = pd.read_csv("HousingBD.csv")
```

Estes são os 5 primeiros itens do banco de dados:

```
In [6]: df.head()
```

```
Out[6]:    Unnamed: 0    price  lotsize  bedrooms  bathrms  stories driveway recroom  \
        0           1  42000.0     5850         3        1        2      yes      no
        1           2  38500.0     4000         2        1        1      yes      no
        2           3  49500.0     3060         3        1        1      yes      no
        3           4  60500.0     6650         3        1        2      yes     yes
        4           5  61000.0     6360         2        1        1      yes      no

          fullbase gashw airco  garagepl prefarea
        0      yes    no    no         1       no
        1       no    no    no         0       no
        2       no    no    no         0       no
        3       no    no    no         0       no
        4       no    no    no         0       no
```

Para comerçarmos a análsie, serão criados dois vetores formados pelas colunas 'price' (preço) e 'lotsize' (tamanho), chamados *"X"* e *"Y"*

```
In [7]: Y = np.array(df['price'])
        X = np.array(df['lotsize'])

In [8]: X=X.reshape(len(X),1)
        Y=Y.reshape(len(Y),1)
```

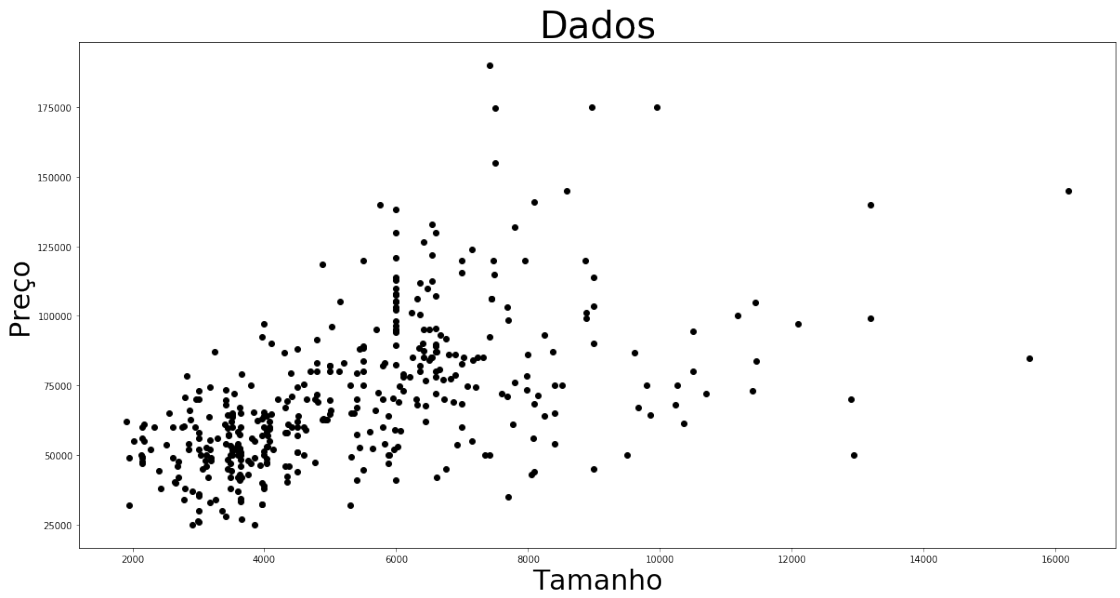
4) Divide-se os dados em conjuntos de **treinamento (75%)** e **teste (25%)**, para X e Y

```
In [9]: #X_train = X[:-375]
        #X_test = X[-125:]
        #n = X.shape[0]     (outro modo)

        n = len(X)

        X_train = X[:round(0.75*n)]
        X_test = X[round(0.25*n):]

In [10]: #Y_train = Y[:-375]
         #Y_test = Y[-125:]
         #n = Y.shape[0]     (outro modo)
         n = len(Y)

         Y_train = Y[:round(0.75*n)]
         Y_test = Y[round(0.25*n):]
```

2) Por fim, os dados são exibidos por meio de um gráfico de pontos **Tamanho x Preço**

```
In [11]: fig = plt.figure(figsize=(20, 10))
         th = fig.add_subplot(1, 1, 1)

         plt.scatter(X_test, Y_test,  color='black')
         plt.title('Dados', fontsize=40)
```

```
plt.xlabel('Tamanho', fontsize=30)
plt.ylabel('Preço', fontsize=30)

plt.show()
```



Tamanho do banco de dados (em número de linhas):

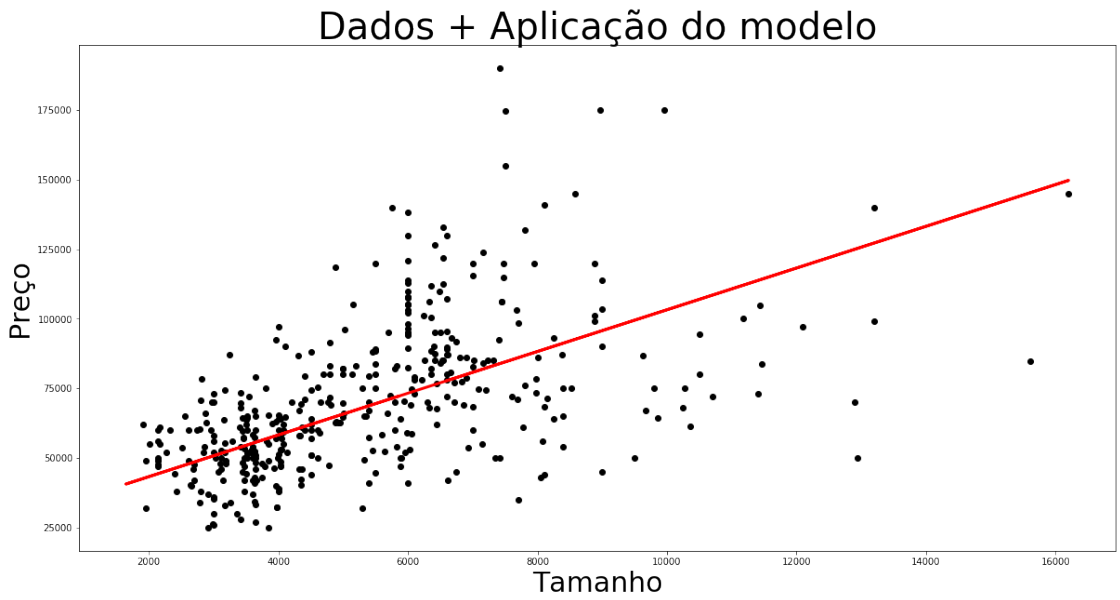
```
In [17]: len(X) #ou len(Y)

Out[17]: 546
```

3) Tendo em vista o gráfico gerado, temos que o comportamento dos dados não possui aparência de uma determinada gunção conhecida, como uma função linear, exponencial, logarítimica, etc., pois os pontos no gráfico possuem uma tendência aleatória, aparentemente. Por isso, foi escolhido o método dos ***mínimos quadrados***.

5) A seguir faremos a análise dos gráficos das partições de treinamento e teste ao ser aplicado o método, respectivamente:

```
In [29]: fig = plt.figure(figsize=(20, 10))
         th = fig.add_subplot(1, 1, 1)

         plt.scatter(X_test, Y_test,  color='black')
         plt.title('Dados + Aplicação do modelo', fontsize=40)
         plt.xlabel('Tamanho', fontsize=30)
         plt.ylabel('Preço', fontsize=30)

         #plt.xticks(())
         #plt.yticks(())
```

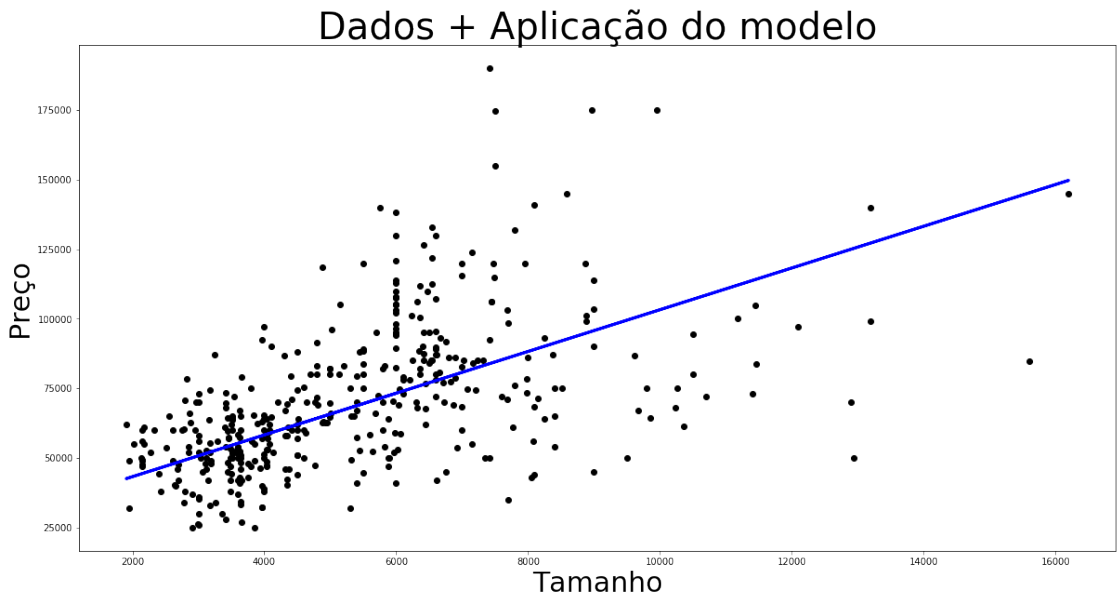
```
        # Cria objeto de regressão linear
        regr = linear_model.LinearRegression()

        # Treine o modelo usando os conjuntos de treinamento
        regr.fit(X_train, Y_train)

        # Saídas de plotagem

        plt.plot(X_train, regr.predict(X_train), color='red',linewidth=3)

        plt.show()
```



Comparando com a partição de treinamento:

```
In [30]: fig = plt.figure(figsize=(20, 10))
         th = fig.add_subplot(1, 1, 1)

         plt.scatter(X_test, Y_test,  color='black')
         plt.title('Dados + Aplicação do modelo', fontsize=40)
         plt.xlabel('Tamanho', fontsize=30)
         plt.ylabel('Preço', fontsize=30)

         #plt.xticks(())
         #plt.yticks(())

         # Cria objeto de regressão linear
         regr = linear_model.LinearRegression()

         # Treine o modelo usando os conjuntos de treinamento
```

```
regr.fit(X_train, Y_train)

# Saídas de plotagem

plt.plot(X_test, regr.predict(X_test), color='blue',linewidth=3)

plt.show()
```



6) Sendo assim, foi aplicada a função de regressão linear (mínimos quadrados), nas partições de treinamento e teste. No gráfico pode ser observada a função em meio os dados dados e constatada eficiência do método.

Pode-se observar que a aplicação do modelo na partição de teste consegue aproximar perfeitamente todos os dados.

7) Deste modo, foi possível constatar através dos gráficos obtidos pelas funções de leitura e manipuçação de dados em Python, além da própria função de regressão linear (mínimos quadrados), que o modelo encontrado mostrado-se eficiente ao ser comparado aos dados originais.