Curso de Data Science

Prof. MSc. Eng. Marcelo Bianchi

Trabalho 1

Grupo 4

Integrantes: Daniel Moreira, Lia Morimoto, Raphael Bezerra, Thainan Abreu

Parte 1: Regressão Linear Simples

Predição: Nota Teste Aptidão Física

Importando as bibliotecas

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

1) Importar o Dataset e aplicar a técnica **Missing Data**

In [2]:
```
df1 = pd.read_csv('dataset1.csv')
df1.columns= ["horas","nota"]
# apontando as variáveis
X = df1.iloc[:,0].values
X = X.reshape(-1, 1)
y = df1.iloc[:,1].values
y = y.reshape(-1, 1)


# Não há valores faltando, logo não há necessidade para o tratamento de missing data
df1.isnull().sum()
```

Out[2]:
```
horas    0
nota     0
dtype: int64
```

In [3]:
```
df1.describe(include = "all")   # resumo do dataset
```

Out[3]:

	horas	nota
count	10.000000	10.000000
mean	10.500000	70.100000
std	5.421152	16.960411
min	1.000000	43.000000
25%	7.250000	56.750000
50%	10.500000	71.500000
75%	14.750000	83.750000
max	19.000000	96.000000

2) Dividir o dataset entre o Training Set e o Test Set

In [4]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

# conjunto de treino 80% e 20% de teste
```

3) Aplicar Feature Scaling (Se for aplicável, se não for então justificar)

Não há necessidade de aplicar Featuring Scale. Não há grandes discrepâncias entre os valores nem das entradas e nem das saídas e o propósito é exatamente explicitar a relação entre as unidades de tempo de treino e a nota no teste físico.

4) Aplicar Dummy Variable (Se for aplicável, se não for então justificar)

Não há necessidade de aplicar Dummy Variable Encoding pois não há variáveis categóricas.

---

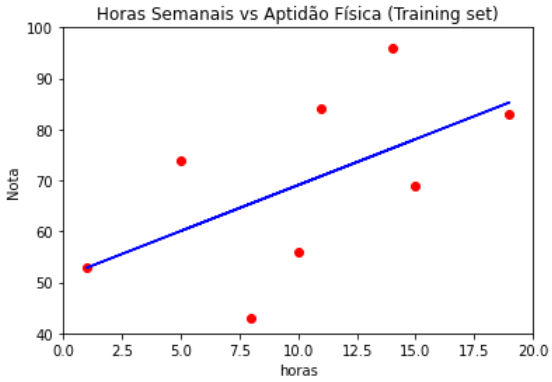
5) Aplicar a Simple Linear Regression

In [5]:
```
# Fitting Simple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)


# Predicting the Test set results
y_pred = regressor.predict(X_test)
```

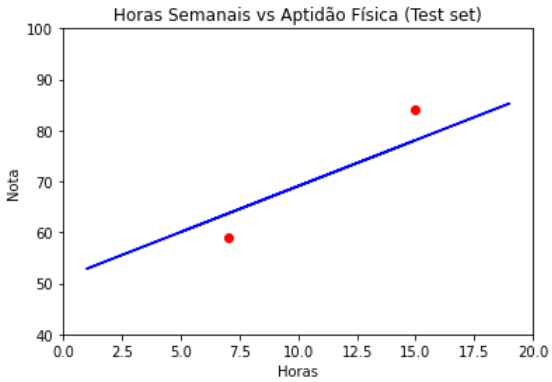
---

6) Construir o Gráfico (Scatter Plot)

In [6]:
```
#Visualizando o gráfico de dispersão dos resultados do conjunto de treino
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')

plt.title('Horas Semanais vs Aptidão Física (Training set)')
plt.xlabel('horas')
plt.xlim(0, 20)

plt.ylabel('Nota')
plt.ylim(40, 100)
plt.show()
```



In [7]:
```
# Visualising the Test set results
#Visualizando o gráfico de dispersão dos resultados do conjunto de teste
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Horas Semanais vs Aptidão Física (Test set)')
plt.xlabel('Horas')
plt.xlim(0, 20)

plt.ylabel('Nota')
plt.ylim(40, 100)
plt.show()
```



---

7) Criar a tabela no banco de dados SQLite

In [8]:
```
import sqlite3 as sq3

#Criando uma conexão com o banco de dados
connection = sq3.connect('database.db')

#Usando o pandas para criar uma tabela no banco de dados
df1.to_sql(name = 'teste_aptidao', con = connection, if_exists = 'append', index = False)
```

8) Aplicar uma consulta em linguagem SQL que irá trazer uma listagem da tabela

In [9]:
```
teste = pd.read_sql('select * from teste_aptidao', connection)

print(teste)
```

```
   horas  nota
0      1    53
1      5    74
2      7    59
3      8    43
4     10    56
5     11    84
6     14    96
7     15    69
8     15    84
9     19    83
```

In [10]:
```
connection.close()
```