T3.1 / N2 e Presença em aula 24

Atividade de Presença para Probest.

Para nossa análise, X são os casos e Y são as mortes.

Essa análise será direta ao ponto. Entregando os resultados omitindo o processo. Caso queira ver todo o processo de raciocínio, olhar o outro arquivo.

In [2]:
```
import pandas as pd
from regressao_linear import RegressaoLinearSimples

# Leitura de Dados:
dados = pd.read_csv('./Dados semana 1 a 20 - Covid 2021 - Página1.csv')
```

Separamos os dados em X impares (Casos) e Y impares (Obitos)

In [3]:
```
impares = dados.loc[dados.Semana % 2 != 0]
X_impares = impares.Casos
Y_impares = impares.Obitos
```

Foi criado um modelo de Regressao Linear com todos os calculos para que possa ser reutilizado. Se quiser ver o arquivo estará sendo enviado com o código fonte. Porém, como disse anteriormente, o cálculo completo estará na análise grande.

In [4]:
```
model = RegressaoLinearSimples()
model.fit(X_impares.values ,Y_impares.values)
```

Aqui Já podemos prever um valor de X:

In [5]:
```
# Quando tiver 450 mil casos, as mortes poderão ser de:
model.prever(450_000)
```

Out[5]:
```
15433.629708967546
```

Vamos agora solucionar as questões:

1: Faça X = Nº de casos informados e Y = Nº de mortes por semana epidemiológica no Ceará (nas semanas de 1 a 20 de 2021). Elabore a distribuição de frequências para variável (Y).

In [53]:
```
n = len(dados.Obitos)
k = sqrt(n) if n >= 25 else 5
print(f'Teremos {k} Classes.')
```

```
Teremos 5 Classes
```

In [157…
```
value = dados.Obitos.min()
classes = []
toAdd = (dados.Obitos.max() - dados.Obitos.min()) / k
while value <= dados.Obitos.max():
    classes.append((value, value + toAdd))
    value += toAdd
print('Classes retiradas', classes)
```

```
Classes retiradas [(6665, 9560.2), (9560.2, 12455.400000000001), (12455.400000000001, 15350.600000000002), (15350.600000000002, 18245.800000000003), (18245.800000000003, 21141.000000000004)]
```

In [124…
```
Y_freq = [] # (min, max, classes)
for min, max in classes:
    item_classe = dados.Obitos.loc[(dados.Obitos >= min) & (dados.Obitos < max)]
    Y_freq.append((min, max, item_classe.values))
```

Mostrar os valores das classes e Das frequências Absolutas e Relativas.

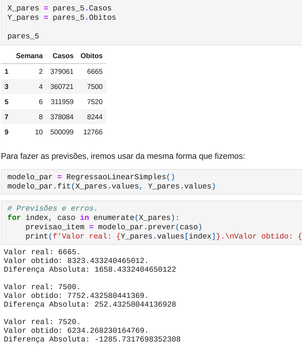
In [156…
```
import numpy as np
for min, max, items in Y_freq:
    to_print_classe = f'Classe: [{int(min)}-{int(max)}[ :'
    tam_classe = len(to_print_classe)


    item_print = f'Itens: {items}'
    tam_item_print = len(item_print)

    freq_abs = f'Frequência Absoluta: {len(items)}'
    tam_freq_abs = len(freq_abs)

    freq_rel = f'Frequência Relativa: {(len(items) / len(dados.Obitos)) * 100}'
    tam_freq_rel = len(freq_rel)

    print(to_print_classe)
    print(item_print)
    print(freq_abs)
    print(freq_rel)
    print('-'*(np.max([tam_classe, tam_item_print, tam_freq_abs, tam_freq_rel])+1))
```

```
Classe: [6665-9560[ :
Itens: [6665 6906 7067 7149 7445 7500 7520 8244]
Frequência Absoluta: 8
Frequência Relativa: 40.0
----------------------------------------------------
Classe: [9560-12455[ :
Itens: [10104]
Frequência Absoluta: 1
Frequência Relativa: 5.0
-----------------------
Classe: [12455-15350[ :
Itens: [12766 13399 13403 14879]
Frequência Absoluta: 4
Frequência Relativa: 20.0
-------------------------------
Classe: [15350-18245[ :
Itens: [15661 16945 17788 17814]
Frequência Absoluta: 4
Frequência Relativa: 20.0
-------------------------------
Classe: [18245-21141[ :
Itens: [19643 20344 21141]
Frequência Absoluta: 3
Frequência Relativa: 15.0
-----------------------------
```

Aproveitando para também plotar os dados em um histograma

In [158…
```
import seaborn as srn
srn.histplot(dados.Obitos, stat='frequency')
```

Out[158…
```
<AxesSubplot:xlabel='Obitos', ylabel='Frequency'>
```



In [61]:
```
(dados.Obitos.max() - dados.Obitos.min()) / k
```

Out[61]:
```
2895.2
```

2: Encontre a reta de regressão linear Y em função de X (para as semanas ímpares das 20 semanas), se ela existir.

In [7]:
```
print('A formula da reta ficou como:', model.formula_str)
```

```
A formula da reta ficou como: Y = -16424.80179962504 + 0.07079651446353907 * X
```

PS: A questão 3 foi pego como referencia os dados impares, em excessão da 3.3, que pede 5 semanas pares.

3.1: Plote em um gráfico;

In [8]:
```
import matplotlib.pyplot as plt
plt.scatter(X_impares, Y_impares)
plt.plot(X_impares,[model.prever(i) for i in X_impares], color='red')
plt.ylabel('Obitos')
plt.xlabel('Casos')
```

Out[8]:
```
Text(0.5, 0, 'Casos')
```

3.2 - Determine os coeficientes de determinação e correlação linear entre as variáveis X e Y;

In [9]:
```
E_X, E_Y, E_X_Y = model.get_mean(X_impares.values, Y_impares.values)

previsoes = [model.prever(i) for i in X_impares]
previsoes_2 = [(p - E_Y)**2 for p in previsoes]

real = Y_impares.values
real_2 = [(r - E_Y)**2 for r in real]
```

In [10]:
```
R_2 = sum(previsoes_2) / sum(real_2)
print('Coeficiente de Determinação:', R_2)
```

```
Coeficiente de Determinação: 0.6771566077941553
```

Sabemos que $R^2 = P^2$

Então:

In [11]:
```
# p = raiz quadrada de R²
from math import sqrt
p = sqrt(R_2)
print('Coeficiente de Correlação linear:', p)
```

```
Coeficiente de Correlação linear: 0.8228952593095644
```

3.3: Escolha 5 semanas pares e faça a previsão do número de mortes por semana, na reta ajustada de Y; determinando o erro cometido no processo para cada semana.

In [12]:
```
pares = dados.loc[dados.Semana % 2 == 0]
pares_5 = pares[:5]

X_pares = pares_5.Casos
Y_pares = pares_5.Obitos

pares_5
```

Out[12]:

	Semana	Casos	Obitos
1	2	379061	6665
3	4	360721	7500
5	6	311959	7520
7	8	378084	8244
9	10	500099	12766

Para fazer as previsões, iremos usar da mesma forma que fizemos:

In [13]:
```
modelo_par = RegressaoLinearSimples()
modelo_par.fit(X_pares.values, Y_pares.values)
```

In [14]:
```
# Previsões e erros.
for index, caso in enumerate(X_pares):
    previsao_item = modelo_par.prever(caso)
    print(f'Valor real: {Y_pares.values[index]}.\nValor obtido: {previsao_item}.\nDiferença Absoluta: {previsao
```

```
Valor real: 6665.
Valor obtido: 8323.433240465012.
Diferença Absoluta: 1658.433240465122

Valor real: 7500.
Valor obtido: 7752.432580441369.
Diferença Absoluta: 252.43258044136928

Valor real: 7520.
Valor obtido: 6234.268230164769.
Diferença Absoluta: -1285.7317698352308

Valor real: 8244.
Valor obtido: 8293.015157321986.
Diferença Absoluta: 49.015157321986641

Valor real: 12766.
Valor obtido: 12091.850791606868.
Diferença Absoluta: -674.1492083931316
```

In [15]:
```
print(f'Formula do Modelo PAR: {modelo_par.formula_str}')
```

```
Formula do Modelo PAR: Y = -3478.316006493654 6 + 0.03113416903073296 * X
```

In [ ]:
```

```