```
#1 BIBLIOTECAS
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from scipy.stats import pearsonr
from sklearn import metrics

#2 IMPORTANDO ARQUIVO
dados=pd.read_csv('insurance.csv')

#3 ANALISANDO OS DADOS I (ANÁLISE EXPLORATÓRIA DOS DADOS - AED)
print(dados.head())
print(dados.shape)
```

```
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520
(1338, 7)
```

```
#4 ANALISANDO OS DADOS II - AED
print(dados.dtypes)
```

```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

```
#5 ANALISANDO OS DADOS III - AED
dados.describe().round(2)
```

	age	bmi	children	charges
count	1338.00	1338.00	1338.00	1338.00
mean	39.21	30.66	1.09	13270.42
std	14.05	6.10	1.21	12110.01
min	18.00	15.96	0.00	1121.87
25%	27.00	26.30	0.00	4740.29
50%	39.00	30.40	1.00	9382.03
75%	51.00	34.69	2.00	16639.91
max	64.00	53.13	5.00	63770.43

```
#6 PRÉ PROCESSSANDO OS DADOS I
#Convertendo as variáveis SEX, SMOKER e REGION em numéricas (ENCODING)
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

#sex
le.fit(dados.sex)
dados.sex = le.transform(dados.sex)

# smoker
le.fit(dados.smoker)
dados.smoker = le.transform(dados.smoker)
```

```
#region
le.fit(dados.region)
dados.region = le.transform(dados.region)

print(dados.head())
print(dados.shape)
```

```
   age  sex     bmi  children  smoker  region      charges
0   19    0  27.900         0       1       3  16884.92400
1   18    1  33.770         1       0       2   1725.55230
2   28    1  33.000         3       0       2   4449.46200
3   33    1  22.705         0       0       1  21984.47061
4   32    1  28.880         0       0       1   3866.85520
(1338, 7)
```

```
#7 ANALISANDO OS DADOS IV - AED
#CORRELAÇÕES
dados.corr().round(2)
```

	age	sex	bmi	children	smoker	region	charges
age	1.00	-0.02	0.11	0.04	-0.03	0.00	0.30
sex	-0.02	1.00	0.05	0.02	0.08	0.00	0.06
bmi	0.11	0.05	1.00	0.01	0.00	0.16	0.20
children	0.04	0.02	0.01	1.00	0.01	0.02	0.07
smoker	-0.03	0.08	0.00	0.01	1.00	-0.00	0.79
region	0.00	0.00	0.16	0.02	-0.00	1.00	-0.01
charges	0.30	0.06	0.20	0.07	0.79	-0.01	1.00

```
#8 FILTRANDO DADOS I
#FILTRO PARA SEPARAR SOMENTE OS FUMANTES
dados = dados[dados['smoker'] == 1]
print(dados.head())
print(dados.shape)
```

```
    age  sex    bmi  children  smoker  region     charges
0    19    0  27.90         0       1       3  16884.9240
11   62    0  26.29         0       1       2  27808.7251
14   27    1  42.13         0       1       2  39611.7577
19   30    1  35.30         0       1       3  36837.4670
23   34    0  31.92         1       1       0  37701.8768
(274, 7)
```

```
#9 FILTRANDO DADOS II
#FILTRO PARA SEPARAR SOMENTE AS II - MULHERES
dados = dados[dados['sex'] == 1]
print(dados.head())
print(dados.shape)
```

```
    age  sex    bmi  children  smoker  region      charges
14   27    1  42.13         0       1       2  39611.75770
19   30    1  35.30         0       1       3  36837.46700
29   31    1  36.30         2       1       3  38711.00000
30   22    1  35.60         0       1       3  35585.57600
34   28    1  36.40         1       1       3  51194.55914
(159, 7)
```

```
#10 ESCOLHA DAS VARIÁVEIS : IMC X GASTO COM SEGURO
X = dados['bmi'].values
Y = dados['charges'].values


print(X)
```

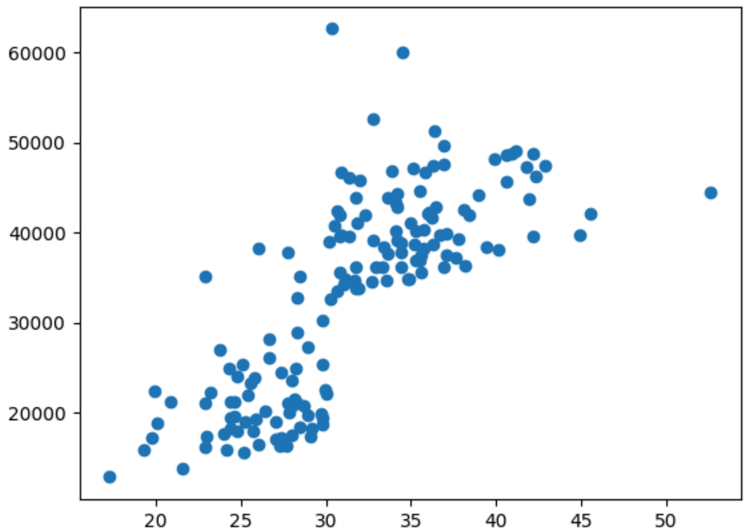
```
[42.13  35.3   36.3   35.6   36.4   36.67  39.9   35.2   28.    34.43
 36.955 31.68  23.98  37.62  22.895 29.83  19.95  19.3   28.025 35.09
 31.35  25.3   28.69  30.495 24.42  25.175 35.53  31.92  41.895 27.74  34.8
 24.64  29.07  17.29  34.21  31.825 33.63  31.92  24.32  36.955 42.35
 19.8   34.2   40.565 45.54  27.7   25.41  34.39  35.97  30.8   36.48
 27.36  32.3   32.9   40.15  30.685 33.88  35.86  32.775 26.695 30.
```

```
 25.1   28.31  28.5   25.7   34.4   23.21  30.25  28.3   26.07  42.13
 25.84  40.565 37.8   25.6   34.1   33.535 26.41  28.31  38.06  32.015
 31.35  35.3   31.13  35.75  24.42  31.73  35.5   29.15  34.105 38.17
 27.1   24.4   20.9   28.5   24.795 31.79  28.025 30.78  24.13  28.93
 28.975 38.94  40.92  31.73  22.895 34.2   29.7   42.9   30.2   27.835
 30.8   34.96  24.795 22.895 25.9   22.99  32.7   28.215 20.13  36.08
 26.03  21.565 37.07  30.685 52.58  30.9   29.8   41.14  37.07  31.68
 36.19  38.39  33.33  35.75  32.8   44.88  27.36  29.81  35.625 33.4
 34.485 41.8   36.96  33.63  29.83  27.3   23.76  31.065 27.06  29.925
 36.3   39.4   34.9   30.36  30.875 27.8   24.605 28.12  26.695]
```

```
print(Y)
```

```
[39611.7577  36837.467   38711.      35585.576   51194.55914 39774.2763
 48173.361   38709.176   23568.272   37742.5757  47496.49445 34303.1672
 17663.1442  37165.1638  21098.55405 30184.9367  22412.6485  15820.699
 17560.37975 47055.5321  39556.4945  18972.495   20745.9891  40720.55105
 21223.6758  15518.18025 36950.2567  43753.33705 20984.0936  34779.615
 19515.5416  17352.6803  12829.4551  44260.7499  41097.16175 43921.1837
 33750.2918  24869.8368  36219.40545 46151.1245  17179.522   42856.838
 48549.17835 42112.2356  16297.846   21978.6769  38746.3551  42124.5153
 35491.64    42760.5022  24393.6224  41919.097   36085.219   38126.2465
 42303.69215 46889.2612  46599.1084  39125.33225 26109.32905 22144.032
 25382.297   28868.6639  35147.52848 17942.106   36197.699   22218.1149
 32548.3405  21082.16    38245.59327 48675.5177  23807.2406  45702.02235
 39241.442   23306.547   40182.246   34617.84065 20149.3229  32787.45859
 42560.4304  45710.20785 46130.5265  40103.89    34806.4677  40273.6455
 19361.9988  36189.1017  44585.45587 18246.4955  43254.41795 36307.7983
 19040.876   18259.216   21195.818   18310.742   17904.52705 43813.8661
 20773.62775 39597.4072  15817.9857  19719.6947  27218.43725 44202.6536
 48673.5588  33732.6867  35069.37452 39047.285   19933.458   47462.894
 38998.546   20009.63365 41999.52    41034.2214  23967.38305 16138.76205
 19199.944   17361.7661  34472.841   24915.22085 18767.7377  42211.1382
 16450.8947  13747.87235 37484.4493  33475.81715 44501.3982  39727.614
 25309.489   48970.2476  39871.7043  34672.1472  41676.0811  41949.2441
 36124.5737  38282.7495  52590.82939 39722.7462  17178.6824  19350.3689
 37465.34375 38415.474   60021.39897 47269.854   49577.6624  37607.5277
 18648.4217  16232.847   26926.5144  34254.05335 17043.3414  22462.04375
 47403.88    38344.566   34828.654   62592.87309 46718.16325 37829.7242
 21259.37795 21472.4788  28101.33305]
```

```
#11 ANÁLISE GRÁFICA - AED
#Gráfico da relação entre IMC x Custo
plt.scatter(X, Y)
plt.show()
```




```
#12 PEARSON
#Calculo do r (Pearson)
r = pearsonr(X, Y)
print(f'Coeficiente de correlação: {r}')
```

```
Coeficiente de correlação: PearsonRResult(statistic=0.7693553500239402, pvalue=2.290057897722594e-32)
```

```
#13 MLS I
#Separar os conjuntos TREINAMENTO e TESTE (70% / 30%)
```

```
x_train, x_test, y_train, y_test = train_test_split(X,Y,test_size=0.3)

#Dados de x (Features)
print(x_train)
```

```
[30.9   24.42  30.8   30.36  20.13  24.42  28.025 23.76  36.955 37.62
 30.78  34.43  37.8   34.1   34.105 19.95  24.13  25.7   52.58  26.03
 21.565 35.86  22.99  40.92  41.895 31.79  38.06  35.2   30.    34.485
 27.06  31.35  35.3   45.54  31.35  25.9   42.35  40.565 33.535 40.565
 32.8   35.09  37.07  22.895 28.93  36.955 32.775 31.68  22.895 35.5
 27.36  33.88  26.695 29.83  31.13  44.88  33.33  20.9   27.7   31.825
 35.75  33.4   26.41  38.94  34.21  34.8   32.3   19.3   40.15  37.07
 27.835 36.3   24.64  27.3   34.4   28.025 24.4   39.4   35.97  35.75
 28.12  29.15  29.81  31.73  28.975 28.69  28.215 27.74  27.36  30.495
 29.7   25.84  41.8   27.1   27.8   42.9   33.63  35.6   38.39  29.925
 36.19  25.1   35.625 30.8   36.96  30.2   23.21  22.895 24.32  33.63
 29.07 ]
```

```
print(x_test)
```

```
[34.2   25.175 34.2   32.015 23.98  24.795 32.9   36.08  30.875 26.07
 36.3   28.31  41.14  24.795 29.8   35.53  30.685 28.    34.9   36.67
 35.3   34.96  28.5   42.13  28.31  32.7   28.3   25.6   31.065 24.605
 38.17  30.685 26.695 31.92  34.39  29.83  28.5   25.41  39.9   36.4
 19.8   31.73  17.29  30.25  25.3   42.13  31.68  36.48 ]
```

```
#Dados de y (Target)
print(y_train)
```

```
[39727.614   19361.9988  35491.64    62592.87309 18767.7377  21223.6758
 17560.37975 26926.5144  36219.40545 37165.1638  39597.4072  37742.5757
 39241.442   40182.246   43254.41795 22412.6485  15817.9857  17942.106
 44501.3982  16450.8947  13747.87235 46599.1084  17361.7661  48673.5588
 43753.33705 43813.8661  42560.4304  38709.176   22144.032   60021.39897
 17043.3414  46130.5265  40103.89    42112.2356  39556.4945  19199.944
 46151.1245  48549.17835 34617.84065 45702.02235 52590.82939 47055.5321
 37484.4493  35069.37452 19719.6947  47496.49445 39125.33225 34303.1672
 16138.76205 44585.45587 17178.6824  46889.2612  28101.33305 18648.4217
 34806.4677  39722.7462  36124.5737  21195.818   16297.846   41097.16175
 38282.7495  38415.474   20149.3229  44202.6536  44260.7499  34779.615
 41919.097   15820.699   38126.2465  39871.7043  20009.63365 47403.88
 19515.5416  16232.847   36197.699   20773.62775 18259.216   38344.566
 42124.5153  40273.6455  21472.4788  18246.4955  19350.3689  33732.6867
 27218.43725 20745.9891  24915.22085 20984.0936  24393.6224  40720.55105
 19933.458   23807.2406  47269.854   19040.876   37829.7242  47462.894
 43921.1837  35585.576   41949.2441  22462.04375 41676.0811  25382.297
 37465.34375 41999.52    49577.6624  38998.546   22218.1149  21098.55405
 24869.8368  37607.5277  17352.6803 ]
```

```
print(y_test)
```

```
[39047.285   15518.18025 42856.838   45710.20785 17663.1442  17904.52705
 36085.219   42211.1382  46718.16325 38245.59327 38711.      28868.6639
 48970.2476  23967.38305 25309.489   36950.2567  42303.69215 23568.272
 34828.654   39774.2763  36837.467   41034.2214  35147.52848 39611.7577
 32787.45859 34472.841   21082.16    23306.547   34254.05335 21259.37795
 36307.7983  33475.81715 26109.32905 33750.2918  38746.3551  30184.9367
 18310.742   21978.6769  48173.361   51194.55914 17179.522   36189.1017
 12829.4551  32548.3405  18972.495   48675.5177  34672.1472  42760.5022 ]
```

```
#14 PRÉ PROCESSSANDO OS DADOS II
# Carregar os dados no modelo de ML
# Transformar os dados de treino e teste em arrays coluna
x_train=x_train.reshape(-1,1)
y_train=y_train.reshape(-1,1)
x_test=x_test.reshape(-1,1)
y_test=y_test.reshape(-1,1)

print(x_train)
```

```
 [33.4  ]
 [26.41 ]
 [38.94 ]
 [34.21 ]
 [34.8  ]
 [32.3  ]
 [19.3  ]
 [40.15 ]
 [37.07 ]
 [27.835]
 [36.3  ]
 [24.64 ]
 [27.3  ]
 [34.4  ]
 [28.025]
 [24.4  ]
 [39.4  ]
 [35.97 ]
 [35.75 ]
 [28.12 ]
 [29.15 ]
 [29.81 ]
 [31.73 ]
 [28.975]
 [28.69 ]
 [28.215]
 [27.74 ]
 [27.36 ]
 [30.495]
 [29.7  ]
 [25.84 ]
 [41.8  ]
 [27.1  ]
 [27.8  ]
 [42.9  ]
 [33.63 ]
 [35.6  ]
 [38.39 ]
 [29.925]
 [36.19 ]
 [25.1  ]
 [35.625]
 [30.8  ]
 [36.96 ]
 [30.2  ]
 [23.21 ]
 [22.895]
 [24.32 ]
 [33.63 ]
 [29.07 ]]
```

```
print(y_train)
```

```
 [19933.458  ]
 [23807.2406 ]
 [47269.854  ]
 [19040.876  ]
 [37829.7242 ]
 [47462.894  ]
 [43921.1837 ]
 [35585.576  ]
 [41949.2441 ]
 [22462.04375]
 [41676.0811 ]
 [25382.297  ]
 [37465.34375]
 [41999.52   ]
 [49577.6624 ]
 [38998.546  ]
 [22218.1149 ]
 [21098.55405]
 [24869.8368 ]
 [37607.5277 ]
 [17352.6803 ]]
```

```
print(x_test)
```

```
[[34.2  ]
 [25.175]
 [34.2  ]
 [32.015]
 [23.98 ]
 [24.795]
 [32.9  ]
 [36.08 ]
 [30.875]
 [26.07 ]
 [36.3  ]
 [28.31 ]
 [41.14 ]
 [24.795]
 [29.8  ]
 [35.53 ]
 [30.685]
 [28.   ]
 [34.9  ]
 [36.67 ]
 [35.3  ]
 [34.96 ]
 [28.5  ]
 [42.13 ]
 [28.31 ]
 [32.7  ]
 [28.3  ]
 [25.6  ]
 [31.065]
 [24.605]
 [38.17 ]
 [30.685]
 [26.695]
 [31.92 ]
 [34.39 ]
 [29.83 ]
 [28.5  ]
 [25.41 ]
 [39.9  ]
 [36.4  ]
 [19.8  ]
 [31.73 ]
 [17.29 ]
 [30.25 ]
 [25.3  ]
 [42.13 ]
 [31.68 ]
 [36.48 ]]
```

```
print(y_test)
```

```
[[39047.285  ]
 [15518.18025]
 [42856.838  ]
 [45710.20785]
 [17663.1442 ]
 [17904.52705]
 [36085.219  ]
 [42211.1382 ]
 [46718.16325]
 [38245.59327]
 [38711.     ]
 [28868.6639 ]
 [48970.2476 ]
```

```
 [23967.38305]
 [25309.489  ]
 [36950.2567 ]
 [42303.69215]
 [23568.272  ]
 [34828.654  ]
 [39774.2763 ]
 [36837.467  ]
 [41034.2214 ]
 [35147.52848]
 [39611.7577 ]
 [32787.45859]
 [34472.841  ]
 [21082.16   ]
 [23306.547  ]
 [34254.05335]
 [21259.37795]
 [36307.7983 ]
 [33475.81715]
 [26109.32905]
 [33750.2918 ]
 [38746.3551 ]
 [30184.9367 ]
 [18310.742  ]
 [21978.6769 ]
 [48173.361  ]
 [51194.55914]
 [17179.522  ]
 [36189.1017 ]
 [12829.4551 ]
 [32548.3405 ]
 [18972.495  ]
 [48675.5177 ]
 [34672.1472 ]
 [42760.5022 ]]
```

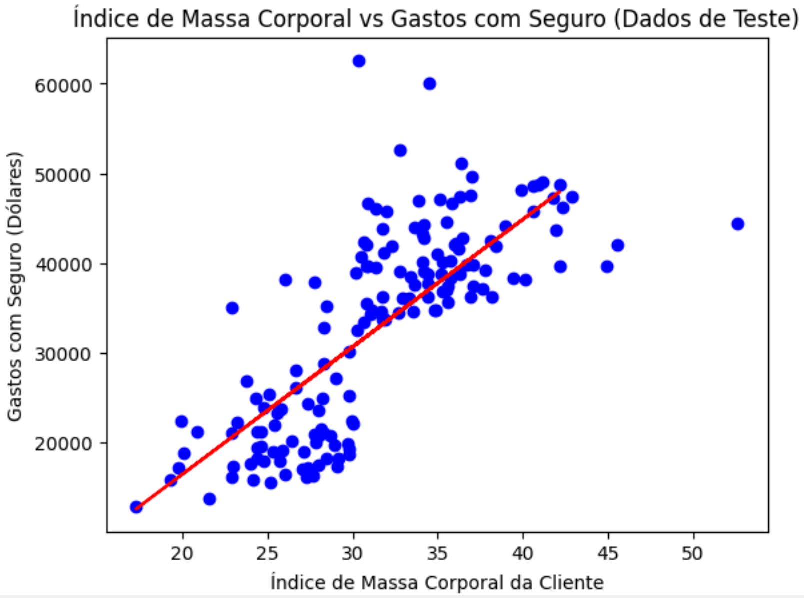
```
# 15 MLS
# Aplicação do Método de MLS (Regressão Linear)
# 15.1 Ajuste do MODELO
reg = LinearRegression()
reg.fit(x_train,y_train)
# 15.2 Predição com o MODELO (TESTE COM x_teste -> pred)
pred = reg.predict(x_test)
print(pred)
```

```
[[36636.58340406]
 [23810.25604725]
 [36636.58340406]
 [33531.26204399]
 [22111.92239723]
 [23270.20015854]
 [34789.02378479]
 [39308.43885345]
 [31911.09437786]
 [25082.22978513]
 [39621.10278902]
 [28265.71712909]
 [46499.70937151]
 [23270.20015854]
 [30383.3046927 ]
 [38526.77901453]
 [31641.06643351]
 [27825.14521988]
 [37631.42319904]
 [40146.94668066]
 [38199.90308189]
 [37716.69518147]
 [28535.74507344]
 [47906.69708156]
 [28265.71712909]
 [34504.78384337]
 [28251.50513202]
 [24414.26592278]
 [32181.12232222]
 [23000.17221419]
 [42278.74624134]
 [31641.06643351]
 [25970.47960208]
 [33396.24807181]
 [36906.61134841]
 [30425.94068392]
 [28535.74507344]
 [24144.23797842]
 [44737.42173467]
```

```
 [39763.22275973]
 [16171.30762145]
 [33126.22012746]
 [12604.09635656]
 [31022.84456091]
 [23987.90601064]
 [47906.69708156]
 [33055.1601421 ]
 [39876.9187363 ]]
```

```
#16 ANÁLISE GRÁFICA - Dados Experimentais x Modelo
plt.scatter(X, Y, color="blue")
plt.plot(x_test, pred, color="red")
plt.title("Índice de Massa Corporal vs Gastos com Seguro (Dados de Teste)")
plt.xlabel("Índice de Massa Corporal da Cliente")
plt.ylabel("Gastos com Seguro (Dólares)")
```

```
Text(0, 0.5, 'Gastos com Seguro (Dólares)')
```




```
#17 CÁLCULO DO R2 (AJUSTE LINEAR)
r_squared = r2_score(y_test, pred)
print(f'Coeficiente r2: {r_squared}')
```

```
Coeficiente r2: 0.6985988031846052
```

```
#18 DETERMINAÇÃO DO AJUSTE (ERRO MÉDIO)
print('MAE (Erro):', metrics.mean_absolute_error(y_test, pred))
```

```
MAE (Erro): 3968.476960701532
```