Open in Colab (https://colab.research.google.com/github/joanby/python-ml-course/blob/master/notebooks/T4%20-%202%20-%20Linear%20Regression%20-%20Regresión%20lineal%20con%20statsmodel-Colab.ipynb)

Regresión lineal simple en Python

El paquete statsmodel para regresión lineal

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
data = pd.read_csv("../datasets/ads/Advertising.csv")
```

In [3]:
```
data.head()
```

Out[3]:

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

In [4]:
```
import statsmodels.formula.api as smf
```

In [5]:
```
lm = smf.ols(formula="Sales~TV", data = data).fit()
```

In [6]:
```
lm.params
```

Out[6]:
```
Intercept    7.032594
TV           0.047537
dtype: float64
```

El modelo lineal predictivo sería Sales = 7.032594 + 0.047537 * TV

In [7]:
```
lm.pvalues
```

Out[7]:
```
Intercept    1.406300e-35
TV           1.467390e-42
dtype: float64
```

In [8]:
```
lm.rsquared
```

Out[8]:
```
0.611875050850071
```

In [9]:
```
lm.rsquared_adj
```

Out[9]:
```
0.6099148238341623
```

In [10]:
```
lm.summary()
```

Out[10]:

OLS Regression Results

Dep. Variable:	Sales	**R-squared:**	0.612
Model:	OLS	**Adj. R-squared:**	0.610
Method:	Least Squares	**F-statistic:**	312.1
Date:	Thu, 14 Jul 2022	**Prob (F-statistic):**	1.47e-42
Time:	01:15:21	**Log-Likelihood:**	-519.05
No. Observations:	200	**AIC:**	1042.
Df Residuals:	198	**BIC:**	1049.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	7.0326	0.458	15.360	0.000	6.130	7.935
TV	0.0475	0.003	17.668	0.000	0.042	0.053

Omnibus:	0.531	**Durbin-Watson:**	1.935
Prob(Omnibus):	0.767	**Jarque-Bera (JB):**	0.669
Skew:	-0.089	**Prob(JB):**	0.716
Kurtosis:	2.779	**Cond. No.**	338.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

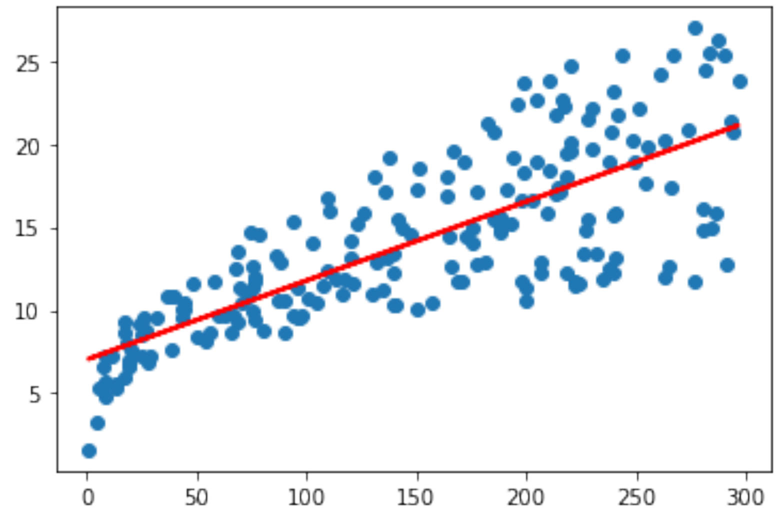
In [52]:
```
sales_pred = lm.predict(pd.DataFrame(data["TV"]))
sales_pred
data.sales_pred = sales_pred
```

In [12]:
```
import matplotlib.pyplot as plt
```

In [13]:
```
#%matplotlib inline
#data.plot(kind = "scatter", x = "TV", y ="Sales")
#plt.plot(pd.DataFrame(data["TV"]), sales_pred, c="red", linewidth = 2
```

In [53]:
```
#corregido
%matplotlib inline
plt.scatter(x = data.TV, y= data.Sales)
plt.plot(data.TV, data.sales_pred, color = 'red',linewidth = 2)
```

Out[53]:
```
[<matplotlib.lines.Line2D at 0x7fabe8b11b50>]
```



In [15]:
```
data["sales_pred"] = 7.032594 + 0.047537*data["TV"]
```

In [16]:
```
data["RSE"] = (data["Sales"]-data["sales_pred"])**2
```

In [17]:
```
SSD = sum(data["RSE"])
SSD
```

Out[17]:
```
2102.5305838896525
```

In [18]:
```
RSE = np.sqrt(SSD/(len(data)-2))
RSE
```

Out[18]:
```
3.258656369238098
```

In [19]:
```
sales_m = np.mean(data["Sales"])
```

In [20]:
```
sales_m
```

Out[20]:
```
14.022500000000003
```

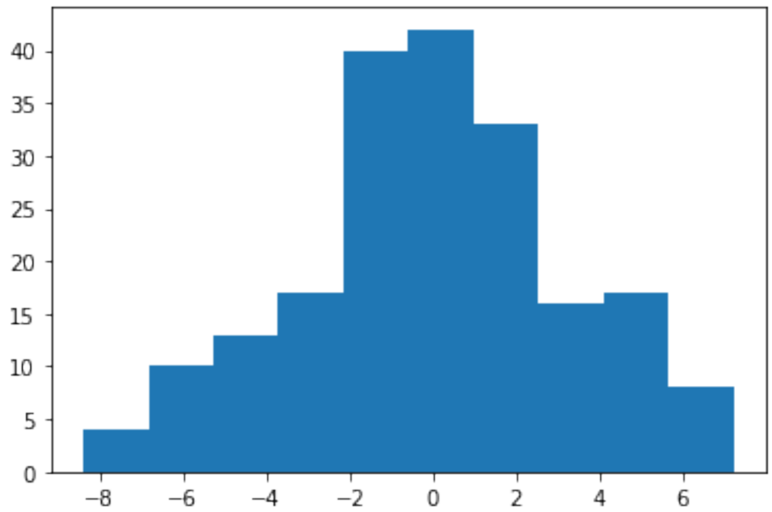
In [21]:
```
error = RSE/sales_m
```

In [22]:
```
error
```

Out[22]:
```
0.2323876890168014
```

In [23]:
```
plt.hist((data["Sales"]-data["sales_pred"]))
```

Out[23]:
```
(array([ 4., 10., 13., 17., 40., 42., 33., 16., 17.,  8.]),
 array([-8.3860819 , -6.82624404, -5.26640618, -3.70656832, -2.146730
46,
        -0.5868926 ,  0.97294526,  2.53278312,  4.09262098,  5.652458
84,
         7.2122967 ]),
 <BarContainer object of 10 artists>)
```



Regresión lineal múltiple en Python

El paquete statsmodel para regresión múltiple

- Sales ~TV
- Sales ~Newspaper
- Sales ~Radio
- Sales ~TV+Newspaper
- Sales ~TV+Radio
- Sales ~Newspaper+Radio
- Sales ~TV+Newspaper+Radio

In [24]:
```
#Añadir el Newspaper al modelo existente
lm2 = smf.ols(formula="Sales~TV+Newspaper", data = data).fit()
```

In [25]:
```
lm2.params
```

Out[25]:
```
Intercept    5.774948
TV           0.046901
Newspaper    0.044219
dtype: float64
```

In [26]:
```
lm2.pvalues
```

Out[26]:
```
Intercept    3.145860e-22
TV           5.507584e-44
Newspaper    2.217084e-05
dtype: float64
```

Sales = $5.774948+0.046901 TV + 0.044219$Newspaper

In [27]:
```
lm2.rsquared
```

Out[27]:
```
0.6458354938293273
```

In [28]:
```
lm2.rsquared_adj
```

Out[28]:
```
0.6422399150864777
```

In [29]:
```
sales_pred = lm2.predict(data[["TV", "Newspaper"]])
```

In [30]:
```
sales_pred
```

Out[30]:
```
0      19.626901
1       9.856348
2       9.646055
3      15.467318
4      16.837102
         ...
195     8.176802
196    10.551220
197    14.359467
198    22.003458
199    17.045429
Length: 200, dtype: float64
```

In [31]:
```
SSD = sum((data["Sales"]-sales_pred)**2)
```

In [32]:
```
SSD
```

Out[32]:
```
1918.5618118968273
```

In [33]:
```
RSE = np.sqrt(SSD/(len(data)-2-1))
```

In [34]:
```
RSE
```

Out[34]:
```
3.120719860252885
```

In [35]:
```
error = RSE / sales_m
```

In [36]:
```
error
```

Out[36]:
```
0.22255089037282116
```

In [37]:
```
lm2.summary()
```

Out[37]:

OLS Regression Results

Dep. Variable:	Sales	**R-squared:**	0.646
Model:	OLS	**Adj. R-squared:**	0.642
Method:	Least Squares	**F-statistic:**	179.6
Date:	Thu, 14 Jul 2022	**Prob (F-statistic):**	3.95e-45
Time:	01:15:26	**Log-Likelihood:**	-509.89
No. Observations:	200	**AIC:**	1026.
Df Residuals:	197	**BIC:**	1036.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	5.7749	0.525	10.993	0.000	4.739	6.811
TV	0.0469	0.003	18.173	0.000	0.042	0.052
Newspaper	0.0442	0.010	4.346	0.000	0.024	0.064

Omnibus:	0.658	**Durbin-Watson:**	1.969
Prob(Omnibus):	0.720	**Jarque-Bera (JB):**	0.415
Skew:	-0.093	**Prob(JB):**	0.813
Kurtosis:	3.122	**Cond. No.**	410.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [38]:
```
#Añadir la Radio al modelo existente
lm3 = smf.ols(formula="Sales~TV+Radio", data = data).fit()
```

In [39]:
```
lm3.summary()
```

Out[39]:

OLS Regression Results

Dep. Variable:	Sales	**R-squared:**	0.897
Model:	OLS	**Adj. R-squared:**	0.896
Method:	Least Squares	**F-statistic:**	859.6
Date:	Thu, 14 Jul 2022	**Prob (F-statistic):**	4.83e-98
Time:	01:15:26	**Log-Likelihood:**	-386.20
No. Observations:	200	**AIC:**	778.4
Df Residuals:	197	**BIC:**	788.3
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.9211	0.294	9.919	0.000	2.340	3.502
TV	0.0458	0.001	32.909	0.000	0.043	0.048
Radio	0.1880	0.008	23.382	0.000	0.172	0.204

Omnibus:	60.022	**Durbin-Watson:**	2.081
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	148.679
Skew:	-1.323	**Prob(JB):**	5.19e-33
Kurtosis:	6.292	**Cond. No.**	425.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [40]:
```
sales_pred = lm3.predict(data[["TV", "Radio"]])
SSD = sum((data["Sales"]-sales_pred)**2)
RSE = np.sqrt(SSD/(len(data)-2-1))
```

In [41]:
```
RSE
```

Out[41]: 1.681360912508001

In [42]:
```
RSE/sales_m
```

Out[42]: 0.11990450436855059

In [43]:
```
#Añadir la Radio al modelo existente
lm4 = smf.ols(formula="Sales~TV+Radio+Newspaper", data = data).fit()
```

In [44]:
```
lm4.summary()
```

Out[44]:

OLS Regression Results

Dep. Variable:	Sales	**R-squared:**	0.897
Model:	OLS	**Adj. R-squared:**	0.896
Method:	Least Squares	**F-statistic:**	570.3
Date:	Thu, 14 Jul 2022	**Prob (F-statistic):**	1.58e-96
Time:	01:15:26	**Log-Likelihood:**	-386.18
No. Observations:	200	**AIC:**	780.4
Df Residuals:	196	**BIC:**	793.6
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	2.9389	0.312	9.422	0.000	2.324	3.554
TV	0.0458	0.001	32.809	0.000	0.043	0.049
Radio	0.1885	0.009	21.893	0.000	0.172	0.206
Newspaper	-0.0010	0.006	-0.177	0.860	-0.013	0.011

Omnibus:	60.414	**Durbin-Watson:**	2.084
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	151.241
Skew:	-1.327	**Prob(JB):**	1.44e-33
Kurtosis:	6.332	**Cond. No.**	454.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [45]:
```
sales_pred = lm4.predict(data[["TV", "Radio","Newspaper"]])
SSD = sum((data["Sales"]-sales_pred)**2)
RSE = np.sqrt(SSD/(len(data)-3-1))
```

In [46]:
```
RSE
```

Out[46]: 1.6855103734147439

In [47]:
```
RSE/sales_m
```

Out[47]: 0.12020041885646236

Multicolinealidad

Factor Inflación de la Varianza

- VIF = 1 : Las variables no están correlacionadas
- VIF < 5 : Las variables tienen una correlación moderada y se pueden quedar en el modelo
- VIF >5 : Las variables están altamente correlacionadas y deben desaparecer del modelo.

In [48]:
```
# Newspaper ~ TV + Radio -> R^2 VIF = 1/(1-R^2)
lm_n = smf.ols(formula="Newspaper~TV+Radio", data = data).fit()
rsquared_n = lm_n.rsquared
VIF = 1/(1-rsquared_n)
VIF
```

Out[48]: 1.1451873787239286

In [49]:
```
# TV ~ Newspaper + Radio -> R^2 VIF = 1/(1-R^2)
lm_tv = smf.ols(formula="TV~Newspaper+Radio", data=data).fit()
rsquared_tv = lm_tv.rsquared
VIF = 1/(1-rsquared_tv)
VIF
```

Out[49]: 1.00461078493965

In [50]:
```
# Radio ~ TV + Newspaper -> R^2 VIF = 1/(1-R^2)
lm_r = smf.ols(formula="Radio~Newspaper+TV", data=data).fit()
rsquared_r = lm_r.rsquared
VIF = 1/(1-rsquared_r)
VIF
```

Out[50]: 1.1449519171055351

In [51]:
```
lm3.summary()
```

Out[51]:

OLS Regression Results

Dep. Variable:	Sales	**R-squared:**	0.897
Model:	OLS	**Adj. R-squared:**	0.896
Method:	Least Squares	**F-statistic:**	859.6
Date:	Thu, 14 Jul 2022	**Prob (F-statistic):**	4.83e-98
Time:	01:15:26	**Log-Likelihood:**	-386.20
No. Observations:	200	**AIC:**	778.4
Df Residuals:	197	**BIC:**	788.3
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	2.9211	0.294	9.919	0.000	2.340	3.502
TV	0.0458	0.001	32.909	0.000	0.043	0.048
Radio	0.1880	0.008	23.382	0.000	0.172	0.204

Omnibus:	60.022	**Durbin-Watson:**	2.081
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	148.679
Skew:	-1.323	**Prob(JB):**	5.19e-33
Kurtosis:	6.292	**Cond. No.**	425.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.