```
In [1]: import warnings
        warnings.filterwarnings('ignore')
In [2]: # Import the numpy and pandas package
        import numpy as np
        import pandas as pd
        # Data Visualisation
        import matplotlib.pyplot as plt
        import seaborn as sns
In [3]: | advertising = pd.DataFrame(pd.read_csv("advertising.csv"))
        advertising.head()
Out[3]:
              TV Radio Newspaper Sales
         0 230.1
                   37.8
                             69.2
                                   22.1
            44.5
                   39.3
                             45.1
                                   10.4
            17.2
                 45.9
                             69.3
                                   12.0
         2
         3 151.5
                 41.3
                             58.5
                                   16.5
         4 180.8
                  10.8
                             58.4
                                   17.9
In [4]: # 1. Missing value Treatment
        advertising.isnull().sum()
Out[4]: TV
                      0
        Radio
                      0
        Newspaper
                      0
        Sales
        dtype: int64
In [5]: # 2. Multicollineartity
        from statsmodels.stats.outliers_influence import variance_inflation_factor
        x = advertising[['TV', 'Radio', 'Newspaper']]
        vif_data= pd.DataFrame()
        vif_data["features"] = x.columns
        vif_data["VIF"] = [variance_inflation_factor(x.values,i)
                           for i in range(len(x.columns))]
        print(vif_data)
             features
                            VIF
        0
                   TV 2.486772
        1
                Radio 3.285462
```

2 Newspaper 3.055245

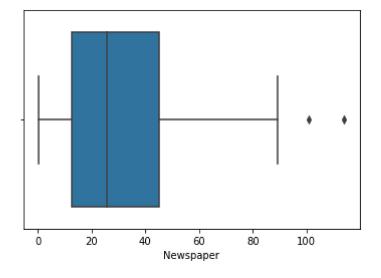
```
In [6]: # 3. Univaraite analysis
advertising.describe()
```

Out[6]:

	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000
mean	147.042500	23.264000	30.554000	15.130500
std	85.854236	14.846809	21.778621	5.283892
min	0.700000	0.000000	0.300000	1.600000
25%	74.375000	9.975000	12.750000	11.000000
50%	149.750000	22.900000	25.750000	16.000000
75%	218.825000	36.525000	45.100000	19.050000
max	296.400000	49.600000	114.000000	27.000000

In [7]: sns.boxplot(advertising['Newspaper'])

Out[7]: <AxesSubplot:xlabel='Newspaper'>



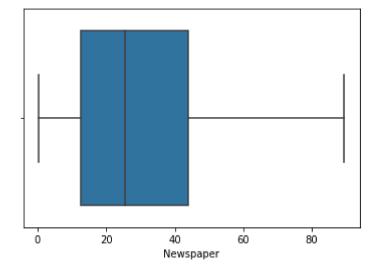
In [8]: advertising = advertising[advertising['Newspaper'] < 100]</pre>

In [9]: advertising.shape

Out[9]: (198, 4)

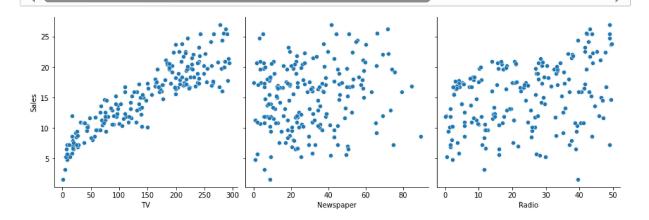
```
In [10]: sns.boxplot(advertising['Newspaper'])
```

Out[10]: <AxesSubplot:xlabel='Newspaper'>



```
In [ ]:
```

In [11]: # 4.Bivariate analysis
sns.pairplot(advertising, x_vars=['TV', 'Newspaper', 'Radio'], y_vars='Sales',
plt.show()



In [12]: advertising.head()

Out[12]:

	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	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

```
In [13]: # 5. Creating two data
         X = advertising[['TV', 'Radio']]
         y = advertising['Sales']
In [14]: X.head()
Out[14]:
              TV Radio
          0 230.1
                   37.8
             44.5
                   39.3
            17.2
                  45.9
          3 151.5
                  41.3
          4 180.8
                  10.8
In [15]: y.head()
Out[15]: 0
              22.1
              10.4
         1
              12.0
         2
              16.5
         3
              17.9
         4
         Name: Sales, dtype: float64
In [16]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.75, test
In [17]: X_train.shape
Out[17]: (148, 2)
In [18]: X_test.shape
Out[18]: (50, 2)
In [19]: # 6 .model building
         # model 1 . Sales = F(TV)
         \# model 2 . sales = F(TV + Radio)
         # model 3. sales = F(TV+Radio+newspaper)
```

In [20]: # model_1
X_train

Out[20]:

	TV	Radio
187	191.1	28.7
43	206.9	8.4
68	237.4	27.5
195	38.2	3.7
76	27.5	1.6
•••		
108	13.1	0.4
14	204.1	32.9
93	250.9	36.5
181	218.5	5.4
104	238.2	34.3

148 rows × 2 columns

```
In [21]: import statsmodels.api as sm

# Add a constant to get an intercept
X_train_sm = sm.add_constant(X_train)

# Fit the resgression line using 'OLS'
lr = sm.OLS(y_train, X_train_sm).fit()

X_train_sm
```

Out[21]:

	const	TV	Radio
187	1.0	191.1	28.7
43	1.0	206.9	8.4
68	1.0	237.4	27.5
195	1.0	38.2	3.7
76	1.0	27.5	1.6
108	1.0	13.1	0.4
14	1.0	204.1	32.9
93	1.0	250.9	36.5
181	1.0	218.5	5.4
104	1.0	238.2	34.3

148 rows × 3 columns

	OLS Regression Results								
=========	======	====	======	====	=====	=========		=======	
= Dep. Variable:			Sa	les	R-sq	uared:		0.89	
2 Model:				OLS	Adj.	R-squared:		0.89	
0						•			
Method: 9		L	east Squa	res	F-st	atistic:		596.	
Date:		Tue.	11 Jul 2	023	Prob	(F-statistic):	•	1.03e-7	
0			11 341 1	0_3		(. 500015010)	•	1,036	
Time:			22:52	: 09	l og-	Likelihood:		-288.4	
5			22.32		208	LIKEIIIIOU.		2001.	
No. Observatio	ns:			148	AIC:			582.	
9 Df Residuals:				145	BIC:			591.	
9									
Df Model:				2					
Covariance Typ	e:		nonrob	ust					
==========		====		====		=========		=======	
=									
	coef	= ;	std err		t	P> t	[0.025	0.97	
5]						• •			
-									
const	4.8447	7	0.349	13	3.870	0.000	4.154	5.53	
5									
TV	0.0532	2	0.002	32	2.010	0.000	0.050	0.05	
6									
Radio	0.1057	7	0.009	11	L.220	0.000	0.087	0.12	
4									
		====	======	====				=======	
=									
Omnibus:			17.	987	Durb	in-Watson:		2.25	
7									
Prob(Omnibus): 1			0.	000	Jarq	ue-Bera (JB):		32.46	
Skew:			-0.	569	Prob	(ЈВ):		8.94e-0	
8									
Kurtosis:			4.	992	Cond	. No.		43	
2.									
==========		====	======	=====				======	
=									

	OLS Regression Results								
=========	=======	:=======	===:	====	=========	======	=======		
= Dep. Variable:		Sales	S	R-sq	uared:		0.89		
2 Model:		OLS	S	Adj.	R-squared:		0.89		
0 Method:		Least Squares	S	F-st	atistic:		596.		
9									
Date:	Τι	ie, 11 Jul 2023	3	Prob	(F-statistic):		1.03e-7		
0 Time:		22:52:09	9	Log-	Likelihood:		-288.4		
5 No. Observatio	ns:	148	8	AIC:			582.		
9									
<pre>Df Residuals: 9</pre>		145	5	BIC:			591.		
Df Model:		2	2						
Covariance Typ	e:	nonrobust	t						
=========	=======	-=======	===:		=========	======	=======		
=	_				- 1.1	F0 00-			
5]	coef	std err		t	P> t	[0.025	0.97		
- const	4.8447	0.349	13	.870	0.000	4.154	5.53		
5 TV	0.0532	0.002	32	.010	0.000	0.050	0.05		
6									
Radio 4	0.1057	0.009	11	.220	0.000	0.087	0.12		
=========	======	.=======	:	====	=========		=======		
=									
Omnibus:		17.987	7	Durb	in-Watson:		2.25		
7 Prob(Omnibus):		0.000	9	Jara	ue-Bera (JB):		32.46		
1									
Skew: 8		-0.569	9	Prob	(JB):		8.94e-0		
Kurtosis:		4.992	2	Cond	. No.		43		
2.									
========	======	==========	===:		=========	======	=======		
=									

OLS Regression Results									
=======================================	=								
Dep. Variable:			9	Sales	5	R-sq	uared:		0.89
2 Model:				OLS		۸di	R-squared:		0.89
0				OLS	,	Auj.	K-3quai eu.		0.09
Method: 9		L	east Sqı	uares	5	F-st	atistic:		596.
Date: 0		Tue,	11 Jul	2023	3	Prob	(F-statistic):		1.03e-7
Time:			22:5	52:09)	Log-	Likelihood:		-288.4
5						6			
No. Observatio	ns:			148	3	AIC:			582.
9 Df Residuals:				145		BIC:			591.
9				143	,	DIC.			291.
Df Model:				2					
Covariance Typ			nonro						
=======================================	======	====	======	=====	===	=====	=========	=======	=======
	coet	= ;	std err			t	P> t	[0.025	0.97
5]									
_									
const	4.8447	7	0.349		13	.870	0.000	4.154	5.53
5									
	0.0532	2	0.002		32	.010	0.000	0.050	0.05
6 Radio	0.1057	7	0.009		11	.220	0.000	0.087	0.12
4	0.103		0.005				0.000	0.007	3.12
=========	======	====	======		===	=====	=========	=======	:======
= Omnibus:			1-	7.987	,	Dunh	in-Watson:		2.25
7			1.	/ .90/	,	Durb.	In-watson:		2.25
Prob(Omnibus):			(a.000)	Jarq	ue-Bera (JB):		32.46
1									
Skew: 8			-(ð.569)	Prob	(JB):		8.94e-0
8 Kurtosis:			2	1.992)	Cond	. No.		43
2.					-	55110			,,,
=========	======	====	======	====	===	=====	========	=======	
=									

OLS Regression Results									
=======================================	======	=====	======	=====	===:	=====	========	=======	=======
Dep. Variable: 2				Sales	5	R-sq	uared:		0.89
Model: 0				OLS	5	Adj.	R-squared:		0.89
Method: 9		Le	east Squ	uares	5	F-st	atistic:		596.
Date:		Tue,	11 Jul	2023	3	Prob	(F-statistic)	•	1.03e-7
0 Time:			22:	52:09)	Log-	Likelihood:		-288.4
5 No. Observation	ns:			148	3	AIC:			582.
9 Df Residuals:				145	5	BIC:			591.
9 Df Model:				2	2				
Covariance Type			nonro						
=======================================				=====	===:	====:			
5]	coef	= :	std err			t	P> t	[0.025	0.97
	4.8447	7	0.349		13	.870	0.000	4.154	5.53
	0.0532	2	0.002		32	.010	0.000	0.050	0.05
	0.1057	7	0.009		11	.220	0.000	0.087	0.12
4	======		======	====		=====		=======	=======
=									
Omnibus: 7			17	7.987	7	Durb:	in-Watson:		2.25
Prob(Omnibus): 1			(ə.00e)	Jarqı	ue-Bera (JB):		32.46
Skew: 8			- (ð.569)	Prob	(JB):		8.94e-0
Kurtosis: 2.			4	4.992	2	Cond	. No.		43
=======================================	=====	====	======	====	==:	====:	=========	=======	=======

```
In [27]: y_test
Out[27]: 66
                 11.0
         116
                 12.2
                 24.4
         17
         143
                10.4
         158
                 7.3
         128
                 24.7
         142
                 20.1
         31
                11.9
         19
                14.6
         169
                20.0
         160
                16.4
         15
                 22.4
         56
                 5.5
         57
                13.2
         117
                9.4
         46
                10.6
         126
                 6.6
         9
                15.6
         149
                10.1
         86
                12.0
         115
                12.6
         163
                18.0
                24.2
         61
         189
                 6.7
         74
                17.0
         69
                 22.3
         121
                 7.0
         165
                16.9
         119
                 6.6
         177
                16.7
         70
                18.3
         77
                14.2
         127
                11.9
         83
                13.6
         25
                17.0
         193
                 19.6
         97
                20.5
         173
                16.7
         20
                18.0
         152
                16.6
         139
                 20.7
         5
                 7.2
         67
                13.4
         140
                10.9
         106
                 7.2
         172
                 7.6
         99
                17.2
                 20.9
         113
         12
                 9.2
         36
                 25.4
         Name: Sales, dtype: float64
```

In [28]: X_test_sm = sm.add_constant(X_test)
X_test_sm

Out[28]:

	const	TV	Radio
66	1.0	31.5	24.6
116	1.0	139.2	14.3
17	1.0	281.4	39.6
143	1.0	104.6	5.7
158	1.0	11.7	36.9
128	1.0	220.3	49.0
142	1.0	220.5	33.2
31	1.0	112.9	17.4
19	1.0	147.3	23.9
169	1.0	284.3	10.6
160	1.0	172.5	18.1
15	1.0	195.4	47.7
56	1.0	7.3	28.1
57	1.0	136.2	19.2
117	1.0	76.4	8.0
46	1.0	89.7	9.9
126	1.0	7.8	38.9
9	1.0	199.8	2.6
149	1.0	44.7	25.8
86	1.0	76.3	27.5
115	1.0	75.1	35.0
163	1.0	163.5	36.8
61	1.0	261.3	42.7
189	1.0	18.7	12.1
74	1.0	213.4	24.6
69	1.0	216.8	43.9
121	1.0	18.8	21.7
165	1.0	234.5	3.4
119	1.0	19.4	16.0
177	1.0	170.2	7.8
70	1.0	199.1	30.6
77	1.0	120.5	28.5
127	1.0	80.2	0.0
83	1.0	68.4	44.5
25	1.0	262.9	3.5

	const	TV	Radio
193	1.0	166.8	42.0
97	1.0	184.9	21.0
173	1.0	168.4	7.1
20	1.0	218.4	27.7
152	1.0	197.6	23.3
139	1.0	184.9	43.9
5	1.0	8.7	48.9
67	1.0	139.3	14.5
140	1.0	73.4	17.0
106	1.0	25.0	11.0
172	1.0	19.6	20.1
99	1.0	135.2	41.7
113	1.0	209.6	20.6
12	1.0	23.8	35.1
36	1.0	266.9	43.8

```
In [29]: y_pred = lr.predict(X_test_sm)
```

```
In [30]: y_pred
Out[30]: 66
                  9.119808
          116
                 13.755840
                 23.989330
          17
          143
                 11.007458
          158
                  9.367648
          128
                 21.735233
          142
                 20.075518
          31
                 12.685557
          19
                 15.201300
          169
                 21.077656
          160
                 15.927673
          15
                 20.274209
          56
                  8.203440
          57
                 14.114391
          117
                  8.990432
          46
                 10.659446
          126
                  9.371775
          9
                 15.740207
          149
                  9.948333
          86
                 11.807791
                 12.536890
          115
          163
                 17.426196
          61
                 23.248616
          189
                  7.117931
          74
                 18.788932
          69
                 21.010024
          121
                  8.138141
          165
                 17.669304
          119
                  7.567441
          177
                 14.716516
          70
                 18.663106
          77
                 14.263016
          127
                  9.107851
          83
                 13.185066
          25
                 19.189514
          193
                 18.151346
          97
                 16.893393
          173
                 14.546832
          20
                 19.382440
          152
                 17.811629
          139
                 19.314339
          5
                 10.476797
          67
                 13.782300
          140
                 10.543598
          106
                  7.336526
          172
                  8.011517
          99
                 16.439893
          113
                 18.164066
          12
                  9.820546
```

36

23.662581

dtype: float64

```
In [31]: from sklearn.metrics import r2_score
    r_squared = r2_score(y_test, y_pred)
    r_squared

Out[31]: 0.9233579214065604

In [32]: 0.89
    0.84
    0.85
```

Out[32]: 0.85

	OLS Regression Results								
_	======	=========	===:	====	=========	======	=======		
= Dep. Variable:		Sales	s	R-sq	uared:		0.89		
2 Model:		OL:	S	Adj.	R-squared:		0.89		
0 Method:		Loost Cauphon	_	Г с+	atistic:		596.		
9		Least Squares	5	r=st	atistit.		590.		
Date:	Tu	ue , 11 Jul 2023	3	Prob	(F-statistic):		1.03e-7		
0 T:		22.52.1/	_	1			200 4		
Time: 5		22:52:10	0	Log-	Likelihood:		-288.4		
No. Observatio	ns:	148	8	AIC:			582.		
Df Residuals:		145	5	BIC:			591.		
9			_						
Df Model:	•		2						
Covariance Typ	e: 	nonrobus† 	L :						
=									
	coef	std err		t	P> t	[0.025	0.97		
5]									
-									
const 5	4.8447	0.349	13	.870	0.000	4.154	5.53		
TV	0.0532	0.002	32	.010	0.000	0.050	0.05		
6	0 1057	0.000	4.4	220	0.000	0 007	0.42		
Radio 4	0.1057	0.009	11	.220	0.000	0.087	0.12		
========	======		===:		=========				
=									
Omnibus:		17.987	7	Durb	in-Watson:		2.25		
7 Prob(Omnibus):		0.000	а	lard	ue-Bera (JB):		32.46		
1		0.00	•	Jul 4	uc bei u (35).		32.10		
Skew:		-0.569	9	Prob	(JB):		8.94e-0		
8 Kurtosis:		4.992	2	Cond	. No.		43		
2.		4.992	_	CONG	• INO •		43		
	======	-========			=========		=======		
=									

OLS Regression Results								
=========	======	:====:	=======		=====	=========	=======	=======
= Dep. Variable:			Sa]	les	R-sq	uared:		0.89
2 Model:			(OLS	Adj.	R-squared:		0.89
0								
Method: 9		Le	east Squar	res	F-st	atistic:		596.
Date:		Tue.	11 Jul 20	923	Prob	(F-statistic):		1.03e-7
0		,				(. 500.015010)		_,,
Time:			22:52:	:10	Log-	Likelihood:		-288.4
5				v	8			
No. Observatio	ns:		1	148	AIC:			582.
9 Df Residuals:			1	145	BIC:			591.
9								
Df Model:				2				
Covariance Typ	e:		nonrobu	ust				
=========	======	====	=======		=====	=========	=======	=======
=								
	coet	:	std err		t	P> t	[0.025	0.97
5]								
const	4.8447	,	0.349	13	.870	0.000	4.154	5.53
5								
TV	0.0532	<u> </u>	0.002	32	.010	0.000	0.050	0.05
6								
Radio	0.1057	'	0.009	11	.220	0.000	0.087	0.12
4								
=========	======	:====:	=======	====	=====	=========	=======	=======
=								
Omnibus:			17.9	987	Durb	in-Watson:		2.25
7								
<pre>Prob(Omnibus): 1</pre>			0.6	900	Jarq	ue-Bera (JB):		32.46
Skew:			-0.5	569	Prob	(JB):		8.94e-0
8			4 6	202	C e1	l No		40
Kurtosis:			4.9	992	cona	. No.		43
2.								
	======	:====:	=======	====	=====	=========	=======	:======
=								

```
In [35]: # Add a constant to X_test
         X_test_sm = sm.add_constant(X_test)
         # Predict the y values corresponding to X_test_sm
         y_pred = lr.predict(X_test_sm)
         from sklearn.metrics import r2_score
         r_squared = r2_score(y_test, y_pred)
         r_squared
Out[35]: 0.9233579214065604
In [36]: X_test.head()
Out[36]:
                 TV Radio
           66
               31.5
                      24.6
          116 139.2
                      14.3
           17 281.4
                      39.6
          143 104.6
                      5.7
          158
               11.7
                      36.9
In [37]: y_test.head()
Out[37]: 66
                 11.0
         116
                 12.2
         17
                 24.4
         143
                 10.4
         158
                 7.3
         Name: Sales, dtype: float64
In [38]: |X_test_sm = sm.add_constant(X_test)
         # Predict the y values corresponding to X_test_sm
         y_pred = lr.predict(X_test_sm)
         y_pred.head()
Out[38]: 66
                 9.119808
         116
                 13.755840
                 23.989330
         17
         143
                 11.007458
         158
                 9.367648
         dtype: float64
In [39]: | from sklearn.metrics import r2_score
         r_squared = r2_score(y_test, y_pred)
         r_squared
Out[39]: 0.9233579214065604
```

Thankyou