

```
#Amit Sharad pawar
#PRN:25070126501
#Multiple Linear Regression
#Asvertising dataset use
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
import seaborn as sns
```

```
df=pd.read_csv('/content/Assignment 2 Advertising (1).csv')
```

```
df.head(20)
```


|    | R&D Spend | Administration | Marketing Spend | State      | Profit    |
|----|-----------|----------------|-----------------|------------|-----------|
| 0  | 165349.20 | 136897.80      | 471784.10       | New York   | 192261.83 |
| 1  | 162597.70 | 151377.59      | 443898.53       | California | 191792.06 |
| 2  | 153441.51 | 101145.55      | 407934.54       | Florida    | 191050.39 |
| 3  | 144372.41 | 118671.85      | 383199.62       | New York   | 182901.99 |
| 4  | 142107.34 | 91391.77       | 366168.42       | Florida    | 166187.94 |
| 5  | 131876.90 | 99814.71       | 362861.36       | New York   | 156991.12 |
| 6  | 134615.46 | 147198.87      | 127716.82       | California | 156122.51 |
| 7  | 130298.13 | 145530.06      | 323876.68       | Florida    | 155752.60 |
| 8  | 120542.52 | 148718.95      | 311613.29       | New York   | 152211.77 |
| 9  | 123334.88 | 108679.17      | 304981.62       | California | 149759.96 |
| 10 | 101913.08 | 110594.11      | 229160.95       | Florida    | 146121.95 |
| 11 | 100671.96 | 91790.61       | 249744.55       | California | 144259.40 |
| 12 | 93863.75  | 127320.38      | 249839.44       | Florida    | 141585.52 |
| 13 | 91992.39  | 135495.07      | 252664.93       | California | 134307.35 |
| 14 | 119943.24 | 156547.42      | 256512.92       | Florida    | 132602.65 |
| 15 | 114523.61 | 122616.84      | 261776.23       | New York   | 129917.04 |
| 16 | 78013.11  | 121597.55      | 264346.06       | California | 126992.93 |
| 17 | 94657.16  | 145077.58      | 282574.31       | New York   | 125370.37 |
| 18 | 91749.16  | 114175.79      | 294919.57       | Florida    | 124266.90 |
| 19 | 86419.70  | 153514.11      | 0.00            | New York   | 122776.86 |

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Unnamed: 0   200 non-null    int64
1   TV           200 non-null    float64
2   Radio        200 non-null    float64
3   Newspaper    200 non-null    float64
4   Sales        200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

```
df=df.iloc[:,1:]
```

```
df.head()
```

|   | 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  |   |

Next steps:

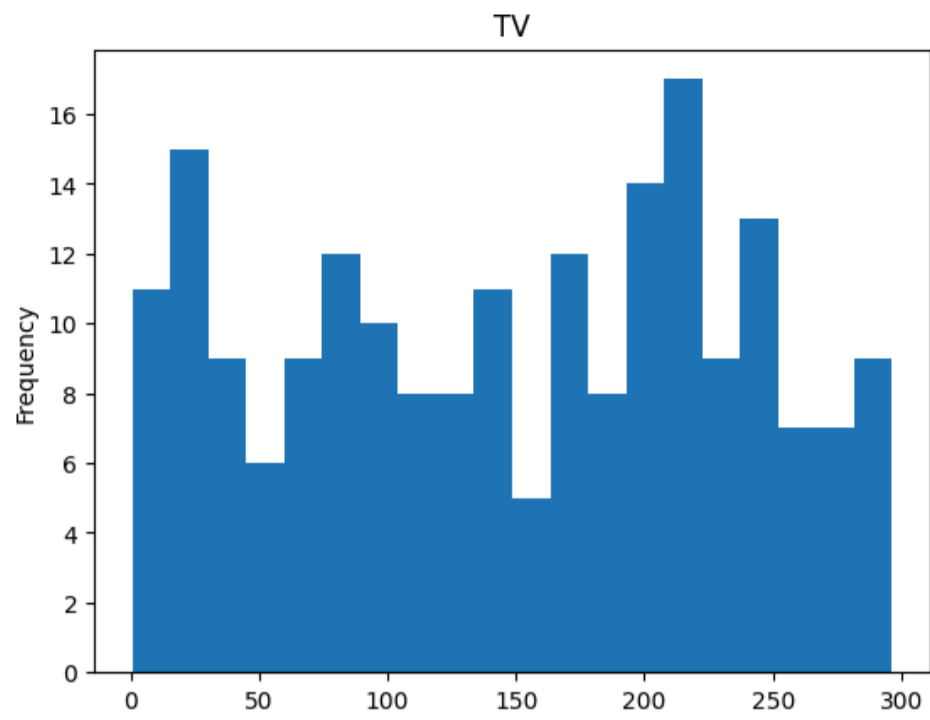
[Generate code with df](#)

[New interactive sheet](#)

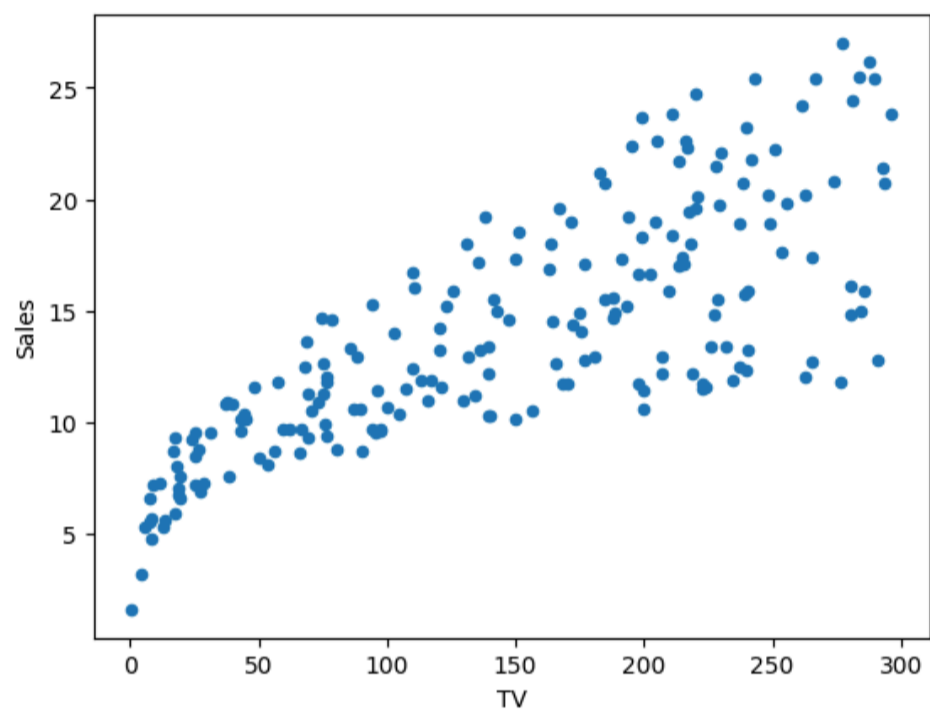
```
df.shape
```

```
(200, 4)
```

```
df['TV'].plot(kind='hist',bins=20,title='TV')
plt.show()
```

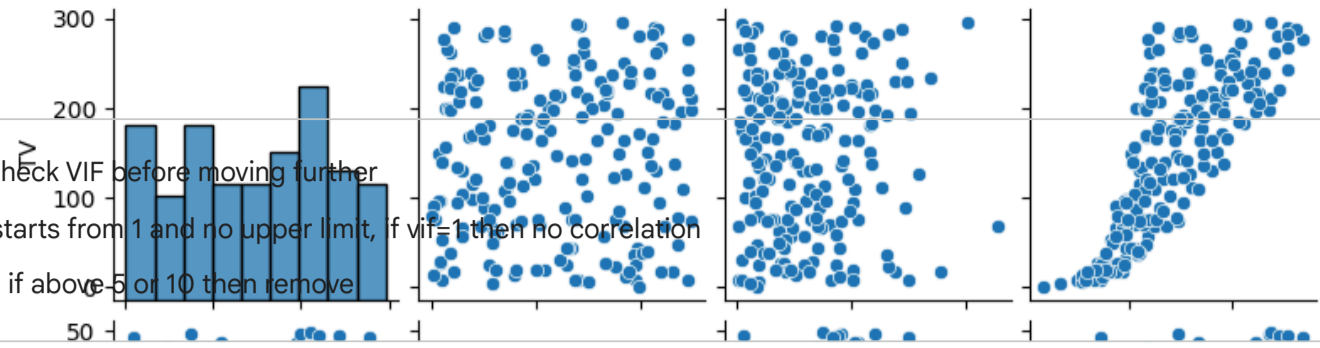


```
df.plot(kind='scatter',x='TV',y='Sales',alpha=1)
plt.show()
```



```
sns.pairplot(data=df,height=2)
```

<seaborn.axisgrid.PairGrid at 0x7b1d008f0ad0>



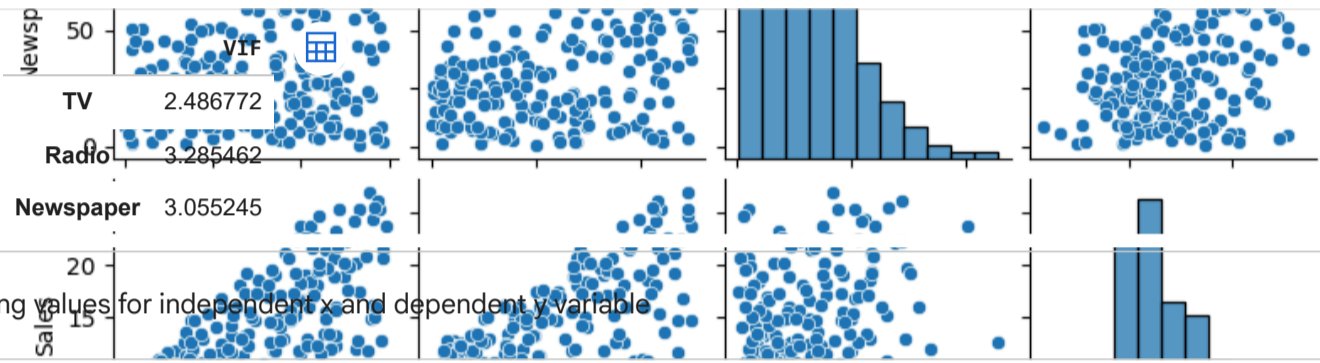
We check VIF before moving further  
VIF starts from 1 and no upper limit, if vif=1 then no correlation

- if above 5 or 10 then remove

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
def calc_vif(X):  
    vif=pd.DataFrame()  
    vif["VIF"] = [variance_inflation_factor(X.values,i)  
                  for i in range(X.shape[1])]  
    vif.index=X.columns # directly assign column names  
    return vif
```

```
X=df.iloc[:, :-1]  
calc_vif(X)
```



Setting values for independent x and dependent y variable

```
x = df[['TV', 'Radio', 'Newspaper']]  
y = df['Sales']
```

```
x.head()
```

|   | TV    | Radio | Newspaper |
|---|-------|-------|-----------|
| 0 | 230.1 | 37.8  | 69.2      |
| 1 | 44.5  | 39.3  | 45.1      |
| 2 | 17.2  | 45.9  | 69.3      |
| 3 | 151.5 | 41.3  | 58.5      |
| 4 | 180.8 | 10.8  | 58.4      |

Next steps: [Generate code with x](#) [New interactive sheet](#)

```
y.head()
```

|   | Sales |
|---|-------|
| 0 | 22.1  |
| 1 | 10.4  |
| 2 | 9.3   |
| 3 | 18.5  |
| 4 | 12.9  |

**dtype:** float64

```
from sklearn.model_selection import train_test_split  
x_train,xtest,ytrain,ytest = train_test_split(x,y,test_size = 0.20,random_state = 42)
```

## Implement Linear Model

```
from sklearn.linear_model import LinearRegression  
mlr = LinearRegression()  
mlr.fit(x_train,ytrain)
```

LinearRegression

LinearRegression()

```
print("Intercept: ",mlr.intercept_)  
print("Coefficients: ",mlr.coef_)  
list(zip(x,mlr.coef_))
```

Intercept: 2.979067338122629  
Coefficients: [0.04472952 0.18919505 0.00276111]

```
[('TV', np.float64(0.044729517468716326)),
 ('Radio', np.float64(0.18919505423437652)),
 ('Newspaper', np.float64(0.0027611143413671935))]
```


Prediction on test set

```
#prediction of test set
y_pred_mlr=mlr.predict(xtest)
print("Prediction for test set: {}".format(y_pred_mlr))
```

```
Prediction for test set: [16.4080242  20.88988209 21.55384318 10.60850256 22.11237326 13.10559172
 21.05719192  7.46101034 13.60634581 15.15506967  9.04831992  6.65328312
 14.34554487  8.90349333  9.68959028 12.16494386  8.73628397 16.26507258
 10.27759582 18.83109103 19.56036653 13.25103464 12.33620695 21.30695132
  7.82740305  5.80957448 20.75753231 11.98138077  9.18349576  8.5066991
 12.46646769 10.00337695 21.3876709  12.24966368 18.26661538 20.13766267
 14.05514005 20.85411186 11.0174441  4.56899622]
```

Combine Actual Values and Predicted value

```
mlr_diff=pd.DataFrame({'Actual Value': ytest, 'Predicted Value': y_pred_mlr})
mlr_diff.head()
```

|     | Actual Value | Predicted Value |  |
|-----|--------------|-----------------|---|
| 95  | 16.9         | 16.408024       |   |
| 15  | 22.4         | 20.889882       |   |
| 30  | 21.4         | 21.553843       |   |
| 158 | 7.3          | 10.608503       |   |
| 128 | 24.7         | 22.112373       |   |

Next steps: [Generate code with mlr\\_diff](#) [New interactive sheet](#)

Evaluating the Model

```
import numpy as np
from sklearn import metrics
meanAbsErr = metrics.mean_absolute_error(ytest,y_pred_mlr)
meanSqErr = metrics.mean_squared_error(ytest,y_pred_mlr)
rootMeanSqErr = np.sqrt(metrics.mean_squared_error(ytest,y_pred_mlr))
print("R square: {:.2f}".format(mlr.score(x,y)*100))
print("Mean Absolute Error: ",meanAbsErr)
print("Mean Square Error: ",meanSqErr)
print("Root Mean Square Error: ",rootMeanSqErr)
```

```
R square: 89.67
Mean Absolute Error:  1.4607567168117603
Mean Square Error:   3.1740973539761033
Root Mean Square Error:  1.78159966153345
```

```
fig, axes=plt.subplots(1,3,figsize=(15,5))

def plot_scatter_with_regression(ax,x_data,y_data,x_label,title):
    ax.scatter(x_data,y_data,alpha=1)

    #calculating reg line
    slope, intercept = np.polyfit(x_data,y_data,1)
    regression_line = slope*x_data + intercept

    ax.plot(x_data,regression_line,color='green',label='Regression Line')
    ax.set_title(title)
    ax.set_xlabel(x_label)
    ax.set_ylabel('Sales')
    ax.legend()

#Scatter plot for tv vs sales
plot_scatter_with_regression(axes[0],df['TV'],df['Sales'],'TV ADVERTISING','TV vs Sales')
# Scatter plot for Radio vs sales
plot_scatter_with_regression(axes[1],df['Radio'],df['Sales'],'Radio Advertising','Radio vs Sales')
# Scatter plot for Newspaper vs sales
plot_scatter_with_regression(axes[2],df['Newspaper'],df['Sales'],'Newspaper Advertising','Newspaper vs Sales')
plt.tight_layout()
plt.show()
```

