S. B. JAIN INSTITUTE OF TECHNOLOGY,
MANAGEMENT & RESEARCH, NAGPUR

Practical No. 1(a)

Aim: Implement the concept of Linear Regression Models in Machine Learning with Kaggle dataset.

Name of Student : Shrutika Pradeep Bagdi

Roll No. : CS22130

Semester/Year : 6th / 3rd

Academic Session : 2024-2025

Date of Performance :

Date of Submission :

AIM: Implement the concept of Linear Regression Models in Machine Learning.

OBJECTIVE/EXPECTED LEARNING OUTCOME:

The objectives and expected learning outcome of this practical are:

- Develop a deeper understanding of the linear regression model and its limitations;
- Know how to diagnose and apply corrections to some problems with the generalized linear model found in real data; discussed;
- Use and understand generalizations of the linear model to binary and count data;
- Develop a greater familiarity with a range of techniques and methods through a diverse set of theoretical and applied readings;
- Know where to go to learn more about the techniques in this class and those called for that were not covered in this class.

THEORY:

Linear Regression:

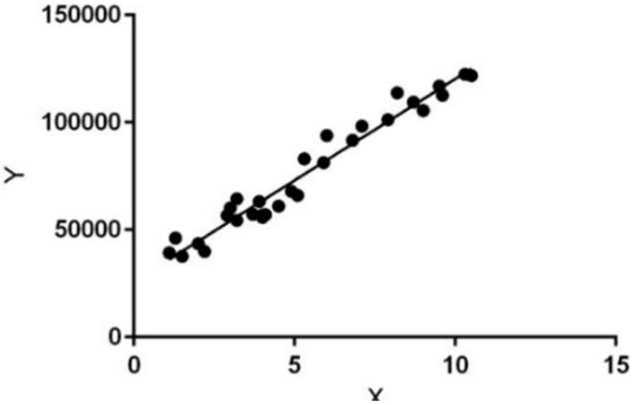
It is the basic and commonly used type for predictive analysis. It is a statistical approach to modeling the relationship between a dependent variable and a given set of independent variables. These are of two types:

1) Simple linear Regression

2) Multiple Linear Regressions

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used. There are many names for a regression's dependent variable. It may be called an outcome variable, criterion variable, endogenous variable, or regressand. The independent variables can be called exogenous variables, predictor variables, or regressors.

Linear regression is used in many different fields, including finance, economics, and psychology, to understand and predict the behavior of a particular variable. For example, in finance, linear regression might be used to understand the relationship between a company's stock price and its earnings, or to predict the future value of a currency based on its past performance.

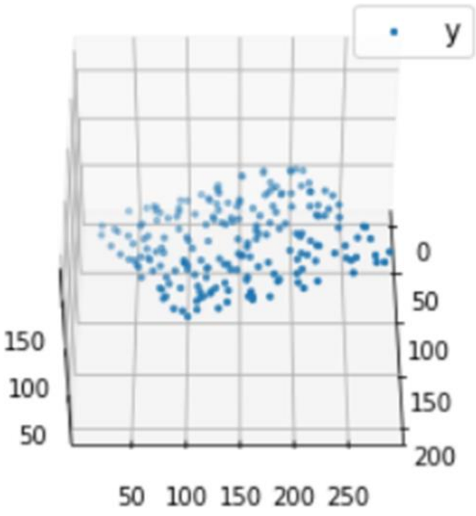


Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x)). Hence, the name is Linear Regression. In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Sample Code(For reference only):

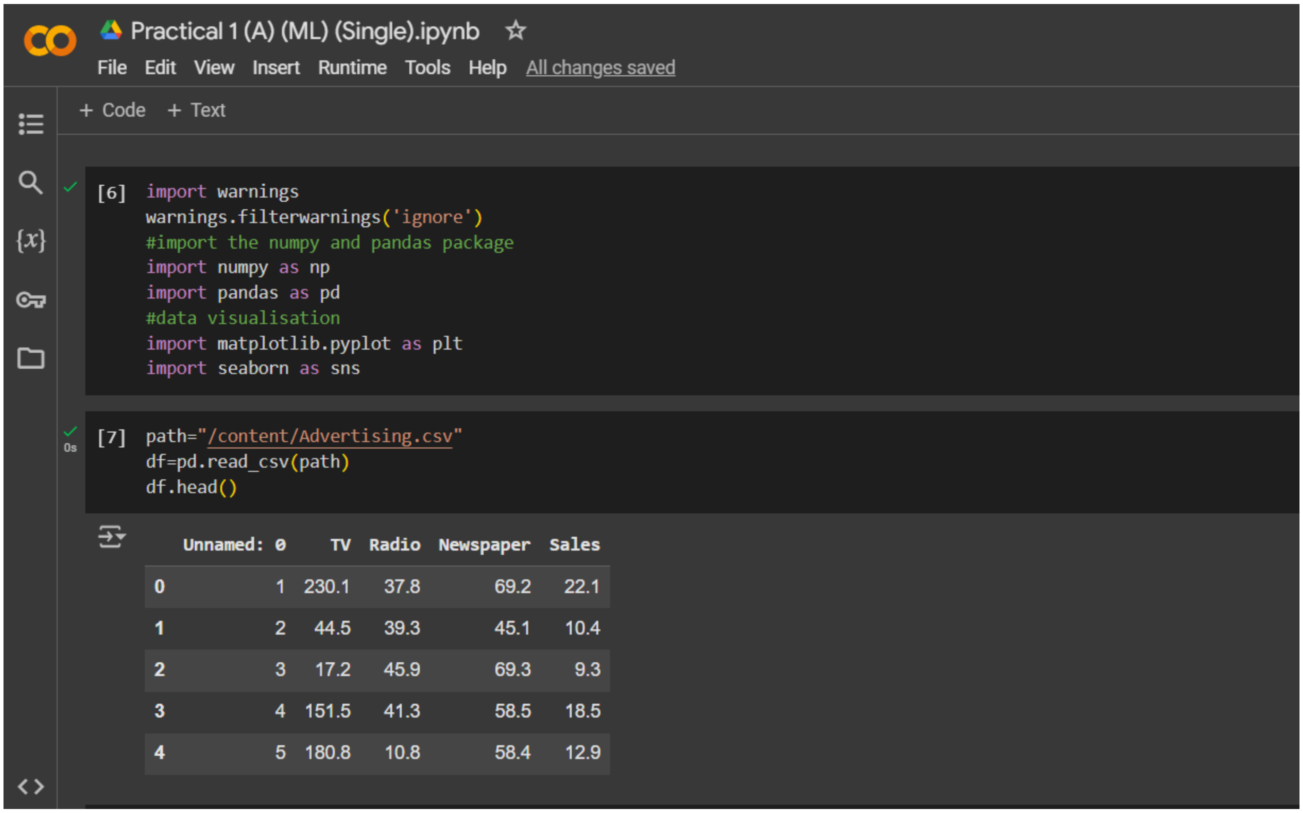
```
import numpy as np
import matplotlib as mpl
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt

def generate_dataset(n):
    x = []
    y = []
    random_x1 = np.random.rand()
    random_x2 = np.random.rand()
    for i in range(n):
        x1 = i
        x2 = i/2 + np.random.rand()*n
        x.append([1, x1, x2])
        y.append(random_x1 * x1 + random_x2 * x2 + 1)
    return np.array(x), np.array(y)

x, y = generate_dataset(200)

mpl.rcParams['legend.fontsize'] = 12

fig = plt.figure()
ax = fig.add_subplot(projection ='3d')

ax.scatter(x[:, 1], x[:, 2], y, label ='y', s = 5)
ax.legend()
ax.view_init(45, 0)

plt.show()
```

OUTPUT (SCREENSHOT):



PROGRAM CODE:

OUTPUT (SCREENSHOT):

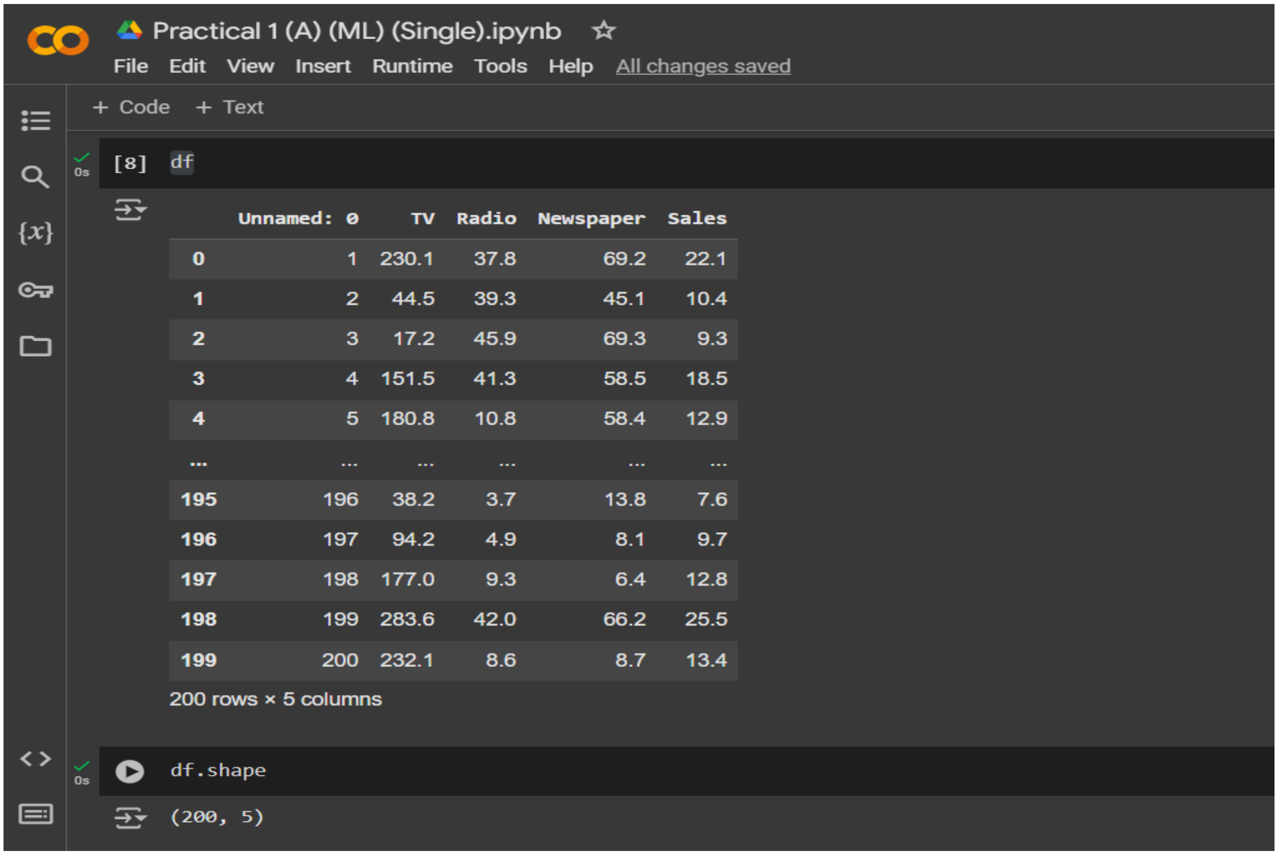


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
[6] import warnings
    warnings.filterwarnings('ignore')
    #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
```

```
[7] path="/content/Advertising.csv"
    df=pd.read_csv(path)
    df.head()
```

```
   Unnamed: 0     TV  Radio  Newspaper  Sales
0           1  230.1   37.8       69.2   22.1
1           2   44.5   39.3       45.1   10.4
2           3   17.2   45.9       69.3    9.3
3           4  151.5   41.3       58.5   18.5
4           5  180.8   10.8       58.4   12.9
```


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

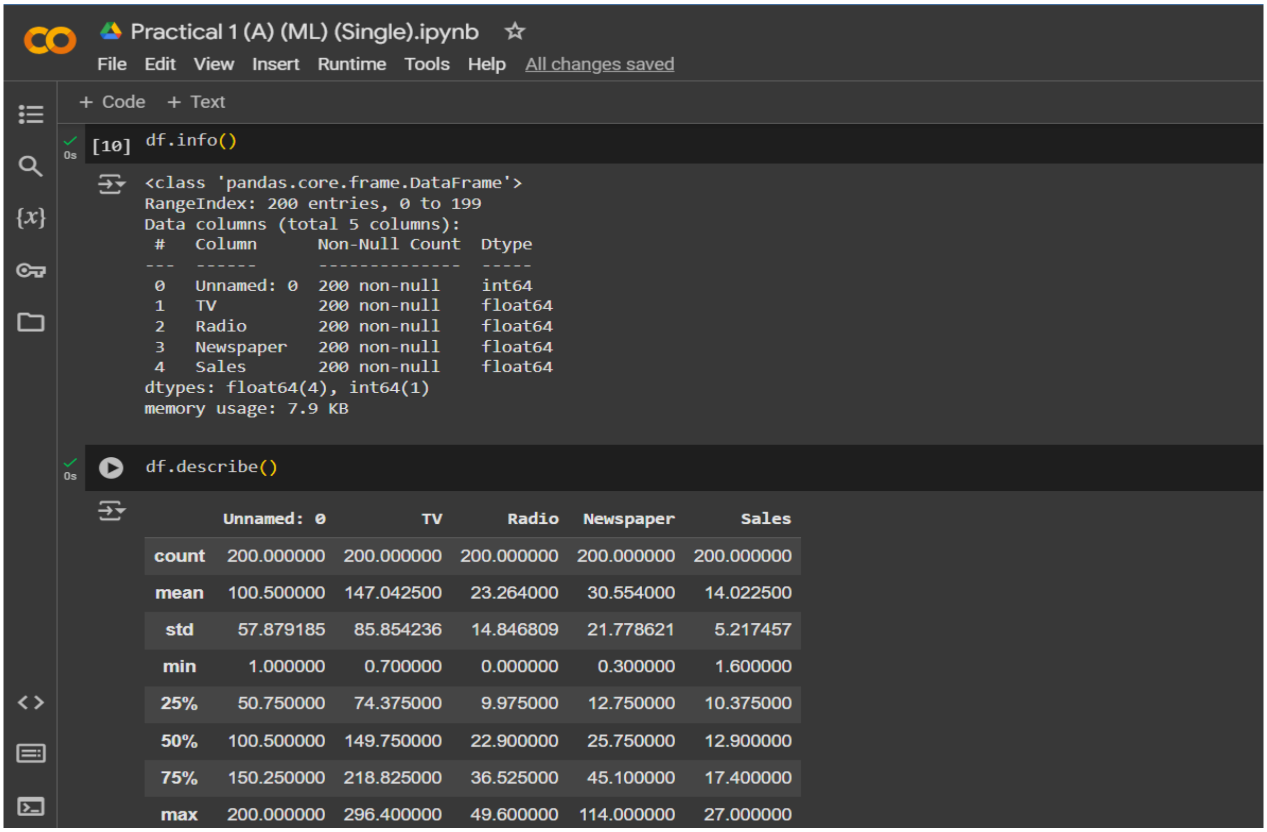
```
[8] df
```

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9
...	...	...	...	...	...
195	196	38.2	3.7	13.8	7.6
196	197	94.2	4.9	8.1	9.7
197	198	177.0	9.3	6.4	12.8
198	199	283.6	42.0	66.2	25.5
199	200	232.1	8.6	8.7	13.4

200 rows × 5 columns

```
df.shape
```

```
(200, 5)
```

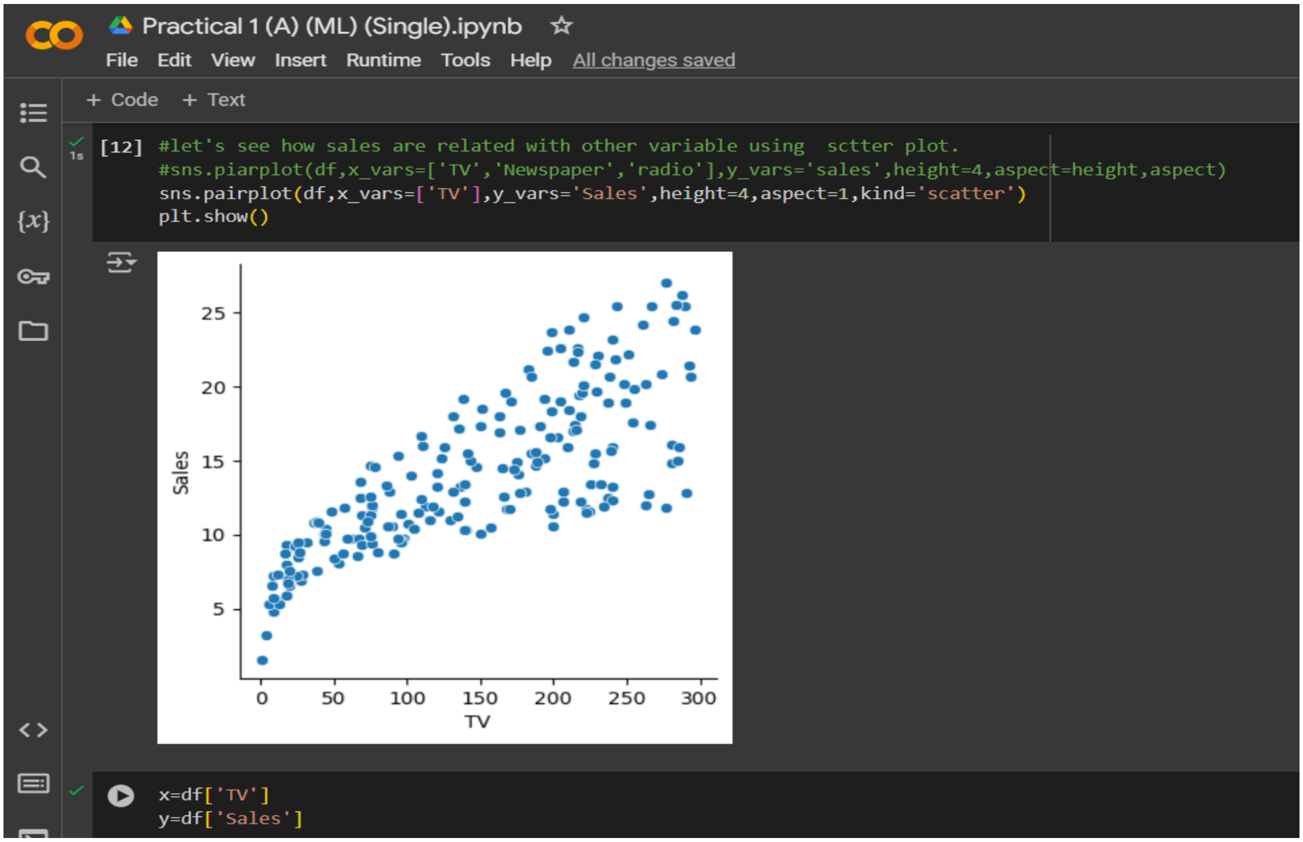

Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
[10] 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
```

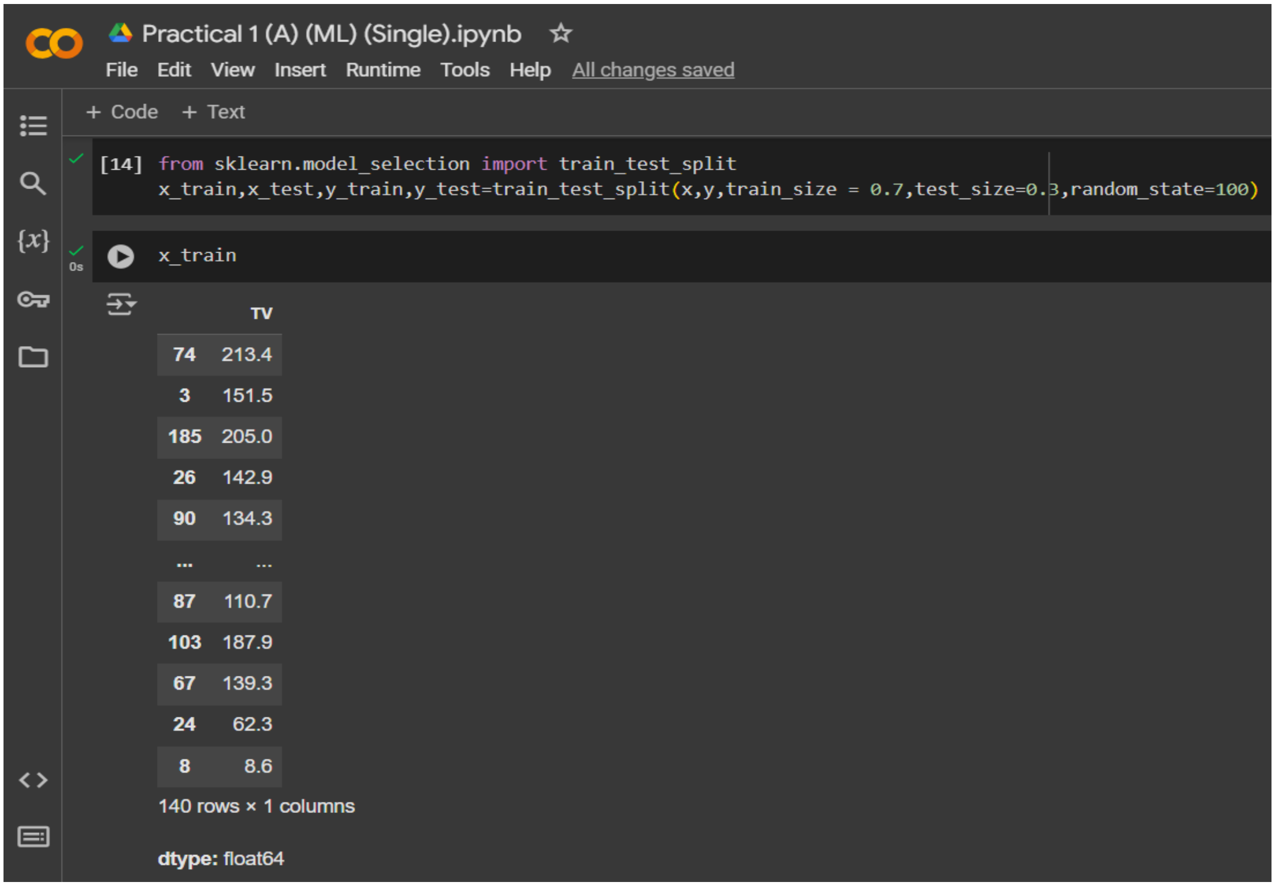
```
df.describe()
```

	Unnamed: 0	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	147.042500	23.264000	30.554000	14.022500
std	57.879185	85.854236	14.846809	21.778621	5.217457
min	1.000000	0.700000	0.000000	0.300000	1.600000
25%	50.750000	74.375000	9.975000	12.750000	10.375000
50%	100.500000	149.750000	22.900000	25.750000	12.900000
75%	150.250000	218.825000	36.525000	45.100000	17.400000
max	200.000000	296.400000	49.600000	114.000000	27.000000


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Sales
TV
25
20
15
10
5
0
50
100
150
200
250
300

```
[12] #let's see how sales are related with other variable using  sctter plot.
     #sns.piarplot(df,x_vars=['TV','Newspaper','radio'],y_vars='sales',height=4,aspect=height,aspect)
     sns.pairplot(df,x_vars=['TV'],y_vars='Sales',height=4,aspect=1,kind='scatter')
     plt.show()
```

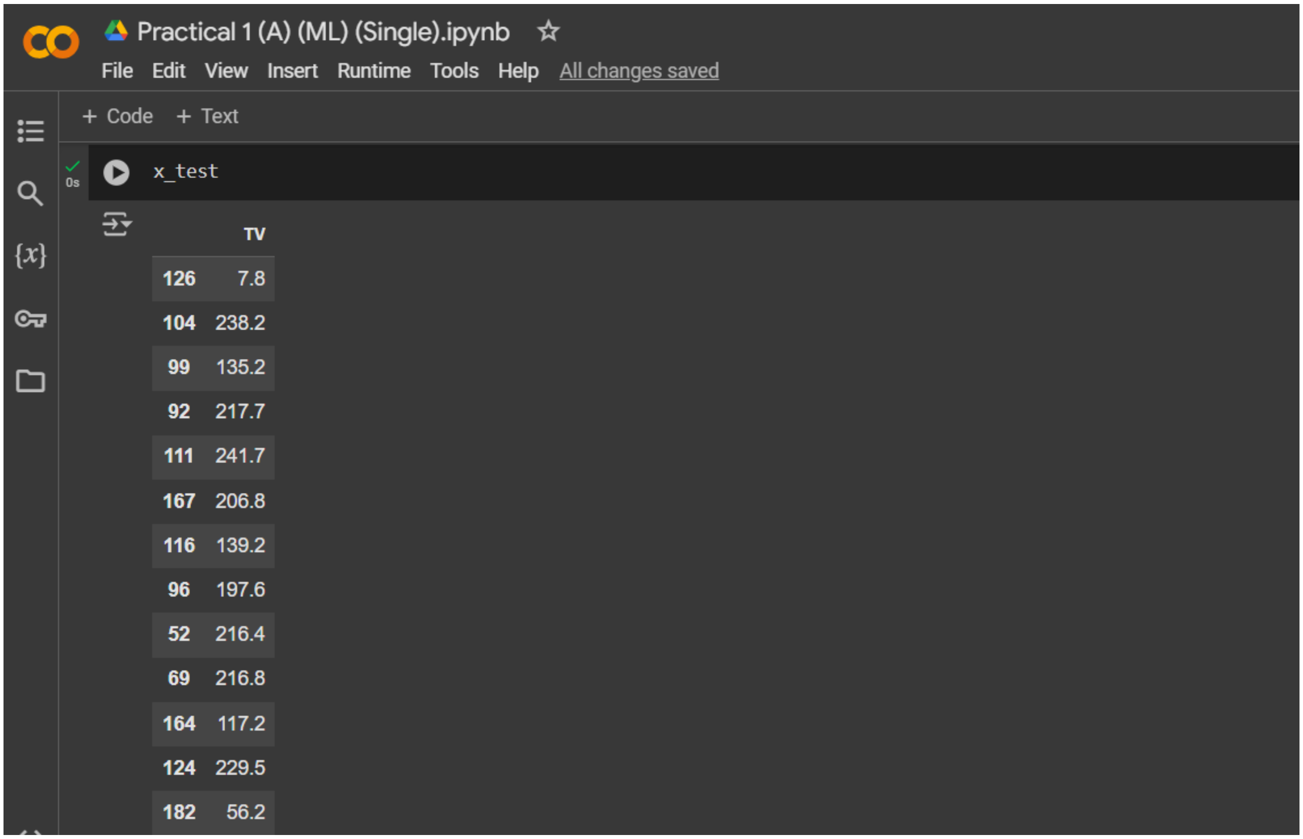
```
x=df['TV']
y=df['Sales']
```


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
[14] from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,train_size = 0.7,test_size=0.3,random_state=100)
```

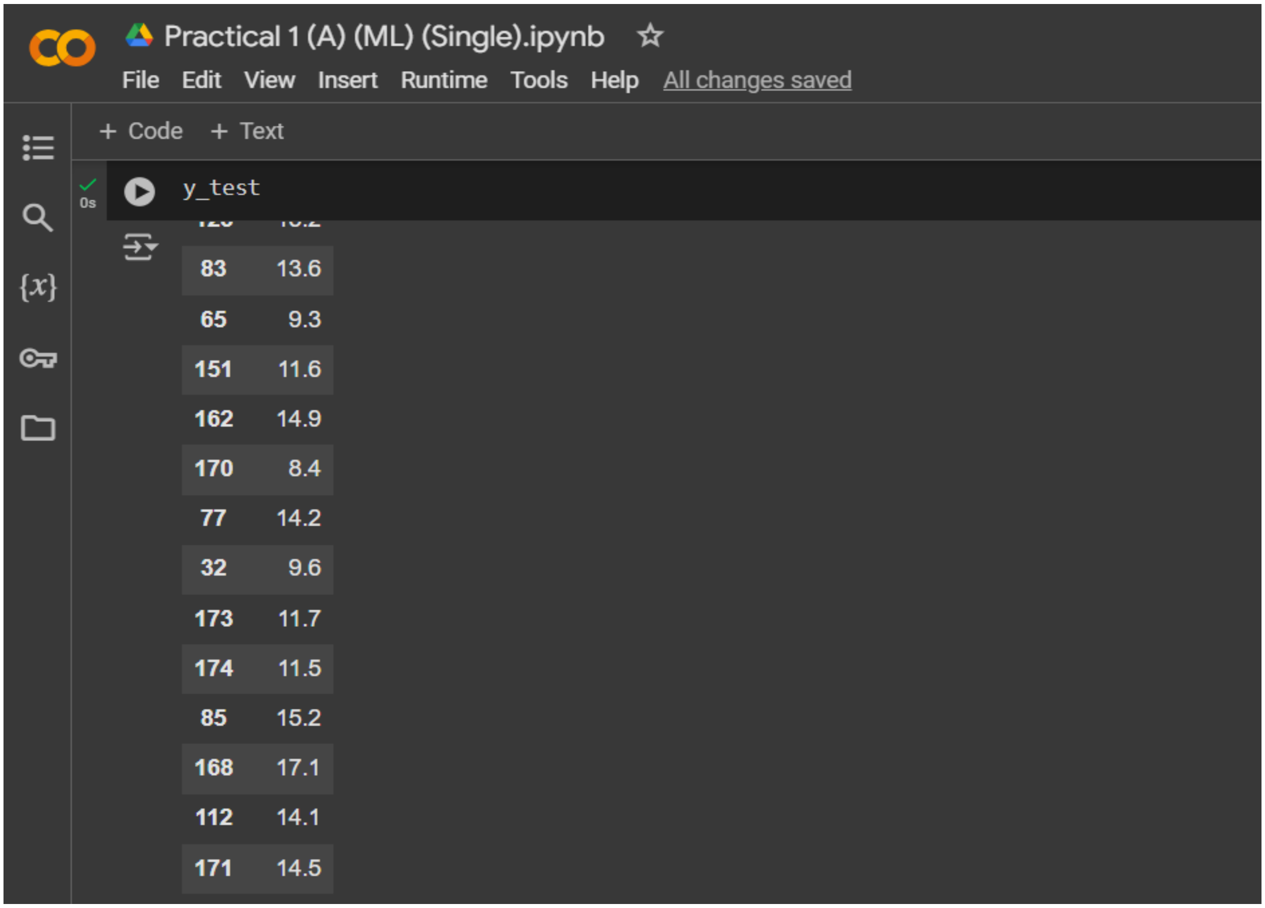
```
x_train
```

```
        TV
 74  213.4
  3  151.5
185  205.0
 26  142.9
 90  134.3
...    ...
 87  110.7
103  187.9
 67  139.3
 24   62.3
  8    8.6

140 rows × 1 columns

dtype: float64
```


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

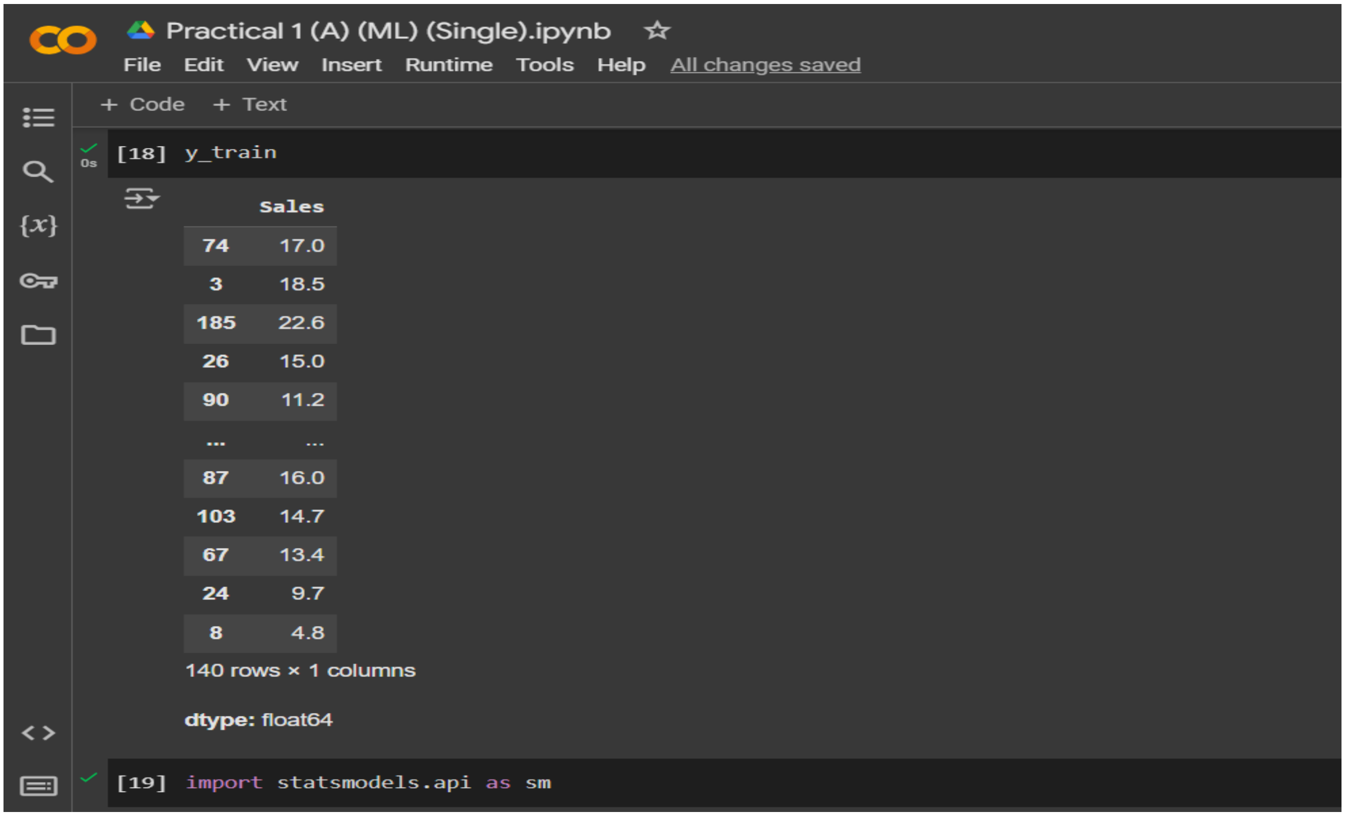
```
x_test
```

```
        TV
126    7.8
104  238.2
 99  135.2
 92  217.7
111  241.7
167  206.8
116  139.2
 96  197.6
 52  216.4
 69  216.8
164  117.2
124  229.5
182   56.2
```


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

```
y_test
```

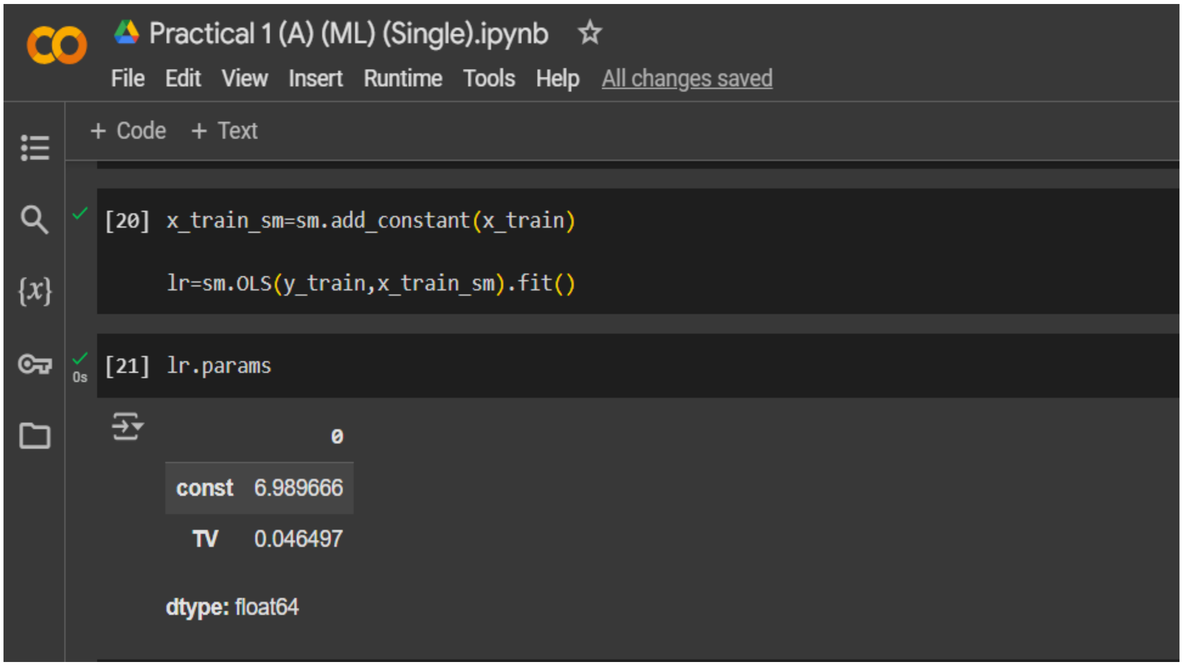
```
 83  13.6
 65   9.3
151  11.6
162  14.9
170   8.4
 77  14.2
 32   9.6
173  11.7
174  11.5
 85  15.2
168  17.1
112  14.1
171  14.5
```


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
[18] y_train
```

```
       Sales
 74     17.0
  3     18.5
185     22.6
 26     15.0
 90     11.2
...      ...
 87     16.0
103     14.7
 67     13.4
 24      9.7
  8      4.8
140 rows × 1 columns

dtype: float64
```

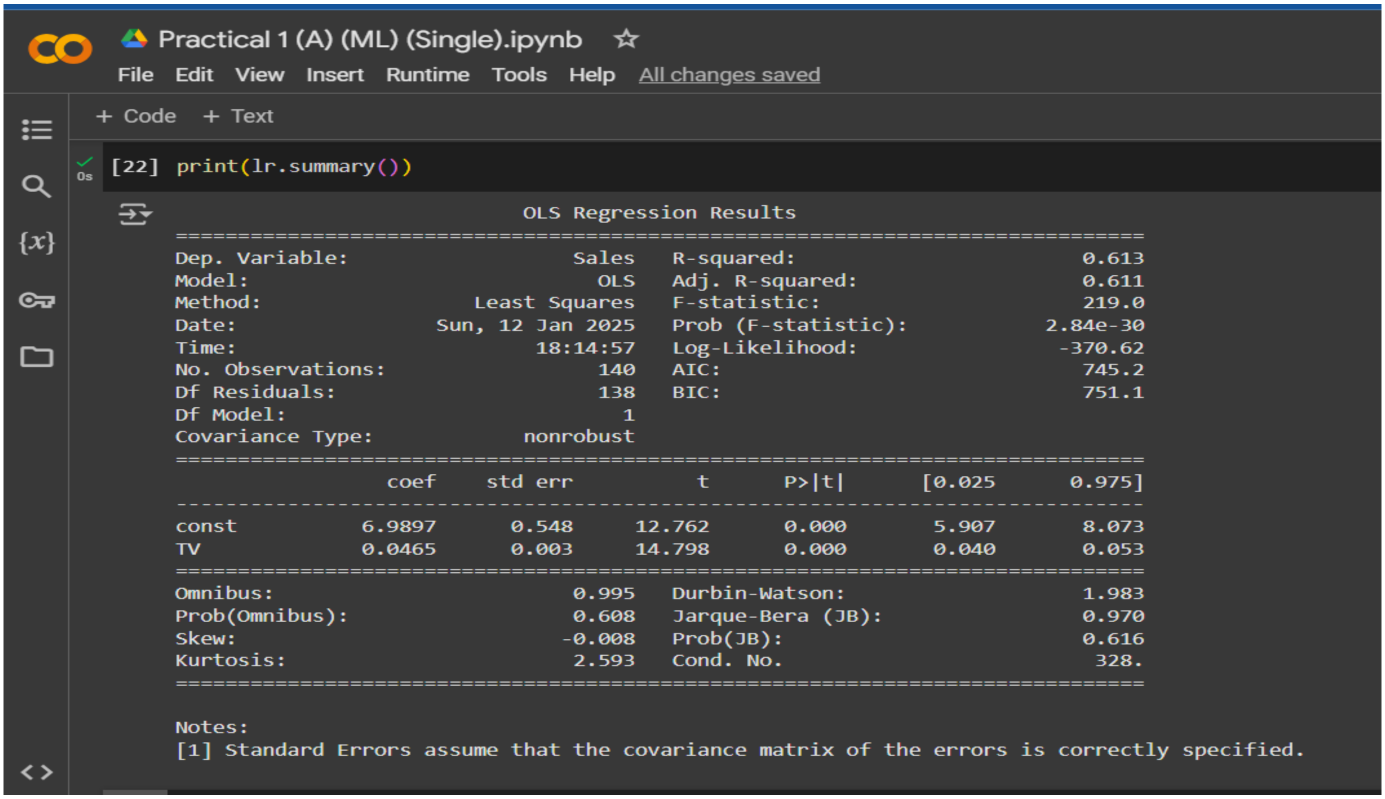
```
[19] import statsmodels.api as sm
```


Practical 1 (A) (ML) (Single).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
[20] x_train_sm=sm.add_constant(x_train)

     lr=sm.OLS(y_train,x_train_sm).fit()
```

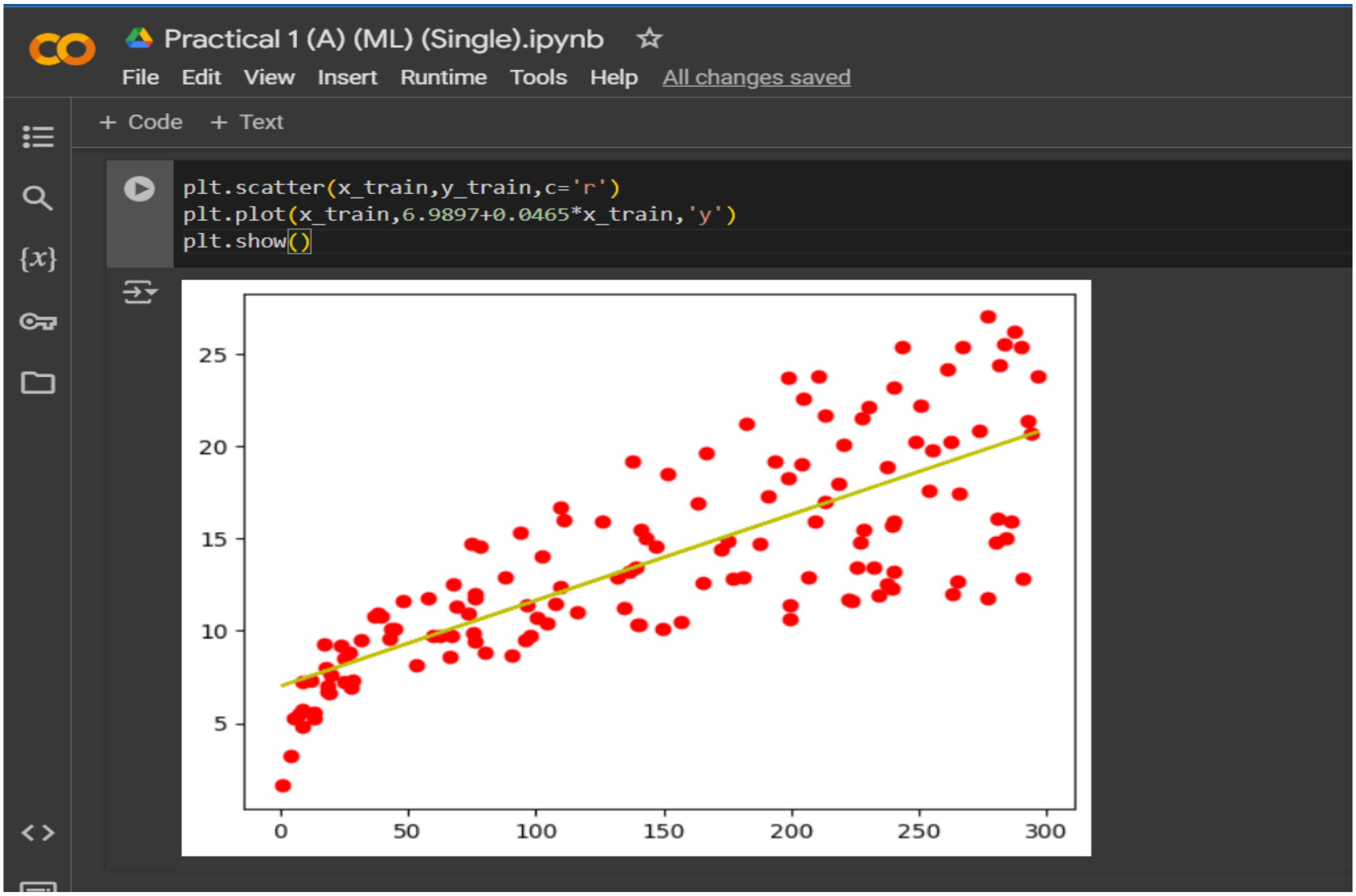
```
[21] lr.params
```

```
              0
const  6.989666
TV     0.046497

dtype: float64
```

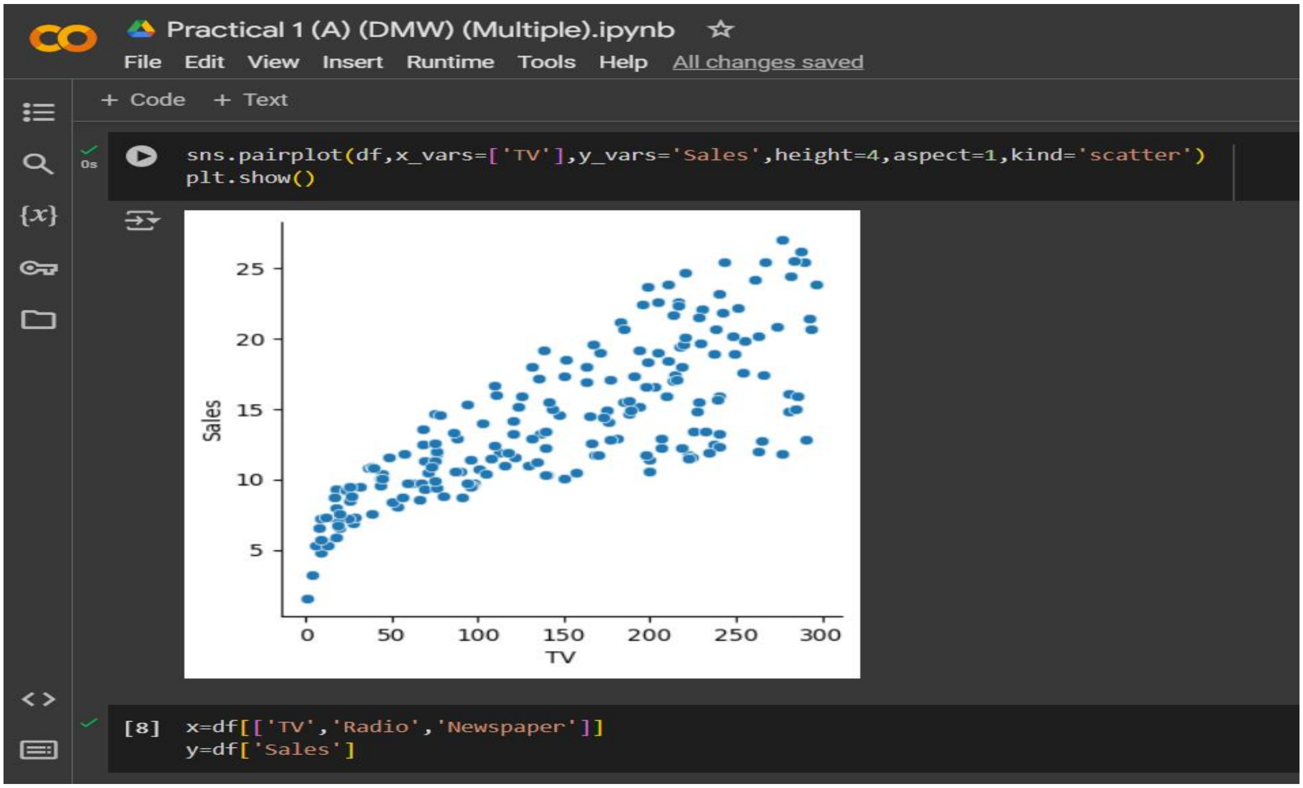


```
[22] print(lr.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Sales   R-squared:                       0.613
Model:                            OLS   Adj. R-squared:                  0.611
Method:                 Least Squares   F-statistic:                     219.0
Date:                Sun, 12 Jan 2025   Prob (F-statistic):           2.84e-30
Time:                        18:14:57   Log-Likelihood:                -370.62
No. Observations:                 140   AIC:                             745.2
Df Residuals:                     138   BIC:                             751.1
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          6.9897      0.548     12.762      0.000       5.907       8.073
TV             0.0465      0.003     14.798      0.000       0.040       0.053
==============================================================================
Omnibus:                        0.995   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.608   Jarque-Bera (JB):                0.970
Skew:                          -0.008   Prob(JB):                        0.616
Kurtosis:                       2.593   Cond. No.                         328.
==============================================================================

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

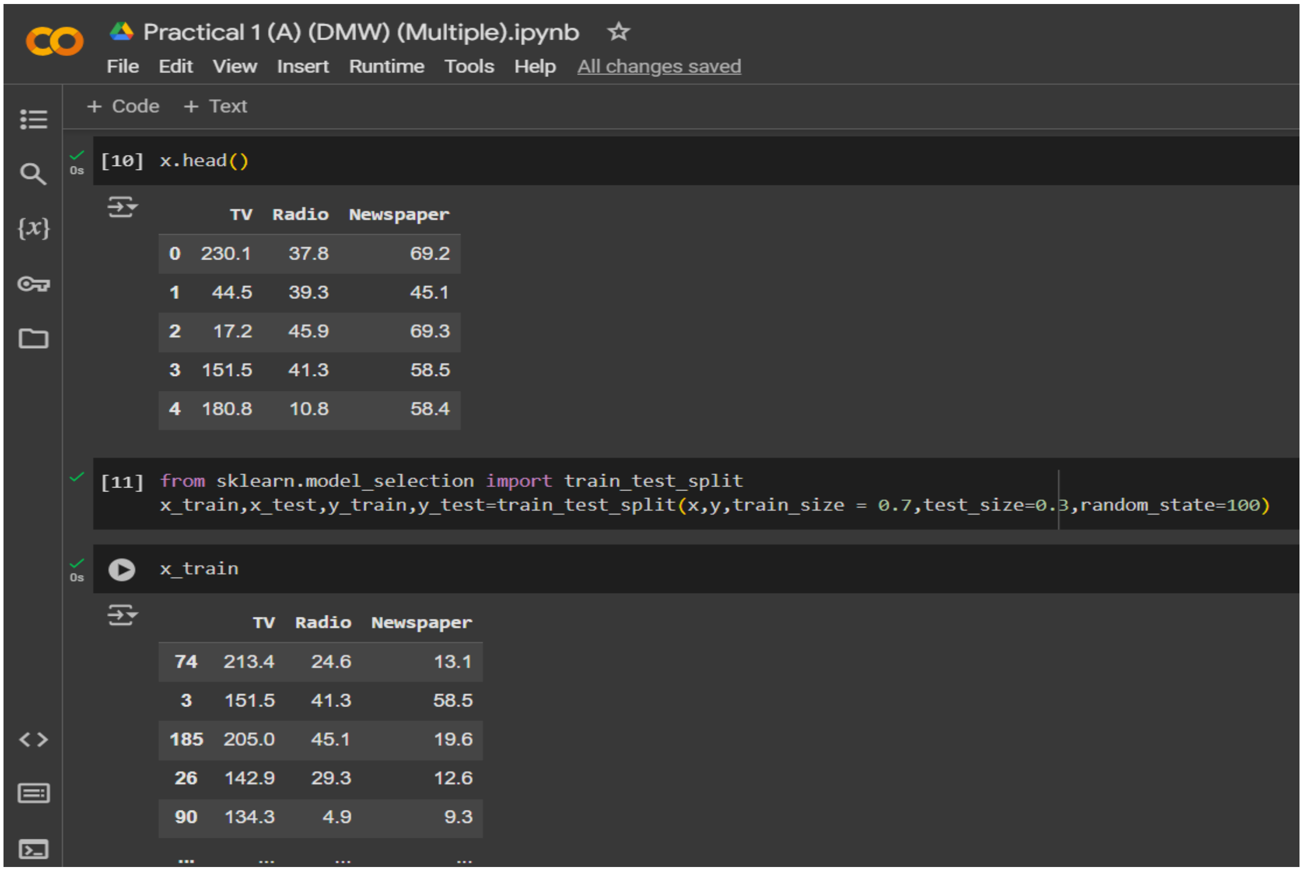


```
plt.scatter(x_train,y_train,c='r')
plt.plot(x_train,6.9897+0.0465*x_train,'y')
plt.show()
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s
Sales
25
20
15
10
5
0
50
100
150
200
250
300
TV

```
sns.pairplot(df,x_vars=['TV'],y_vars='Sales',height=4,aspect=1,kind='scatter')
plt.show()
```

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


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s
0s

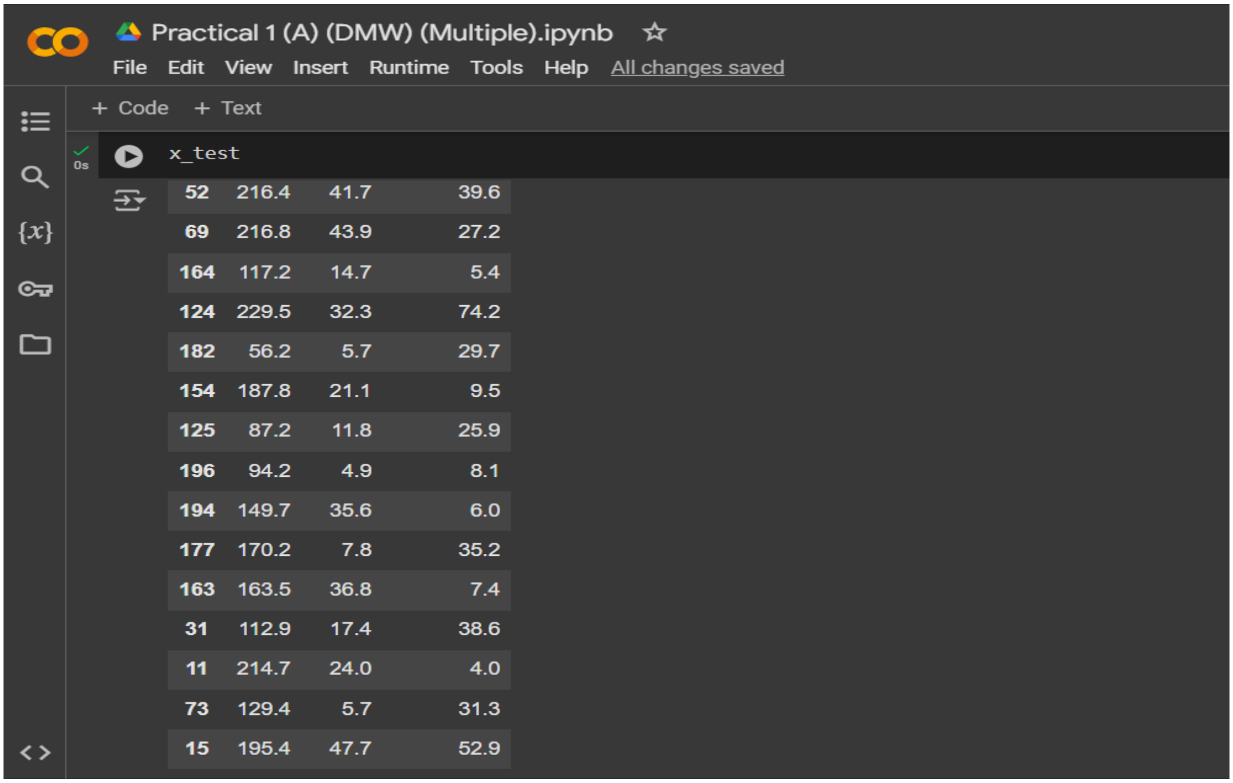
```
[10] x.head()
```

```
      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
```

```
[11] from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,train_size = 0.7,test_size=0.3,random_state=100)
```

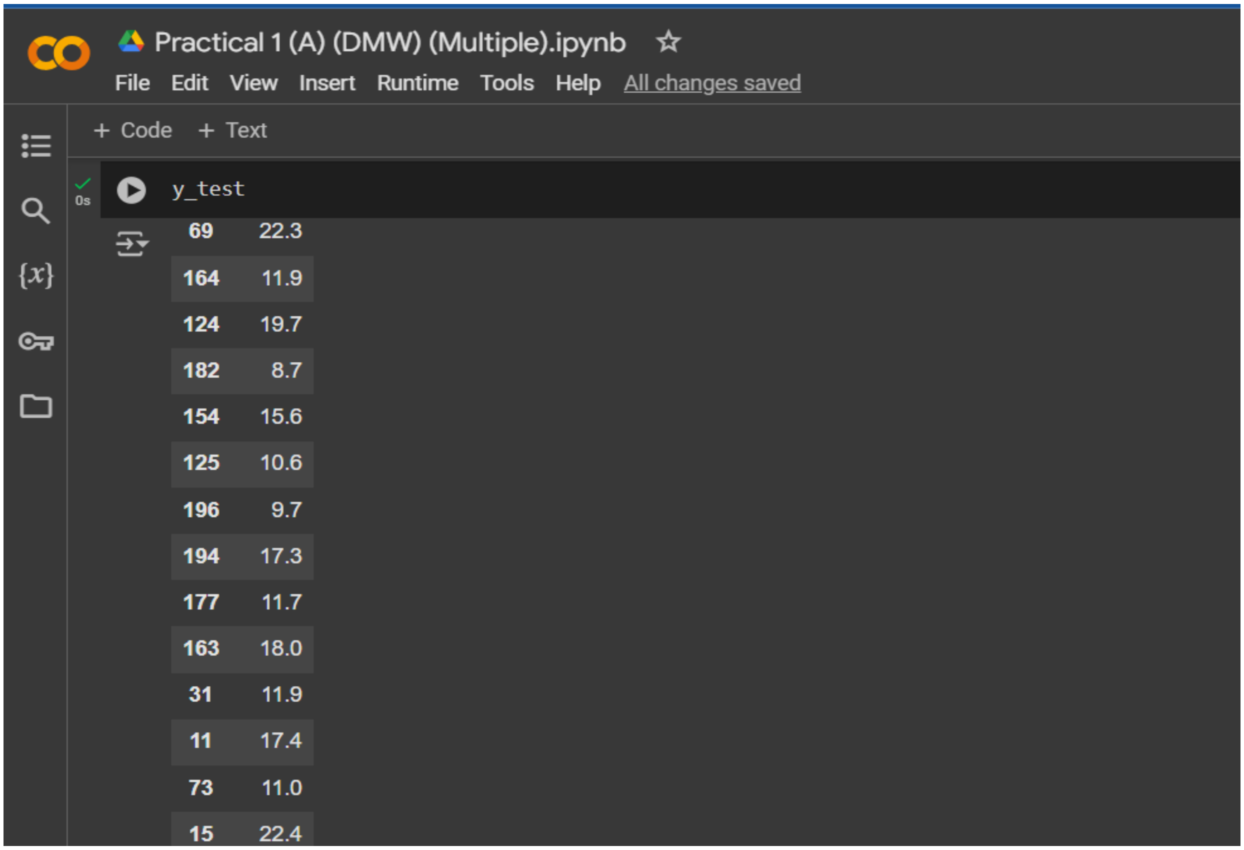
```
x_train
```

```
        TV  Radio  Newspaper
74   213.4   24.6       13.1
3    151.5   41.3       58.5
185  205.0   45.1       19.6
26   142.9   29.3       12.6
90   134.3    4.9        9.3
...    ...    ...        ...
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

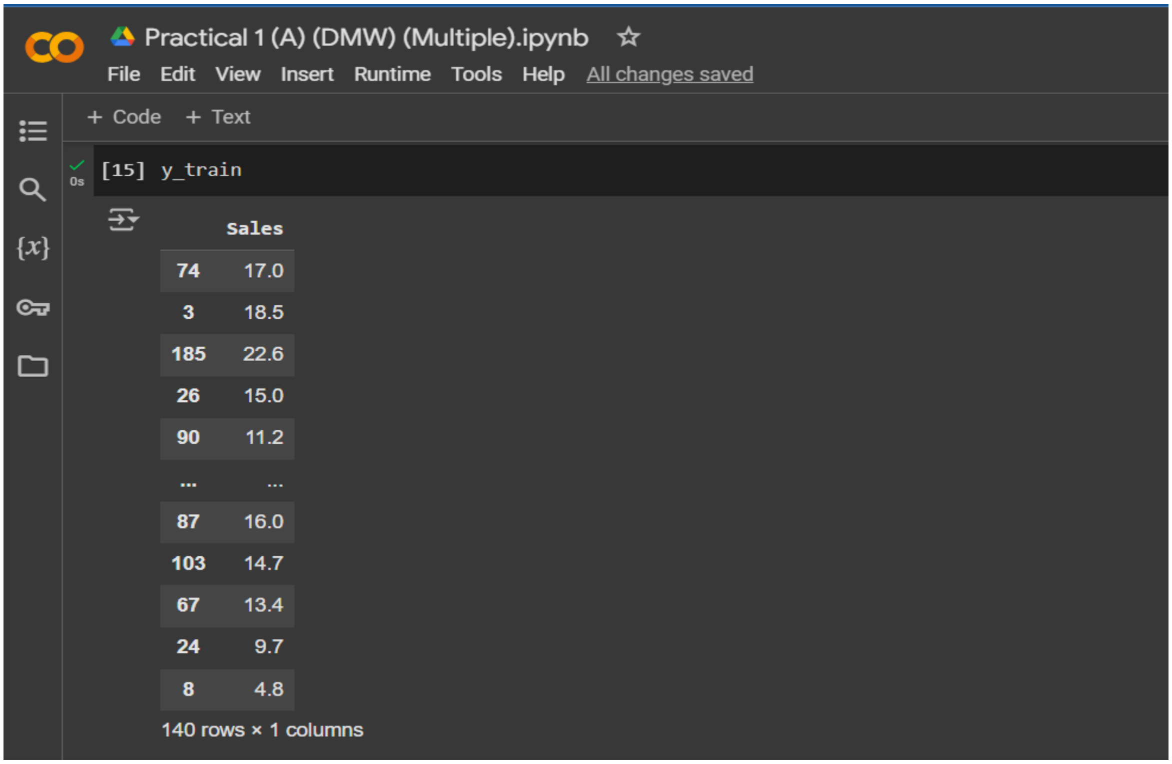
```
x_test
```

```
 52  216.4  41.7  39.6
 69  216.8  43.9  27.2
164  117.2  14.7   5.4
124  229.5  32.3  74.2
182   56.2   5.7  29.7
154  187.8  21.1   9.5
125   87.2  11.8  25.9
196   94.2   4.9   8.1
194  149.7  35.6   6.0
177  170.2   7.8  35.2
163  163.5  36.8   7.4
 31  112.9  17.4  38.6
 11  214.7  24.0   4.0
 73  129.4   5.7  31.3
 15  195.4  47.7  52.9
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

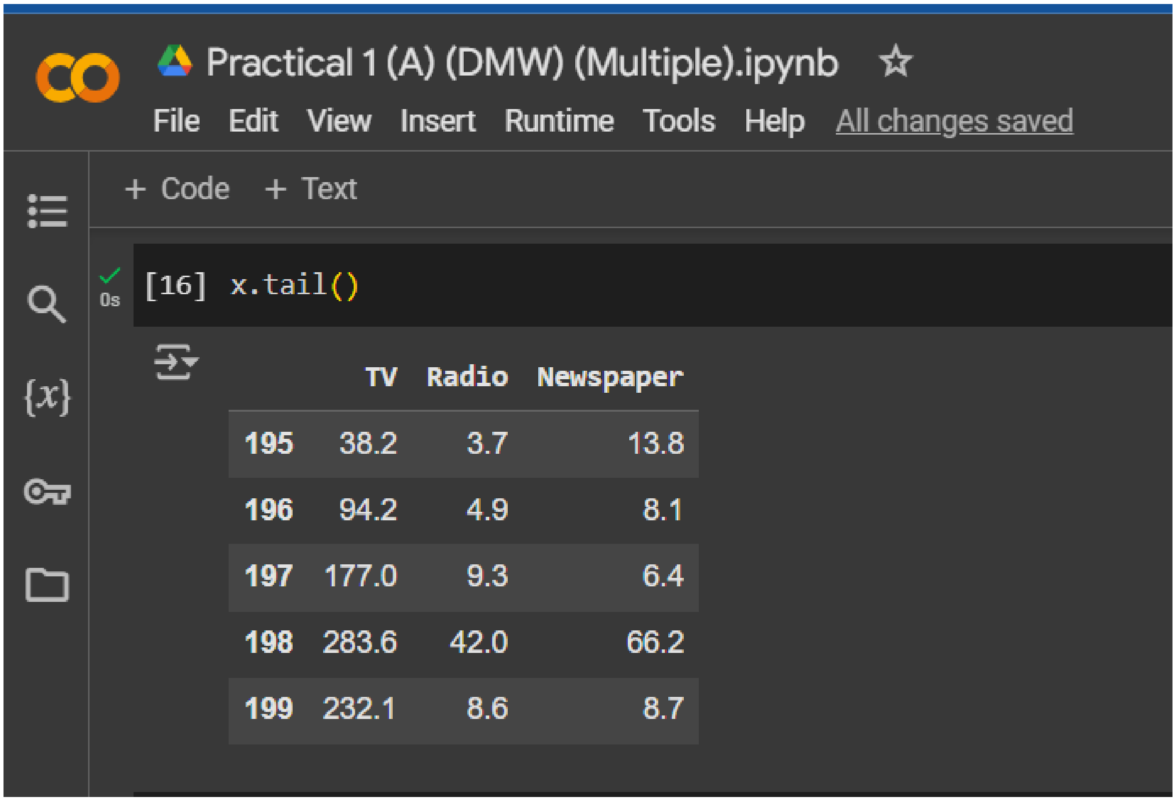
```
y_test
```

```
 69  22.3
164  11.9
124  19.7
182   8.7
154  15.6
125  10.6
196   9.7
194  17.3
177  11.7
163  18.0
 31  11.9
 11  17.4
 73  11.0
 15  22.4
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

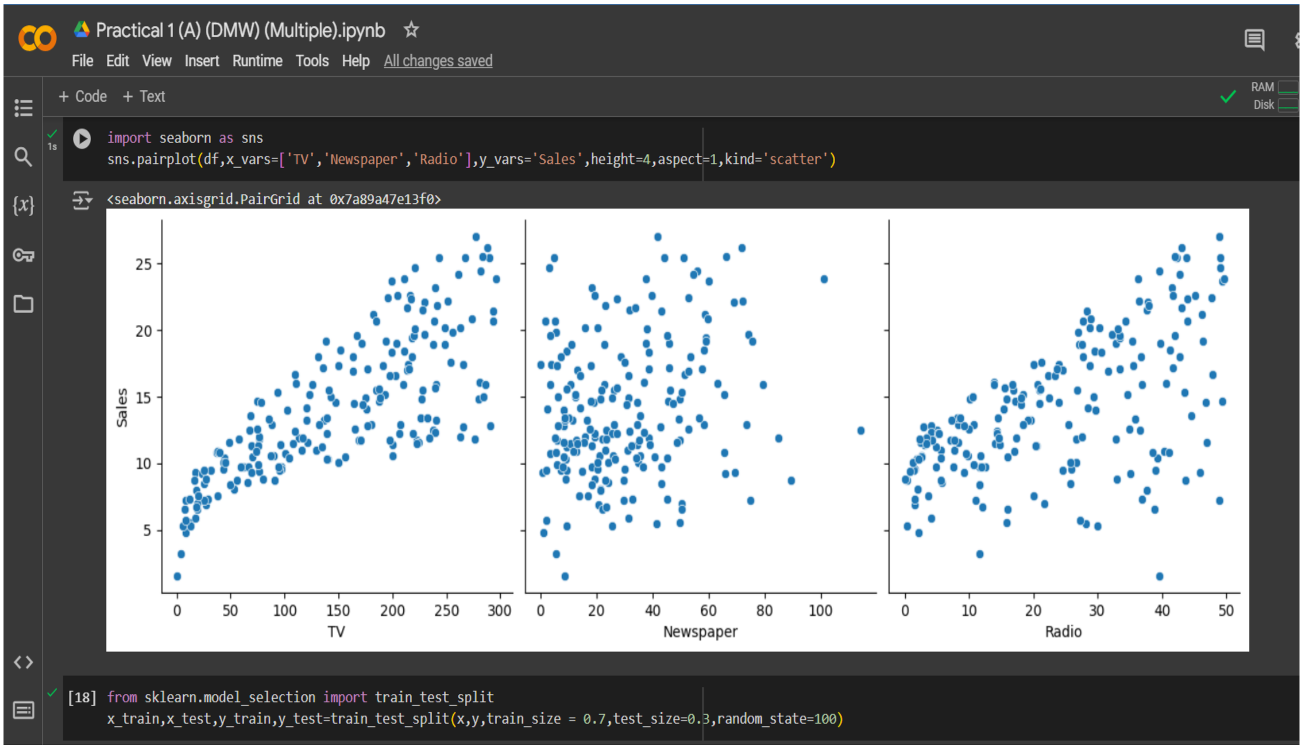
```
[15] y_train
```

```
      Sales
 74    17.0
  3    18.5
185    22.6
 26    15.0
 90    11.2
...     ...
 87    16.0
103    14.7
 67    13.4
 24     9.7
  8     4.8
140 rows × 1 columns
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

```
[16] x.tail()
```

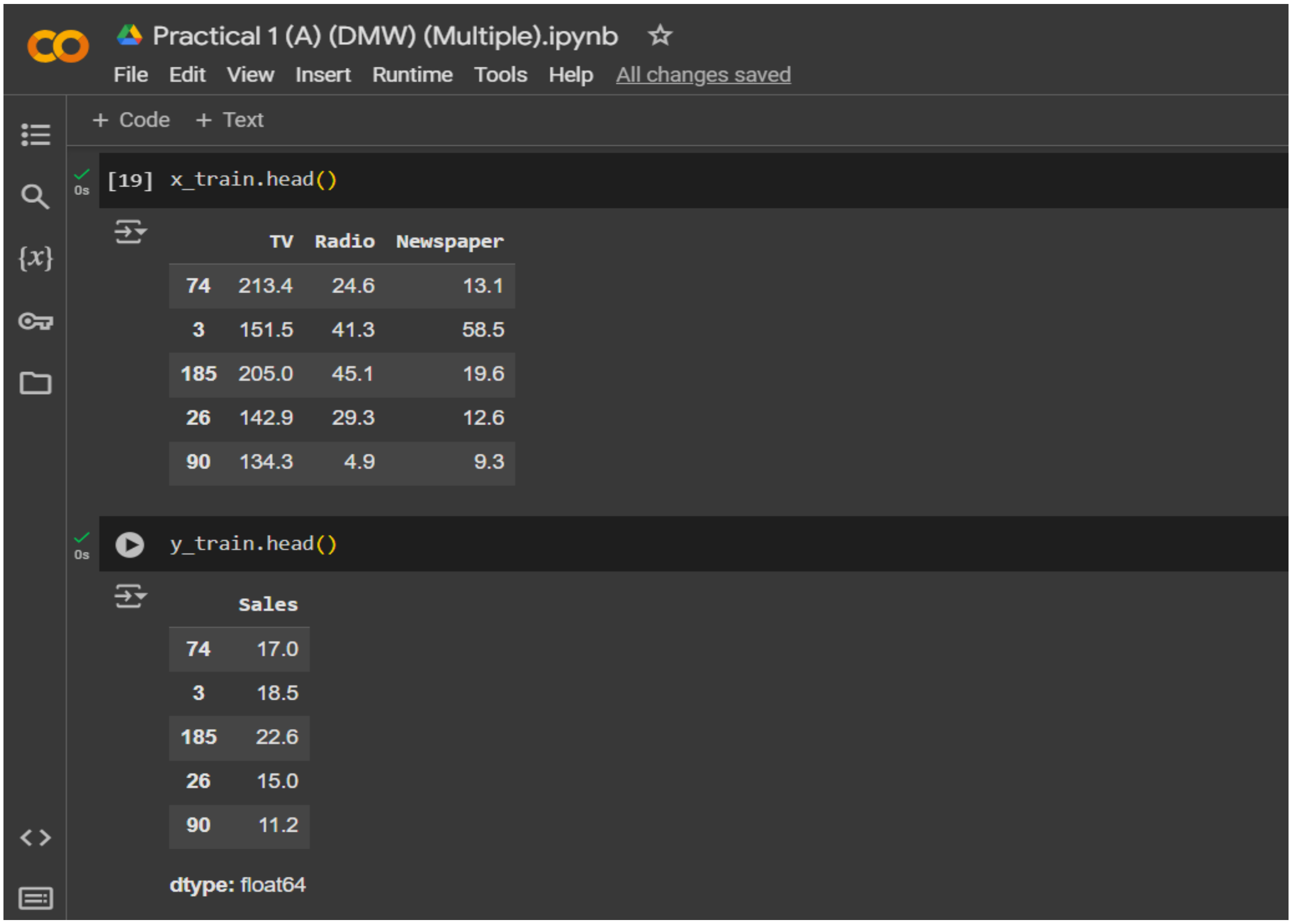
```
        TV  Radio  Newspaper
195   38.2    3.7       13.8
196   94.2    4.9        8.1
197  177.0    9.3        6.4
198  283.6   42.0       66.2
199  232.1    8.6        8.7
```



```
import seaborn as sns
sns.pairplot(df,x_vars=['TV','Newspaper','Radio'],y_vars='Sales',height=4,aspect=1,kind='scatter')
```

```
<seaborn.axisgrid.PairGrid at 0x7a89a47e13f0>
```

```
[18] from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,train_size = 0.7,test_size=0.3,random_state=100)
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

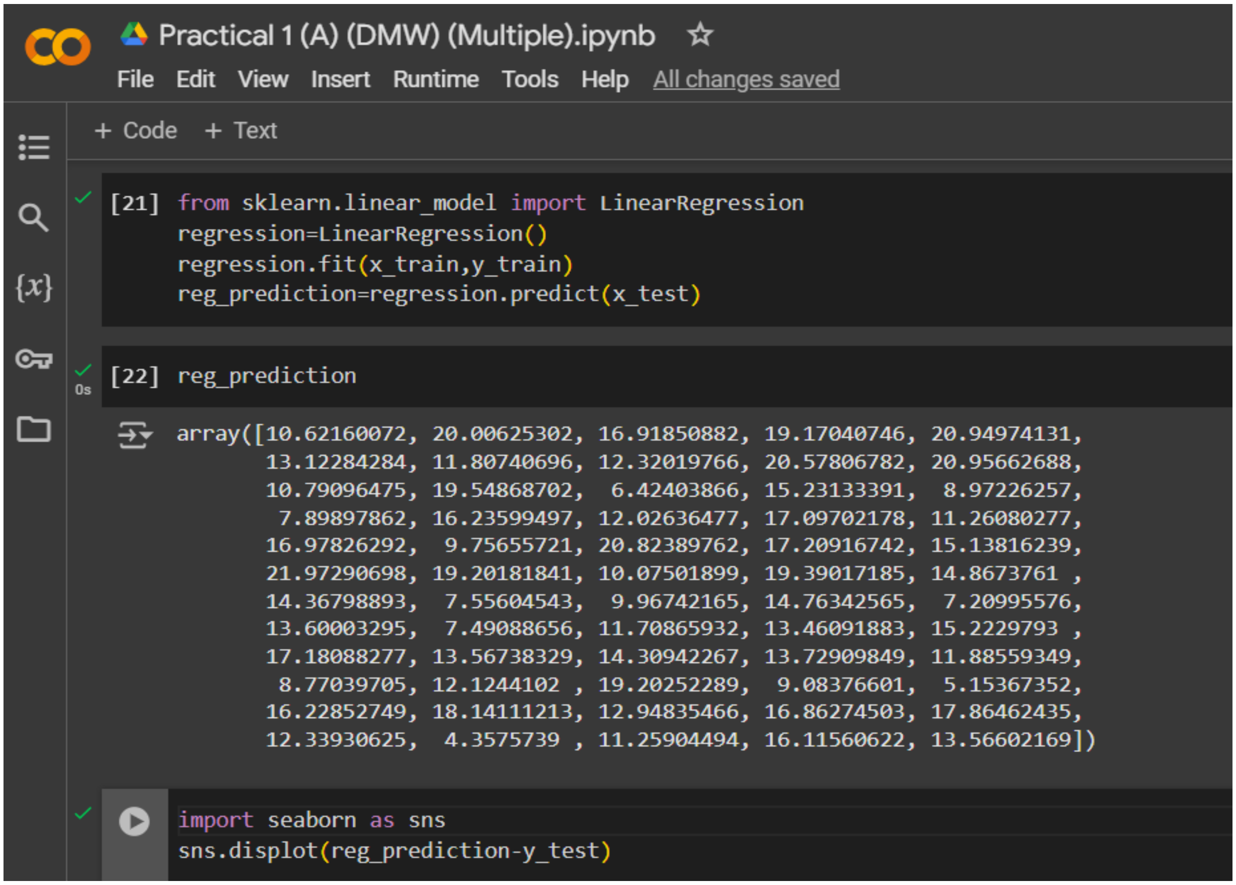
```
[19] x_train.head()
```

	TV	Radio	Newspaper
74	213.4	24.6	13.1
3	151.5	41.3	58.5
185	205.0	45.1	19.6
26	142.9	29.3	12.6
90	134.3	4.9	9.3

```
y_train.head()
```

	Sales
74	17.0
3	18.5
185	22.6
26	15.0
90	11.2

dtype: float64

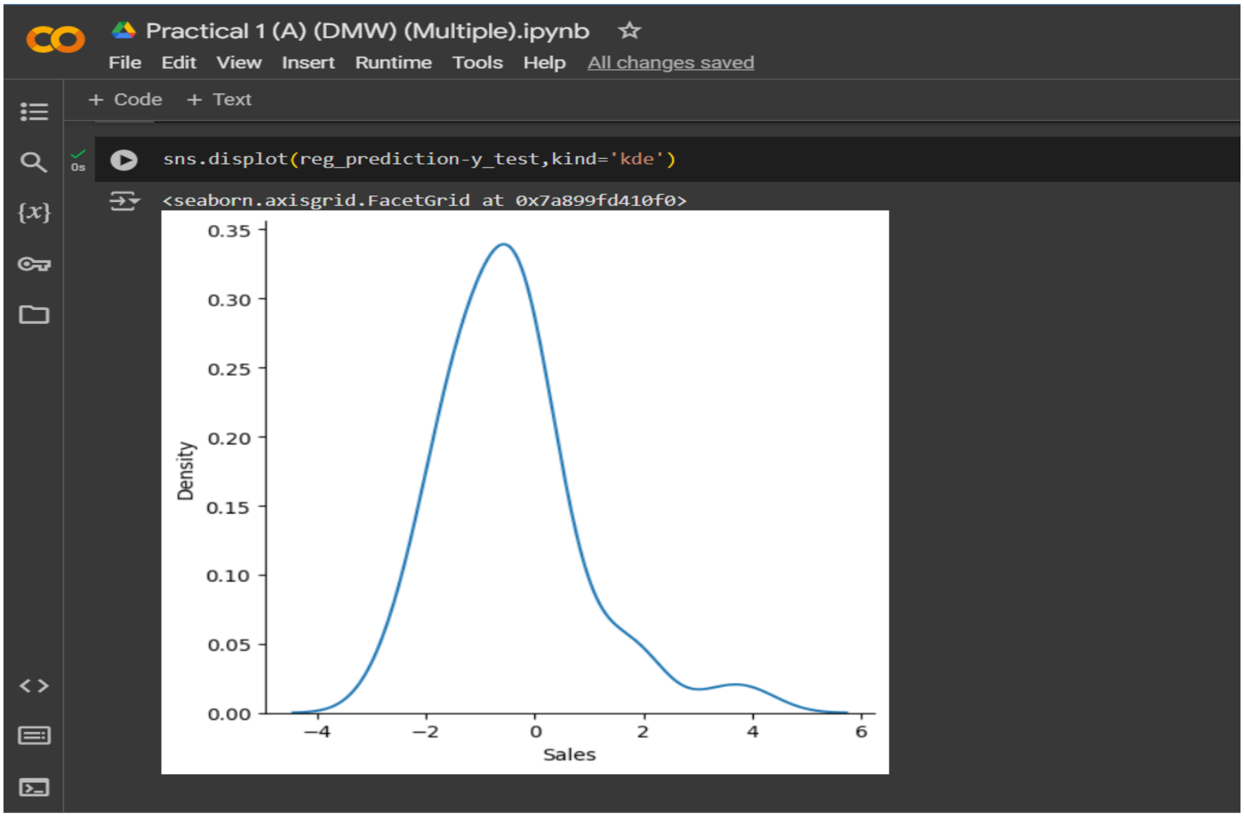

Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s

```
[21] from sklearn.linear_model import LinearRegression
     regression=LinearRegression()
     regression.fit(x_train,y_train)
     reg_prediction=regression.predict(x_test)
```

```
[22] reg_prediction
```

```
array([10.62160072, 20.00625302, 16.91850882, 19.17040746, 20.94974131,
       13.12284284, 11.80740696, 12.32019766, 20.57806782, 20.95662688,
       10.79096475, 19.54868702,  6.42403866, 15.23133391,  8.97226257,
        7.89897862, 16.23599497, 12.02636477, 17.09702178, 11.26080277,
       16.97826292,  9.75655721, 20.82389762, 17.20916742, 15.13816239,
       21.97290698, 19.20181841, 10.07501899, 19.39017185, 14.8673761 ,
       14.36798893,  7.55604543,  9.96742165, 14.76342565,  7.20995576,
       13.60003295,  7.49088656, 11.70865932, 13.46091883, 15.2229793 ,
       17.18088277, 13.56738329, 14.30942267, 13.72909849, 11.88559349,
        8.77039705, 12.1244102 , 19.20252289,  9.08376601,  5.15367352,
       16.22852749, 18.14111213, 12.94835466, 16.86274503, 17.86462435,
       12.33930625,  4.3575739 , 11.25904494, 16.11560622, 13.56602169])
```

```
import seaborn as sns
sns.displot(reg_prediction-y_test)
```


Practical 1 (A) (DMW) (Multiple).ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
0s
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Density
−4
−2
0
2
4
6
Sales

```
sns.displot(reg_prediction-y_test,kind='kde')
```

```
<seaborn.axisgrid.FacetGrid at 0x7a899fd410f0>
```

CONCLUSION:

_ _

_ _

_ _

_ _ _ _ _ _ _ _ _ ___ _

_ _ _ _ _ _ _ _ _ ___ _

DISCUSSION AND VIVA VOCE:

- What is linear regression in machine learning example?
- What algorithm is used in linear regression?
- What is the application of linear regression?
- What are some real life examples of linear regression?
- What is linear regression and its types?

REFERENCE:

- https://www.google.com/search?q=linear+regression+machine+learning&oq=Linear+Regression&aqs=chrome.1.0i433i512j0i512j69i59j0i131i433i512l2j69i60l3.4688j0j7&sourceid=chrome&ie=UTF-8
- https://pubs.acs.org/doi/10.1021/acsomega.2c00362
- https://colab.research.google.com/drive/1gbyjW-RgdgVkUNyRkHAWmjMr7RUCUrYX