MOHAMMED YOUNUS

Sales Predictions using Advertising Costs by using Machine Learning Model

<u>Problem Statement</u> – In the business world, advertising is a crucial element for any company looking to promote its products or services. However, advertising costs can be substantial, and businesses need to determine the effectiveness of their advertising campaigns. This is where sales prediction comes in – it's a critical aspect of advertising that helps companies understand how much revenue they can expect from their advertising campaigns. In this project I build a model which predicts sales based on the money spent on different advertising platforms for marketing.

🔶 <u>Steps to be taken in the Project is sub-divided into the following sections.</u>
<u>These are:</u>

- Loading necessary libraries such as numpy , pandas , sklearn etc.
- Loading the dataset as CSV file and showing first 10 rows.
- Drop the unnecessary columns from dataset.
- Calculate statistical values and round them up to 3 decimal places. ☐ Checking for null values and return their sum of numbers of true values in ☐ each column.
- Handle the null by mean of all values fill into them.
- Extracting all information about data.
- Checking Shape of Data.
- Visualization of <u>Sales by different source of Advertisement cost</u> using Python data visualization.
- Data preprocessing or (Data cleaning) performed by the one hot encoding in this process we change categorical data into numerical data and the technique is called feature Engineering.
- Splitting the cleaned data into dependent and independent variables.
- Splitting the data into train and test sets with train_test_split using sklearn library.
- Import different kind of Regression Models and Train that model with the help of .fit().
- Predicting the trained models and then checking their accuracy of the model using accuracy score.
- Then recall the train_test_split and split the data into training and testing set with different models.
- Then predicting the trained models and checking the accuracy of model and check the accuracy difference.

MOHAMMED YOUNUS

☐ <u>Step-1</u> – Loading Necessary Libraries used in machine learning.

Import Necessary Libraries.

```
[ ]  import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as snr
     from sklearn.preprocessing import OneHotEncoder, PolynomialFeatures
     from sklearn.metrics import r2_score
```

☐ <u>Step-2</u> – Loading the dataset as csv file and showing first ten rows.

Load the data and show first 10 rows of data.

```
data=pd.read_csv("/content/advertising.csv")
data.head(10)
```

	sno	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	12.0
3	4	151.5	41.3	58.5	16.5
4	5	180.8	10.8	58.4	17.9
5	6	8.7	48.9	75.0	7.2
6	7	57.5	32.8	23.5	11.8
7	8	120.2	19.6	11.6	13.2
8	9	8.6	2.1	1.0	4.8
9	10	199.8	2.6	21.2	15.6

✓ 0s completed at 10:43 PM

☐ <u>Step-3</u> – Drop the unnecessary columns from dataset.

Drop Unnecessary columns from data

```
data_new=data.drop(['sno'],axis=1)
data_new.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	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

☐ <u>Step-4</u> – Calculate statistical values and round them up to 3 decimal places.

Calculates statistical values and rounds them to 3 decimal places.

```
data_new.describe().round(3)
```

	TV	Radio	Newspaper	Sales
count	200.000	200.000	200.000	200.000
mean	147.042	23.264	30.554	15.131
std	85.854	14.847	21.779	5.284
min	0.700	0.000	0.300	1.600
25%	74.375	9.975	12.750	11.000
50%	149.750	22.900	25.750	16.000
75%	218.825	36.525	45.100	19.050
max	296.400	49.600	114.000	27.000

☐ <u>Step-5</u> – Checking for null values and return their sum of numbers of true values in each column.

Mark null values as True and returns sum of number of True values in each column

```
[5] data_new.isnull().sum()
```

```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

☐ <u>Step-6</u> – Extracting all information about data.
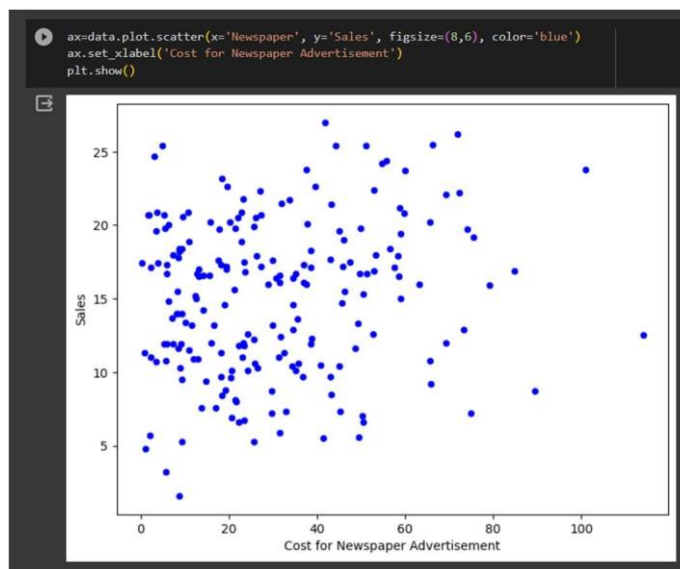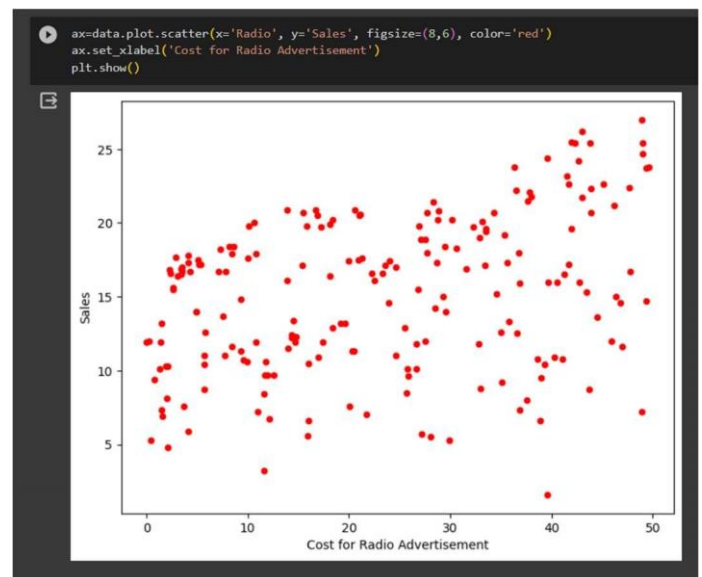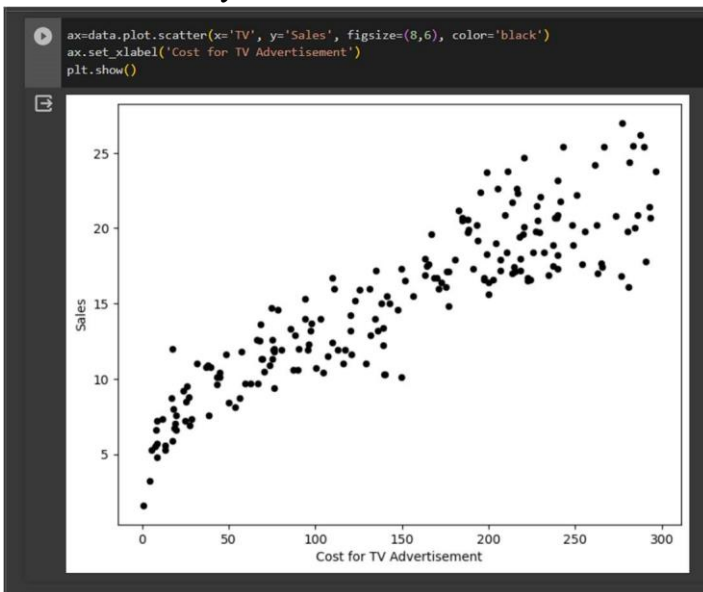
Extracting all information about data.

```
data_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   TV         200 non-null    float64
 1   Radio      200 non-null    float64
 2   Newspaper  200 non-null    float64
 3   Sales      200 non-null    float64
dtypes: float64(4)
memory usage: 6.4 KB
```

☐ <u>Step-7</u> – Checking shape of data.

MOHAMMED YOUNUS

Shape of data.

```
[7] data_new.shape
    (200, 4)
```

☐ Step-8 – Visualization of <u>Sales by different source of Advertisement cost</u> using Python data visualization.

```
ax=data.plot.scatter(x='TV', y='Sales', figsize=(8,6), color='black')
ax.set_xlabel('Cost for TV Advertisement')
plt.show()
```



```
ax=data.plot.scatter(x='Radio', y='Sales', figsize=(8,6), color='red')
ax.set_xlabel('Cost for Radio Advertisement')
plt.show()
```



```
ax=data.plot.scatter(x='Newspaper', y='Sales', figsize=(8,6), color='blue')
ax.set_xlabel('Cost for Newspaper Advertisement')
plt.show()
```

☐ <u>Step-9</u> – Splitting the cleaned data into dependent and independent variables.

```
[ ] x=data_new.drop(['Sales'], axis=1)
    y=data_new['Sales']
    y.head()

    0    22.1
    1    10.4
    2    12.0
    3    16.5
    4    17.9
    Name: Sales, dtype: float64

[ ] 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

··· Connecting to Python 3 Google Compute Engine backend

☐ <u>Step-10</u> – Splitting the data into train and test sets with train_test_split using sklearn library.

Deviding the cleaned data into training and testing sets.

```
[ ] from sklearn.model_selection import train_test_split
    x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.8)
```

☐ <u>Step-11</u> – Import first regression model 'Linear Regression'.

creating first Machine Learning Model 'Linear Regression'.

```
[ ] from sklearn.linear_model import LinearRegression
    linear=LinearRegression()
```

☐ <u>Step-12</u> – Train the model using .fit() function.

Train the model.

```
[ ] linear.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

☐ <u>Step-13</u> – Make predictions on model.

Make predictions on model.

```
predictions=linear.predict(x_test)
print(predictions)
```

```
[19.35348128 18.57298689 15.31853389 21.06983525 19.39310167 12.21635064
 18.40278288 10.92286818 18.73897836 14.41175839 14.22131534 18.8457162
 15.79597205 11.31088609 18.83097089  9.39386536 12.13137023 17.06984975
 20.35856822 20.71744625 20.13587073 13.89716702 19.58417107 14.49220512
 20.84303016 10.78204944  7.61614664 15.80804073 15.02082892  5.55821089
 16.88226665 15.22576091  8.42227541  8.09757199 11.50310765  8.44260674
 18.04940455  9.25708646 13.80913848 15.7802781 ]
```

☐ <u>Step-14</u> – Check the accuracy of model by r2 score.

Check accuracy score.

```
print(r2_score(y_test, predictions))
```

```
0.8718000770602493
```

☐ <u>Step-15</u> – Import the Second Machine Learning Model Polynomial Regression and train model and then make prediction.

Import the second machine learning model 'Polynomial regression'.

```
from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree=2, include_bias=True)
x_train_trans=poly.fit_transform(x_train)
x_test_trans=poly.transform(x_test)
linear=LinearRegression()
```

Train the Model

```
linear.fit(x_train_trans,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

Make predictions on model

```
y_predictions=linear.predict(x_test_trans)
```

☐ <u>Step-16</u> – Check the accuracy of model by r2 score.

Check Accuracy Score.

```
[ ]  print(r2_score(y_test,y_predictions))

     0.9314403710799105
```

☐ <u>Step-17</u> – Import the Third Machine Learning Model Support Vector Regressor and train model and then make prediction.

Import the third Machine Learning Model 'SVM regressor'.

```
[ ]  from sklearn.svm import SVR
     svr=SVR()
```

Train the Model.

```
[ ]  svr.fit(x_train,y_train)
     ▾ SVR
     SVR()
```

Make predictions on Model.

```
⊙  pred_svr=svr.predict(x_test)
   print(pred_svr)

�  [18.91524718 19.03795713 15.70225069 20.17643533 19.43567759 12.96051786
   17.6334425  12.03172711 18.77167786 14.41677104 14.81849883 19.00729881
   16.49120221 12.3286714  18.66950268  8.66966378 12.35991829 17.81155391
   19.52486664 19.66652726 19.82072458 14.73773859 18.52731034 13.48188886
   19.20688192 10.78510718  8.10000268 14.73713063 14.43131344  7.52355828
   17.04944771 14.90598098  9.37576039  8.32735203 10.204765    8.33843055
   18.51699395  8.62951333 14.7056213  17.055878  ]
```

☐ <u>Step-18</u> – Check the accuracy score.

Check the accuracy score of model.

```
⊙  print(r2_score(y_test, pred_svr))

   0.8963929828568145
```

Conclusion – In this project, we have demonstrated in detail how to apply linear regression, polynomial regression and support vector regressor model for predicting sales from data of spend cost for advertising. By carefully selecting the right variables, preparing and cleaning the data, and selecting an appropriate regression models, businesses can accurately predict sales from advertising cost ads.

project analysis resulted in a good R-squared value of 0.87100, 0.93144 and 0.89639 respectively, which indicates that the linear regression model has a decent fit for the data and gave 87%, 93% and 89% accuracy respectively. This level of accuracy can provide

businesses with valuable insights into the effectiveness of their advertising campaigns and enable them to make informed decisions about how to allocate their resources.

thank you