Muhumuza Stephen 00526 linearregression

March 25, 2024

[51]:
```
# importing the necesary libraries
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error,r2_score
import matplotlib.pyplot as plt
```

[52]:
```
# Loading the dataset
data=pd.read_csv('C:\\Users\\lynda\\Desktop\\company.csv')
data.head()
```

[52]:
```
   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
```

[53]:
```
#checking for null values
data.isna().sum()
```

[53]:
```
R&D Spend          0
Administration     0
Marketing Spend    0
State              0
Profit             0
dtype: int64
```

[54]:
```
# defining the x(independent valu)
x=data[['R&D Spend','Administration','Marketing Spend']]
x.head()
```

[54]:
```
   R&D Spend  Administration  Marketing Spend
0  165349.20       136897.80        471784.10
1  162597.70       151377.59        443898.53
2  153441.51       101145.55        407934.54
```

```
3  144372.41       118671.85        383199.62
4  142107.34        91391.77        366168.42
```

[55]:

```
# y(dependent values)
y=data['Profit']
y.head()
```

[55]:

```
0    192261.83
1    191792.06
2    191050.39
3    182901.99
4    166187.94
Name: Profit, dtype: float64
```

[56]:

```
# spliting my data into training and testing set
x_train,x_test,y_train,y_test = train_test_split(x,y, test_size=0.2,␣
 ↪random_state=42)
```

[57]:

```
# Building my Ordinary Linear regression
model = LinearRegression()
model.fit(x_train, y_train)
```

[57]:

```
LinearRegression()
```

[58]:

```
#calculating accuracy score
y_pred = model.predict(x_test)
y_pred
```

[58]:

```
array([126703.02716461,  84894.75081556,  98893.41815974,  46501.70815036,
       129128.39734381,  50992.69486261, 109016.5536578 , 100878.4641454 ,
        97700.59638629, 113106.15292226])
```

[59]:

```
r2=r2_score(y_test,y_pred)
r2
```

[59]:

```
0.900065308303732
```

[62]:

```
# Optimization of Linear Regression
model_opt=LinearRegression()
model_opt
```

[62]:

```
LinearRegression()
```

[87]:

```
#define parameters
params={
    'fit_intercept':[True,False],
    'copy_X':[True,False],
    'n_jobs':[True,False],
    'positive':[True,False],

}
```

```
[88]: #initializing the gridesearch
      grid_search=GridSearchCV(model_opt,params,cv=5)
      grid_search
```

```
[88]: GridSearchCV(cv=5, estimator=LinearRegression(),
                   param_grid={'copy_X': [True, False],
                               'fit_intercept': [True, False],
                               'n_jobs': [True, False], 'positive': [True, False]})
```

```
[89]: # training optimized model_
      grid_search.fit(x_train,y_train)
```

```
[89]: GridSearchCV(cv=5, estimator=LinearRegression(),
                   param_grid={'copy_X': [True, False],
                               'fit_intercept': [True, False],
                               'n_jobs': [True, False], 'positive': [True, False]})
```

```
[90]: #finding params
      best_params=grid_search.best_params_
      best_params
```

```
[90]: {'copy_X': True, 'fit_intercept': True, 'n_jobs': True, 'positive': True}
```

```
[91]: #finding the best model
      best_model=LinearRegression(**best_params)
      best_model
```

```
[91]: LinearRegression(n_jobs=True, positive=True)
```

```
[92]: #fitting model
      best_model.fit(x_train,y_train)
```

```
[92]: LinearRegression(n_jobs=True, positive=True)
```

```
[93]: #making predictions
      y_pred_opt=best_model.predict(x_test)
      y_pred_opt
```

```
[93]: array([127521.38604123,  82615.07411457,  97683.2462344 ,  46400.65677644,
             130782.53611917,  45967.0205249 , 109813.19061887, 101612.68921418,
              97023.64013854, 113241.36575804])
```

```
[94]: #evaluting my model
      r2=r2_score(y_test,y_pred_opt)
```

```
r2
```

```
[94]: 0.9168381183550245
```

```
[ ]:
```