PROFIT PREDICTION MODEL

DATA SCIENCE

SUBMITTED BY

DEEPAK KUMAR

NATIONAL INSTITUTE OF TECHNOLOGY SIKKIM

UNDER THE GUIDANCE OF EXPOSYS

DATA LABS

CONTENS

- INTRODUCTION TO MODEL
- LIBRARY IMPORTING
- DATA CLEANING
- Data Visualization
- Feature exploration
- Model development
- Model evaluation

INTRODUCTION TO PROJECT

- In this project , we will be predicting the profit from the startup's dataset with the features available to us. We're using the 50-startups dataset for this problem statement and we will be using the concept of Multiple linear regression to predict the profit of startups companies
- his particular dataset holds data from **50 startups** The features in this dataset are **R&D spending, Administration Spending, Marketing Spending, and location features,** while the target variable is: **Profit.**

. **R&D spending:** The amount which startups are spending on Research and development.

. **Administration spending:** The amount which startups are spending on the Admin panel.

. **Marketing spending:** The amount which startups are spending on marketing strategies.
. **Profit:** How much profit that particular startup is mak

IMPORTING LIBRARIES

- import numpy as np # for performing mathematical calculations behind ML algorithm
  import matplotlib.pyplot as plt # for visualization
- import pandas as pd # for handling and cleaning the dataset
- import seaborn as sns # for visualization
- import sklearn # for model evaluation and development
- dataset = pd.read_csv('50_Startups.csv')
- These all operations performed on vs code

- Analyzing the data
- dataset.head()

NUMERICAL/STATISTICAL ANALYSIS OF THE DATASET

code

dataset.describe()

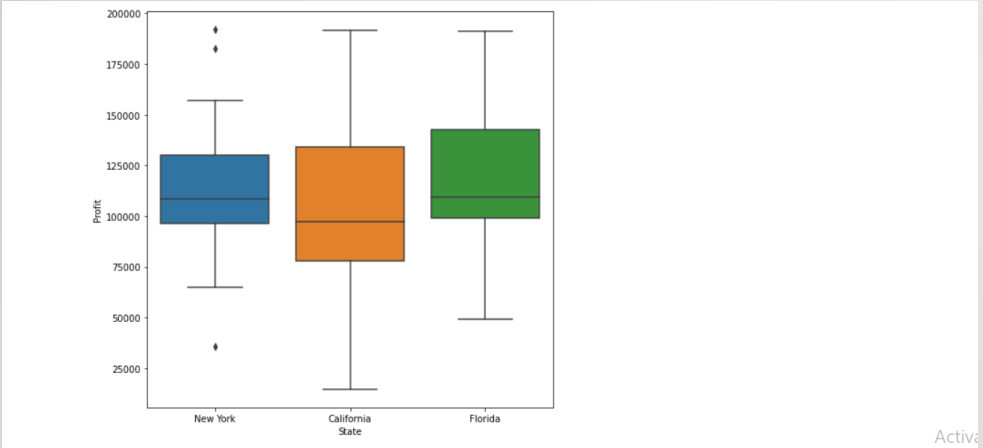
	R&D Spend	Administration	Marketing Spend	Profit
count	50.000000	50.000000	50.000000	50.000000
mean	73721.615600	121344.639600	211025.097800	112012.639200
std	45902.256482	28017.802755	122290.310726	40306.180338
min	0.000000	51283.140000	0.000000	14681.400000
25%	39936.370000	103730.875000	129300.132500	90138.902500
50%	73051.080000	122699.795000	212716.240000	107978.190000
75%	101602.800000	144842.180000	299469.085000	139765.977500
max	165349.200000	182645.560000	471784.100000	192261.830000

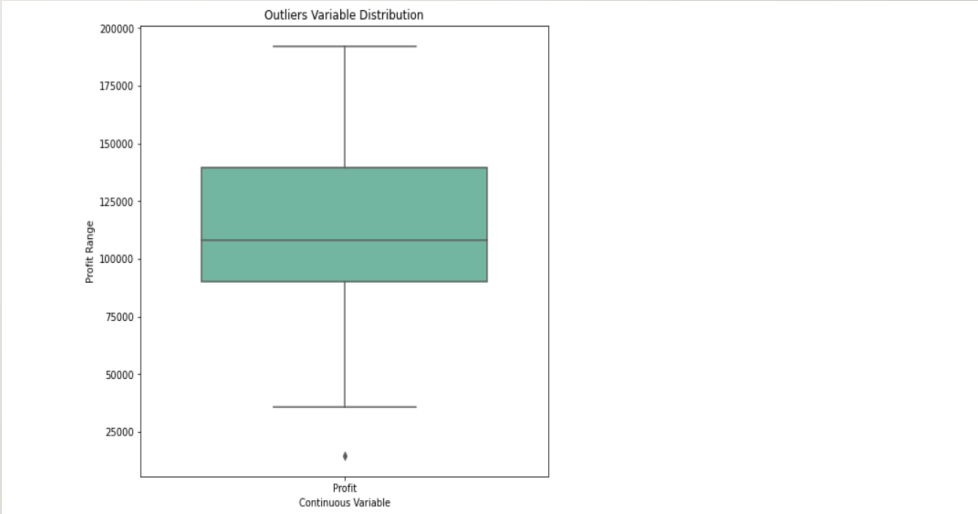
OUTLIERS DETECTION IN THE TARGET VARIABLE

- utliers = ['Profit']
- plt.rcParams['figure.figsize'] = [8,8]
- sns.boxplot(data=dataset[outliers], orient="v"
- palette="Set2" , width=0.7) # orient = "v" :
- vertical boxplot ,
- # orient = "h" :
- hotrizontal boxplot plt.title("Outliers Variable Distribution")
- plt.ylabel("Profit Range")
- plt.xlabel("Continuous Variable") plt.show()

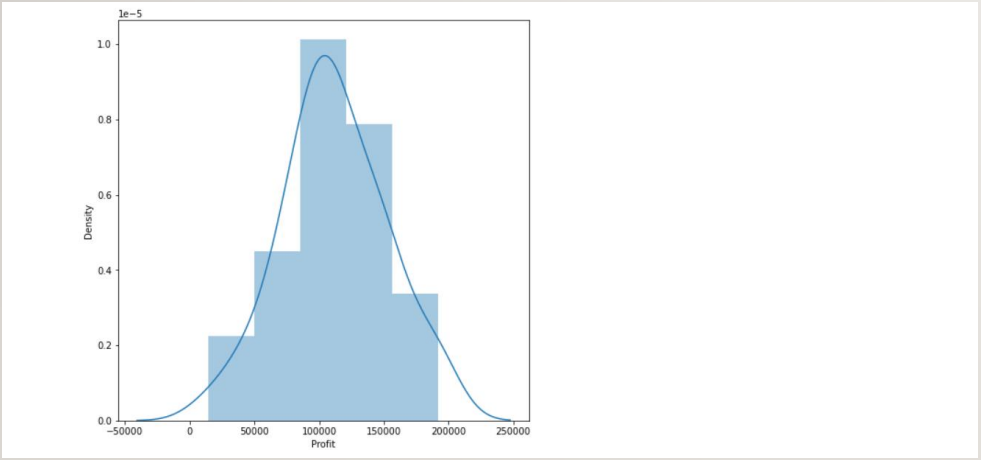
OUTPUT

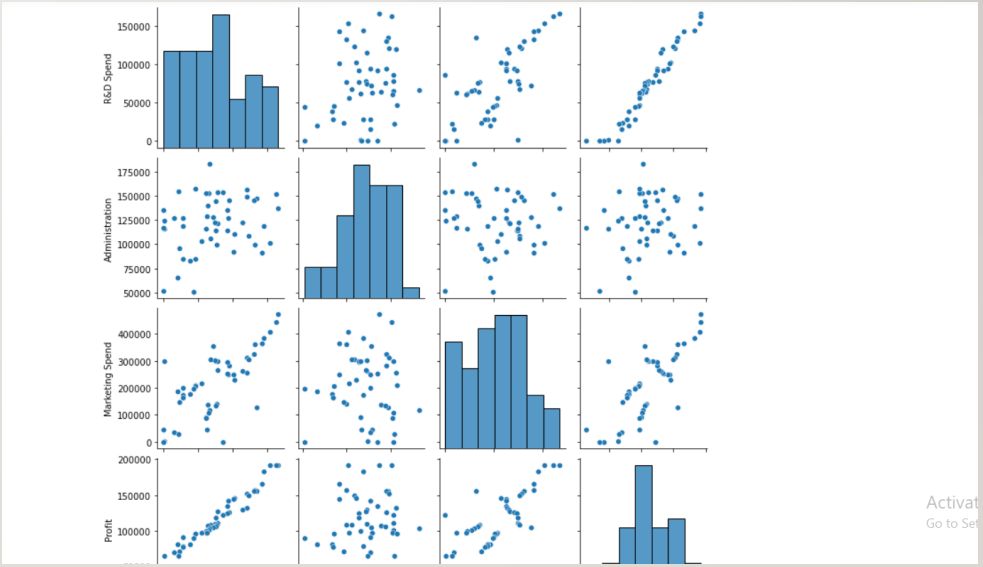
WHILE LOOKING AT THE **BOXPLOT** WE CAN SEE THE **OUTLIERS IN THE PROFIT(TARGET VARIABLE)**, BUT THE AMOUNT OF DATA IS NOT MUCH (JUST 50 ENTRIES) SO IT WON'T CREATE MUCH NEGATIVE IMPACT





HISTOGRAM ON PROFIT
SNS.DISTPLOT(DATASET['PROFIT'],BINS=5,KDE=TRUE) PLT.SHOW(





MODEL DEVELOPMENT

- # spliting Dataset in Dependent & Independent Variables
- X = dataset.iloc[:, :-1].values
- y = dataset.iloc[:, 4].values
- labelencoder = LabelEncoder()
- X[:, 3] = labelencoder.fit_transform(X[:, 3])
- X1 = pd.DataFrame(X) X1.head()

SPLIT THE DATA INTO TRAINING AND TESTING DATA

- 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,random_state=0) x_train
- train Output: array([[130298.13, 145530.06, 323876.68, 1], [119943.24, 156547.42, 256512.92, 1], [1000.23, 124153.04, 1903.93, 2], [542.05, 51743.15, 0.0, 2], [65605.48, 153032.06, 107138.38, 2], [114523.61, 122616.84, 261776.23, 2], [61994.48, 115641.28, 91131.24, 1], [63408.86, 129219.61, 46085.25, 0], [78013.11, 121597.55, 264346.06, 0], [23640.93, 96189.63, 148001.11, 0], [76253.86, 113867.3, 298664.47, 0], [15505.73, 127382.3, 35534.17, 2], [120542.52, 148718.95, 311613.29, 2], [91992.39, 135495.07, 252664.93, 0], [64664.71, 139553.16, 137962.62, 0], [131876.9, 99814.71, 362861.36, 2], [94657.16, 145077.58, 282574.31, 2], [28754.33, 118546.05, 172795.67, 0], [0.0, 116983.8, 45173.06, 0], [162597.7, 151377.59, 443898.53, 0], [93863.75, 127320.38, 249839.44, 1], [44069.95, 51283.14, 197029.42, 0], [77044.01, 99281.34, 140574.81, 2], [134615.46, 147198.87, 127716.82, 0], [67532.53, 105751.03, 304768.73, 1], [28663.76, 127056.21, 201126.82, 1], [78389.47, 153773.43, 299737.29, 2], [8641 9.7, 153514.11, 0.0, 2], [123334.88, 108679.17, 304981.62, 0], [38558.51, 82982.09, 174999.3, 0], [1315.46, 115816.21, 297114.46, 1], [144372.41, 118671.85, 383199.62, 2], [165349.2, 136897.8, 471784.1, 2], [0.0, 135426.92, 0.0, 0], [22177.74, 154806.14, 28334.72, 0]], dtype=object

TESTING THE MODEL USING THE PREDICT FUNCTION

- y_pred = model.predict(x_test) y_pred
- Testing scores testing_data_model_score = model.score(x_test, y_test) print("Model Score/Performance on Testing data",testing_data_model_score) training_data_model_score = model.score(x_train, y_train) print("Model Score/Performance on Training data",training_data_model_score)
- Output: Model Score/Performance on Testing data 0.9355139722149948 Model Score/Performance on Training data 0.9515496105627431

MODEL EVALUATION

- Model evaluation 1. R2 score: R2 score – R squared score.
- It is one of the statistical approaches by which we can find the variance or the spread of the target and feature data.
- from sklearn.metrics import r2_score r2Score = r2_score(y_pred, y_test) print("R2 score of model is :" ,r2Score*100)
- Output: R2 score of model is: 93.39448007716636
- MSE: MSE – Mean Squared Error. By using this approach we can find that how much the regression best fit line is close to all the residual. from sklearn.metrics import mea

VALIDATION

- Root Mean Squared Error is: 788954.7666974603
- MAE: MAE – Mean Absolute Error. By using this approach we can find the difference between the actual values and predicted values but that difference is absolute i.e. the difference is positive.
- from sklearn.metrics import mean_absolute_error mae = mean_absolute_error(y_pred,y_test) print("Mean Absolute Error is :" ,mae)
- Output: Mean Absolute Error is: 6503.577323580025 Conclusion So, the mean absolute error is 6503.577323580025. Therefore our predicted value can be 6503.577323580025 units more or less than the actual value.

CONCLUSIONS AND RESULT

- predicted value is close to the actual values i.e the one present in the testing set, Hence we can use this model for prediction.
- The marketing spend seems to be directly proportional (though a little bit outliers are there) with the profit
- . Therefore our predicted value can be **6503.577323580025**
- his machine learning model will be quite helpful in such a situation where we need to find a profit , based on the amount which we spend from the 50 startups dataset