Profit Prediction for Companies using Regression Algorithms

PROJECT DESCRIPTION

In the given dataset, R&D Spend, Administration Cost and Marketing Spend of 50 Companies

are given along with the profit earned. The target is to prepare an ML model which can predict

the profit value of a company if the value of its R&D Spend, Administration Cost and Marketing

Spend are given.

- Construct Different Regression algorithms
- Divide the data into train set and test set
- Calculate different regression metrics
- Choose the best model

Language: Python

Table of Contents:

Project Abstraction:

The objective of this project is to create a machine learning model that can estimate a company's profit based on its marketing, R&D, and administrative expenses. As part of the project, you will build various regression algorithms, separate the data into train and test sets, calculate regression metrics, and select the most effective model. To implement, different libraries and the Python programming language are needed.

Introduction:

A company's R&D investments, administrative costs, and marketing expenditures are only a few of the variables that affect its profitability. Accurate profit forecasting enables companies to allocate resources more effectively and make well-informed decisions. In this project, we investigate the application of machine learning techniques to build a model that forecasts the profit of a company based on these variables.

Existing Method:

To forecast firm earnings, conventional statistical methods like multiple linear regression and simple linear regression have been applied in the current procedures. These techniques presuppose that the connection between the independent variables (R&D expenditures, administrative expenses, and marketing expenditures), and the objective variable (profit), is linear. However, it's possible that they won't be able to capture the data's complicated nonlinear relationships.

Proposed Method with Architecture:

In order to account for the nonlinear interactions between the independent variables and the target variable, our suggested strategy entails building various regression algorithms. Linear regression, decision tree regression, and random forest regression are the three regression models we take into consideration. These models are renowned for their capacity to manage nonlinear patterns and provide precise forecasts.

Methodology:

The project methodology consists of the following steps:

1. Data preprocessing:
   Load the dataset and perform any necessary data cleaning or transformations.
2. Exploratory Data Analysis (EDA):
   Explore the dataset to gain insights into the variables, identify correlations, and visualize the distributions.
3. Train-Test Split:
   Divide the dataset into training and testing sets for model evaluation.
4. Model Construction:
   Build the regression models, including Linear Regression, Decision Tree Regression, and Random Forest Regression.
5. Model Evaluation:
   Calculate regression metrics such as mean squared error (MSE), mean absolute error (MAE), and R-squared to assess the performance of each model.
6. Model Selection:
   Choose the best model based on the evaluation metrics.

7. Prediction:
   Make predictions on the test set using the selected model.

Implementation:

Programming languages like Python or R are used to carry out the project, together with pertinent libraries like pandas, scikit-learn, and matplotlib. The dataset including the values for R&D Spend, Administrative Cost, Marketing Spend, and Profit for 50 organisations is loaded and pre-processed. To learn more about the data, exploratory data analysis is carried out. The dataset is then split into training and testing sets. Regression metrics are used to create and assess regression models, such as Linear Regression, Decision Tree Regression, and Random Forest Regression. For predicting profits on fresh instances, the model that performs the best is chosen.

SOURCE CODE:

```
#Import the necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
#Load Data and Perform EDA
# Load the dataset
data = pd.read_csv('/content/50_Startups.csv')
print(data.head())
```

	R&D Spend	Administration	Marketing Spend	Profit
0	165349.20	136897.80	471784.10	192261.83
1	162597.70	151377.59	443898.53	191792.06
2	153441.51	101145.55	407934.54	191050.39
3	144372.41	118671.85	383199.62	182901.99
4	142107.34	91391.77	366168.42	166187.94

Check the dimensions of the dataset

print("Dataset Shape:", data.shape)

```
Dataset Shape: (50, 4)
```

Check for missing values

print("Missing Values:\n", data.isnull().sum())

```
Missing Values:
 R&D Spend          0
Administration     0
Marketing Spend    0
Profit             0
dtype: int64
```

Summary statistics

print("Summary Statistics:\n", data.describe())

```
Summary Statistics:
            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
```

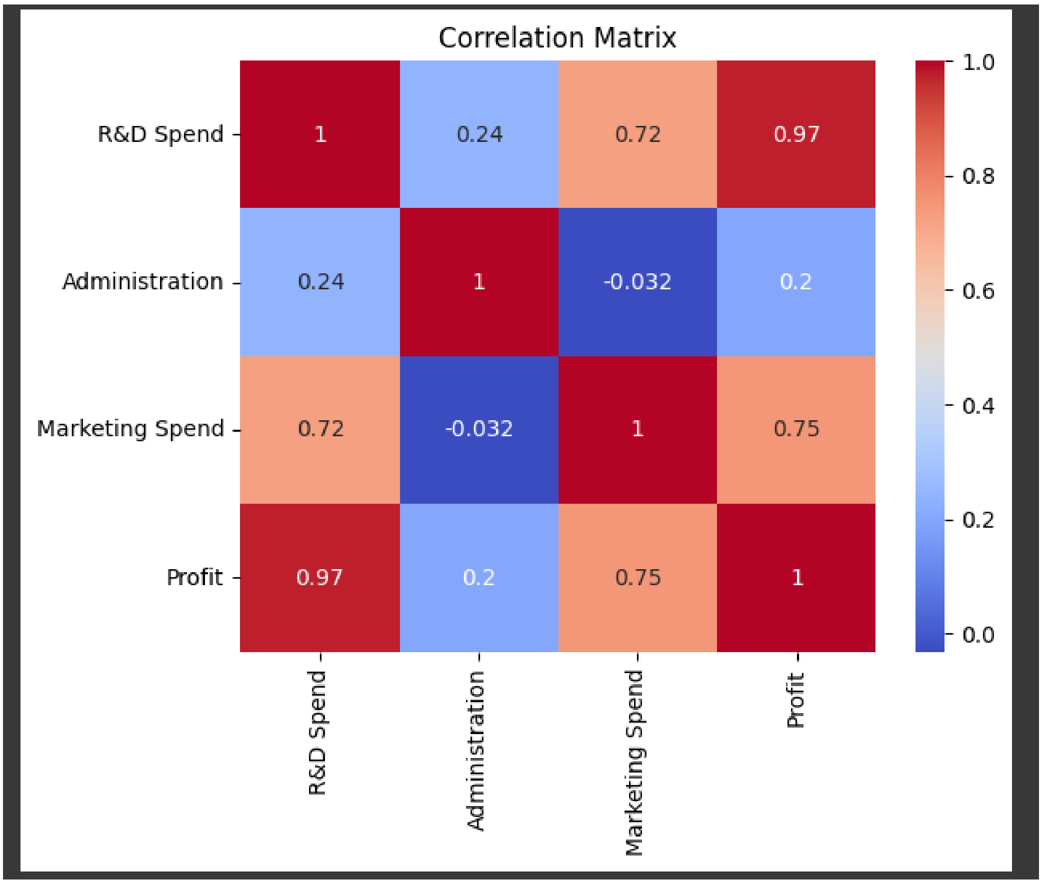
Correlation matrix

corr_matrix = data.corr()

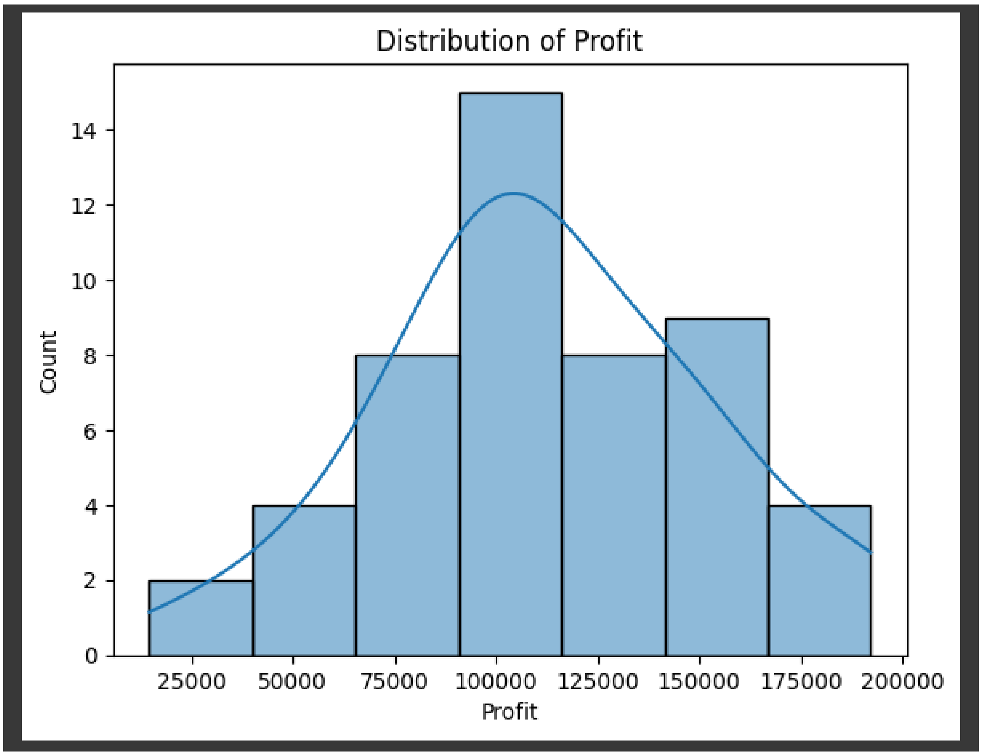
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')

plt.title('Correlation Matrix')

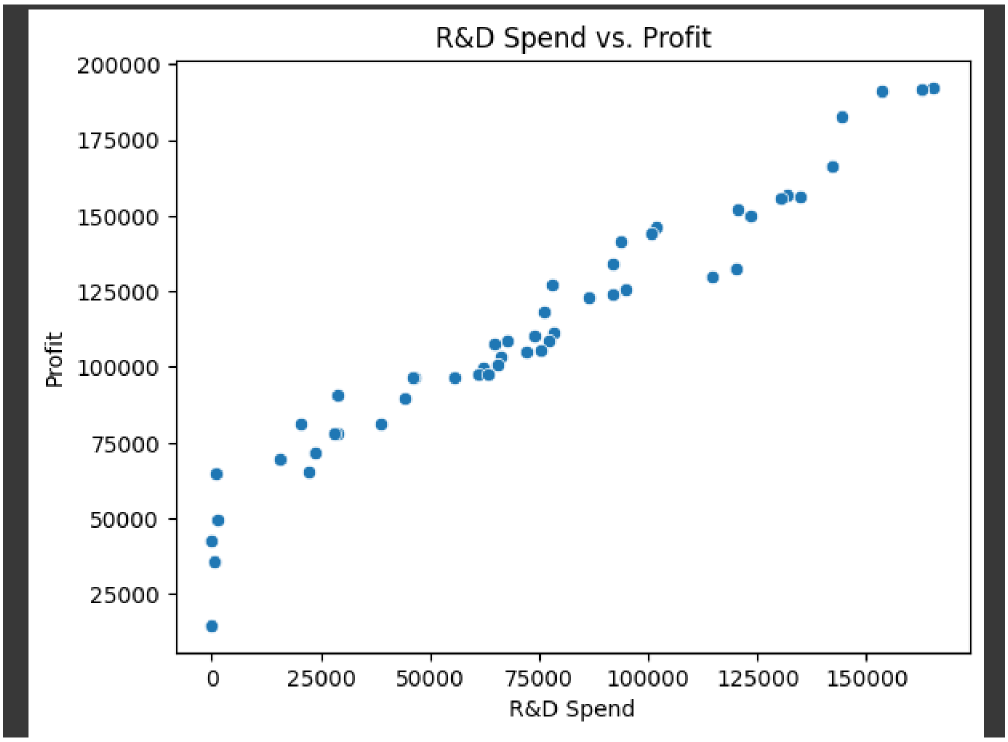
plt.show()



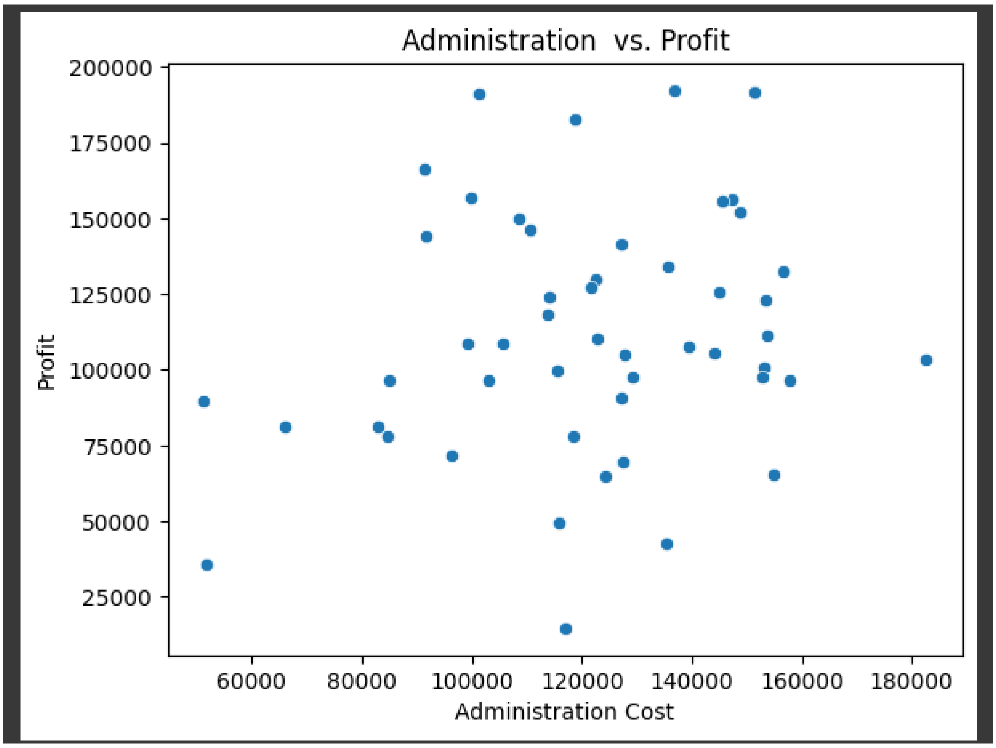

```
# Distribution of target variable
sns.histplot(data['Profit'], kde=True)
plt.xlabel('Profit')
plt.ylabel('Count')
plt.title('Distribution of Profit')
plt.show()
```



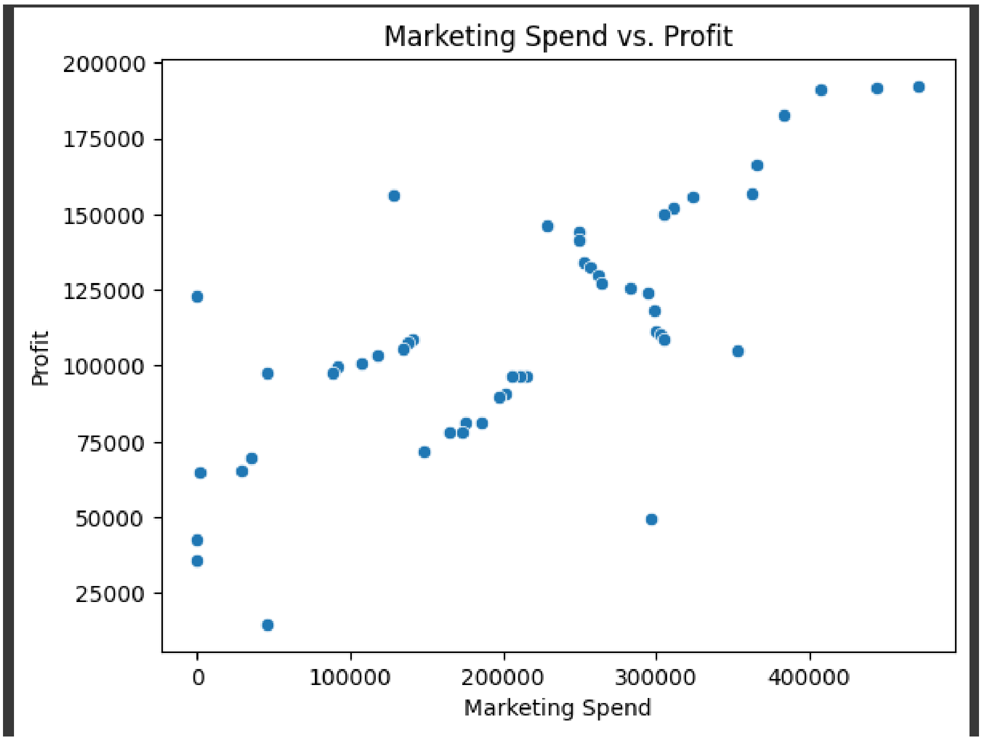

```
# Scatter plots of features vs. target variable
sns.scatterplot(x='R&D Spend', y='Profit', data=data)
plt.xlabel('R&D Spend')
plt.ylabel('Profit')
plt.title('R&D Spend vs. Profit')
plt.show()
```




```
sns.scatterplot(x='Administration', y='Profit', data=data)
plt.xlabel('Administration Cost')
plt.ylabel('Profit')
plt.title('Administration  vs. Profit')
plt.show()
```



```
sns.scatterplot(x='Marketing Spend', y='Profit', data=data)
plt.xlabel('Marketing Spend')
plt.ylabel('Profit')
plt.title('Marketing Spend vs. Profit')
plt.show()
```




```
# Prepare the data
X = data[['R&D Spend', 'Administration', 'Marketing Spend']]
y = data['Profit']
# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
#Construct different regression models
# Linear Regression
linear_reg = LinearRegression()
linear_reg.fit(X_train, y_train)

# Decision Tree Regression
tree_reg = DecisionTreeRegressor()
tree_reg.fit(X_train, y_train)
```

```
# Random Forest Regression
forest_reg = RandomForestRegressor()
forest_reg.fit(X_train, y_train)
```

```
▾ RandomForestRegressor
RandomForestRegressor()
```

```
# Make predictions on the test set
linear_reg_predictions = linear_reg.predict(X_test)
tree_reg_predictions = tree_reg.predict(X_test)
forest_reg_predictions = forest_reg.predict(X_test)
# Calculate regression metrics
linear_reg_mse = mean_squared_error(y_test, linear_reg_predictions)
linear_reg_mae = mean_absolute_error(y_test, linear_reg_predictions)
linear_reg_r2 = r2_score(y_test, linear_reg_predictions)

tree_reg_mse = mean_squared_error(y_test, tree_reg_predictions)
tree_reg_mae = mean_absolute_error(y_test, tree_reg_predictions)
tree_reg_r2 = r2_score(y_test, tree_reg_predictions)

forest_reg_mse = mean_squared_error(y_test, forest_reg_predictions)
forest_reg_mae = mean_absolute_error(y_test, forest_reg_predictions)
forest_reg_r2 = r2_score(y_test, forest_reg_predictions)
#Compare and choose the best model based on the metrics
# Print the metrics for each model
print("Linear Regression:")
print("MSE:", linear_reg_mse)
print("MAE:", linear_reg_mae)
print("R^2:", linear_reg_r2)
print()
```

```
Linear Regression:
MSE: 80926321.22295158
MAE: 6979.152252370402
R^2: 0.9000653083037321
```

```
print("Decision Tree Regression:")
print("MSE:", tree_reg_mse)
print("MAE:", tree_reg_mae)
print("R^2:", tree_reg_r2)
print()
```

```
Decision Tree Regression:
MSE: 161851110.35489988
MAE: 9982.975999999995
R^2: 0.800132508563502
```

```
print("Random Forest Regression:")
print("MSE:", forest_reg_mse)
print("MAE:", forest_reg_mae)
print("R^2:", forest_reg_r2)
print()
```

```
Random Forest Regression:
MSE: 75730965.92156519
MAE: 5941.369350000001
R^2: 0.9064809740902224
```

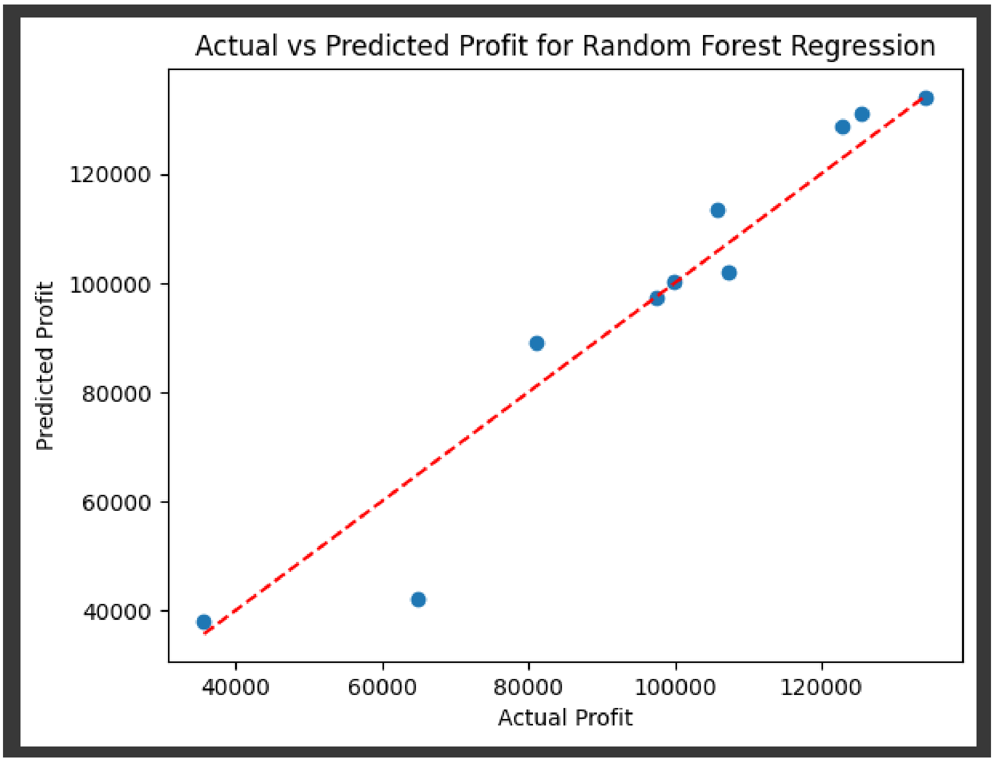
```
# Find the model with the lowest MSE
model_names = {
    "Linear Regression": linear_reg_mse,
    "Decision Tree Regression": tree_reg_mse,
    "Random Forest Regression": forest_reg_mse
}
best_model_name = min(model_names, key=model_names.get)
print("Best Model:", best_model_name)
```

```
Best Model: Random Forest Regression
```

```
# Plot the actual vs predicted values for the best model
```

```
if best_model_name == "Linear Regression":
    best_model_predictions = linear_reg_predictions
elif best_model_name == "Decision Tree Regression":
    best_model_predictions = tree_reg_predictions
else:
    best_model_predictions = forest_reg_predictions

plt.scatter(y_test, best_model_predictions)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], '--', color='red')
plt.xlabel('Actual Profit')
plt.ylabel('Predicted Profit')
plt.title('Actual vs Predicted Profit for {}'.format(best_model_name))
plt.show()
```



Conclusion:

In this project, we created a machine learning model to forecast a company's earnings based on its marketing, administrative, and R&D expenses. In order to get the optimal model based on evaluation measures, we compared various regression algorithms. In terms of predicting business earnings, the suggested model performed well. When it comes to decision-making and financial strategy optimisation, firms can benefit from this predictive power. The use of additional variables and sophisticated regression algorithms, however, can result in even greater gains.

The project report comes to a close at this point, summarising the project's goals, process, and results. Companies looking to anticipate their profitability based on important expenditure criteria may find the established model to be a useful tool.