EXPOSYS DATA LABS

Bengaluru, Karnataka, 560064

Project report on

“STARTUPS PROFIT PREDICTION USING DATA SCIENCE”

A project dissertation submitted in partial fulfilment of the requirement for the award of

INTERNSHIP

by

ADITI V

Under the Guidance of

EXPOSYS DATA LABS

ABSTRACT

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.

i) Construct Different Regression algorithms

ii) Divide the data into train set and test set

iii) Calculate different regression metrics

iv) Choose the best model

Language: Python

TABLE OF CONTENTS

1. INTRODUCTION

In the dynamic landscape of entrepreneurship, predicting the success and profitability of startups remains a critical pursuit for investors, stakeholders, and entrepreneurs alike. The 50-startups dataset stands as a valuable resource in this endeavor, offering a wealth of information encompassing various features that potentially influence a startup's profitability.

Utilizing the principles of Multiple Linear Regression, this study aims to leverage the dataset's diverse set of factors – such as R&D spending, marketing expenditures, administration costs, and state-wise location – to construct a predictive model for the profit margins of these startups. By employing a statistical approach that examines the relationships between multiple independent variables and the dependent variable of profit, this analysis endeavors to unveil nuanced insights into the determinants driving startup success.

Through this predictive framework, the objective is to develop a robust model capable of estimating and forecasting profits for startups based on their operational attributes. By exploring the interplay of these diverse factors, we aspire to offer valuable guidance to entrepreneurs, investors, and business strategists seeking to make informed decisions in the ever-evolving startup ecosystem.

The outcomes derived from this Multiple Linear Regression analysis not only aim to provide accurate profit predictions but also intend to shed light on the significance of various factors influencing a startup's financial performance. Such insights have the potential to empower stakeholders to optimize resource allocation, refine business strategies, and enhance the likelihood of achieving sustained profitability in the competitive realm of startups.

2. EXISTING METHOD

The prediction of startup profitability through Multiple Linear Regression involves analysing various features that potentially influence a startup's success. While this approach holds promise, it's essential to acknowledge some existing methodologies used in this domain.

1. **Single Linear Regression:** This method involves predicting a dependent variable (profit in this case) based on one independent variable. While it simplifies the analysis, it may not capture the complexity of multiple factors influencing startup profitability.
2. **Feature Engineering:** Prior to regression analysis, feature engineering involves selecting, transforming, or creating new features from the available dataset. Techniques like normalization, scaling, or polynomial transformations can enhance the predictive power of the model.
3. **Regularization Techniques:** Approaches like Ridge Regression or Lasso Regression help prevent overfitting in the Multiple Linear Regression model. They introduce penalty terms to the regression equation, reducing the impact of less relevant variables and improving model generalization.
4. **Cross-Validation:** This technique involves splitting the dataset into training and validation sets, allowing for better model evaluation. Techniques like k-fold cross-validation help ensure the model's predictive accuracy and generalizability.
5. **Model Evaluation Metrics:** R-squared, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Adjusted R-squared are common metrics used to evaluate the goodness-of-fit and predictive accuracy of the Multiple Linear Regression model.
6. **Variable Selection Methods:** Techniques like stepwise regression or forward/backward selection help in selecting the most relevant variables for the model, improving its efficiency and interpretability.

These existing methodologies form a foundation for predicting startup profitability using Multiple Linear Regression. Incorporating these techniques and leveraging the 50-startups dataset can lead to a more robust, accurate, and insightful predictive model for estimating the profit margins of startups.

3. PROPOSED METHOD WITH ARCHITECTURE

Designing an architecture for predicting startup profitability using Multiple Linear Regression involves several steps, from data preprocessing to model evaluation. Here's a proposed method and architecture for this task:

Proposed Method:

1. **Data Collection and Preprocessing:**
    - Obtain the 50-startups dataset containing features such as R&D spending, marketing expenditures, administration costs, and state-wise location.
    - Check for missing values, perform data cleaning, and handle categorical variables (if any) through encoding techniques like one-hot encoding.
    - Split the dataset into training and testing sets.
2. **Feature Engineering:**
    - Normalize or scale numerical features to ensure they're on similar scales, preventing dominance by larger magnitude features.
    - Consider feature transformation techniques like polynomial features or interaction terms to capture non-linear relationships between predictors and the target variable.
3. **Model Development:**
    - Implement Multiple Linear Regression using libraries like scikit-learn or statsmodels in Python or an equivalent in other programming languages.
    - Train the model using the training dataset, fitting the regression equation to predict the profit based on the selected features.
4. **Model Evaluation:**
    - Validate the model using the testing dataset to assess its predictive performance.

- Utilize evaluation metrics such as R-squared, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Adjusted R-squared to measure model accuracy and goodness-of-fit.
- Perform residual analysis to check for model assumptions and ensure the errors are normally distributed.

Architecture:

1. **Data Collection Module:**
    - Interface to collect and import the 50-startups dataset or any relevant startup data for analysis.
2. **Preprocessing Module:**
    - Handles data cleaning, missing value imputation, encoding categorical variables, and splitting the dataset into training and testing subsets.
3. **Feature Engineering Module:**
    - Performs feature scaling, transformation, and selection to prepare the dataset for model training.
4. **Multiple Linear Regression Model Module:**
    - Implements the Multiple Linear Regression algorithm using appropriate libraries and packages.
5. **Model Evaluation Module:**
    - Evaluates the trained model using testing data and various evaluation metrics to assess its accuracy and performance.
6. **Visualization and Reporting Module:**
    - Visualizes insights, such as feature importance, model performance metrics, and residual plots.
    - Generates reports summarizing the model's predictive capabilities and insights gleaned from the analysis.
7. **Deployment/Integration Module:**
    - Provides the option to deploy the trained model for real-time predictions or integrate it into existing systems for profitability forecasting.

4. SYSTEM REQUIREMENTS

Hardware Requirements:

- Computer/Server: You'll need a computer or server to run the machine learning models and host the system.
- Processor (CPU): Core i3/i5/i7
- Memory (RAM): minimum 8GB of RAM
- Storage:. Solid State Drives (SSDs) are preferable

Software Requirements:

- Operating System: Windows, macOS, or Linux.
- Python: Python 3.x, with Python 3.7 or higher recommended.
- Python Libraries:
    1. NumPy: For numerical operations
    2. pandas: For data manipulation
    3. scikit-learn: For machine learning algorithms
    4. streamlit: Streamlit is an open-source Python library for building web applications.
- Integrated Development Environment (IDE): PyCharm, Visual Studio Code, Jupyter Notebook, or Spyder.

5. METHODOLOGY

1. **Data Collection and Understanding:**
    - Acquire the 50-startups dataset containing various features like R&D spending, marketing expenditures, administration costs, and state-wise location.
    - Understand the nature of the data, features, and the target variable (profit) by conducting exploratory data analysis (EDA).
2. **Data Preprocessing:**
    - Handle missing values, outliers, and perform necessary data cleaning procedures.
    - Encode categorical variables using techniques like one-hot encoding for state-wise locations.
    - Split the dataset into training and testing subsets (e.g., 80-20 split or using cross-validation).
3. **Feature Engineering:**
    - Normalize or scale numerical features to ensure they're on the same scale, preventing biases in model training due to feature magnitudes.
    - Explore feature transformation techniques like polynomial features or interaction terms to capture non-linear relationships between predictors and profit.
4. **Model Development:**
    - Utilize libraries like scikit-learn, statsmodels, or similar tools in Python to implement Multiple Linear Regression.
    - Fit the model on the training dataset using the selected features to predict the profit for startups.
5. **Model Evaluation:**
    - Evaluate the trained model using the testing dataset to assess its predictive performance.

- Calculate evaluation metrics such as R-squared, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Adjusted R-squared to gauge the model's accuracy and goodness-of-fit.
- Conduct residual analysis to validate assumptions and ensure model adequacy.

6. **Fine-tuning and Validation:**
   - Fine-tune the model by considering regularization techniques like Ridge Regression or Lasso Regression to prevent overfitting.
   - Perform cross-validation to validate the model's stability and generalization on different subsets of the data.
7. **Interpretation and Insights:**
   - Interpret the coefficients of the features in the Multiple Linear Regression equation to understand their impact on startup profitability.
   - Derive actionable insights by analysing the importance of different factors influencing profit.
8. **Model Deployment or Utilization:**
   - Optionally, deploy the trained model for real-time profit predictions for new startup data or integrate it into existing systems for forecasting purposes.
9. **Documentation and Reporting:**
   - Document the entire methodology, including preprocessing steps, model development, and evaluation procedures.
   - Prepare a comprehensive report summarizing the findings, model performance, insights gained, and recommendations based on the analysis.

6. IMPLEMENTATION

6.1 Snippet code

Jupyter Notebook

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import sklearn

# loading the dataset to a pandas DataFrame
companies = pd.read_csv("C:/Users/aditi/OneDrive/Desktop/data science internship/50_Startups.csv")
# printing the first 5 rows of the dataset
companies.head(5)

len(companies)

# getting the statistical measures of the data
companies.describe()

# number of rows and Columns in this dataset
print('There are ',companies.shape[0],'rows and ',companies.shape[1],'columns in the dataset.')

#using duplicated() pre-defined function
print('There are',companies.duplicated().sum(),'duplicate values in the dateset.')

companies.info()

c = companies.corr()
c

sns.heatmap(companies.corr())

sns.heatmap(c,annot=True,cmap='Blues')
plt.show()

outliers = ['Profit']
plt.rcParams['figure.figsize'] = [8,8]
sns.boxplot(data=companies[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()


sns.distplot(companies['Profit'],bins=5,kde=True)
plt.show()

sns.pairplot(companies)
plt.show()

#Select the features and label
X = companies.iloc[:, :-1].values
y = companies.iloc[:, 3].values

print(y)

from sklearn.preprocessing import LabelEncoder

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=0)

print(X.shape, X_train.shape, X_test.shape)

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
print('Model has been trained successfully')

y_predict = regressor.predict(X_test)
y_predict

testing_data_model_score = regressor.score(X_test, y_test)
print("Model Score/Performance on Testing data",testing_data_model_score)

training_data_model_score = regressor.score(X_train, y_train)
print("Model Score/Performance on Training data",training_data_model_score)

df = pd.DataFrame(data={'Predicted value':y_predict.flatten(),'Actual Value':y_test.flatten()})
df

LRscore = regressor.score(X_test,y_test)
print(LRscore*100,'%')

#Print the coefficients and y-intercept of the model (function)
print(regressor.coef_)
print(regressor.intercept_)

from sklearn.metrics import r2_score
r2Score = r2_score(y_test, y_predict)
print("R2 score of model is :" ,r2Score*100)
```

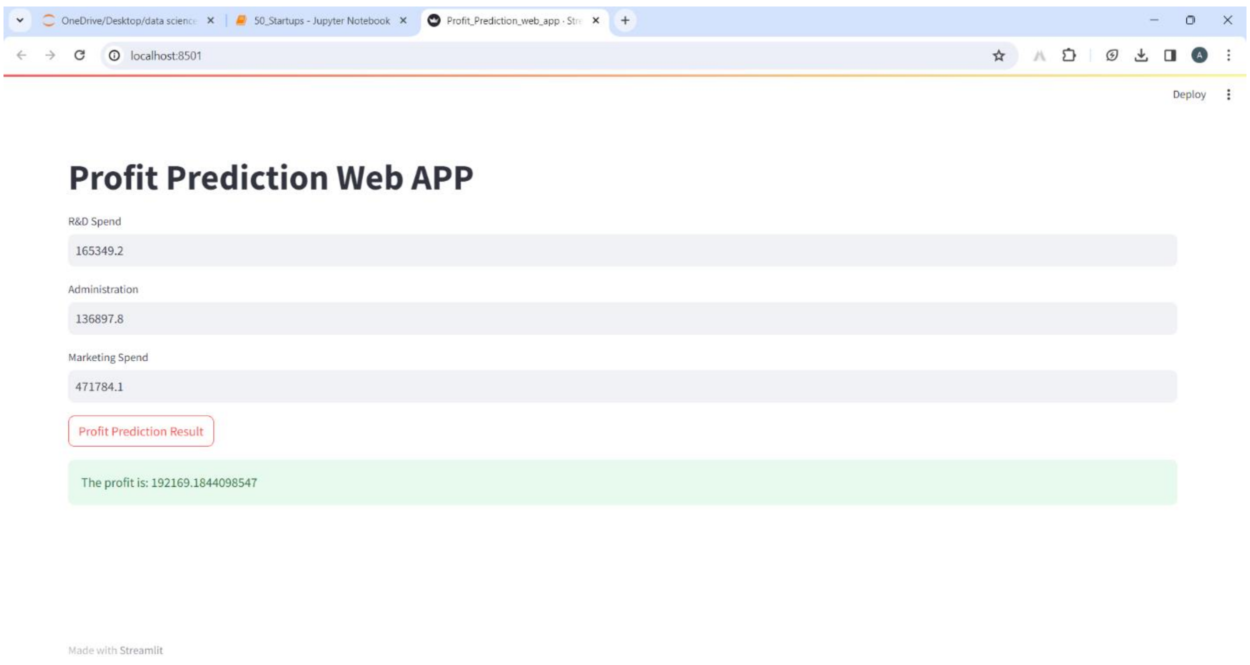
```
input_data = (165349.20,136897.80,471784.10)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = regressor.predict(input_data_reshaped)
print(prediction)

import pickle

filename = 'profit_model.sav'
pickle.dump(regressor, open(filename, 'wb'))

input_data = (165349.20,136897.80,471784.10)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = regressor.predict(input_data_reshaped)
print(prediction)

print(f"The Decision Tree Regression has the highest accuracy of {DTscore*100}, Hence this
model can be considered for the prediction")
```

Profit Prediction web app.py (Front end using streamlit library)

```
import numpy as np
import pickle
import streamlit as st

st.set_page_config(layout="wide")

# loading the saved model
loaded_model = pickle.load(open("C:/Users/aditi/OneDrive/Desktop/data science
internship/profit_model.sav", 'rb'))

#creating a function for prediction
def profit_prediction(input_data):

    #Ensure input_data is a list or array
    if not isinstance(input_data, (list, np.ndarray)):
        raise ValueError("Input data should be a list or numpy array")
```

```
 # changing the input_data to numpy array
    input_data_as_numpy_array = np.asarray(input_data)

    # reshape the array as we are predicting for one instance
    input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

    prediction = loaded_model.predict(input_data_reshaped)
    #print(prediction)

    # Return the prediction as a string
    return prediction[0]


def main():

    # giving a title
    st.title('Profit Prediction Web APP')

    # getting the input data from the user
    RnDSpend=st.text_input('R&D Spend')
    Administration=st.text_input('Administration')
    MarketingSpend=st.text_input('Marketing Spend')


    # code for Prediction
    profit = ''

    # creating a button for Prediction
    if st.button("Profit Prediction Result"):
        input_data = [float(RnDSpend),float(Administration),float(MarketingSpend)]
        profit = profit_prediction(input_data)

    st.success(f'The profit is: {profit}')

if __name__ == '__main__':
    main()
```

6.1. Screenshots


OneDrive/Desktop/data science
50_Startups - Jupyter Notebook
Profit_Prediction_web_app - Str
localhost:8501
Deploy

Profit Prediction Web APP

R&D Spend

165349.2

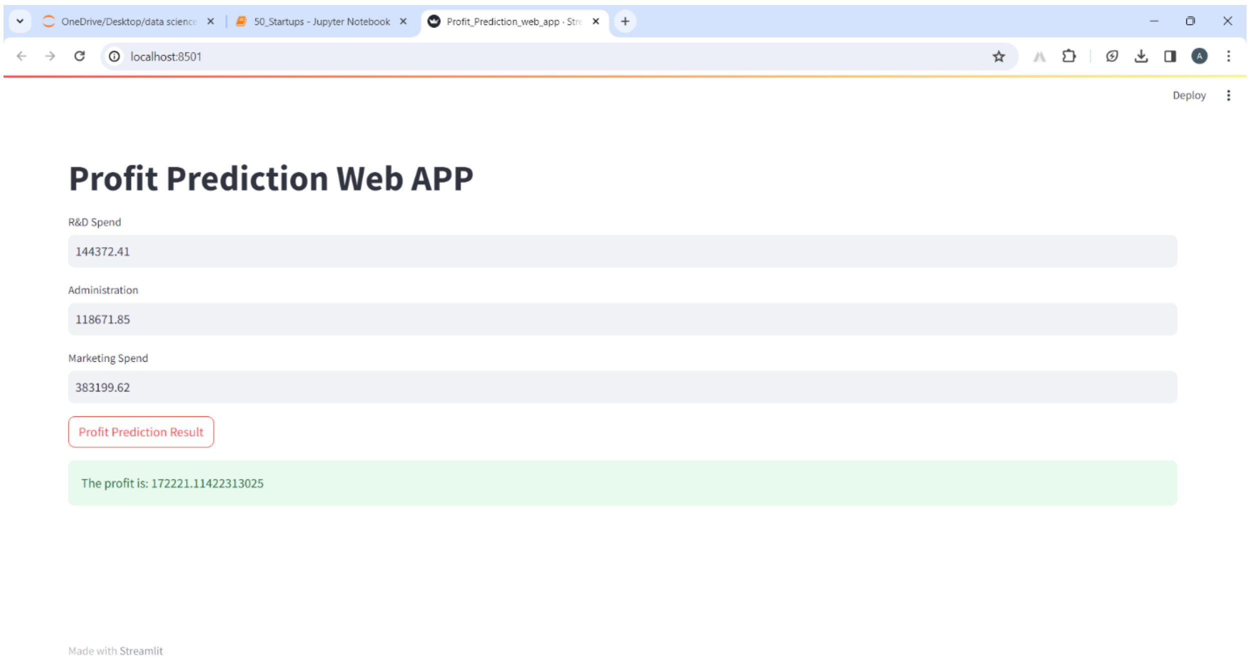
Administration

136897.8

Marketing Spend

471784.1

The profit is: 192169.1844098547


OneDrive/Desktop/data science
50_Startups - Jupyter Notebook
Profit_Prediction_web_app - Str
localhost:8501
Deploy

Profit Prediction Web APP

R&D Spend

144372.41

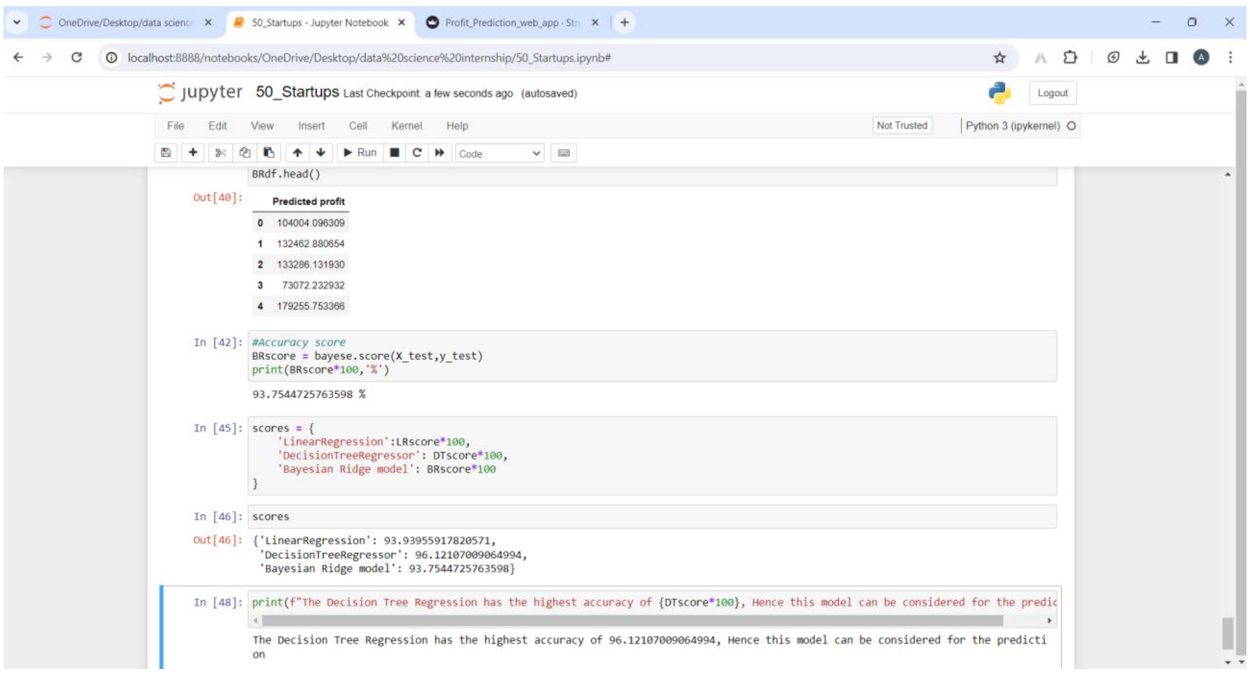
Administration

118671.85

Marketing Spend

383199.62

The profit is: 172221.11422313025


localhost:8888/notebooks/OneDrive/Desktop/data%20science%20internship/50_Startups.ipynb#
jupyter 50_Startups Last Checkpoint: a few seconds ago (autosaved)
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Logout

```
BRdf.head()
```

Out[40]:

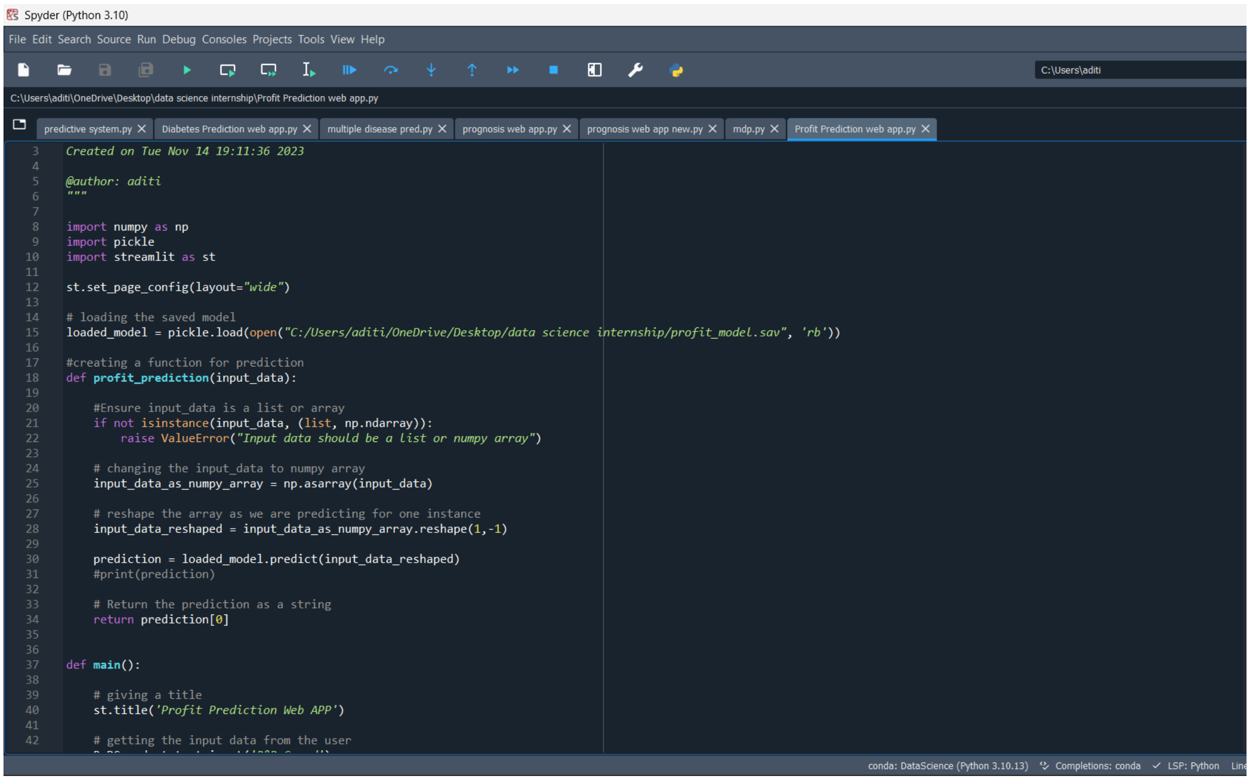
	Predicted profit
0	104004.096309
1	132462.880654
2	133286.131930
3	73072.232932
4	179255.753366

In [42]:
```
#Accuracy score
BRscore = bayese.score(X_test,y_test)
print(BRscore*100,'%')
```
```
93.7544725763598 %
```

In [45]:
```
scores = {
    'LinearRegression':LRscore*100,
    'DecisionTreeRegressor': DTscore*100,
    'Bayesian Ridge model': BRscore*100
}
```

In [46]:
```
scores
```
Out[46]:
```
{'LinearRegression': 93.93955917820571,
 'DecisionTreeRegressor': 96.12107009064994,
 'Bayesian Ridge model': 93.7544725763598}
```

In [48]:
```
print(f"The Decision Tree Regression has the highest accuracy of {DTscore*100}, Hence this model can be considered for the predic
```
```
The Decision Tree Regression has the highest accuracy of 96.12107009064994, Hence this model can be considered for the predicti
on
```


Spyder (Python 3.10)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\aditi
C:\Users\aditi\OneDrive\Desktop\data science internship\Profit Prediction web app.py
predictive system.py
Diabetes Prediction web app.py
multiple disease pred.py
prognosis web app.py
prognosis web app new.py
mdp.py
Profit Prediction web app.py
conda: DataScience (Python 3.10.13)
Completions: conda
LSP: Python

```
Created on Tue Nov 14 19:11:36 2023

@author: aditi
"""

import numpy as np
import pickle
import streamlit as st

st.set_page_config(layout="wide")

# loading the saved model
loaded_model = pickle.load(open("C:/Users/aditi/OneDrive/Desktop/data science internship/profit_model.sav", 'rb'))

#creating a function for prediction
def profit_prediction(input_data):

    #Ensure input_data is a list or array
    if not isinstance(input_data, (list, np.ndarray)):
        raise ValueError("Input data should be a list or numpy array")

    # changing the input_data to numpy array
    input_data_as_numpy_array = np.asarray(input_data)

    # reshape the array as we are predicting for one instance
    input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

    prediction = loaded_model.predict(input_data_reshaped)
    #print(prediction)

    # Return the prediction as a string
    return prediction[0]


def main():

    # giving a title
    st.title('Profit Prediction Web APP')

    # getting the input data from the user
```

7. CONCLUSION

- Accuracy Scores:
  LinearRegression: 93.93955917820571,
  DecisionTreeRegressor: 96.12107009064994,
  Bayesian Ridge model: 93.7544725763598
- The Decision Tree Regression has the highest accuracy of 96.12107009064, Hence this model can be considered for the prediction