ABDELMALEK ESSAADI University

Faculty of Legal, Economic and Social Sciences

Machine Learning Project Report

Create and Deploy Machine Learning Model using Flask Docker and Azure

Realized by : M'SAFRI Kaouthar
Supervised by : Prof. TALI Abdelhak

2023/04/27

Problem Statement

In this project, we will create a web application that predicts the profit a startup can make by providing some information about its spendings. This webapp is built by Flask Framework based on a machine learning model then pushed to Docker and finally deployed on Azure Server.

Getting Started

Getting Data

I got the data from a GitHub repository .It contains Research and Development Spendings , Administration Spendings, Marketing Spendings, The State and finally The Profit for each of the 50 startups .The data that we are using is clean and ready to use so we don't have to do anything to it.To see the data please scan th Qr code below :

Pick a model

Multiple regression analysis is a statistical technique used to analyze the relationship between a dependent variable and two or more independent variables. In my project, I chose to use multiple regression because I needed to understand the impact of several independent variables on a single dependent variable. By using this technique, I was able to determine which independent variables were most strongly associated with the dependent variable and how they affected its value. Multiple regression analysis also allowed me to test the significance of each independent variable, control for confounding factors, and make predictions based on the relationship between the variables. Overall, multiple regression analysis was an appropriate choice for my project because it provided a comprehensive analysis of the relationship between the dependent and independent variables.

Train the model

Now after we picked our model we will start to train it and for that we used JUPYTER Notebook : we started by importing the libraries that we will need like pandas , numpy ,sklearn.Then we load our data by specifying the dependant variable and the indepant variables .

```
In [2]: import pandas as pd
        import numpy as np
        from sklearn.preprocessing import LabelEncoder,OneHotEncoder
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import r2_score
        from sklearn.compose import ColumnTransformer
        from sklearn.model_selection import train_test_split
```

```
In [3]: data=pd.read_csv('50_startups.csv')
        X=data.iloc[:,:-1].values
        y=data.iloc[:,4].values
```

```
In [4]: data.head()
```

Then as we know we have a column named 'state' that have characters so our model will not work if we don't change it to numeric values for that we used One hot encoding to encode it , it will become 3 columns with each column representing one state (NewYork,California and Florida) .So if a startup is located in california the columns of 'NewYork' and 'Florida' will take 0 as a value and 'California' will take 1.As presented below :

```
In [7]: # One Hot Encoding categorical data
        labelencoder = LabelEncoder()
        X[:, 3] = labelencoder.fit_transform(X[:, 3])

        ct = ColumnTransformer(
            [('one_hot_encoder', OneHotEncoder(categories='auto'), [3])],
            remainder='passthrough'  # drop the other columns
        )

        X = ct.fit_transform(X)
```

```
In [8]: X
```

```
Out[8]: array([[0.0, 0.0, 1.0, 165349.2, 136897.8, 471784.1],
               [1.0, 0.0, 0.0, 162597.7, 151377.59, 443898.53],
```

After that we split the data so we can start the training , we used 0.2 of the data to test , then we trained the data with our model .Then we tested it and save the result in ypred and we displayed the test data with the predicted one to compare it .Also we did calculte r2 and it was 0.93 which means that our model is very good.

In [9]:
```
#split the dataset
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

In [10]:
```
regression=LinearRegression()
regression.fit(X_train,y_train)
```

Out[10]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the not
On GitHub, the HTML representation is unable to render, please try loading this page with nbviev

In [11]:
```
y_pred=regression.predict(X_test)
```

In [13]:
```
r2_score(y_test,y_pred)
```

Out[13]:
```
0.9347068473282303
```

In [14]:
```
df=pd.DataFrame(data=y_test,columns=['y_test'])
df['y_pred']=y_pred
df
```

Out[14]:

	y_test	y_pred
0	103282.38	103015.201598
1	144259.40	132582.277608
2	146121.95	132447.738452
3	77798.83	71976.098513
4	191050.39	178537.482211

Finally we saved our model as "multiple linear model.pkl"

Flask app

Now that our model is ready to use we will start coding our Flask app . So for our app we will use two pages ;the first one to input the information needed and the second one gives us the result of the prediction. To check the code you can find it in the repository below :

Docker image

Our app is working so we have to build an image of it on Docker :

- we should have Docker Desktop installed.
- we open the powershell or comand prompt and we execute some comands :
- we open the folder that contains our app
- we create the Dockerfile and requirements.txt
- docker build -t flaskmlmodel :latest .
- docker run -p 5000 :5000 flaskmlmodel :latest
- After we should push into our Dockerhub :
- we create a repository on the docker hub
- docker tag flaskmlmodel kaoutharr/cloudtest
- docker push kaoutharr/cloudtest

So after these steps we are almost done because we built our model , the webapp by flask is working we pushed on the Dockerhub we still need to deploy on Azure Server .

Deployement on Azure Server

This last step is very easy we need to create a student account on portal.azure.com , then we will create a web app and we will put the Dockerhub image in there and that' s it . To access to our webapp please scan the Qrcode below :