MODEL DEPLOYMENT ON FLASK

Name: Mohamed Derbeli
Batch Code: LISUM12
Date of submission: 29/08/2022
Submitted to: Data Glacier

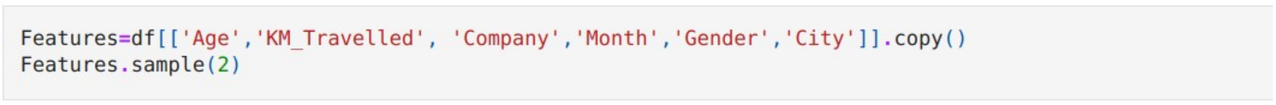
Model Deployment steps

Step1: Choosing the data

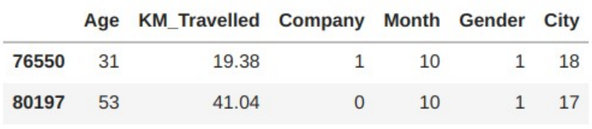
I have used the data of G2M insight for Cab Investment firm

Step2: Build and save the model

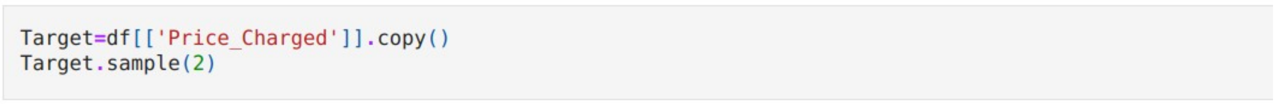
First I have choose the features and target.



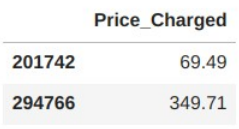
```
Features=df[['Age','KM_Travelled', 'Company','Month','Gender','City']].copy()
Features.sample(2)
```



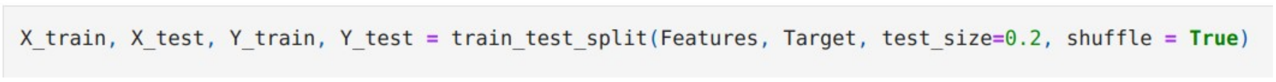
	Age	KM_Travelled	Company	Month	Gender	City
76550	31	19.38	1	10	1	18
80197	53	41.04	0	10	1	17



```
Target=df[['Price_Charged']].copy()
Target.sample(2)
```

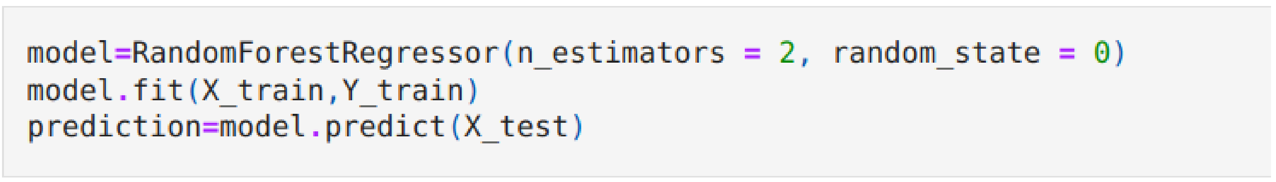


	Price_Charged
201742	69.49
294766	349.71



```
X_train, X_test, Y_train, Y_test = train_test_split(Features, Target, test_size=0.2, shuffle = True)
```

After the data split I have used Random Forest Regressor since it has provided high prediction accuracy.

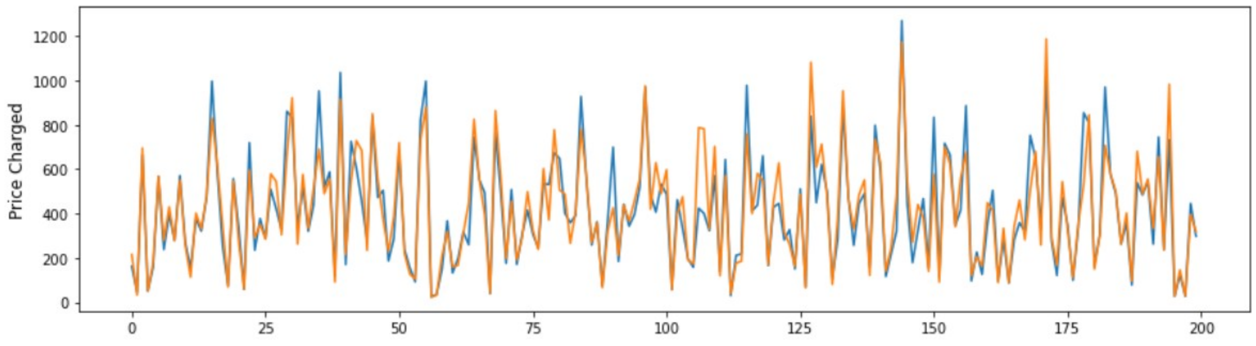


```
model=RandomForestRegressor(n_estimators = 2, random_state = 0)
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

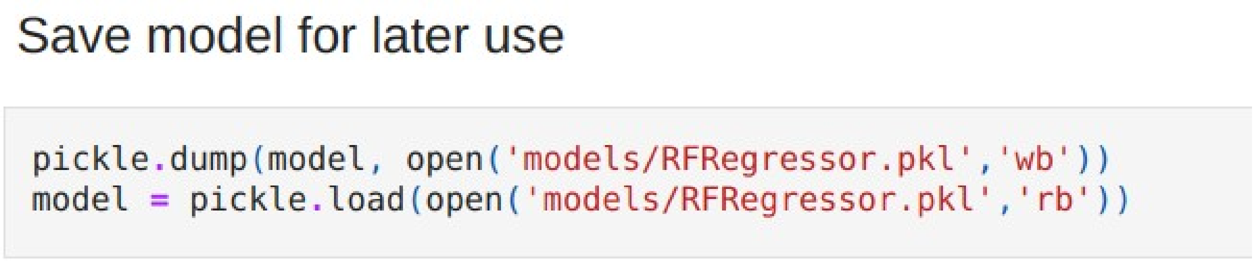
The following figure shows the actual and predicted data.

```
Y_test['Predictions']=prediction
Price_test=Y_test[:200]
fig, ax = plt.subplots()
fig.set_size_inches(15, 4)
plt.plot(Price_test[['Price_Charged','Predictions']].values)
plt.ylabel('Price Charged', fontsize = 12)
```

```
Text(0, 0.5, 'Price Charged')
```



Save the model:

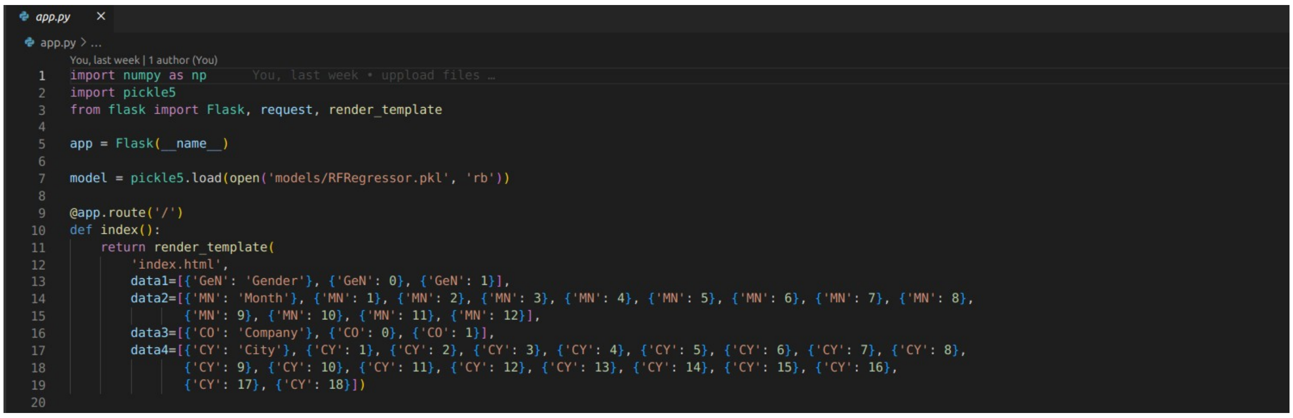


Save model for later use

```
pickle.dump(model, open('models/RFRegressor.pkl','wb'))
model = pickle.load(open('models/RFRegressor.pkl','rb'))
```

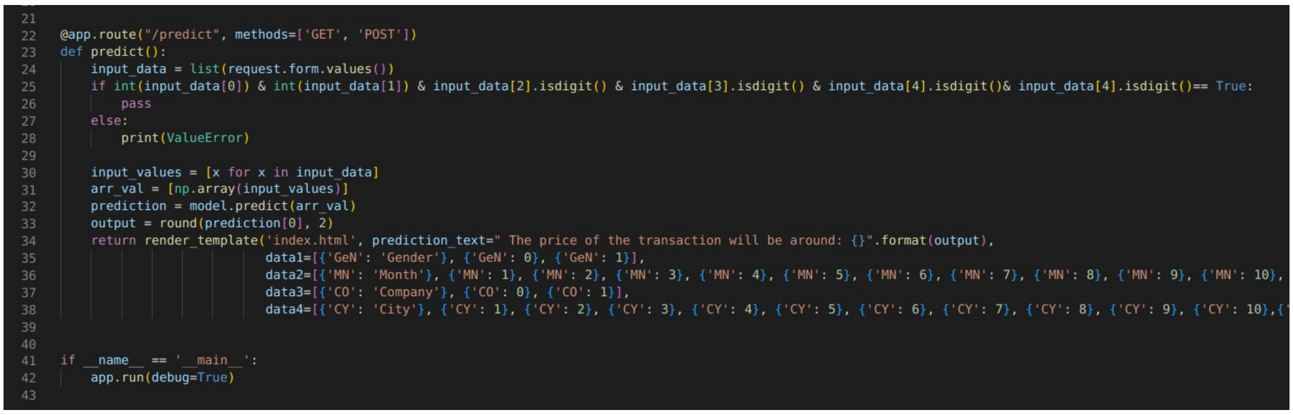
Step3: Deployment using Flask

Firstly, App.py is built, a flask app that used the deserialized model to accept new data and predict a the price charged.



```
import numpy as np      You, last week • uppload files …
import pickle5
from flask import Flask, request, render_template

app = Flask(__name__)

model = pickle5.load(open('models/RFRegressor.pkl', 'rb'))

@app.route('/')
def index():
    return render_template(
        'index.html',
        data1=[{'GeN': 'Gender'}, {'GeN': 0}, {'GeN': 1}],
        data2=[{'MN': 'Month'}, {'MN': 1}, {'MN': 2}, {'MN': 3}, {'MN': 4}, {'MN': 5}, {'MN': 6}, {'MN': 7}, {'MN': 8},
               {'MN': 9}, {'MN': 10}, {'MN': 11}, {'MN': 12}],
        data3=[{'CO': 'Company'}, {'CO': 0}, {'CO': 1}],
        data4=[{'CY': 'City'}, {'CY': 1}, {'CY': 2}, {'CY': 3}, {'CY': 4}, {'CY': 5}, {'CY': 6}, {'CY': 7}, {'CY': 8},
               {'CY': 9}, {'CY': 10}, {'CY': 11}, {'CY': 12}, {'CY': 13}, {'CY': 14}, {'CY': 15}, {'CY': 16},
               {'CY': 17}, {'CY': 18}])

```

Now the function below accepts the data and return the predicted percentage



```python

@app.route("/predict", methods=['GET', 'POST'])
def predict():
    input_data = list(request.form.values())
    if int(input_data[0]) & int(input_data[1]) & input_data[2].isdigit() & input_data[3].isdigit() & input_data[4].isdigit()& input_data[4].isdigit()== True:
        pass
    else:
        print(ValueError)

    input_values = [x for x in input_data]
    arr_val = [np.array(input_values)]
    prediction = model.predict(arr_val)
    output = round(prediction[0], 2)
    return render_template('index.html', prediction_text=" The price of the transaction will be around: {}".format(output),
                           data1=[{'GeN': 'Gender'}, {'GeN': 0}, {'GeN': 1}],
                           data2=[{'MN': 'Month'}, {'MN': 1}, {'MN': 2}, {'MN': 3}, {'MN': 4}, {'MN': 5}, {'MN': 6}, {'MN': 7}, {'MN': 8}, {'MN': 9}, {'MN': 10},
                           data3=[{'CO': 'Company'}, {'CO': 0}, {'CO': 1}],
                           data4=[{'CY': 'City'}, {'CY': 1}, {'CY': 2}, {'CY': 3}, {'CY': 4}, {'CY': 5}, {'CY': 6}, {'CY': 7}, {'CY': 8}, {'CY': 9}, {'CY': 10},{'


if __name__ == '__main__':
    app.run(debug=True)

```

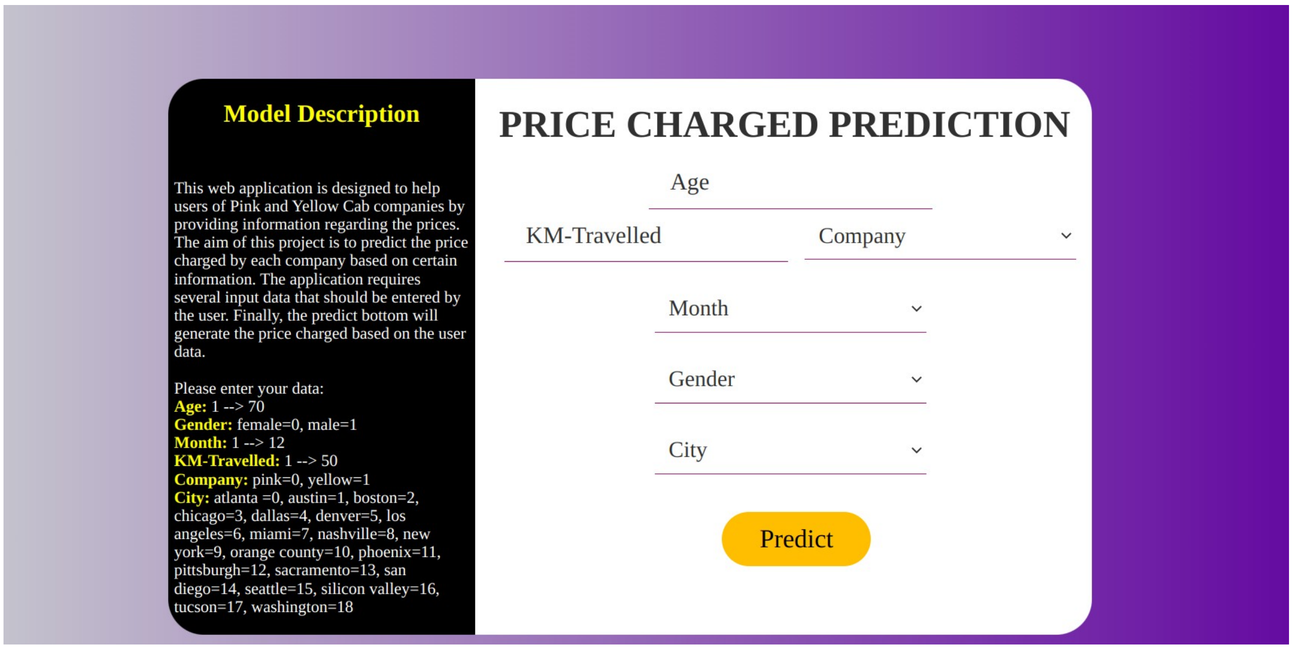
The index.html is a file that contains the structure of the web app design and AppStyle.css is used to beautify the web design.

Then, create a new app in Heroku and and push the repository to Heroku using the terminal. If everything goes well, a link will generated.

Inside the folder of your project, open terminal and implement the following code:

```
git init
git add .
git commit -m “first commit”
heroku git:remote -a name_of_app
git push heroku master
```

Finally, clicking on the link will direct you to the flask web application interface shown below.



Model Description

This web application is designed to help users of Pink and Yellow Cab companies by providing information regarding the prices. The aim of this project is to predict the price charged by each company based on certain information. The application requires several input data that should be entered by the user. Finally, the predict bottom will generate the price charged based on the user data.

Please enter your data:
Age: 1 --> 70
Gender: female=0, male=1
Month: 1 --> 12
KM-Travelled: 1 --> 50
Company: pink=0, yellow=1
City: atlanta =0, austin=1, boston=2, chicago=3, dallas=4, denver=5, los angeles=6, miami=7, nashville=8, new york=9, orange county=10, phoenix=11, pittsburgh=12, sacramento=13, san diego=14, seattle=15, silicon valley=16, tucson=17, washington=18

PRICE CHARGED PREDICTION