MODEL DEVELOPMENT ON FLASK

Name: Ece Kurnaz
Batch Code: LISUM01
Submission Date: 04.07.2021
Submitted To: Data Glacier

CREATING THE MODEL:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
import seaborn as sn
import matplotlib.pyplot as plt
import pickle

candidates = {'SAT': [1560,1500,1380,1420,1360,1460,1380,1440,1480,1380,1220,1380,1420,1360,1540,1220,1160,1300,1080,1180,1240,1200,1100,1100,1140,1340,1320,1160,1300,
              'GPA': [3.9,3.8,3.2,3.6,4.0,3.6,2.2,3.4,3.3,1.7,2.7,3.7,3.7,3.3,3.3,3,2.7,3.7,2.7,2.3,3.3,2,2.3,2.7,3,3.3,3.7,2.3,3.7,3.3,3,2.7,4,3.3,3.3,2.3,2.7,3.3,1.7
              'Experience': [3,4,3,5,4,6,1,5,4,3,3,5,6,4,3,1,4,6,2,3,2,2,3,1,2,5,2,5,6,2,2,3,6,4,1,3,1,5,5,5],
              'Accepted': [1,1,0,1,0,1,0,1,1,0,0,1,1,0,1,0,0,1,0,0,1,0,0,0,0,1,1,0,1,1,0,0,1,1,1,0,0,0,0,1]
              }

df = pd.DataFrame(candidates,columns= ['SAT', 'GPA','Experience','Accepted'])
```

```
df.head(5)
```

	SAT	GPA	Experience	Accepted
0	1560	3.9	3	1
1	1500	3.8	4	1
2	1380	3.2	3	0
3	1420	3.6	5	1
4	1360	4.0	4	0

```
df.isnull().sum().sum()
```

```
0
```

```
X = df[['SAT','GPA','Experience']]
y = df['Accepted']
```

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.25,random_state=0)

logistic_regression= LogisticRegression()
logistic_regression.fit(X_train,y_train)
y_pred=logistic_regression.predict(X_test)
print('Accuracy: ',metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy:  0.7
```

```
filename = 'model_.pkl'
pickle.dump(logistic_regression, open(filename, 'wb'))
```

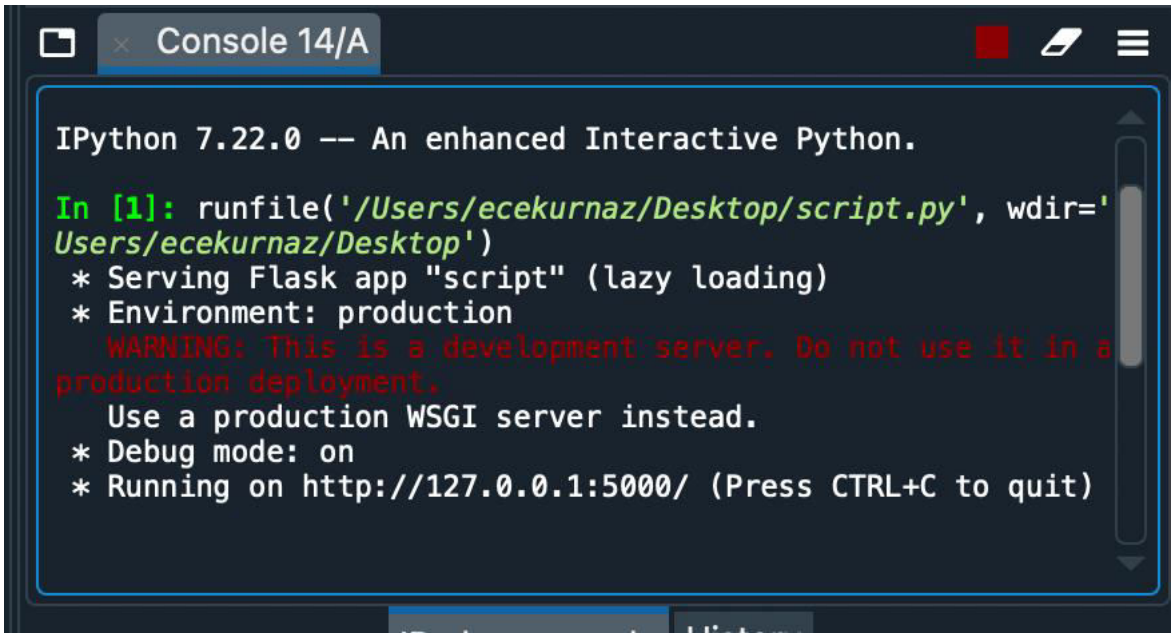
```
filename = 'model_.pkl'
model = pickle.load(open(filename, 'rb'))
result =model.score(X_test, y_test)
print(result)
```

```
0.7
```

```
new_df= {'SAT':[1600,400,450], 'GPA':[2.5,2.1,3.5], 'Experience':[2,2,2] }
```

```
new_df = pd.DataFrame(new_df,columns= ['SAT','GPA' ,'Experience'])
y_pred=model.predict(new_df)

print (new_df)
print (y_pred)
```

```python
import os
import numpy as np
import flask
import pickle
from flask import Flask, render_template, request

#creating instance of the class
app=Flask(__name__)
model = pickle.load(open('checkpoints/model_.pkl','rb'))

#to tell flask what url shoud trigger the function index()
@app.route('/')
def home():
    return flask.render_template('index3.html')

@app.route('/predict',methods = ['POST'])
def predict():
    '''
    DEFINE PREDICTION FUCTION
    '''
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = round(prediction[0],2)

    return render_template("index.html", prediction_text='The final decision of the University should be {}'.format(output))

if __name__=="__main__":
    app.run(port=5000, debug=True,use_reloader=False)
```

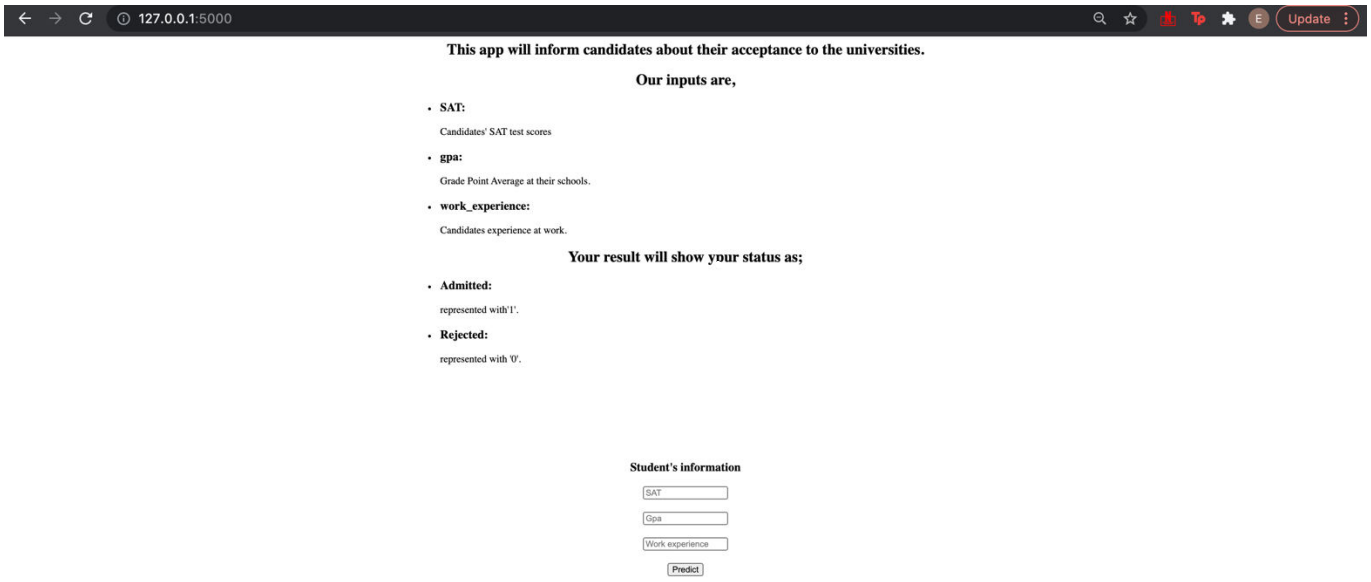


DESIGN OF THE PAGE:

```
<html>

<head>
  <meta charset="utf-8">
  <title>University Acceptance</title>
  <!-- <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Pacifico" type="text/css"/>
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Arimo" type="text/css"/>
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Hind:300" type="text/css"/>
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300" type="text/css"/>-->
  <style>
h1 {text-align: center;}
h2 {text-align: center;}
div {text-align: center;}
ul {
text-align: left;
width: 40%;
position:relative;
left: 30%;
}
body {
  background-image: url('static/images/img.jpg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}

</style>

<link rel="stylesheet" href="{{ url_for('static', filename='css/template.css') }}">

</head>

<body>



    <h2>This app will inform candidates about their acceptance to the universities.</h2>


    <div class="row">

  <div class="column">
    <h2>Our inputs are,</h2>

    <ul>
      <li>
      <h3>SAT:</h3> Candidates' SAT test scores
      </li>
      <li>
      <h3>gpa:</h3> Grade Point Average at their schools.
      </li>
      <li>
      <h3>work_experience:</h3> Candidates experience at work.
    </li>
    </ul>
```

```
    <h2>Your result will show ypur status as;</h2>

<ul>
  <li>
    <h3>Admitted:</h3> represented with'1'.
  </li>
  <li>
    <h3>Rejected:</h3> represented with '0'.
  </li>
</ul>

</div>


<div class="column">

  <div class="login">

    <br />
  <br />
  <br />
  <br />
  <br />
  <br />

<h3>Student's information</h3>

<form action="{{ url_for('predict') }}" method="POST">
  <input type="text" name="SAT" placeholder="SAT" required="required" />
  <br />
  <br />
  <input type="text" name="GPA" placeholder="Gpa" required="required" />
  <br />
  <br />
  <input type="text" name="Experience" placeholder="Work experience" required="required" />
  <br />
  <br />
  <button class="btn btn-primary btn-block btn-large" type="submit">Predict</button>
</form>

<br />


{{ prediction_text}}
```

WEB PAGE:


127.0.0.1:5000
Update
This app will inform candidates about their acceptance to the universities.
Our inputs are,
SAT:
Candidates' SAT test scores
gpa:
Grade Point Average at their schools.
work_experience:
Candidates experience at work.
Your result will show your status as;
Admitted:
represented with'1'.
Rejected:
represented with '0'.
Student's information
SAT
Gpa
Work experience
Predict