Name: Hiten Chadha
Batch Code: LISUM10: 30
Submission Date: 05/07/2022
Submitted to: DataGlacier

Cloud based Deployment using Heroku

Step 1: Serialization and Deserialization of ML Model to create .pkl file in order to create our model in flask(I have made a new model with a simpler dataset for this assignment)

```
import numpy as np
import pandas as pd
import pickle

dataset = pd.read_csv("Salary Prediction based on Position and Experience.csv
X = dataset.iloc[:,[1,2]].values
y = dataset.iloc[:,-1].values

from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor()

regressor.fit(X,y)
pickle.dump(regressor, open('model.pkl', 'wb'))

model = pickle.load(open('model.pkl', 'rb'))
```

Step 2: Creating index.html file for home page

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>

</head>

<body>
 <div class="login">
    <h1>Predict Salary Analysis</h1>

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
        <input type="text" name="experience" placeholder="Experience" required="required" />
        <input type="text" name="test score" placeholder="Test Score" required="required" />

        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
    </form>

   <br>
   <br>
   {{ prediction_text }}

 </div>
```

Step 3: Creating the app.py file and extracting the features from the client and render the result back to the html page as done previously to create a web app using flask

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

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():

    features = [int(x) for x in request.form.values()]
    final_features = [np.array(features)]
    prediction = model.predict(final_features)

    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text="Salary is $ {}".format(output))

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

Step 4: Create the Proc file as well as the requirements.txt file for Heroku App Deployment

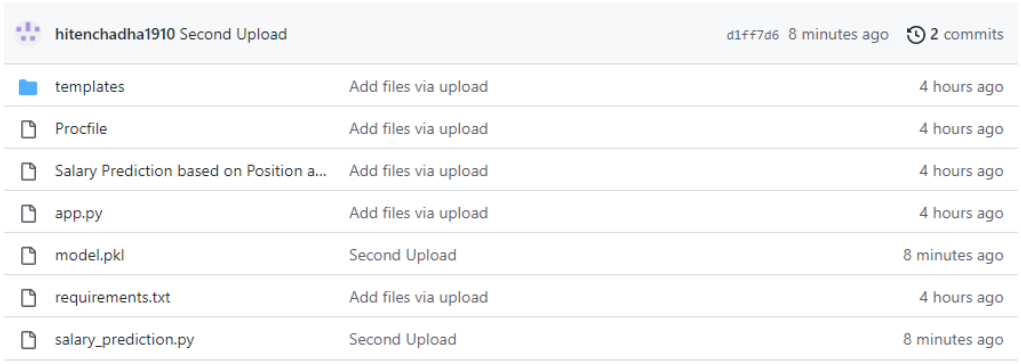
Procfile

```
web: gunicorn app:app
```

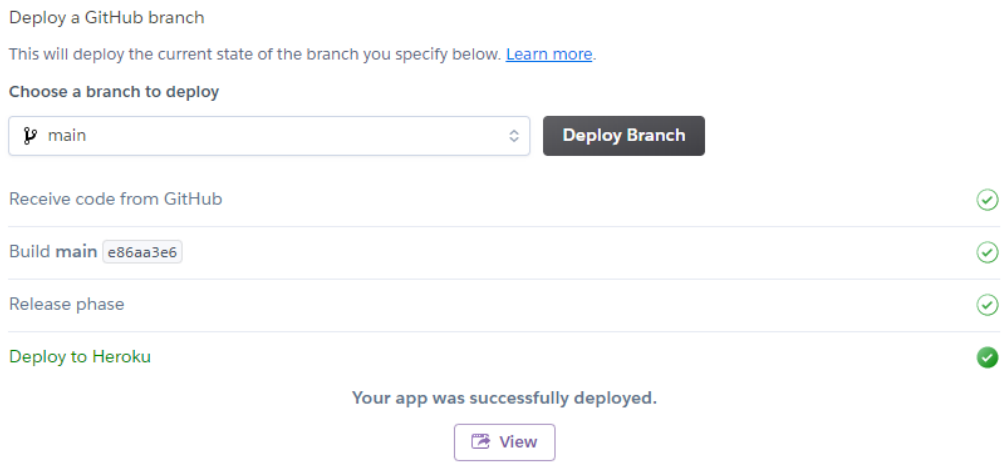
requirements.txt

```
Flask==2.1.2
gunicorn==20.1.0
itsdangerous==2.1.2
Jinja2==3.1.2
MarkupSafe==2.1.1
Werkzeug==2.1.2
numpy>=1.19.2
scipy>=1.8.0
scikit-learn>=1.0.2
matplotlib>=3.3.2
pandas>=1.4.3
```

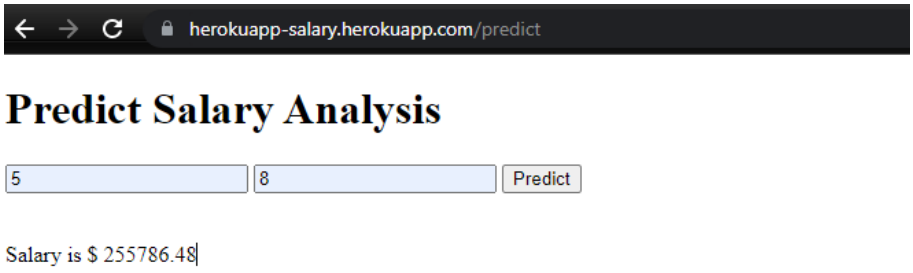
Step 5: Upload all the files to the GitHub repository



Step 6: Deploy the python app to Heroku and choose the Git connect as the deployment method and Deploy the branch to build the app.



Step 7: Run the app with the link provided or simply click on “View”



API based deployment on POSTMAN

Step 1: Create the model and the .pkl file

```
import numpy as np
import pandas as pd
import pickle

dataset = pd.read_csv("Salary Prediction based on Position and Experience.csv"
X = dataset.iloc[:,[1,2]].values
y = dataset.iloc[:,-1].values

from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor()

regressor.fit(X,y)
pickle.dump(regressor, open('model.pkl', 'wb'))
```

Step 2: Create the API endpoints to get the inputs as key,value pairs

```
app = Flask(__name__)

@app.route('/', methods = ['GET', 'POST'])
def home():
    if(request.method == "GET"):

        data = "Hello"
        return jsonify({'data': data})


@app.route('/predict/')
def predict():

    model = pickle.load(open('model.pkl', 'rb'))
    experience = request.args.get('Experience')
    test_score = request.args.get('Test Score')

    df = pd.DataFrame({'experience':[experience], 'test_score':[test_score]})

    prediction = model.predict(df)

    return jsonify({'Salary predicted': str(prediction)})

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

Step 3: Open POSTMAN and generate a request with key, value pairs and SEND request