

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

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

```
requirements.txt
1 Flask==2.1.2
2 gunicorn==20.1.0
3 itsdangerous==2.1.2
4 Jinja2==3.1.2
5 MarkupSafe==2.1.1
6 Werkzeug==2.1.2
7 numpy>=1.19.2
8 scipy>=1.8.0
9 scikit-learn>=1.0.2
10 matplotlib>=3.3.2
11 pandas>=1.4.3
```

Step 5: Upload all the files to the GitHub repository

hitenchadha1910 Second Upload		d1ff7d6 8 minutes ago	🕒 2 commits
📁 templates	Add files via upload	4 hours ago	
📄 Procfile	Add files via upload	4 hours ago	
📄 Salary Prediction based on Position a...	Add files via upload	4 hours ago	
📄 app.py	Add files via upload	4 hours ago	
📄 model.pkl	Second Upload	8 minutes ago	
📄 requirements.txt	Add files via upload	4 hours ago	
📄 salary_prediction.py	Second Upload	8 minutes ago	

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.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

🔗 main

Deploy Branch

Receive code from GitHub



Build main e86aa3e6



Release phase



Deploy to Heroku



Your app was successfully deployed.

🔗 View

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

← → ↻ 🔒 herokuapp-salary.herokuapp.com/predict

Predict Salary Analysis

5 8 Predict

Salary is \$ 255786.48

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

http://127.0.0.1:5000/predict/?Experience=7.5&Test Score=8 Save

GET ⌵ http://127.0.0.1:5000/predict/?Experience=7.5&Test Score=8 Send ⌵

Params ● Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	Experience	7.5			
<input checked="" type="checkbox"/>	Test Score	8			
	Key	Value	Description		

Body Cookies Headers (5) Test Results 🌐 200 OK 54 ms 202 B Save Response ⌵

Pretty Raw Preview Visualize JSON ⌵

```
1 {  
2   "Salary predicted": "[235500.]"  
3 }
```