

MODEL DEPLOYMENT ON THE CLOUD HEROKU

Name: MEMUDU Alimatou sadia

Batch Code: LISUM02

Date of submission: 07/08/2021

Submitted to:

Model Deployment Stages

Stage1: Choosing a simple data

This is the [data](#) used for this project. The dataset contains several parameters which are considered important during the application for Masters Programs.

The parameters included are:

GRE Scores (out of 340)

TOEFL Scores (out of 120)

University Rating (out of 5)

Statement of Purpose and Letter of Recommendation Strength (out of 5)

Undergraduate GPA (out of 10)

Research Experience (either 0 or 1)

Chance of Admit (ranging from 0 to 1)

Stage2: Build and save a model using Flask

The model's goal is to predict the chance of a student to get admitted into a university. We used Pickle to serialize the model for future use in the [admission_model.py](#) file.

```
# Splitting the data
x = data.iloc[:, :-1].values
y = data.iloc[:, 7].values

# split dataset

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

# Fitting linear regression Regression to the dataset
lin_regressor = LinearRegression(normalize=True)
lin_regressor.fit(X_train, y_train)

# To save the model to the disk (serialization) for future use
pickle.dump(lin_regressor, open('admission_model.pkl', 'wb'))
```

Stage3: Deployment using Flask

Firstly, [App.py](#) is built, a flask app that used the deserialized model to accept new data and predict a student percentage to get admission.

```
1 import numpy as np
2 import pickle
3 from flask import Flask, request, render_template
4
5 app = Flask(__name__)
6
7 # Performing deserialization using pickle
8 model = pickle.load(open("admission_model.pkl", 'rb'))
9
10
11 @app.route('/')
12 def index():
13     return render_template(
14         'index.html',
15         data=[{'UR': 'University Rating'}, {'UR': 1}, {'UR': 2}, {'UR': 3}, {'UR': 4}, {'UR': 5}],
16         data1=[{'ReS': 'Research'}, {'ReS': 0}, {'ReS': 1}])
17
```

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

```
19 @app.route("/predict", methods=['GET', 'POST'])
20 def predict():
21     input_data = list(request.form.values())
22     if int(input_data[0]) & int(input_data[1]) & input_data[3].isdigit() & input_data[4].isdigit() & input_data[5].isdigit() == True:
23         pass
24     else:
25         print(ValueError)
26
27     input_values = [x for x in input_data]
28     arr_val = [np.array(input_values)]
29     prediction = model.predict(arr_val)
30     output = round(prediction[0], 2)*100
31     return render_template('index.html', prediction_text=" The Chance of Getting into the University is {} %".format(output),
32         data=[{'UR': 'University Rating'}, {'UR': 1}, {'UR': 2}, {'UR': 3}, {'UR': 4}, {'UR': 5}],
33         data1=[{'ReS': 'Gender'}, {'ReS': 0}, {'ReS': 1}])
34
35
36 if __name__ == '__main__':
37     app.run(debug=True)
38
```

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.

Secondly, you have to write: “python app.py” in the terminal to run the flask application then a link will display.

```
Terminal Local + -
PS C:\Users\Alimat sadia\my pyPrograms> cd "data science"
PS C:\Users\Alimat sadia\my pyPrograms\data science> cd "ADMISSION WEB APPLICATION"
PS C:\Users\Alimat sadia\my pyPrograms\data science\ADMISSION WEB APPLICATION> python app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 683-954-664
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

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

Model Description

This web application is a chance of getting into university prediction system. The aim of this project is to predict a student's change of getting into university based on its previous exam scores. After inputting the corresponding data, a predicted chance will display.

The exam score needed are :
The GRE Score
TOEFL Score
University Rating
SOP- Statement of Purpose
LOR- Letter of Recommendation
CGPA- Cumulative Grade Point
Average
Research-- Made a research?
(Yes=1/No=0)

ADMISSION CHANCE PREDICTION

Your GRE score _____

Your TOEFL score _____ University Rating _____

Enter your SOP score _____

Enter your LOR score _____

Enter your CGPA _____ Research _____

Predict

On this interface, a description of the web app's function is explained by the left and on the right there is the input section that collects the user's data.

This data will be fed into the deserialized model which will provide an output (percentage of change to get admitted) as illustrated below.

Model Description

This web application is a chance of getting into university prediction system. The aim of this project is to predict a student's change of getting into university based on its previous exam scores. After inputting the corresponding data, a predicted chance will display.

The exam score needed are :
The GRE Score
TOEFL Score
University Rating
SOP- Statement of Purpose
LOR- Letter of Recommendation
CGPA- Cumulative Grade Point
Average
Research-- Made a research?
(Yes=1/No=0)

ADMISSION CHANCE PREDICTION

Your GRE score 250

Your TOEFL score 100 University Rating 4

Enter your SOP score 1.5

Enter your LOR score 4.2

Enter your CGPA 3.2 Research 0

Predict

Finally, clicking on the predict button will displayed the predicted value as show below.

Model Description

This web application is a chance of getting into university prediction system. The aim of this project is to predict a student's change of getting into university based on its previous exam scores. After inputting the corresponding data, a predicted chance will display.

The exam score needed are :
The GRE Score
TOEFL Score
University Rating
SOP- Statement of Purpose
LOR- Letter of Recommendation
CGPA- Cumulative Grade Point
Average
Research-- Made a research?
(Yes=1/No=0)

ADMISSION CHANCE PREDICTION

Your GRE score _____

Your TOEFL score _____ University Rating _____

Enter your SOP score _____

Enter your LOR score _____

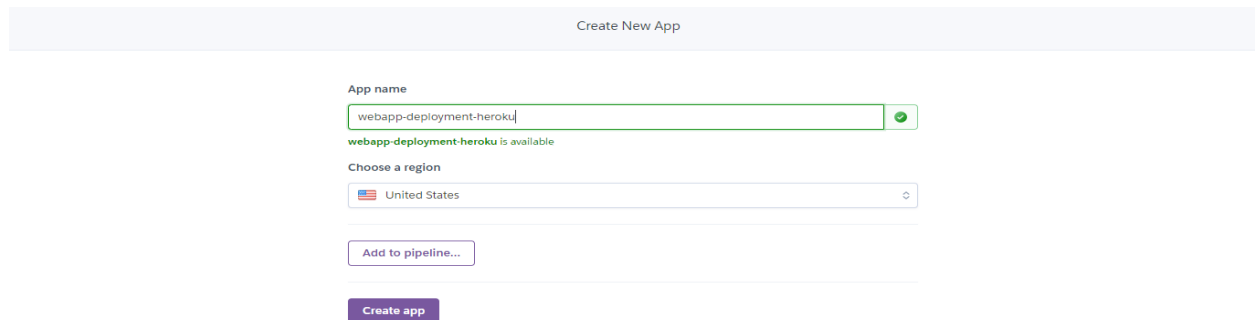
Enter your CGPA _____ Gender _____

Predict

The Chance of Getting into the University is 53.0 %

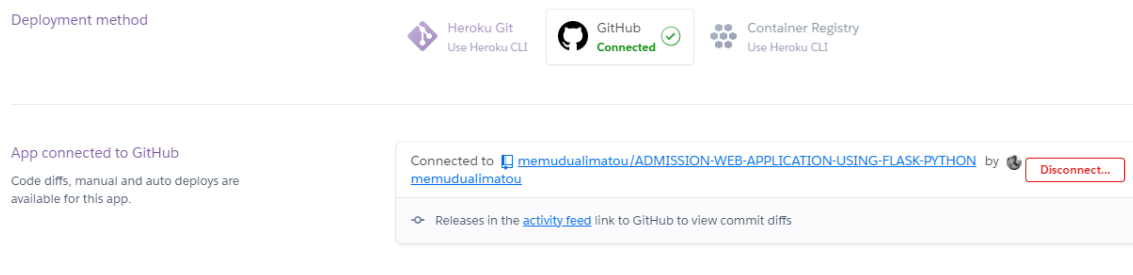
Step4: Deployment of the flask API on the cloud

I created a Heroku account after the login then by the right corner, I click on “New App” to create a new app to deploy.



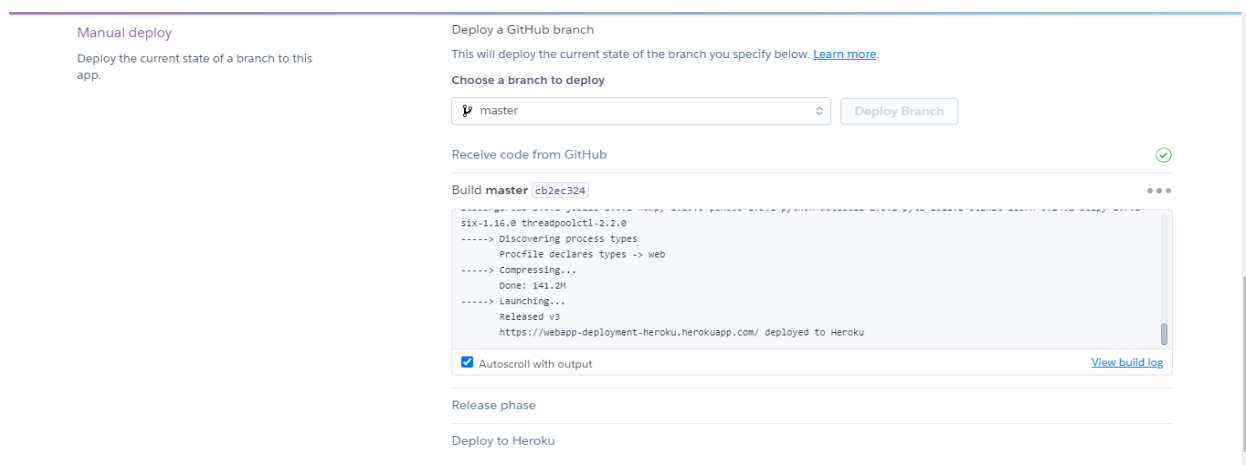
The screenshot shows the 'Create New App' form on Heroku. At the top, it says 'Create New App'. Below that, there's a section for 'App name' with a text input field containing 'webapp-deployment-heroku' and a green checkmark icon. A message below the input says 'webapp-deployment-heroku is available'. Underneath, there's a 'Choose a region' section with a dropdown menu showing 'United States' and a small flag icon. At the bottom of this section, there's a button that says 'Add to pipeline...'. Finally, at the very bottom, there's a purple button that says 'Create app'.

After inserting the project name, I clicked on Create app then connected this [project repository](#) to the deployment method in Heroku.



The screenshot shows the 'Deployment method' selection screen on Heroku. At the top, it says 'Deployment method'. Below that, there are three options: 'Heroku Git' with a purple icon and the text 'Use Heroku CLI', 'GitHub' with a black icon, a green checkmark, and the text 'Connected', and 'Container Registry' with a grey icon and the text 'Use Heroku CLI'. Below these options, there's a section titled 'App connected to GitHub'. It says 'Code diffs, manual and auto deploys are available for this app.' To the right of this text, there's a box containing the text 'Connected to [memudualimatou/ADMISSION-WEB-APPLICATION-USING-FLASK-PYTHON](#) by [memudualimatou](#)' with a 'Disconnect...' button. Below this, there's a link that says 'Releases in the [activity feed](#) link to GitHub to view commit diffs'.

After a successful connection, I deployed the project manually, this step is all about the virtual installation of libraries required to run the web app. This project's library dependencies and their different versions are listed in the [requirements.txt](#) file. The [link](#) presents in the Build master section below directs to the web app deployed in the cloud.



The screenshot shows the 'Manual deploy' screen on Heroku. On the left, there's a section titled 'Manual deploy' with the text 'Deploy the current state of a branch to this app.' On the right, there's a section titled 'Deploy a GitHub branch' with the text 'This will deploy the current state of the branch you specify below. [Learn more.](#)'. Below this, there's a 'Choose a branch to deploy' section with a dropdown menu showing 'master' and a 'Deploy Branch' button. Underneath, there's a 'Receive code from GitHub' section with a green checkmark icon. Below that, there's a 'Build master (cb2ec324)' section with a terminal output showing the build process. The output includes the following text: 'six-1.16.0 threadpoolctl-2.2.0', '----> Discovering process types', 'Procfile declares types -> web', '----> Compressing...', 'Done: 141.2M', '----> Launching...', 'Released v3', and 'https://webapp-deployment-heroku.herokuapp.com/ deployed to Heroku'. At the bottom of this section, there's a checkbox labeled 'Autoscroll with output' which is checked, and a link that says 'View build log'. Below the build section, there's a 'Release phase' section with the text 'Deploy to Heroku'.

A deployment is garentee after a succesfull installation of all requirement libraries as shown below.

The screenshot shows the Heroku deployment interface. On the left, under 'Manual deploy', it says 'Deploy the current state of a branch to this app.' On the right, under 'Deploy a GitHub branch', it says 'This will deploy the current state of the branch you specify below. [Learn more](#).' Below this, there is a dropdown menu for 'Choose a branch to deploy' with 'master' selected, and a 'Deploy Branch' button. A progress bar shows the following steps: 'Receive code from GitHub' (checked), 'Build master' (checked, with commit hash 'cb2ec324'), 'Release phase' (checked), and 'Deploy to Heroku' (checked). Below the progress bar, it says 'Your app was successfully deployed.' and there is a 'View' button.

Clicking on the “View” button directs to the deployed web app [link](#). The admission Flask API will display. Check the URL section of my web browser the “webapp-deployment-heroku” has been successfully deployed into cloud using Heroku.

The screenshot shows a web browser displaying the 'webapp-deployment-heroku.herokuapp.com' website. The page has a purple background. On the left, there is a 'Model Description' section with text about the application's purpose and a list of required exam scores. On the right, there is a form titled 'ADMISSION CHANCE PREDICTION' with input fields for 'Your GRE score', 'Your TOEFL score', 'University Rating', 'Enter your SOP score', 'Enter your LOR score', 'Enter your CGPA', and 'Research'. A 'Predict' button is at the bottom of the form.

Model Description

This web application is a chance of getting into university prediction system. The aim of this project is to predict a student's chance of getting into university based on its previous exam scores. After inputting the corresponding data, a predicted chance will display.

The exam score needed are :

- The GRE Score
- TOEFL Score
- University Rating
- SOP- Statement of Purpose
- LOR- Letter of Recommendation
- CGPA- Cumulative Grade Point Average
- Research-- Made a research? (Yes=1/No=0)

ADMISSION CHANCE PREDICTION

Your GRE score

Your TOEFL score University Rating

Enter your SOP score

Enter your LOR score

Enter your CGPA Research

Predict

Testing the web app by inserting some data and view the result.

The screenshot shows a web application titled "ADMISSION CHANCE PREDICTION". On the left, a "Model Description" box explains the application's purpose and lists the required exam scores: GRE, TOEFL, University Rating, SOP, LOR, CGPA, Average, and Research. The main form on the right contains input fields for these scores: GRE (250), TOEFL (100), University Rating (3), SOP (1.5), LOR (4.2), CGPA (9.5), and Research (1). A "Predict" button is located at the bottom of the form.

The predicted chance of getting admitted into a university is displayed.

This screenshot shows the same web application after a prediction. The input fields are now labeled with their respective categories: "Your GRE score", "Your TOEFL score", "University Rating", "Enter your SOP score", "Enter your LOR score", "Enter your CGPA", and "Gender". The "Predict" button is still present. Below the button, the result is displayed: "The Chance of Getting into the University is 69.0 %".

Project repository: <https://github.com/memudualimatou/ADMISSION-WEB-APPLICATION-USING-FLASK-PYTHON>

Deployed web app link: <https://webapp-deployment-heroku.herokuapp.com>