

Week 5: Deployment on Flask and Cloud & API Heroku Deployment

Name: Ayah Ibrahim

Batch Code: LISUM22

Submission Date: 7/5/2023

Submitted to: Data Glacier

This is a sample of the toy data (from Week 4) I chose to work with. The data represents the 2016 World Happiness by country and region using a variety of different variables such as Life Expectancy, Family, Economy, Generosity, Freedom, etc.

| A | B | C | D | E | F | G | H | I | J | K | L | M |
|---------------|---------------------------------|----------------|-----------------|---------------------------|---------------------------|--------------------------|---------|--------------------------|---------|--------------------|------------|-------------------|
| Country | Region | Happiness Rank | Happiness Score | Lower Confidence Interval | Upper Confidence Interval | Economy (GDP per Capita) | Family | Health (Life Expectancy) | Freedom | Trust (Government) | Generosity | Dystopia Residual |
| Denmark | Western Europe | 1 | 7.526 | 7.46 | 7.592 | 1.44178 | 1.16374 | 0.79504 | 0.57941 | 0.44453 | 0.36171 | 2.73939 |
| Switzerland | Western Europe | 2 | 7.509 | 7.428 | 7.59 | 1.52733 | 1.14524 | 0.86303 | 0.58557 | 0.41203 | 0.28083 | 2.69463 |
| Iceland | Western Europe | 3 | 7.501 | 7.333 | 7.669 | 1.42666 | 1.18326 | 0.86733 | 0.56624 | 0.14975 | 0.47678 | 2.83137 |
| Norway | Western Europe | 4 | 7.498 | 7.421 | 7.575 | 1.57744 | 1.1269 | 0.79579 | 0.59609 | 0.35776 | 0.37895 | 2.66465 |
| Finland | Western Europe | 5 | 7.413 | 7.351 | 7.475 | 1.40598 | 1.13464 | 0.81091 | 0.57104 | 0.41004 | 0.25492 | 2.82596 |
| Canada | North America | 6 | 7.404 | 7.335 | 7.473 | 1.44015 | 1.0961 | 0.8276 | 0.5737 | 0.31329 | 0.44834 | 2.70485 |
| Netherlands | Western Europe | 7 | 7.339 | 7.284 | 7.394 | 1.46468 | 1.02912 | 0.81231 | 0.55211 | 0.29927 | 0.47416 | 2.70749 |
| New Zealand | Australia and New Zealand | 8 | 7.334 | 7.264 | 7.404 | 1.36066 | 1.17278 | 0.83096 | 0.58147 | 0.41904 | 0.49401 | 2.47553 |
| Australia | Australia and New Zealand | 9 | 7.313 | 7.241 | 7.385 | 1.44443 | 1.10476 | 0.8512 | 0.56837 | 0.32331 | 0.47407 | 2.5465 |
| Sweden | Western Europe | 10 | 7.291 | 7.227 | 7.355 | 1.45181 | 1.08764 | 0.83121 | 0.58218 | 0.40867 | 0.38254 | 2.54734 |
| Israel | Middle East and Northern Africa | 11 | 7.267 | 7.199 | 7.335 | 1.33766 | 0.99537 | 0.84917 | 0.36432 | 0.08728 | 0.32288 | 3.31029 |
| Austria | Western Europe | 12 | 7.119 | 7.045 | 7.193 | 1.45038 | 1.08383 | 0.80565 | 0.54355 | 0.21348 | 0.32865 | 2.69343 |
| United States | North America | 13 | 7.104 | 7.02 | 7.188 | 1.50796 | 1.04782 | 0.779 | 0.48163 | 0.14868 | 0.41077 | 2.72782 |
| Costa Rica | Latin America and Caribbean | 14 | 7.087 | 6.999 | 7.175 | 1.06879 | 1.02152 | 0.76146 | 0.55225 | 0.10547 | 0.22553 | 3.35168 |

In this next step, I have performed all required code on Jupyter Notebook needed to import and read my toy data using **Pandas** and extract features needed to build the model, train my data using a linear regression machine learning model using **Sklearn**, and finally save my model using **Pickle** as “happiness_model”.

In [1]: *# Import the libraries needed:*

```
# pandas to read the CSV file, scikit-learn for training the model, and pickle to save the trained model  
  
import pandas as pd  
from sklearn.linear_model import LinearRegression  
from sklearn.model_selection import train_test_split  
import pickle
```

In [2]: *# Read my toy data*

```
data = pd.read_csv(r"C:\Users\Aya K\Desktop\world_happiness_2016.csv")
```

In [3]: *# Extract the features (independent variables) and the target variable (dependent variable) from the data*

```
X = data[['Happiness Rank', 'Lower Confidence Interval', 'Upper Confidence Interval', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corru  
y = data['Happiness Score']
```

In [4]: *# We will divide the data into two parts, training and testing sets: one for training the model and the other for evaluating its performance*

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [5]: *# We will use the training data to train a machine learning model. In this case, we'll use linear regression*

```
model = LinearRegression()  
model.fit(X_train, y_train)
```

Out[5]:

▼ LinearRegression

LinearRegression()

In [6]: *# We will calculate the model's accuracy or any other appropriate metric to assess its performance*

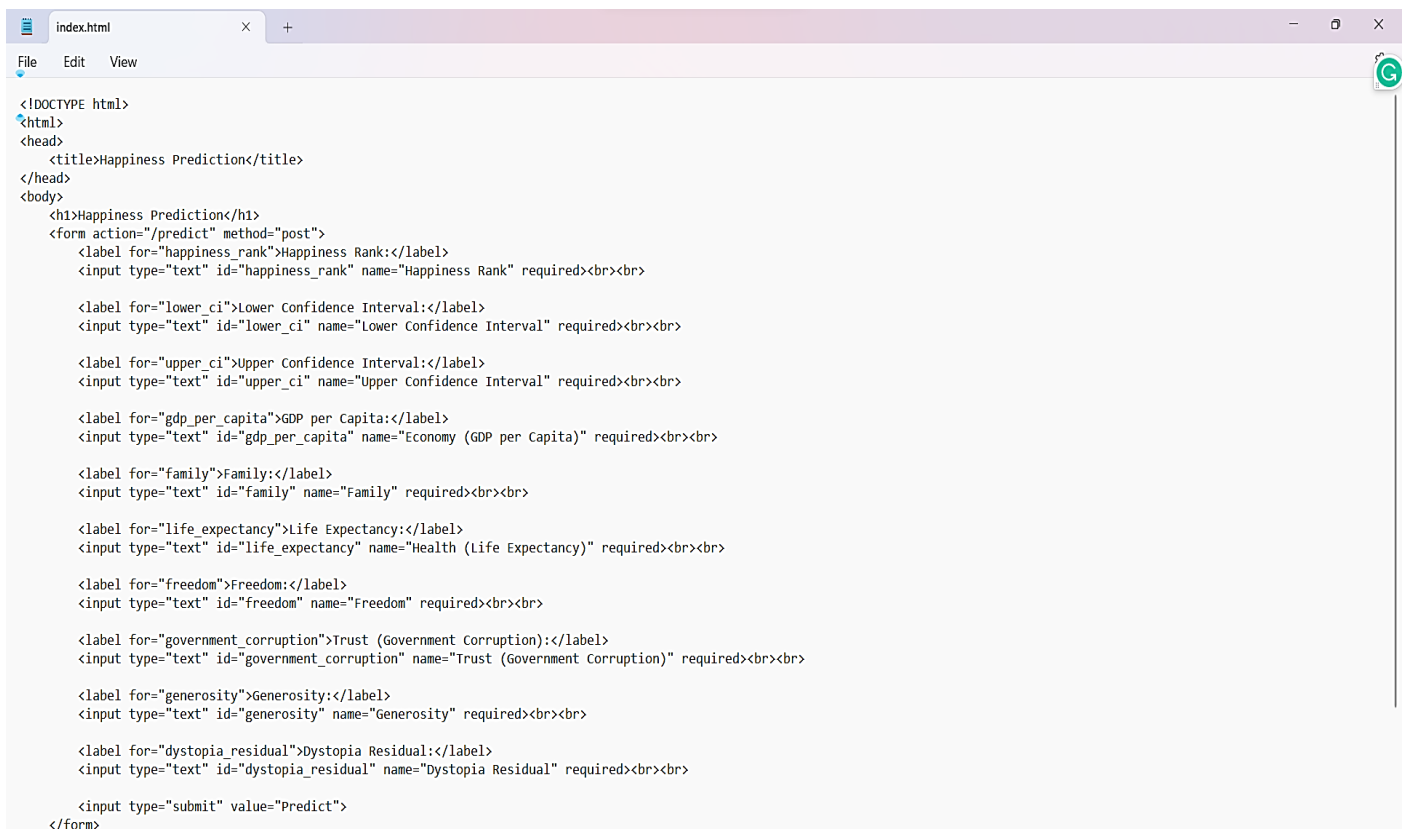
```
accuracy = model.score(X_test, y_test)  
print("Model Accuracy:", accuracy)
```

Model Accuracy: 1.0

In [7]: *# Save the model using pickle*

```
pickle.dump(model, open('happiness_model.pkl', 'wb'))
```

Before using **Flask** to deploy my web app, I will create a file as **index.html** to help create the web design and application and also include some of the variables from the toy data so the user can predict them.

A screenshot of a web browser window with a single tab titled 'index.html'. The browser's address bar is empty, and the page content is displayed as raw HTML code. The code defines a form titled 'Happiness Prediction' with a POST action to '/predict'. It includes ten text input fields, each with a label and a name attribute: 'Happiness Rank', 'Lower Confidence Interval', 'Upper Confidence Interval', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)', 'Generosity', and 'Dystopia Residual'. All inputs are marked as required. A submit button with the value 'Predict' is at the bottom of the form.

```
<!DOCTYPE html>
<html>
<head>
  <title>Happiness Prediction</title>
</head>
<body>
  <h1>Happiness Prediction</h1>
  <form action="/predict" method="post">
    <label for="happiness_rank">Happiness Rank:</label>
    <input type="text" id="happiness_rank" name="Happiness Rank" required><br><br>

    <label for="lower_ci">Lower Confidence Interval:</label>
    <input type="text" id="lower_ci" name="Lower Confidence Interval" required><br><br>

    <label for="upper_ci">Upper Confidence Interval:</label>
    <input type="text" id="upper_ci" name="Upper Confidence Interval" required><br><br>

    <label for="gdp_per_capita">GDP per Capita:</label>
    <input type="text" id="gdp_per_capita" name="Economy (GDP per Capita)" required><br><br>

    <label for="family">Family:</label>
    <input type="text" id="family" name="Family" required><br><br>

    <label for="life_expectancy">Life Expectancy:</label>
    <input type="text" id="life_expectancy" name="Health (Life Expectancy)" required><br><br>

    <label for="freedom">Freedom:</label>
    <input type="text" id="freedom" name="Freedom" required><br><br>

    <label for="government_corruption">Trust (Government Corruption):</label>
    <input type="text" id="government_corruption" name="Trust (Government Corruption)" required><br><br>

    <label for="generosity">Generosity:</label>
    <input type="text" id="generosity" name="Generosity" required><br><br>

    <label for="dystopia_residual">Dystopia Residual:</label>
    <input type="text" id="dystopia_residual" name="Dystopia Residual" required><br><br>

    <input type="submit" value="Predict">
  </form>
```

```
index.html
File Edit View
<h1>Happiness Prediction</h1>
<form action="/predict" method="post">
  <label for="happiness_rank">Happiness Rank:</label>
  <input type="text" id="happiness_rank" name="Happiness Rank" required><br><br>

  <label for="lower_ci">Lower Confidence Interval:</label>
  <input type="text" id="lower_ci" name="Lower Confidence Interval" required><br><br>

  <label for="upper_ci">Upper Confidence Interval:</label>
  <input type="text" id="upper_ci" name="Upper Confidence Interval" required><br><br>

  <label for="gdp_per_capita">GDP per Capita:</label>
  <input type="text" id="gdp_per_capita" name="Economy (GDP per Capita)" required><br><br>

  <label for="family">Family:</label>
  <input type="text" id="family" name="Family" required><br><br>

  <label for="life_expectancy">Life Expectancy:</label>
  <input type="text" id="life_expectancy" name="Health (Life Expectancy)" required><br><br>

  <label for="freedom">Freedom:</label>
  <input type="text" id="freedom" name="Freedom" required><br><br>

  <label for="government_corruption">Trust (Government Corruption):</label>
  <input type="text" id="government_corruption" name="Trust (Government Corruption)" required><br><br>

  <label for="generosity">Generosity:</label>
  <input type="text" id="generosity" name="Generosity" required><br><br>

  <label for="dystopia_residual">Dystopia Residual:</label>
  <input type="text" id="dystopia_residual" name="Dystopia Residual" required><br><br>

  <input type="submit" value="Predict">
</form>

{% if prediction_text %}
<h2>{{ prediction_text }}</h2>
{% endif %}
</body>
</html>
```

After preparing all files needed, I will start my deploying code on Spyder using Flask and Pickle.

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

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

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

@app.route('/predict', methods=['POST'])
def predict():
    happiness_rank = float(request.form['Happiness Rank'])
    lower_ci = float(request.form['Lower Confidence Interval'])
    upper_ci = float(request.form['Upper Confidence Interval'])
    gdp_per_capita = float(request.form['Economy (GDP per Capita)'])
    family = float(request.form['Family'])
    life_expectancy = float(request.form['Health (Life Expectancy)'])
    freedom = float(request.form['Freedom'])
    government_corruption = float(request.form['Trust (Government Corruption)'])
    generosity = float(request.form['Generosity'])
    dystopia_residual = float(request.form['Dystopia Residual'])

    final_features = [[happiness_rank, lower_ci, upper_ci, gdp_per_capita, family, life_expectancy, freedom, government_corruption, generosity, dystopia_residual]]
    prediction = model.predict(final_features)

    output = round(prediction[0], 2)

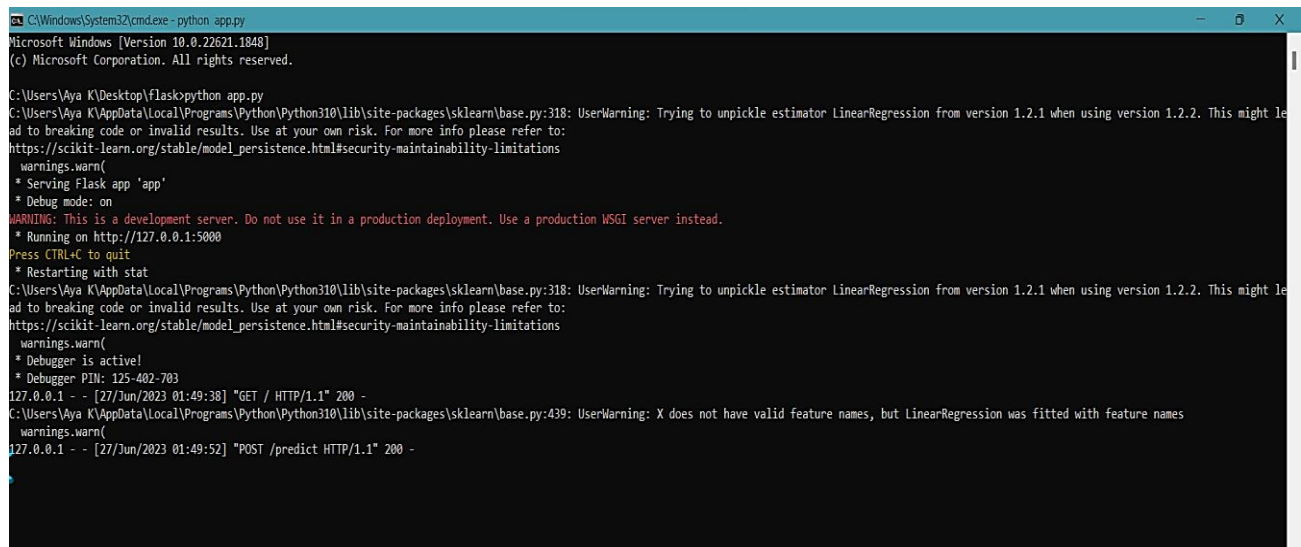
    return render_template('index.html', prediction_text='Predicted Happiness Score: {}'.format(output))

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

Using the command prompt for my specified directory, I will get the HTTP access to my trained web app.

As per the picture below, I will copy this link to my Chrome browser to access the web app.

<http://127.0.0.1:5000/>



```
C:\Windows\System32\cmd.exe - python app.py
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aya K\Desktop\flask>python app.py
C:\Users\Aya K\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:318: UserWarning: Trying to unpickle estimator LinearRegression from version 1.2.1 when using version 1.2.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
C:\Users\Aya K\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:318: UserWarning: Trying to unpickle estimator LinearRegression from version 1.2.1 when using version 1.2.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
* Debugger is active!
* Debugger PIN: 125-402-703
127.0.0.1 - - [27/Jun/2023 01:49:38] "GET / HTTP/1.1" 200 -
C:\Users\Aya K\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
127.0.0.1 - - [27/Jun/2023 01:49:52] "POST /predict HTTP/1.1" 200 -
```


This is the displayed result upon access to the link:

Happiness Prediction

Happiness Rank:

Lower Confidence Interval:

Upper Confidence Interval:

GDP per Capita:

Family:

Life Expectancy:

Freedom:

Trust (Government Corruption):

Generosity:

Dystopia Residual:

I tried to test the accuracy of my web app results by using the data from Denmark.

My web app shows high accuracy of predicted results.

Happiness Prediction

Happiness Rank:

Lower Confidence Interval:

Upper Confidence Interval:

GDP per Capita:

Family:

Life Expectancy:

Freedom:

Trust (Government Corruption):

Generosity:

Dystopia Residual:

Predicted Happiness Score: 7.53

After using Flask for deployment, I will use a cloud-based app, [Heroku](#), to undergo another deployment.

I will need 3 essential files to run this operation:

requirements.txt: lists all required libraries or buildpacks needed to be installed in Python.

Procfile: refers to my web app module and name

runtime.txt (optional): lists my installed Python version

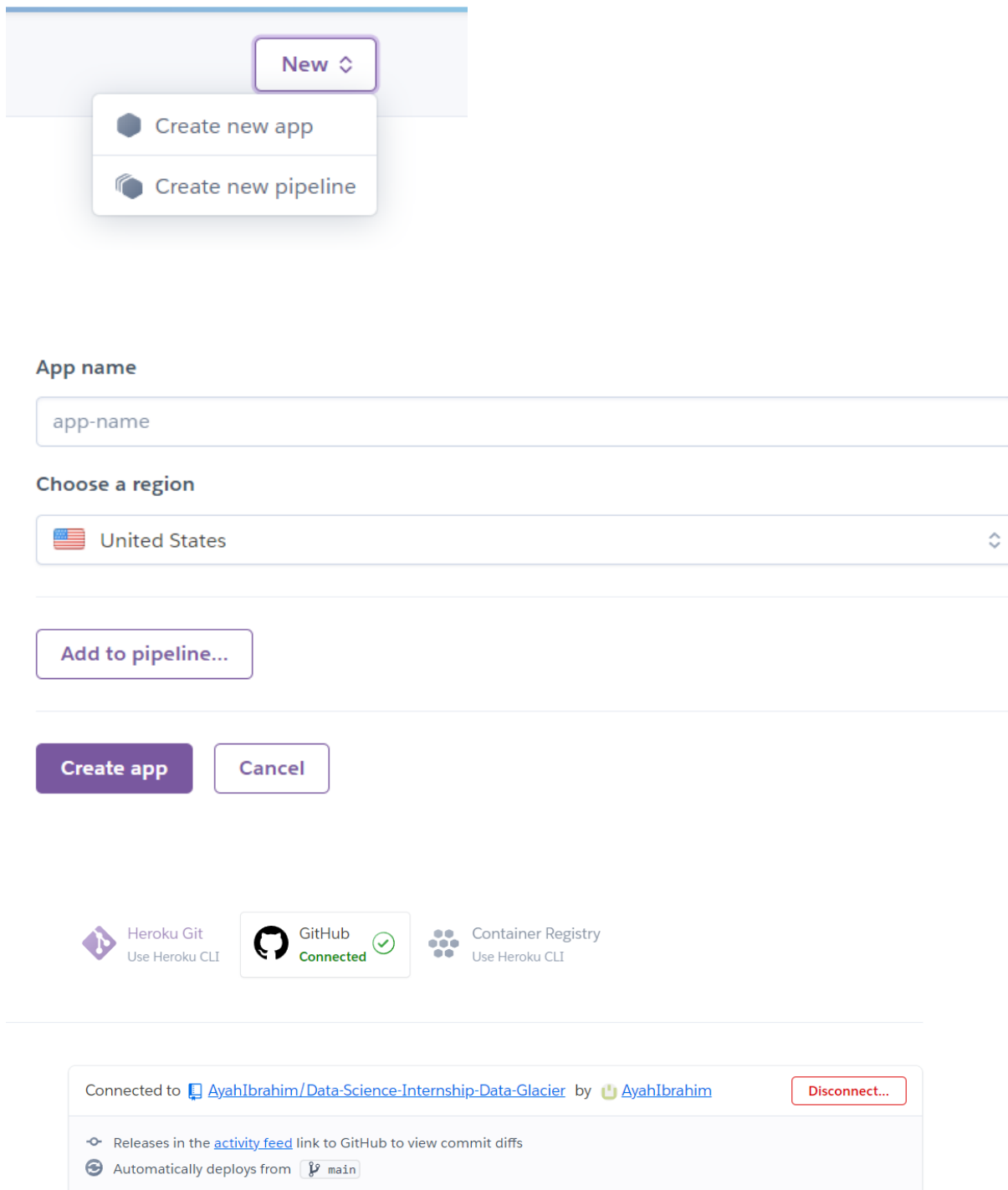
```
Flask==1.1.1
gunicorn==19.9.0
itsdangerous==1.1.0
Jinja2==2.10.1
MarkupSafe==1.1.1
Werkzeug==0.15.5
numpy>=1.9.2
scipy>=0.15.1
scikit-learn>=0.18
matplotlib>=1.4.3
pandas>=0.19
```

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

```
python-3.10.10
```

I will then navigate to [Heroku.com](https://heroku.com) to create an account.

I will then click on create a new app, name it, and connect it to GitHub with my designated code repository.



The image shows the Heroku 'New' dropdown menu and the 'Create new app' form. The 'New' button is highlighted with a purple border. The dropdown menu shows two options: 'Create new app' (selected) and 'Create new pipeline'. Below the dropdown, the 'App name' field contains 'app-name'. The 'Choose a region' dropdown shows 'United States' with a US flag icon. Below this is an 'Add to pipeline...' button. At the bottom of the form are two buttons: 'Create app' (purple) and 'Cancel' (white with purple border). Below the form, there are three status indicators: 'Heroku Git Use Heroku CLI', 'GitHub Connected' (with a green checkmark), and 'Container Registry Use Heroku CLI'. At the very bottom, a connected state bar shows 'Connected to AyahIbrahim/Data-Science-Internship-Data-Glacier by AyahIbrahim' with a 'Disconnect...' button. Below this bar, it says 'Releases in the activity feed link to GitHub to view commit diffs' and 'Automatically deploys from main'.

New ▾

- Create new app
- Create new pipeline

App name

app-name

Choose a region

United States ▾

Add to pipeline...

Create app Cancel

Heroku Git Use Heroku CLI

GitHub Connected

Container Registry Use Heroku CLI

Connected to [AyahIbrahim/Data-Science-Internship-Data-Glacier](#) by [AyahIbrahim](#) Disconnect...

Releases in the [activity feed](#) link to GitHub to view commit diffs


Automatically deploys from [main](#)

I will deploy the branch and wait for it to run on its own:

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

This is what the run code will look like:

Activity Feed > Build Log ID 63a838b0-9b02-4afb-89cd-a650ea6de154

```
-----> Building on the Heroku-22 stack
-----> Using buildpack: heroku/python
-----> Python app detected
-----> Using Python version specified in runtime.txt
!
!   A Python security update is available! Upgrade as soon as possible to: python-3.10.12
!   See: https://devcenter.heroku.com/articles/python-runtimes
!
-----> No change in requirements detected, installing from cache
-----> Using cached install of python-3.10.10
-----> Installing pip 23.1.2, setuptools 67.8.0 and wheel 0.40.0
-----> Installing SQLite3
-----> Installing requirements with pip
-----> Discovering process types
Procfile declares types -> web
-----> Compressing...
Done: 136.2M
-----> Launching...
Build finished
```

It should end with this line saying it was successfully
deployed to Heroku:

```
Done: 136.2M
-----> Launching...
Released v4
https://heroku-deployment1-fc3b9f175fcd.herokuapp.com/ deployed to Heroku
```

You can also double check on your GitHub repository by making sure this appears on the far right side whenever you access the repo:

Environments 1



heroku-deployment1

Active