NAME: MOHAMMAD SHEHZAR KHAN

BATCH CODE: LISUM26

SUBMISSION DATE: 04-NOV-2023

SUBMITTED TO: DATA GLACIER

TOPIC: MACHINE LEARNING MODEL DEPLOYMENT ON HEROKU

Dataset Info: The dataset consists of two features, YearsExperience(Years of Experience) and Salary. We will use years of experience to predict salary.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     float64
dtypes: float64(2)
memory usage: 720.0 bytes
   YearsExperience   Salary
0              1.2  39344.0
1              1.4  46206.0
2              1.6  37732.0
3              2.1  43526.0
4              2.3  39892.0
[[72098.0155738]]
```

STEP 1: We created a model2.py file in which we used a salalry dataset to train our model. This model uses years of experience to predict the expected salary.

STEP 2: Then we saved the model using pickle into a file named model2.pkl

model2.py | app.py | index.html

```python
# importing libraries
import pandas as pd
import pickle
import warnings
warnings.filterwarnings("ignore")


dataset = pd.read_csv("Salary_dataset.csv", index_col=(0))
dataset.info()
print(dataset.head())

# X is the independent variable and y is the dependent variable
X = dataset[["YearsExperience"]]
y = dataset[["Salary"]]


from sklearn.linear_model import LinearRegression
regressor = LinearRegression()

#fitting the linear regression model
regressor.fit(X, y)

#saving the linear regression model to disk
pickle.dump(regressor, open('model2.pkl', 'wb'))

#loading model to compare the results
model = pickle.load(open('model2.pkl', 'rb'))

#predicting salary for 5 years of experience
print(model.predict([[5]]))

```

STEP 3: Then we create a new file app.py which loads the model using the saved pickle file.

We start by creating a flask application. Then we load the model.

Then we endpoints for homepage and prediction. The '/predict' route is used to make prediction when we deploy our model on Heroku. The '/api_predict/' route is used when we use rest api. We can use postman to get prediction and test our model.

C:\Users\Shez Khan\Desktop\Data Glacier Internship\Week5\app.py

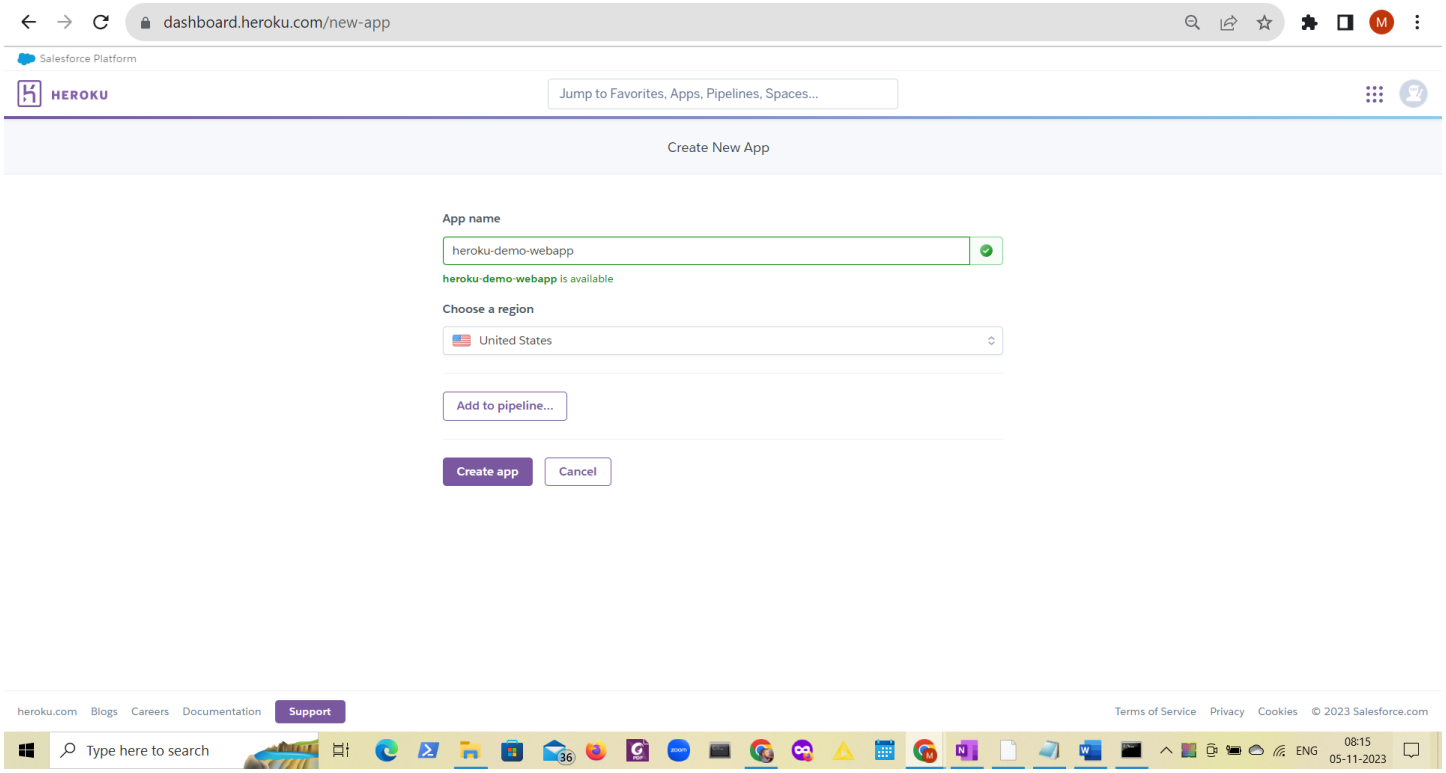
model2.py | app.py | index.html

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

#creating an application by calling flask
app = Flask(__name__)

#loading the model already saved
model = pickle.load(open('model2.pkl', 'rb'))

#creating endpoints for homepage (default) and prediction
@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI

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

    prediction = float(prediction[0])

    output = round(prediction, 2)

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

@app.route('/api_predict/')
def price_predict():
    years = request.args.get('YearsExperience')

    test_df = pd.DataFrame({'YearsExperience':[years]})

    pred_salary = model.predict(test_df)
    return jsonify({'Your salary should be $': str(np.round(pred_salary[0, 0],2))})

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

STEP 4: Once our app.py file is ready we can save it and move to model deployment.

We can go to Heroku.com and signup (for first time users).

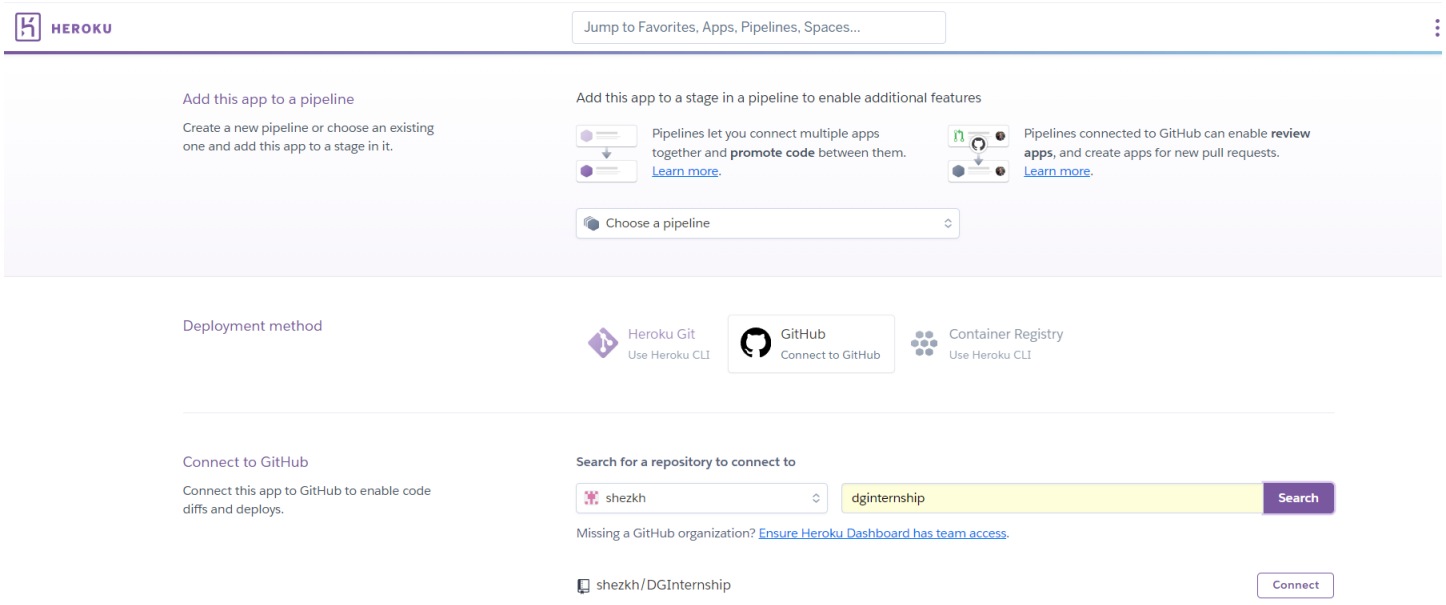
Then we click on “Create new app” on the Heroku website. Then give any unique app name and choose appropriate region and click “create app”. Your app is now created.



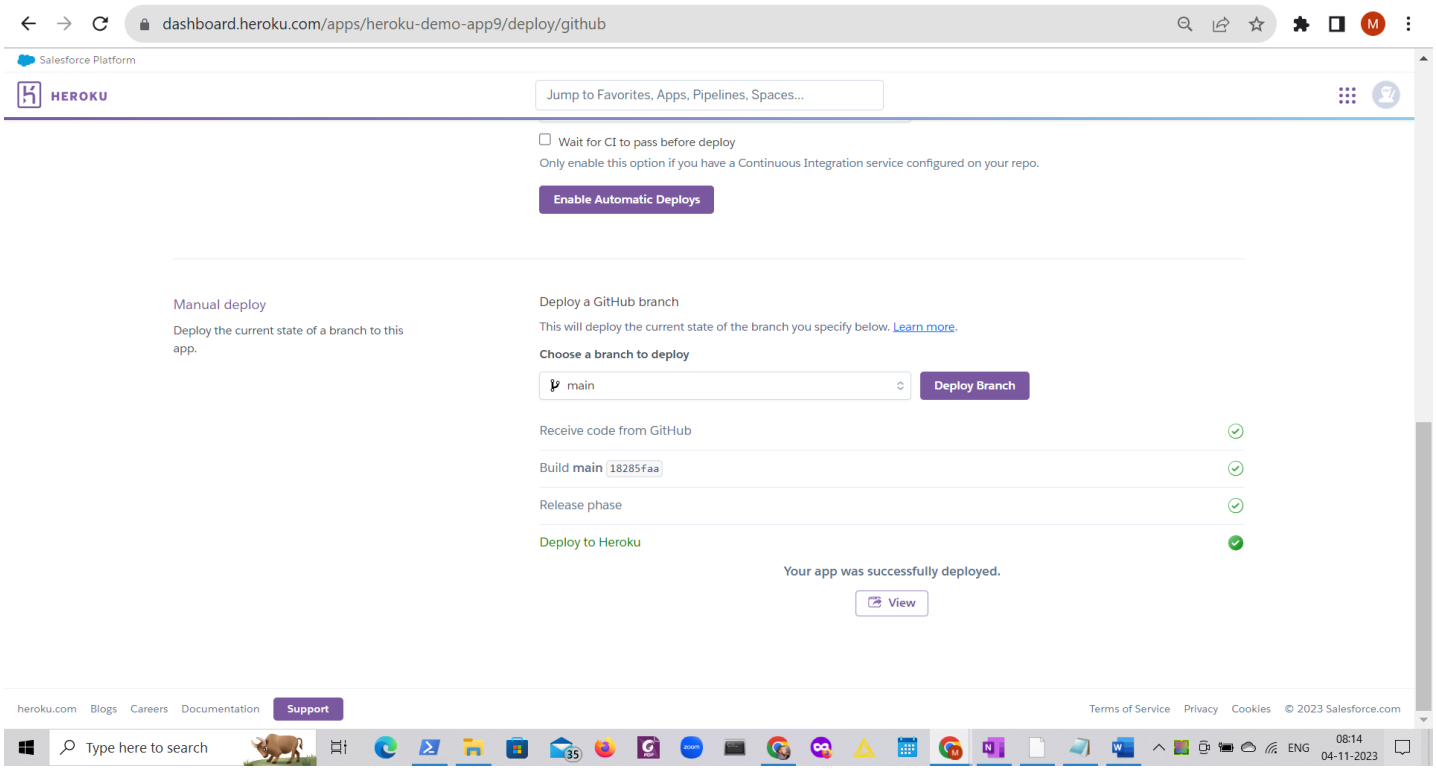
Now we need to go to out GitHub account and push all our necessary files on a GitHub repository.

Now we need to add our GitHub account to Heroku. In the webapp, scroll down to Deployment method, click on GitHub.

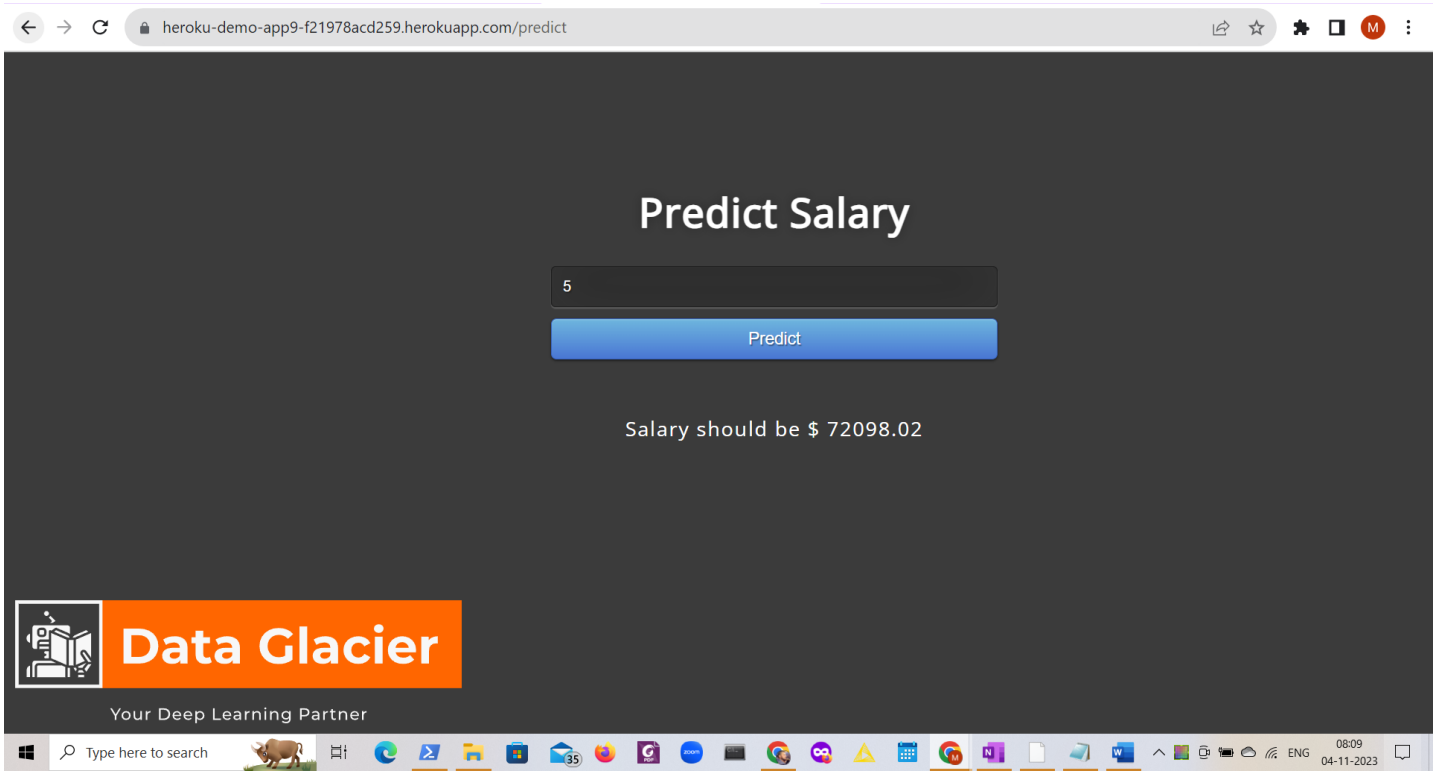
Search for the repository where you have all the required files. Click connect.



Now, scroll down to Manual deploy and click deploy branch. Wait for some time and you will see the message “Your app was successfully deployed.”



Now click on view button and you will be redirected to a URL which can be accessed from anywhere and a page is opened where you can input years of experience to get a estimated salary.



You can also test your model by API call. For this you need to use terminal and run app.py file in your system.

This will deploy your model locally on your machine. You will get a URL which can be used to access the model and test it.

```
C:\Windows\System32\cmd.exe - python  app.py
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Shez  Khan\Desktop\Data Glacier Internship\Week5>venv\Scripts\activate

(venv) C:\Users\Shez  Khan\Desktop\Data Glacier Internship\Week5>python app.py
C:\Users\Shez  Khan\Desktop\Data Glacier Internship\Week5\venv\Lib\site-packages\sklearn\base.py:348: InconsistentVersio
nWarning: Trying to unpickle estimator LinearRegression from version 1.2.2 when using version 1.3.2. 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\Shez  Khan\Desktop\Data Glacier Internship\Week5\venv\Lib\site-packages\sklearn\base.py:348: InconsistentVersio
nWarning: Trying to unpickle estimator LinearRegression from version 1.2.2 when using version 1.3.2. 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: 734-343-466
```

For accessing and testing the model we are using POSTMAN.

Firstly, we open postman.co website. Create an account. Then click on New Request.

Now a different page will open where you can add the local URL that you got in the terminal followed by"/api_predict/"(the route that we gave in the app.py file).

Then in the key, we will give YearsExperience and its value for which you want to test model prediction. Click send.

Then, you will get a response in the below box as in the screenshot.