

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 salary 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 x app.py x index.html x
1  # importing libraries
2  import pandas as pd
3  import pickle
4  import warnings
5  warnings.filterwarnings("ignore")
6
7
8  dataset = pd.read_csv("Salary_dataset.csv", index_col=(0))
9  dataset.info()
10 print(dataset.head())
11
12 # X is the independent variable and y is the dependent variable
13 X = dataset[["YearsExperience"]]
14 y = dataset[["Salary"]]
15
16
17 from sklearn.linear_model import LinearRegression
18 regressor = LinearRegression()
19
20 #fitting the linear regression model
21 regressor.fit(X, y)
22
23 #saving the linear regression model to disk
24 pickle.dump(regressor, open('model2.pkl', 'wb'))
25
26 #loading model to compare the results
27 model = pickle.load(open('model2.pkl', 'rb'))
28
29 #predicting salary for 5 years of experience
30 print(model.predict([[5]]))
31
```

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 X app.py X index.html X

1  import numpy as np
2  import pandas as pd
3  from flask import Flask, request, render_template, jsonify
4  import pickle
5
6  #creating an application by calling flask
7  app = Flask(__name__)
8
9  #loading the model already saved
10 model = pickle.load(open('model2.pkl', 'rb'))
11
12 #creating endpoints for homepage (default) and prediction
13 @app.route('/')
14 def home():
15     return render_template('index.html')
16
17 @app.route('/predict', methods=['POST'])
18 def predict():
19     ...
20     For rendering results on HTML GUI
21     ...
22
23     int_features = [int(x) for x in request.form.values()]
24     final_features = [np.array(int_features)]
25     prediction = model.predict(final_features)
26
27     prediction = float(prediction[0])
28
29     output = round(prediction, 2)
30
31     return render_template('index.html', prediction_text='Salary should be $ {}'.format(output))
32
33 @app.route('/api_predict/')
34 def price_predict():
35     years = request.args.get('YearsExperience')
36
37     test_df = pd.DataFrame({'YearsExperience': [years]})
38
39     pred_salary = model.predict(test_df)
40     return jsonify({'Your salary should be $': str(np.round(pred_salary[0, 0], 2))})
41
42 if __name__ == "__main__":
43     app.run(debug=True)
44
```

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.

← → ↻ dashboard.heroku.com/new-app

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Create New App

App name

heroku-demo-webapp

heroku-demo-webapp is available

Choose a region

United States

Add to pipeline...

Create app Cancel

Now we need to go to our 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.

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Add this app to a pipeline

Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and promote code between them. [Learn more.](#)

Pipelines connected to GitHub can enable review apps, and create apps for new pull requests. [Learn more.](#)

Choose a pipeline

Deployment method

Heroku Git Use Heroku CLI

GitHub Connect to GitHub

Container Registry Use Heroku CLI

Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.

Search for a repository to connect to

shezkh dginternship Search

Missing a GitHub organization? [Ensure Heroku Dashboard has team access.](#)

shezkh/DGInternship 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.”

The screenshot shows the Heroku dashboard for an application named 'heroku-demo-app9'. The browser address bar shows the URL 'dashboard.heroku.com/apps/heroku-demo-app9/deploy/github'. The page has a purple header with the Heroku logo and a search bar. Below the header, there's a section for 'Manual deploy' with a description: 'Deploy the current state of a branch to this app.' To the right, there's a 'Deploy a GitHub branch' section. It includes a checkbox for 'Wait for CI to pass before deploy' (unchecked) and a button 'Enable Automatic Deploys'. Below this, there's a 'Choose a branch to deploy' section with a dropdown menu showing 'main' and a 'Deploy Branch' button. A progress bar shows the deployment steps: 'Receive code from GitHub', 'Build main 18285faa', 'Release phase', and 'Deploy to Heroku', all with green checkmarks. A message states 'Your app was successfully deployed.' with a 'View' button. The footer includes links for 'heroku.com', 'Blogs', 'Careers', 'Documentation', 'Support', 'Terms of Service', 'Privacy', 'Cookies', and '© 2023 Salesforce.com'. The Windows taskbar at the bottom shows various application icons and the system clock indicating 08:14 on 04-11-2023.

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.

The screenshot shows a web application titled 'Predict Salary'. It has a dark grey background. At the top, the title 'Predict Salary' is displayed in white. Below the title, there's a text input field containing the number '5'. Underneath the input field is a blue button labeled 'Predict'. Below the button, the text 'Salary should be \$ 72098.02' is displayed. At the bottom of the page, there's a logo for 'Data Glacier' with the tagline 'Your Deep Learning Partner'. The browser address bar shows the URL 'heroku-demo-app9-f21978acd259.herokuapp.com/predict'. The Windows taskbar at the bottom shows various application icons and the system clock indicating 08:09 on 04-11-2023.

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: InconsistentVersionWarning: 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: InconsistentVersionWarning: 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.

