

Name: Guangyan Huang

Batch code: LISUM 15

Submission date: Dec 5th 2022

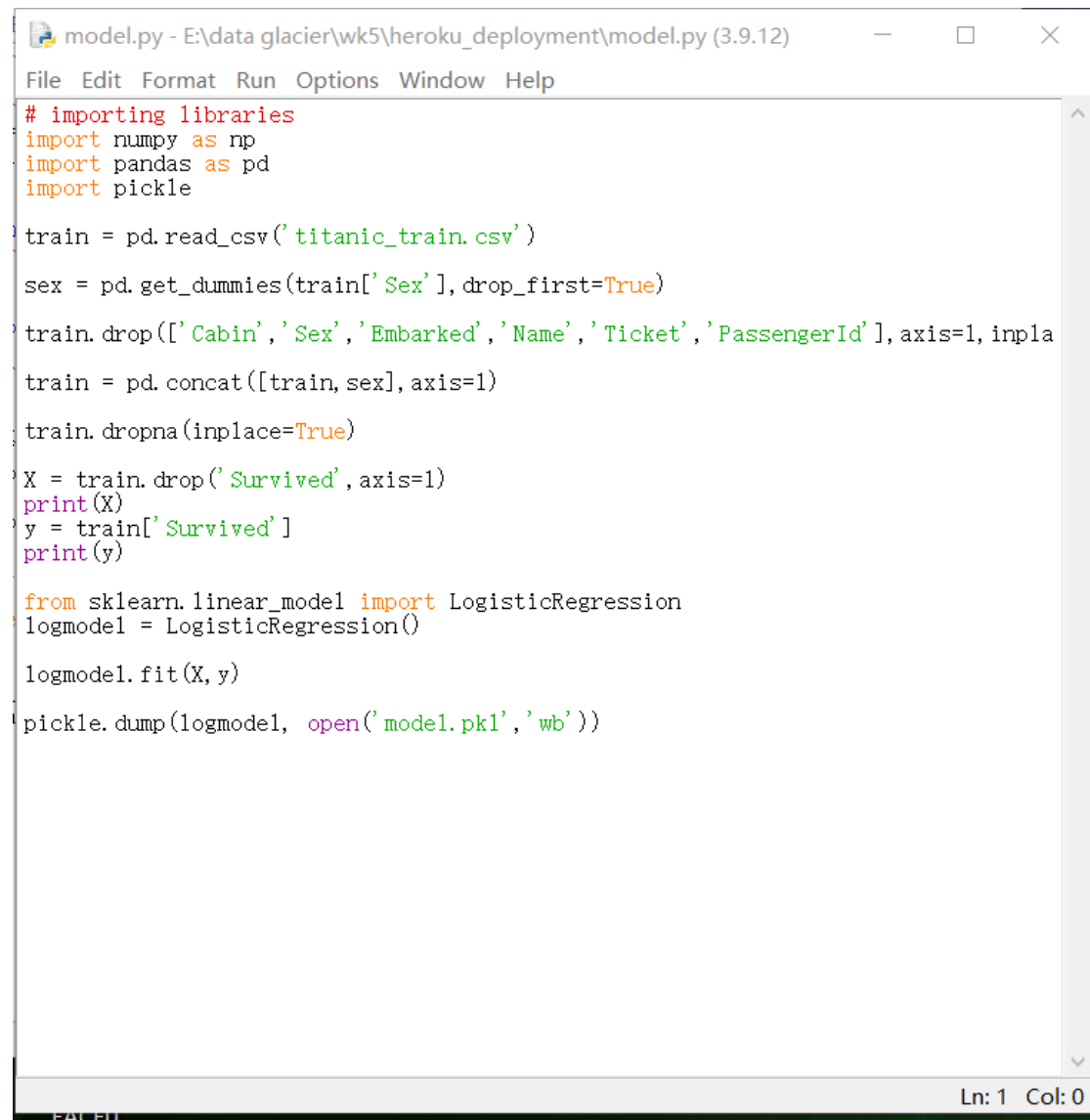
Submitted to: week 5 deployment on Heroku cloud

Model description:

This file records snapshots of deploying a logistic regression model on flask. The dataset is the record of personal information for passengers of the Titanic, as well as whether they survived through the hazard. For this model, by entering user's personal information, the model will predict whether the user can survive from the Titanic shipwreck.

Step 1:

Firstly we need to train the model by our dataset titanic_train.csv. Here is the model.py function which uses logistic regression.



```
model.py - E:\data glacier\wk5\heroku_deployment\model.py (3.9.12)
File Edit Format Run Options Window Help
# importing libraries
import numpy as np
import pandas as pd
import pickle

train = pd.read_csv('titanic_train.csv')
sex = pd.get_dummies(train['Sex'], drop_first=True)
train.drop(['Cabin', 'Sex', 'Embarked', 'Name', 'Ticket', 'PassengerId'], axis=1, inplace=True)
train = pd.concat([train, sex], axis=1)
train.dropna(inplace=True)
X = train.drop('Survived', axis=1)
print(X)
y = train['Survived']
print(y)

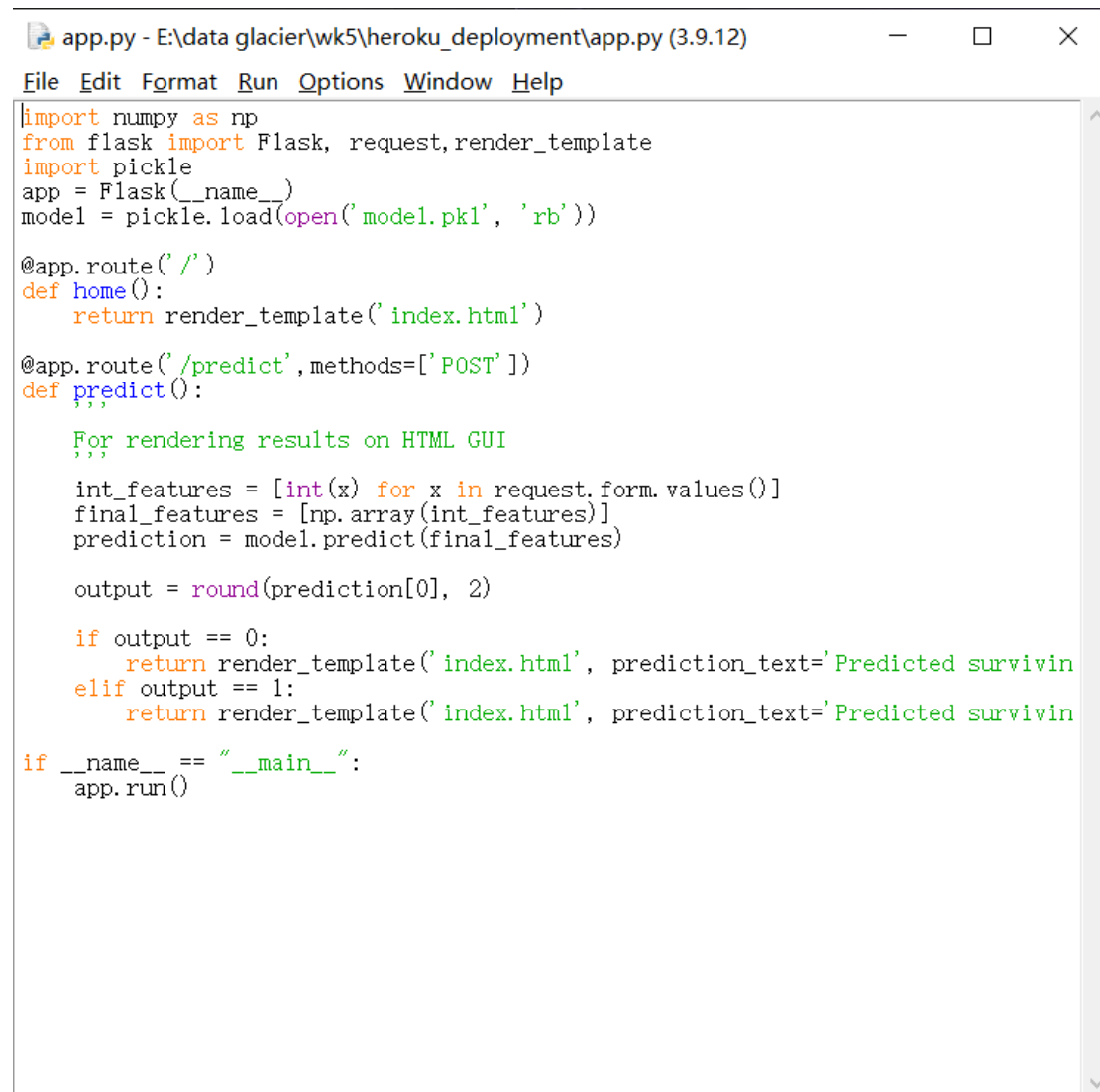
from sklearn.linear_model import LogisticRegression
logmodel = LogisticRegression()

logmodel.fit(X, y)

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

Ln: 1 Col: 0

Step 2: after we are done with the model training, we create the app that enables us to actually run the model and make predictions. Other files, including the folder static and template, are from the online google drive.



```
app.py - E:\data glacier\wk5\heroku_deployment\app.py (3.9.12)
File Edit Format Run Options Window Help
import numpy as np
from flask import Flask, request, 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():
    """ 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)

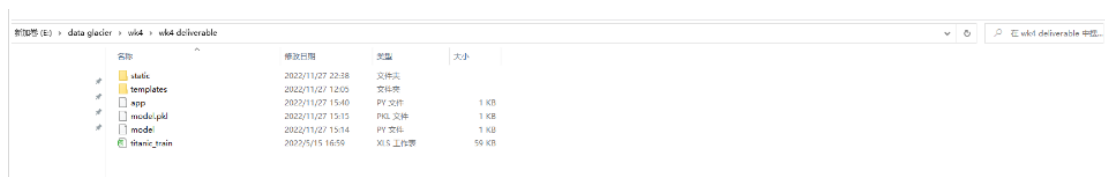
    output = round(prediction[0], 2)

    if output == 0:
        return render_template('index.html', prediction_text='Predicted survivin')
    elif output == 1:
        return render_template('index.html', prediction_text='Predicted survivin')

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

Step 3: After all the files are ready, we start by navigating to the local directory:

```
C:\Users\HGY>E:
E:>>cd E:\data glacier\wk4\wk4 deliverable
E:\data glacier\wk4\wk4 deliverable>
```



File Explorer window showing the directory 'E:\data glacier\wk4\wk4 deliverable'. The table below lists the files and folders in this directory.

名称	修改日期	类型	大小
static	2022/11/27 22:38	文件夹	
templates	2022/11/27 12:05	文件夹	
app	2022/11/27 15:40	Python 文件	1 KB
model.pkl	2022/11/27 15:15	Pick 文件	1 KB
model	2022/11/27 15:14	Python 文件	1 KB
train_train	2022/11/15 16:59	XML 文件	56 KB

Step 4: Then we run the app.py file:


```
E:\data glacier\wk4\wk4 deliverable>python app.py
E:\python\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator LogisticRegression from version 1.1.1 when using version 1.1.3. This might lead to breaking code or invalid results. Use at your own risk. For more information please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
* Serving Flask app 'app'
* Debug mode: on
```

Step 5: The flask app is up and running, let's enter the homepage via Chrome:

```
E:\data glacier\wk4\wk4 deliverable>python app.py
E:\python\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator LogisticRegression from version 1.1.1 when using version 1.1.3. This might lead to breaking code or invalid results. Use at your own risk. For more information 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
```

Predict surviving status of Titanic passengers


Passenger Class
Age
Number of siblings
Number of parents and children
Fare price
Gender (male=1,1 for yes)
Predict

 **Data Glacier**
Your Deep Learning Partner

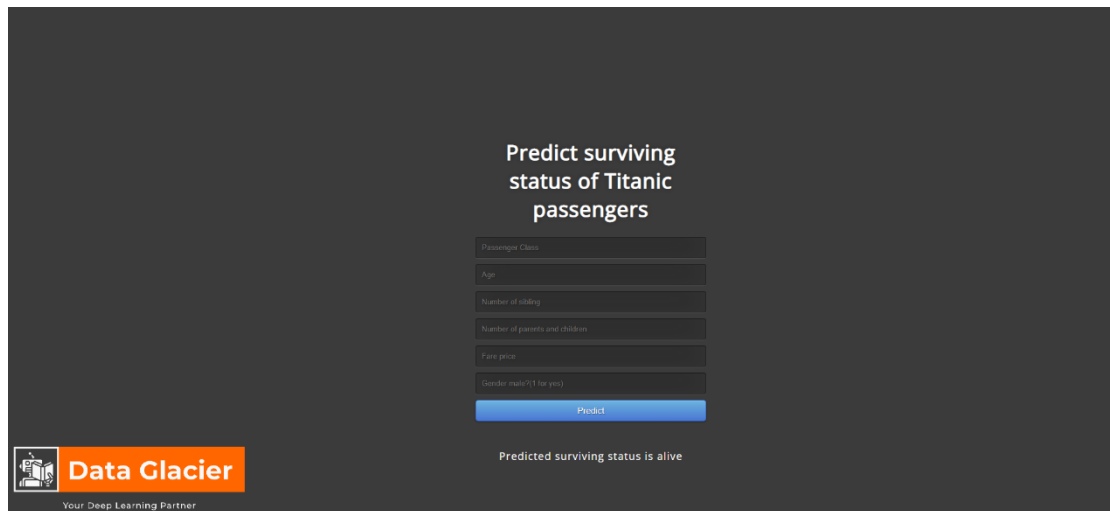
Step 6: Enter passenger class, age, number of siblings, number of parents and children, ticket fare as well as gender to the model. Then click predict.

Predict surviving status of Titanic passengers

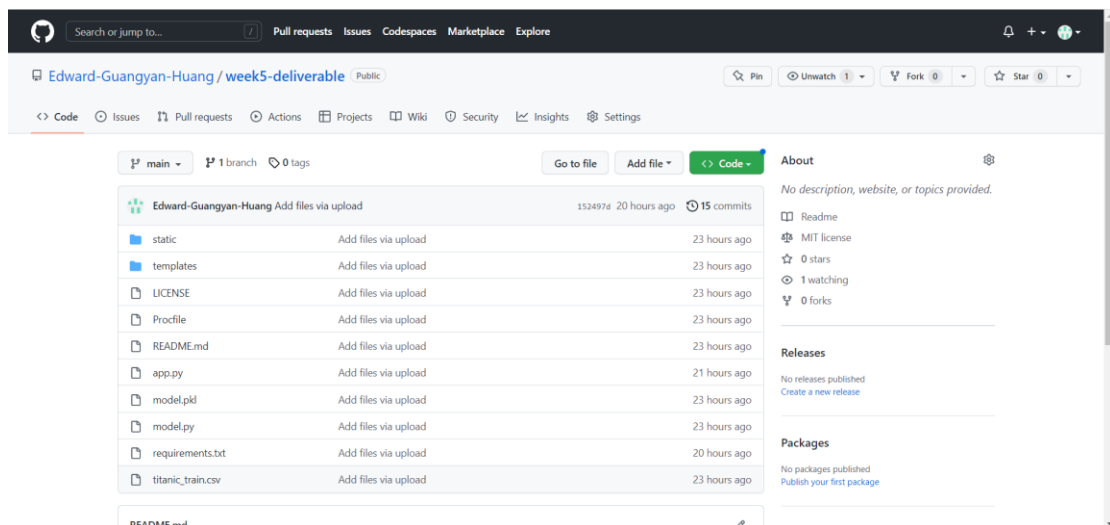
3
27
0
3
22
0
Predict

 **Data Glacier**
Your Deep Learning Partner

Step 7: The prediction outcome shows up on the screen.



Step 8: Given the file is good to run on our local device, we upload it to Heroku for cloud deployment. Before this we need to upload all relevant files to Github.



The file Procfile and requirements are from google drive with crucial information for deployment on Heroku.

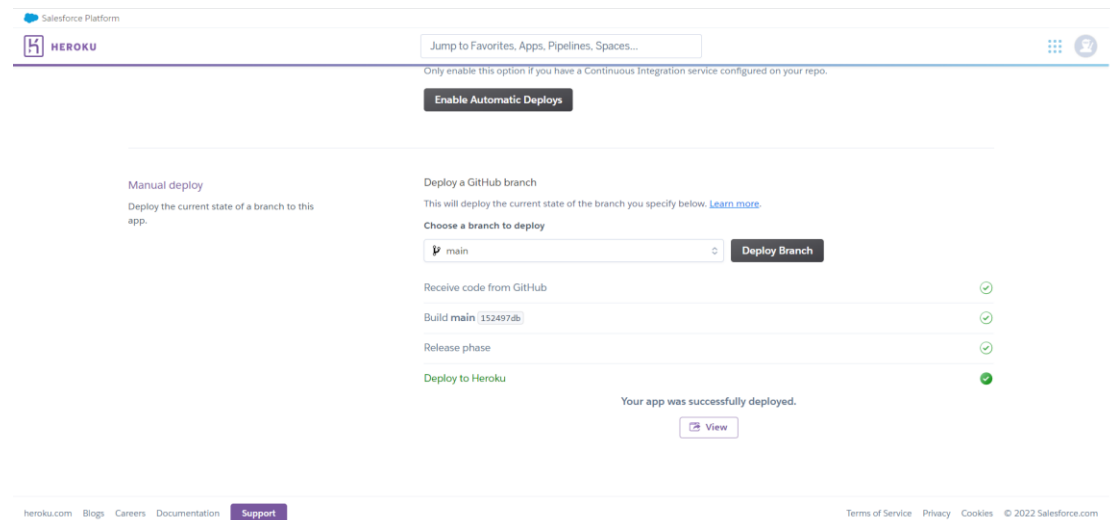
Step 9: After creating the Github repository, we create an app on Heroku and link this repository to the app.

The screenshot shows the Heroku dashboard interface. At the top, there's a navigation bar with the Heroku logo, a search bar, and a user profile icon. Below the navigation bar, a purple banner welcomes the user and provides a 'Dismiss' button. The main content area is divided into several sections:

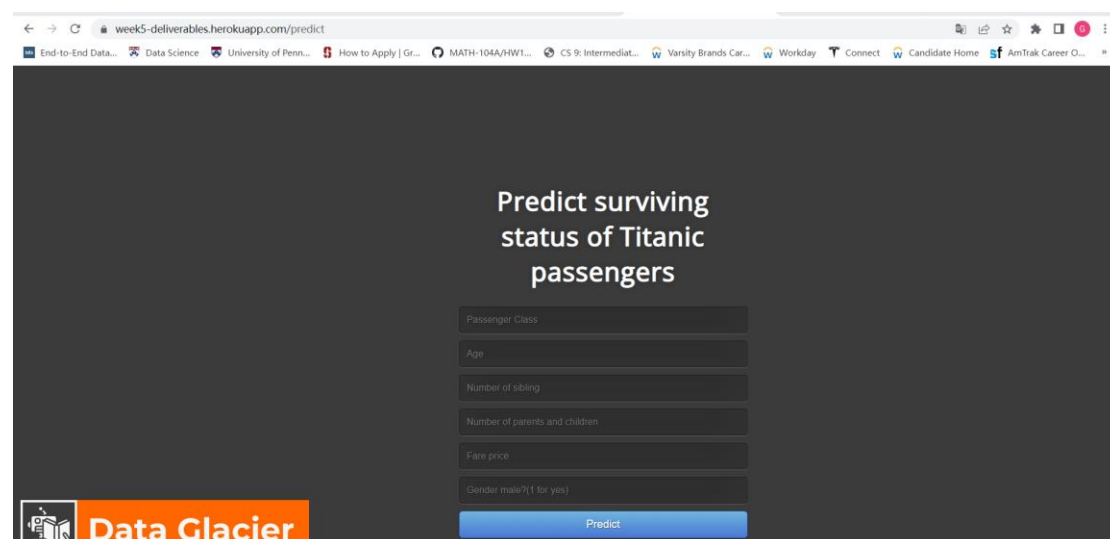
- Filter apps and pipelines:** A search bar with the text 'Filter apps and pipelines'.
- week5-deliverables:** A card showing the app name, its buildpacks (Python, heroku-22), and its region (United States).
- Add this app to a pipeline:** A section with instructions on how to add an app to a pipeline and a 'Choose a pipeline' dropdown menu.
- Deployment method:** A section showing three options: Heroku Git (Use Heroku CLI), GitHub (Connected), and Container Registry (Use Heroku CLI).
- App connected to GitHub:** A section showing the app is connected to the GitHub repository 'EdwardGuangyanHuang/week5-deliverable' by 'EdwardGuangyanHuang'. It includes a 'Disconnect...' button and a link to the activity feed.
- Automatic deploys:** A section with a tip: 'You can now change your main deploy branch from "master" to "main" for both manual and automatic deploys, please follow the instructions [here](#).'

The bottom of the page features a footer with links to heroku.com, Blogs, Careers, Documentation, and Support, as well as Terms of Service, Privacy, Cookies, and a copyright notice for 2022 Salesforce.com.

Step 10: at the ‘manual deploy’ part, click ‘Deploy branch’. After clicking ‘Deploy Branch’, wait a few minutes and the cloud app will be ready to view by clicking the ‘view’ button.



Step 11: after clicking ‘view’, the app we deployed locally will pop up, notice now its url is a url from Heroku, not 127.0.0.5000




Step 12: Put in data for the new prediction and click button 'predict', the result will show up.

Predict surviving status of Titanic passengers

3
27
0
3
22
0

Predict


 **Data Glacier**
Your Deep Learning Partner

Predict surviving status of Titanic passengers

Passenger Class
Age
Number of sibling
Number of parents and children
Fare price
Gender male(1 for yes)

Predict

Predicted surviving status is alive

 **Data Glacier**
Your Deep Learning Partner