Name: Yuchi Chen

Batch code: LISUM14

Submission date: 10/19/2022

Submitted to: https://github.com/ycchen00/DataGlacier-W5-Heroku_Cloud_API

Model train and save (model.py)

```
# Importing the libraries
import pickle
import numpy as np
import pandas as pd
from sklearn import preprocessing
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split


bank = pd.read_csv("bank_data_sk.csv")

nbank = bank.drop('CustomerID', axis=1)

nbank = nbank.dropna()

# one_hot_encoded=pd.get_dummies(nbank.State)
# nbank=pd.concat([nbank,one_hot_encoded],axis=1)

train_set, test_set = train_test_split(nbank, train_size=0.8)

cols = ['Age', 'Balance', 'IsActiveMember', 'CheckingAcct']
x_cols = ['Age', 'Balance', 'IsActiveMember']
y_col = 'CheckingAcct'

train_set = train_set[cols]
test_set = test_set[cols]

# nprmalization
ncolumns = ['Age', 'Balance']

scaler = preprocessing.MinMaxScaler()

nbank[ncolumns] = scaler.fit_transform(nbank[ncolumns])

X_train, X_test, y_train, y_test = train_test_split(nbank[x_cols],
nbank.CheckingAcct, train_size=0.8)

lr = LogisticRegression().fit(X_train, y_train)
```

```py
# Saving model to disk
pickle.dump(lr, open('model.pkl', 'wb'))
pickle.dump(scaler, open('scaler.pkl', 'wb'))
```

Deploy the model on flask (app.py)

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

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))
scaler = pickle.load(open('scaler.pkl', 'rb'))
@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    features = [float(x) for x in request.form.values()]
    test_df = pd.DataFrame([features], columns=['Age', 'Balance',
'IsActiveMember'])
    # final_features = [np.array(int_features)]
    # age = int(request.args.get("Age"))
    # balance = float(request.args.get("Balance"))
    # isActiveMember = int(request.args.get("IsActiveMember"))
    # test_df = pd.DataFrame({'Age': [age], 'Balance': [balance],
'IsActiveMember': [isActiveMember]})

    ncolumns = ['Age', 'Balance']
    test_df[ncolumns] = scaler.fit_transform(test_df[ncolumns])

    prediction = model.predict(test_df)

    prediction_text = "This person will sign up for a checking account" if
prediction == 1 else "This person will not sign up for a checking account"

    return render_template('index.html', prediction_text=prediction_text)

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

index.html

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="UTF-8">
  <title>ML API</title>

       <link href='https://fonts.googleapis.com/css?family=Pacifico'
                      rel='stylesheet' type='text/css'>

<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
                               type='text/css'>

       <link href='https://fonts.googleapis.com/css?family=Hind:300'
                      rel='stylesheet' type='text/css'>

                <link href='https://fonts.googleapis.com/css?
     family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css')
                                    }}">

</head>

<body>
  <div class="login">
    <h1>Predict a customer will sign up a checking account</h1>

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}" method="post">

<input type="text" name="Age" placeholder="Age" required="required" />

<input type="text" name="Balance" placeholder="Balance" required="required"
                                     />

 <input type="text" name="IsActiveMember" placeholder="Is active member or
                      not (1/0)" required="required" />

      <button type="submit" class="btn btn-primary btn-block btn-large">
Predict</button>
    </form>

    <br>
    <br>
    {{ prediction_text }}

  </div>
```

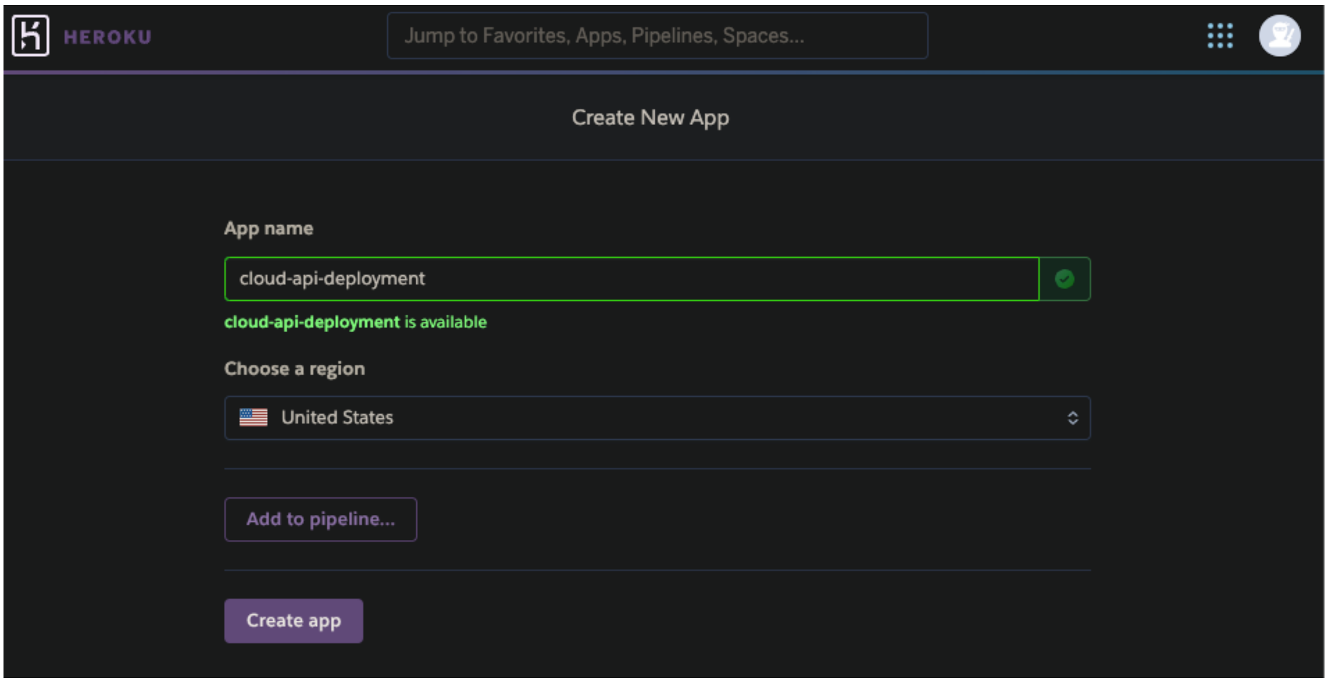
```html
    <img src="/static/images/Original.svg" style="width: 400px;position:
                              absolute;bottom: 10px;left: 10px;"
                                       alt="Company Logo" />

</body>

</html>
```

Heroku

Create new app



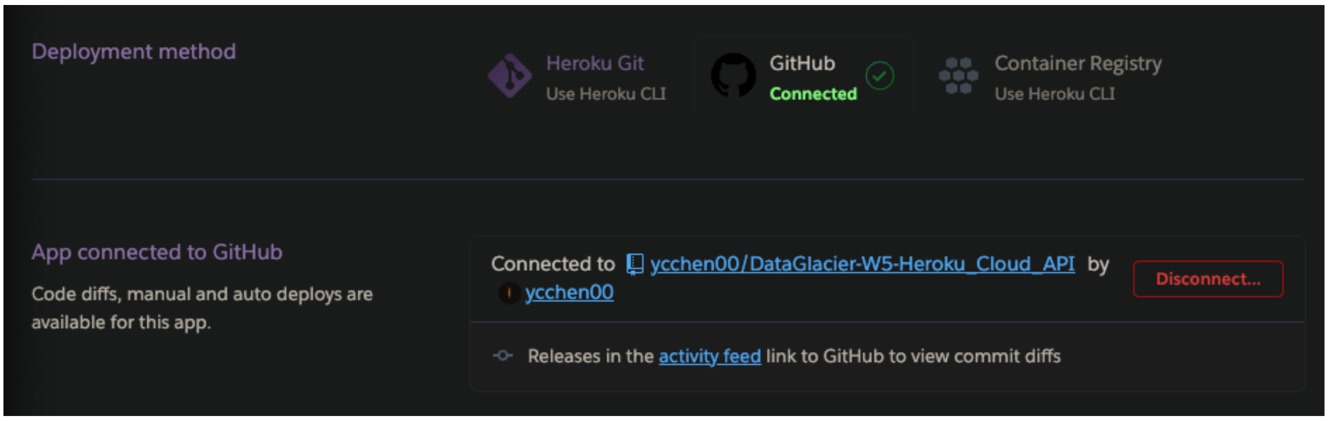
Create New App

App name: cloud-api-deployment

cloud-api-deployment is available

Choose a region: United States

Connect to Github



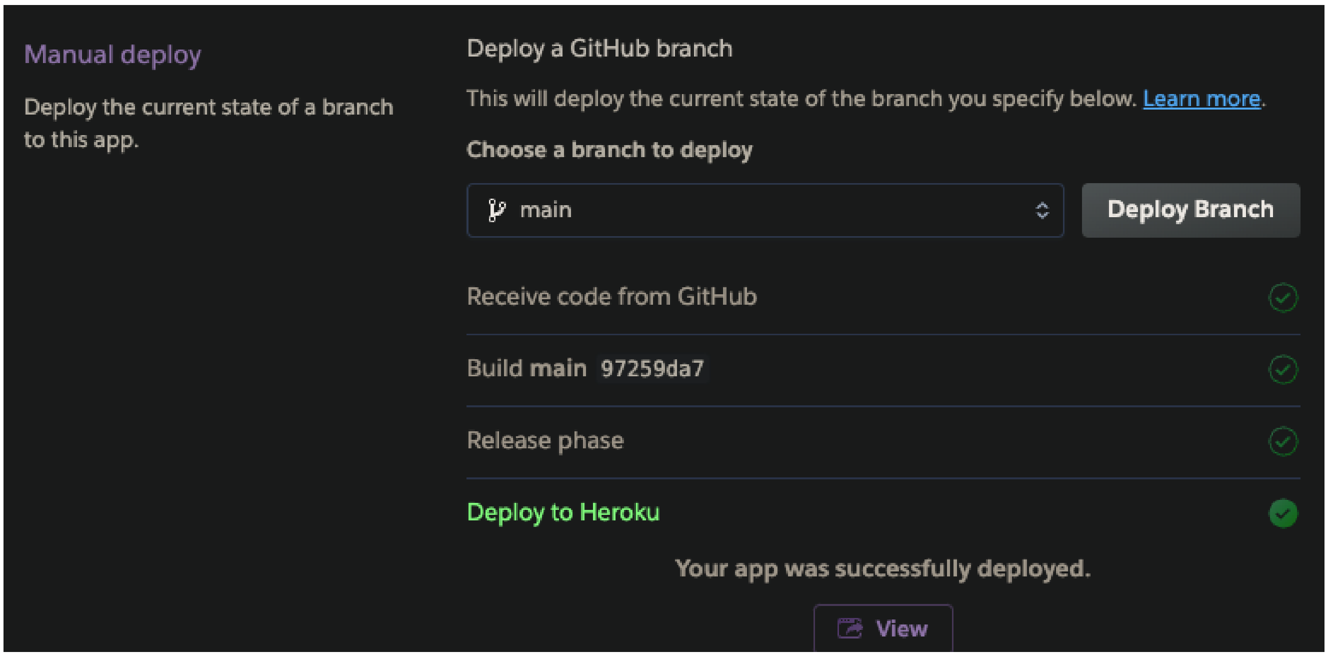
App connected to GitHub

Code diffs, manual and auto deploys are available for this app.

Connected to ycchen00/DataGlacier-W5-Heroku_Cloud_API by ycchen00

Releases in the activity feed link to GitHub to view commit diffs

Deploy to Heroku



Manual deploy

Deploy the current state of a branch to this app.

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

Receive code from GitHub ✓

Build main 97259da7 ✓

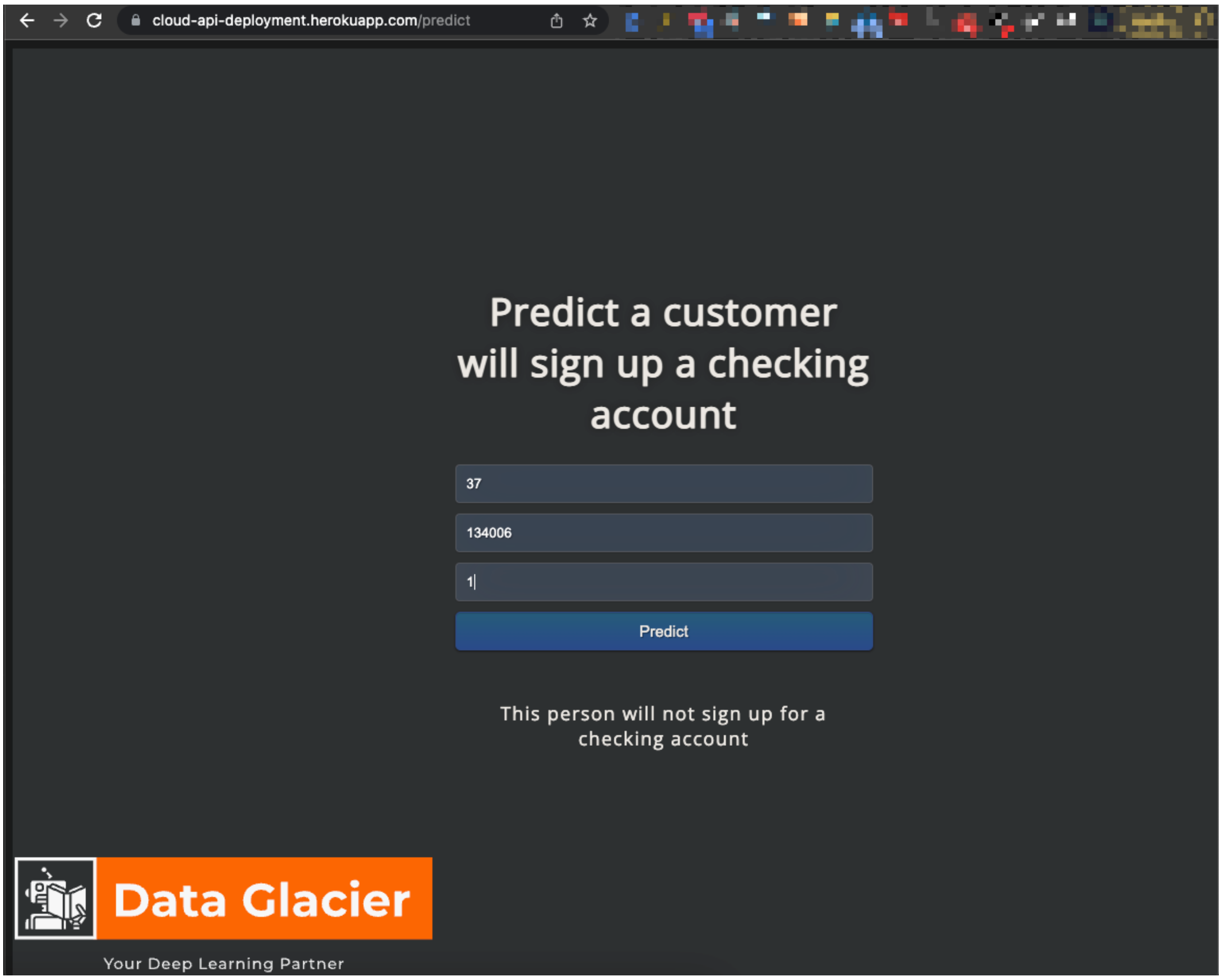
Release phase ✓

Deploy to Heroku ✓

Your app was successfully deployed.

View

Open APP



cloud-api-deployment.herokuapp.com/predict

Predict a customer will sign up a checking account

37

134006

1

Predict

This person will not sign up for a checking account

Data Glacier

Your Deep Learning Partner