Cloud and API Deployment

Name: Anastasiia Trofymova
Batch code: LISUM28
Submission date: 05 January 2023
Submitted to: Data Glacier

1. Select any toy data (simple data).

iris.data.csv (4.55 kB)

Detail Compact Column

# 5.1	# 3.5	# 1.4	# 0.2	Iris-setosa
4.8	3.0	1.4	0.3	Iris-setosa
5.1	3.8	1.6	0.2	Iris-setosa
4.6	3.2	1.4	0.2	Iris-setosa
5.3	3.7	1.5	0.2	Iris-setosa
5.0	3.3	1.4	0.2	Iris-setosa
7.0	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4.0	1.3	Iris-versicolor
6.5	2.8	4.6	1.5	Iris-versicolor
5.7	2.8	4.5	1.3	Iris-versicolor
6.3	3.3	4.7	1.6	Iris-versicolor

2. Save the model.

model.py

```
# Load Iris Dataset
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data
y = iris.target

# Split to Train and Test Datasets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20)

# Transform Features
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)

# Build and Train Model
from sklearn.neural_network import MLPClassifier
mlp = MLPClassifier(hidden_layer_sizes=(10, 10, 10), max_iter=1000)
mlp.fit(X_train, y_train)
predictions = mlp.predict(X_test)

# Evaluation of the Model
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,predictions))
print(classification_report(y_test,predictions))
```

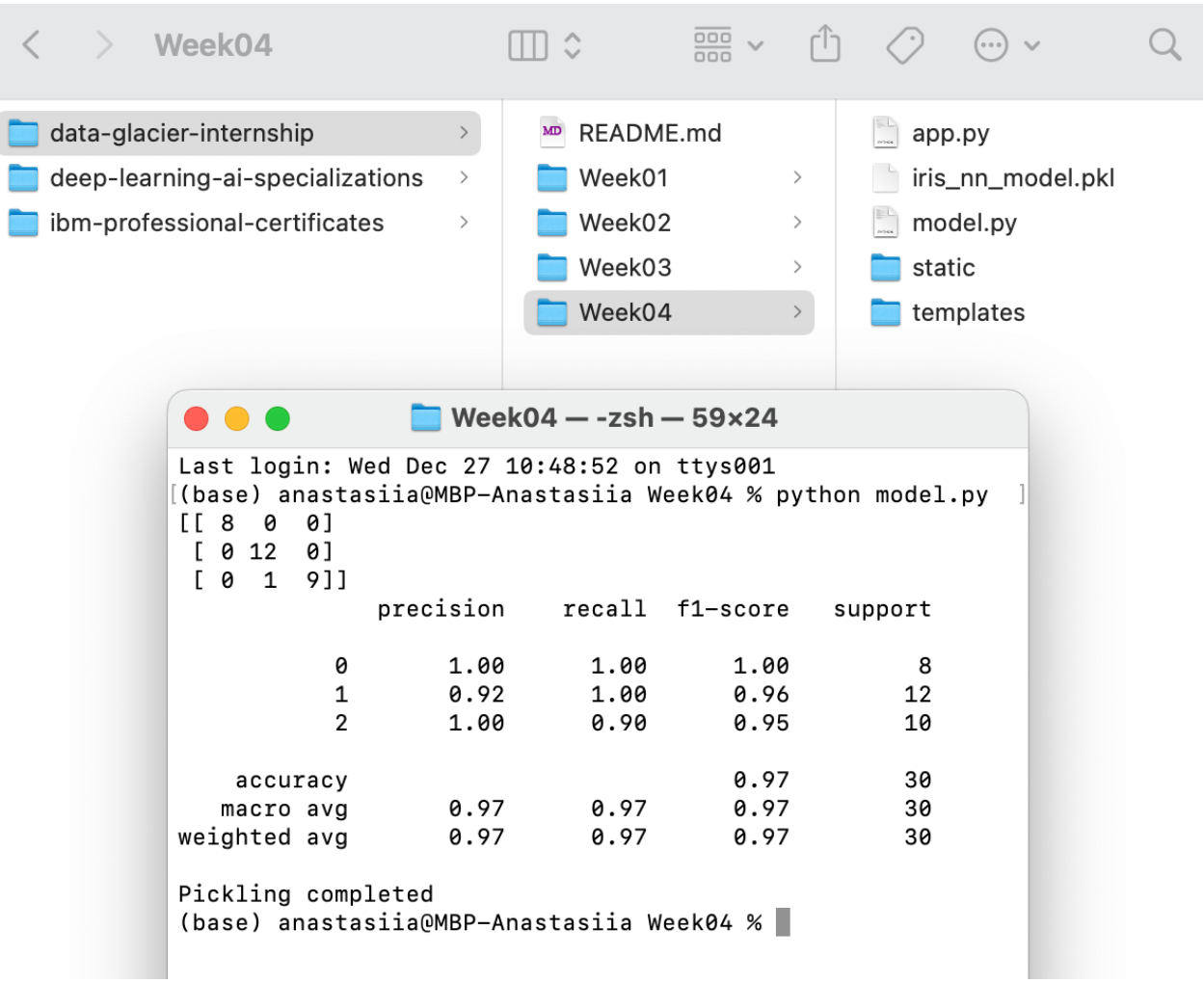
Object Serialization by using Pickle:

```
import pickle
```

model

```
# Save the model to a file
with open('iris_nn_model.pkl', 'wb') as f:
    pickle.dump(mlp, f)
    print("Pickling completed")
```

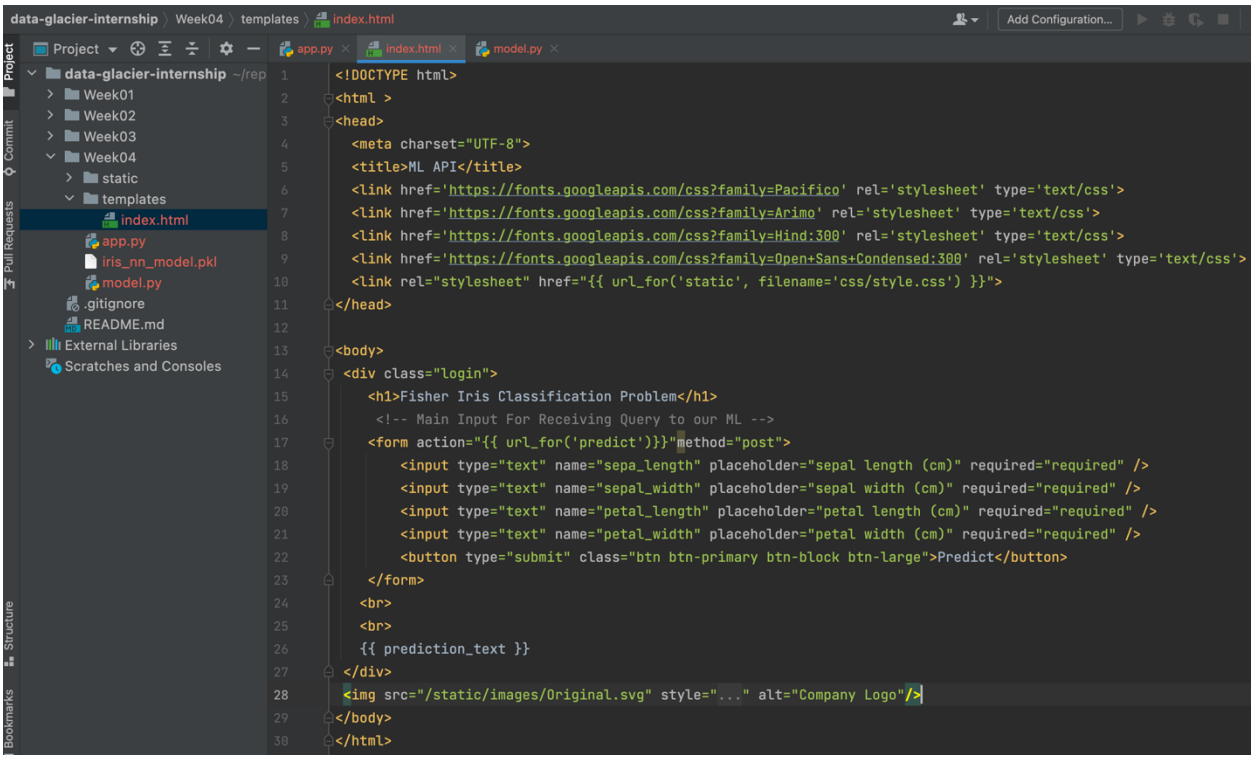
In Terminal:



Now we have file iris_nn_model.pkl

3. Deploy the model on flask (web app).

Create the file index.html for the model.



We use deserialization with Pickle.

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

app = Flask(__name__)
model = pickle.load(open('iris_nn_model.pkl', 'rb'))


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


@app.route('/predict', methods=['POST'])
def predict():
    # for rendering results on HTML GUI
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    iris_names = ['Setosa', 'Versicolor', 'Virginica']
    predicted_name = iris_names[int(prediction[0])]
```

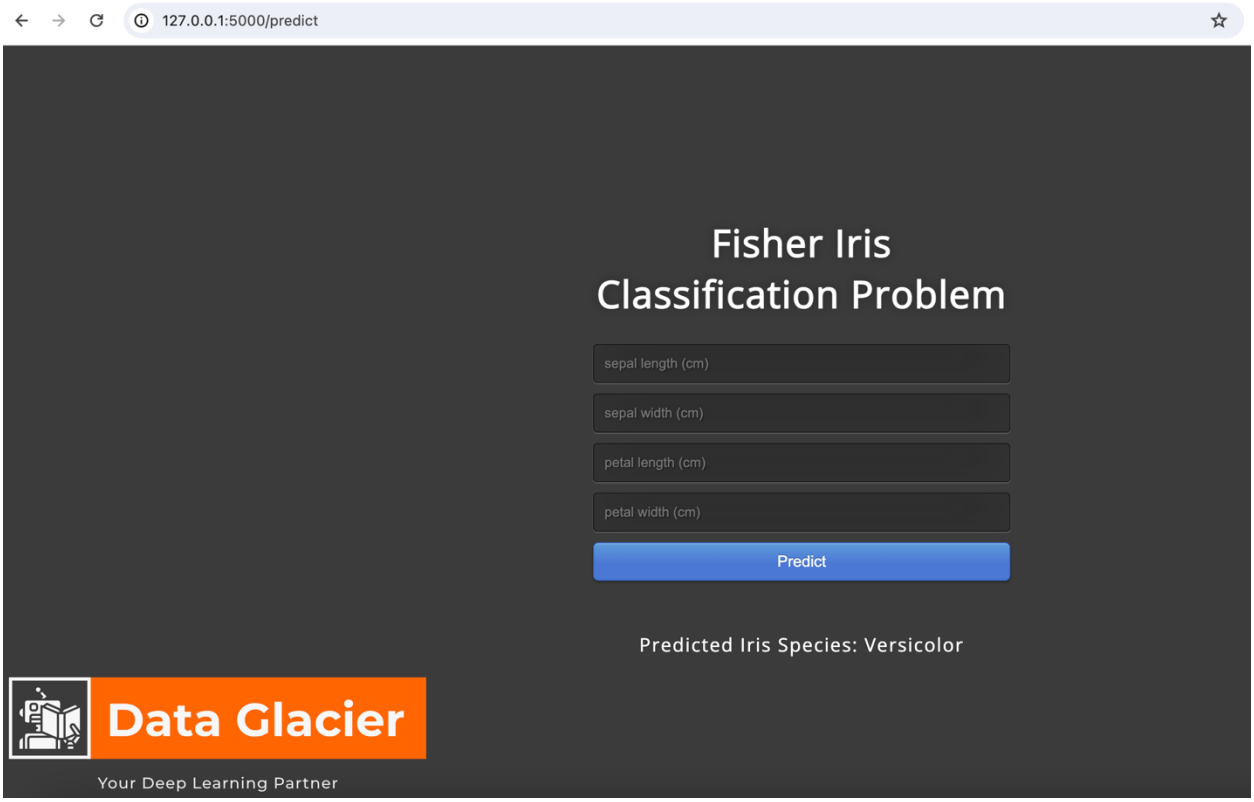
```
    return render_template('index.html', prediction_text='Predicted Iris
Species: {}'.format(predicted_name))


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

In Terminal:

```
(base) anastasiia@MBP-Anastasiia Week04 % python app.py
 * 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 watchdog (fsevents)
 * Debugger is active!
 * Debugger PIN: 515-690-735
```

Copy: http://127.0.0.1:5000



4. Cloud and API Deployment (Heroku).

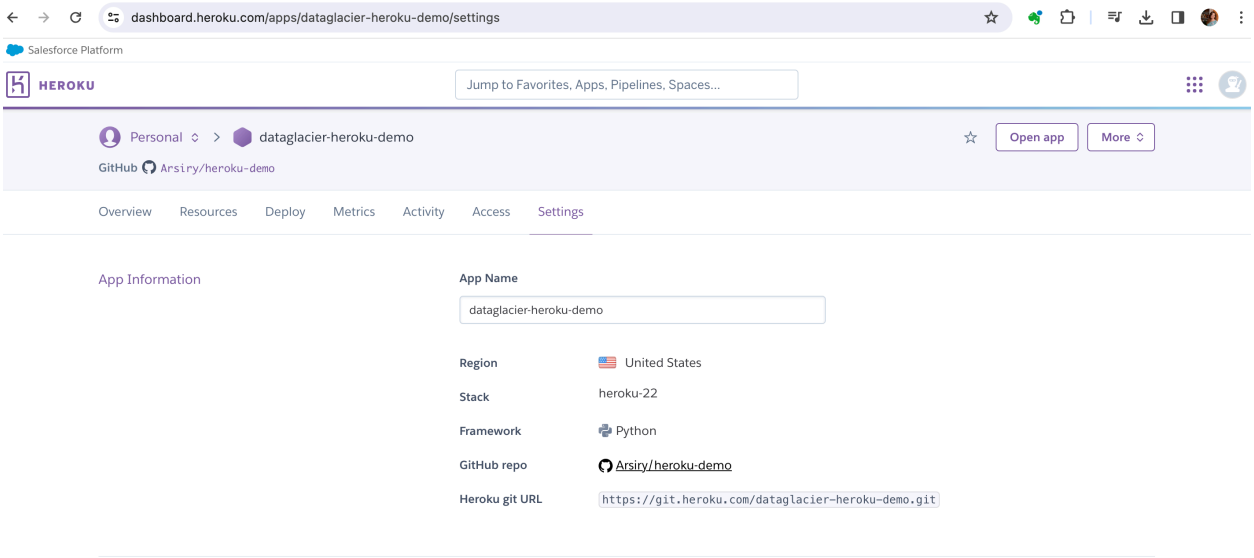
Create the file requirements.txt for virtual environment.

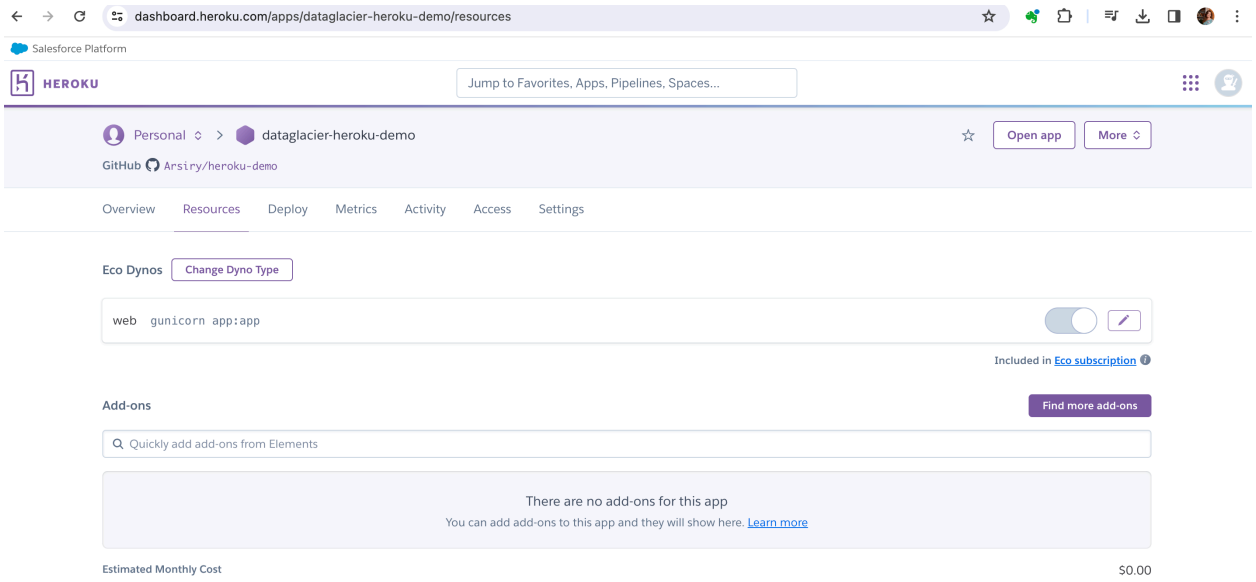
```
Flask==2.2.3
gunicorn==21.2.0
itsdangerous==2.1.2
Jinja2==3.1.2
joblib==1.2.0
MarkupSafe==2.1.2
scikit-learn==1.2.2
Werkzeug==2.2.3
```

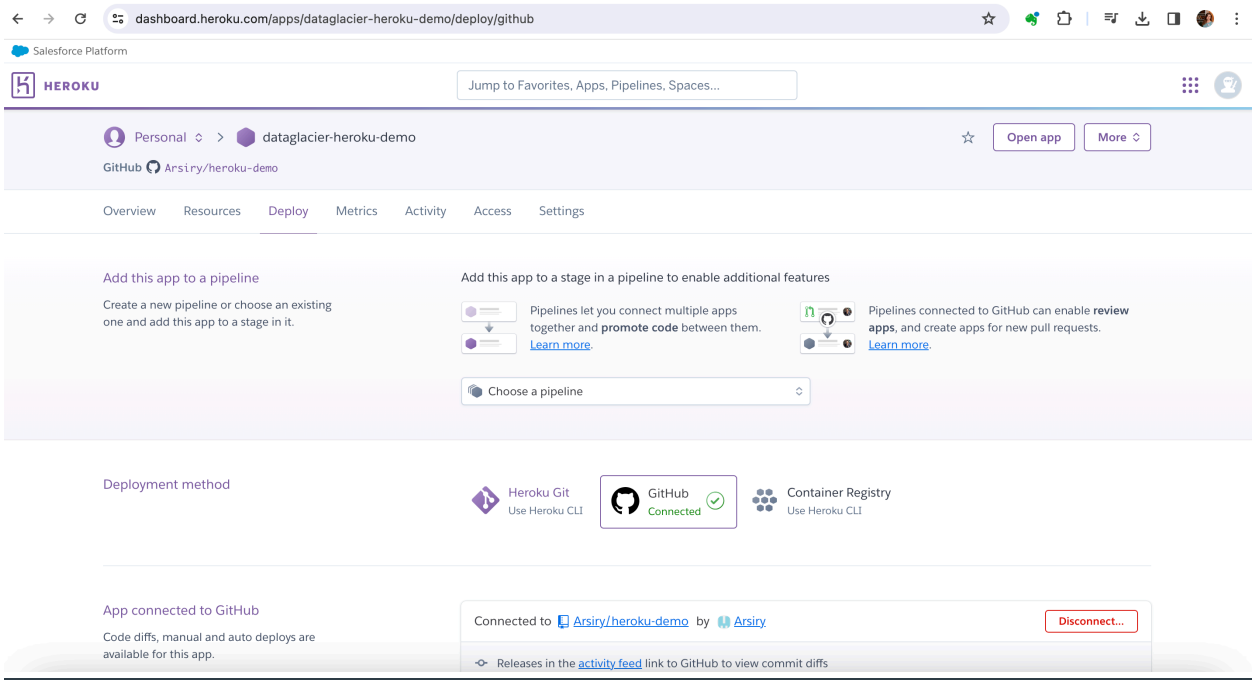
Create the file Procfile.txt for notice how to execute the app.

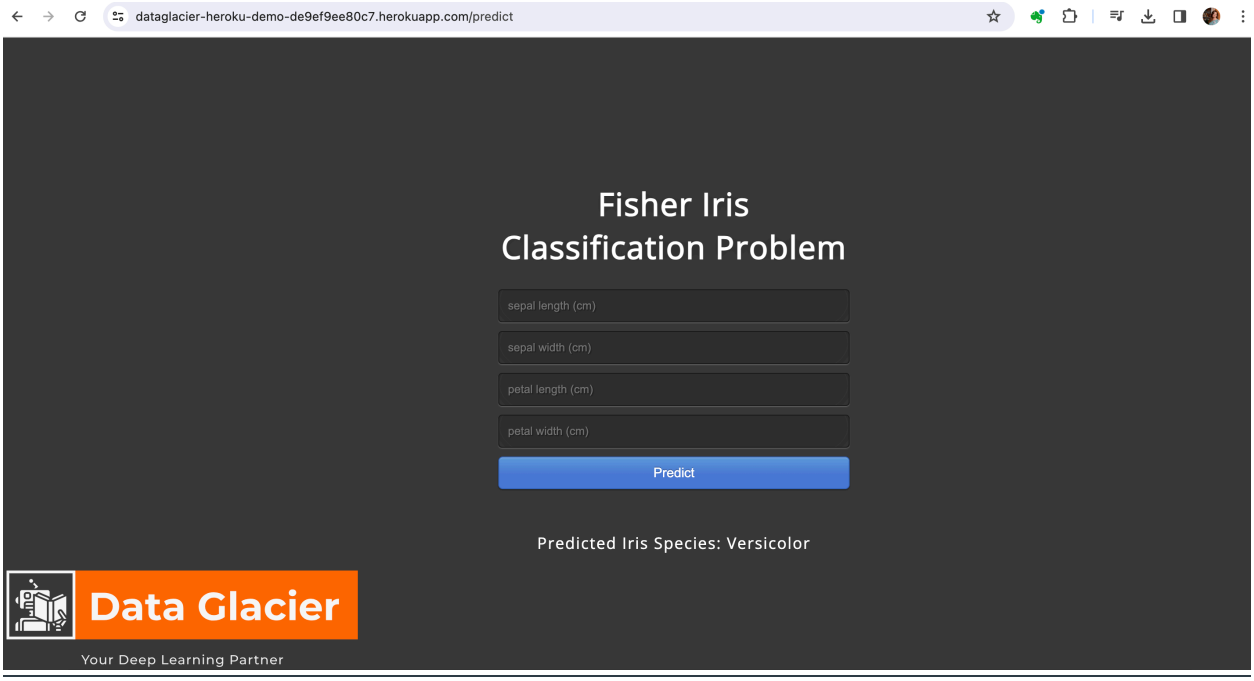
```
web: gunicorn app:app
```

In Heroku:




dashboard.heroku.com/apps/dataglacier-heroku-demo/resources
Salesforce Platform
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
dataglacier-heroku-demo
Open app
More
GitHub Arsiry/heroku-demo
Overview
Resources
Deploy
Metrics
Activity
Access
Settings
Eco Dynos
Change Dyno Type
web gunicorn app:app
Included in Eco subscription
Add-ons
Find more add-ons
Quickly add add-ons from Elements
There are no add-ons for this app
You can add add-ons to this app and they will show here. Learn more
Estimated Monthly Cost
$0.00


dashboard.heroku.com/apps/dataglacier-heroku-demo/deploy/github
Salesforce Platform
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
dataglacier-heroku-demo
Open app
More
GitHub Arsiry/heroku-demo
Overview
Resources
Deploy
Metrics
Activity
Access
Settings
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
Connected
Container Registry
Use Heroku CLI
App connected to GitHub
Code diffs, manual and auto deploys are available for this app.
Connected to Arsiry/heroku-demo by Arsiry
Disconnect...
Releases in the activity feed link to GitHub to view commit diffs


dataglacier-heroku-demo-de9ef9ee80c7.herokuapp.com/predict
Fisher Iris
Classification Problem
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
Predict
Predicted Iris Species: Versicolor
Data Glacier
Your Deep Learning Partner