Name: Clyde Wawire

Batch code: LISUM06

Submission date:7th March 2021

Submitted to: Week 5: Cloud and API deployment

https://github.com/clydewawire/Data-Glacier-Cloud-API-deployment (on GitHub)

Note: the app was deployed on the cloud with Heroku in week 4 but is documented here also.

App URL: https://predict3iris.herokuapp.com/

Select data (iris dataset), create and save a simple model (knn classifier):

Create html and css files:

index.html - Notepad

File Edit Format View Help

```
<html>
<head>
<link rel= "stylesheet" type= "text/css" href= "{{ url_for('static',filename='styles/styles.css') }}">
<link rel="stylesheet" type="text/css" href="//fonts.googleapis.com/css?family=Playfair+Display" />
<title> Predict the type of iris flower </title>
</head>

<body>
<h1> Predict the type of iris flower (Setosa, Versicolor, or Virginica) using a K nearest neighbors classifier (k=3)</h1>
<div class="wrapper">
<div class="form">

<form action = "{{ url_for('predict')}}" method="post">
        <input type="text" name="sepal_length" placeholder= "Sepal Length(cm)" required="required" /> <br> <br>
        <input type="text" name="sepal_width" placeholder= "Sepal Width(cm)" required="required" /> <br> <br>
        <input type="text" name="petal_length" placeholder= "Petal Length(cm)" required="required" /> <br> <br>
        <input type="text" name="petal_width" placeholder= "Petal Width(cm)" required="required" /> <br> <br>
        <button type="submit"> Predict </button>

<br>
<br>
{{ prediction_text }}
</form>

</div>
        <div class="image">
 <img src="{{ iris }}" alt="">
        </div>
```

*styles.css - Notepad

File Edit Format View Help

```
* {
 font-family:"Playfair Display";

}
body {
  background-color: lightblue;

}

h1 {
font-size:3.5em;
margin-left:5%;
margin-right:5%;
}

form input, button {
font-size:1.5em;
}
form  {
font-size:1.5em;
}
```

jupyter Model_Deployment Last Checkpoint: Last Wednesday at 22:22 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted

Press F11 to exit full screen

Run Code

In [1]:
```
import flask
import pickle
```

In [2]:
```
from sklearn import datasets
import pandas as pd
data=datasets.load_iris()
df = pd.DataFrame(data.data,columns=data.feature_names)
df['target']=data['target']
```

In [3]:
```
X=df.loc[:,df.columns!="target"] #let the feature dataframe contain every column of df, except the value we are predicting
y=df.loc[:,df.columns=="target"].values.ravel() #let the target dataframe contain only the value we are predicting
```

In [4]:
```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.25, shuffle=True)
```

In [5]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X, y)
pred=knn.predict(X_test)
acc=accuracy_score(y_test, pred)
```

In [10]:
```
from joblib import dump, load
dump(knn, 'model.joblib')
```

Out[10]:
```
['model.joblib']
```

In [11]:
```
l_m = load('model.joblib')
```

In [12]:
```
print(l_m)
```

```
KNeighborsClassifier(n_neighbors=3)
```

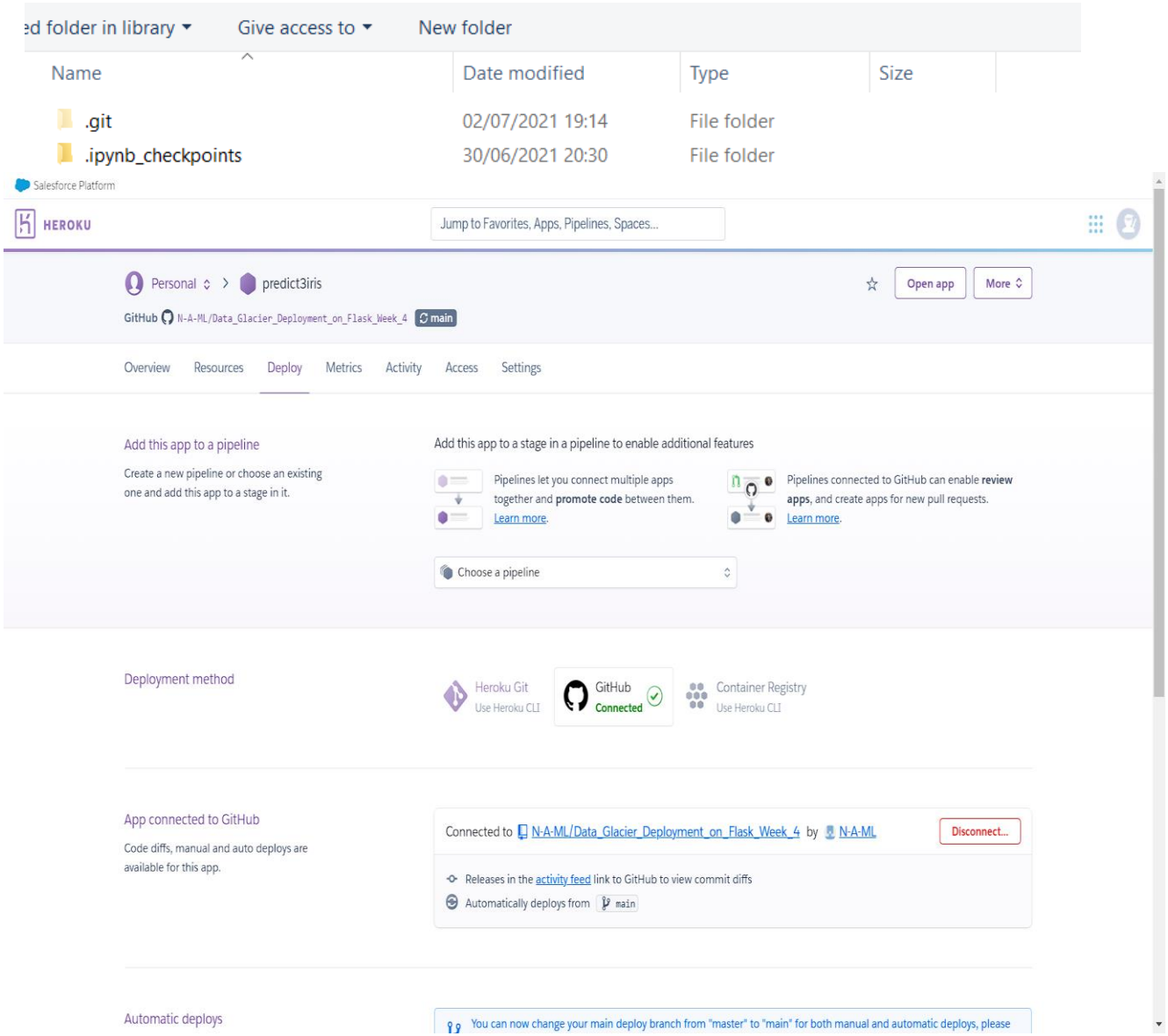
jupyter Deployment_flask.py a few seconds ago

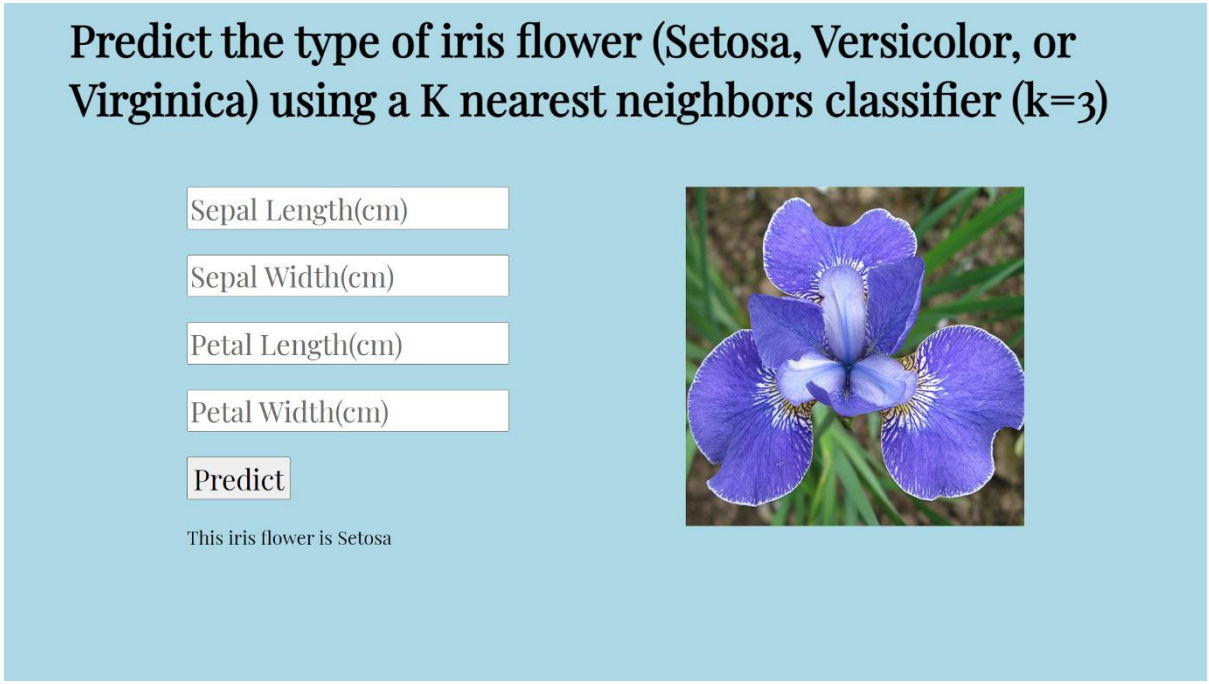
File Edit View Language

Press F11 to exit full screen

```
#!/usr/bin/env python
# coding: utf-8

# In[ ]:
import numpy as np
from flask import Flask, request, render_template
import joblib
from joblib import load
from sklearn.neighbors import KNeighborsClassifier
import os
images_folder=os.path.join('static', 'images')
app=Flask(__name__)
app.config['UPLOAD_FOLDER'] = images_folder
model=load('model.joblib')

@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict', methods=['POST'])
def predict():
    features=[float(x) for x in request.form.values()]
    final_features=[np.array(features)]
    prediction=model.predict(final_features)
    pred_round=round(prediction[0])
    output=""
    if pred_round==0:
        output+="Setosa"
        file = os.path.join(app.config['UPLOAD_FOLDER'], 'setosa.jpg')
    elif pred_round==1:
        output+="Versicolor"
        file = os.path.join(app.config['UPLOAD_FOLDER'], 'versicolor.jpg')
    else:
        output+="Virginica"
        file = os.path.join(app.config['UPLOAD_FOLDER'], 'virginica.jpg')

    return render_template('index.html', prediction_text='This iris flower is {}'.format(output),
                           iris=file
                          )
if __name__=="__main__":
    app.run(port=5000, debug=True, use_reloader=False)

# In[16]:
```

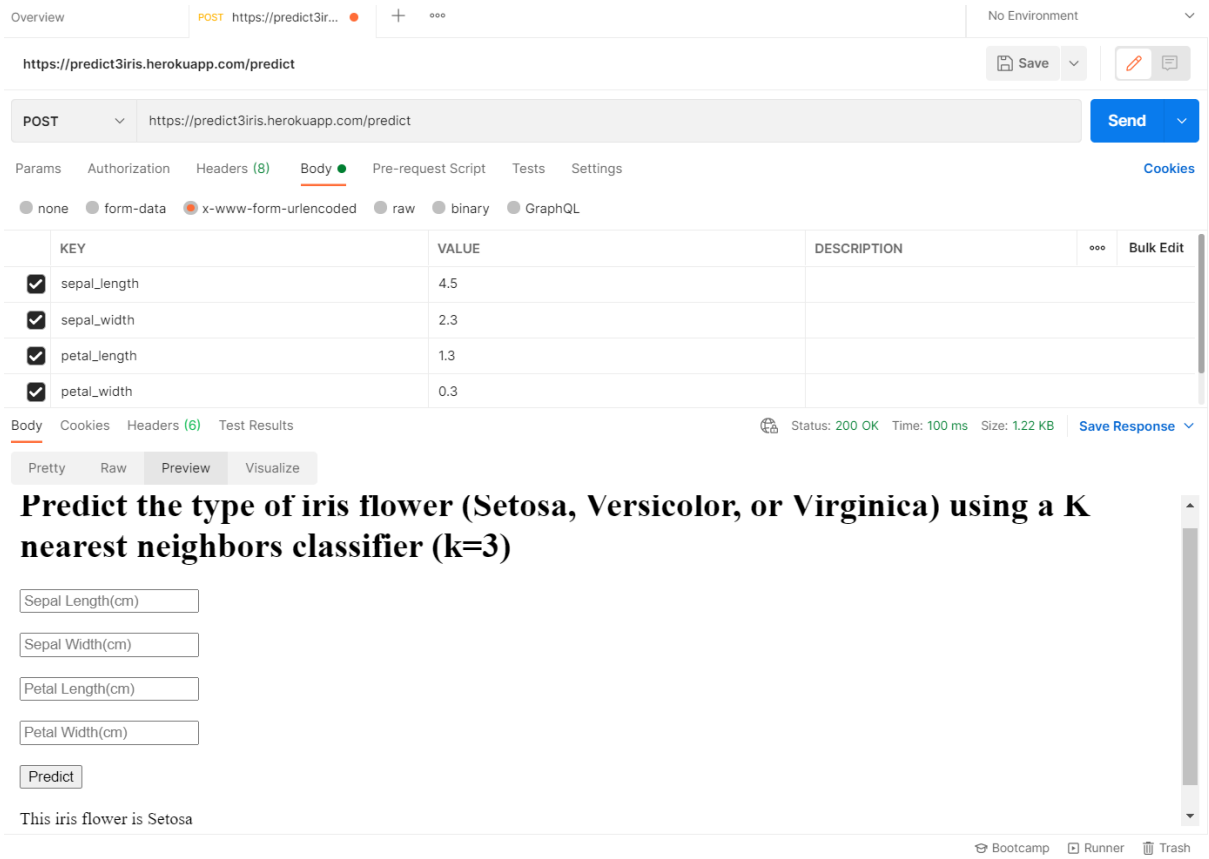
Generate Procfile (and enter the name of the app), requirements.txt and runtime.txt, and structure the files and folders correctly:

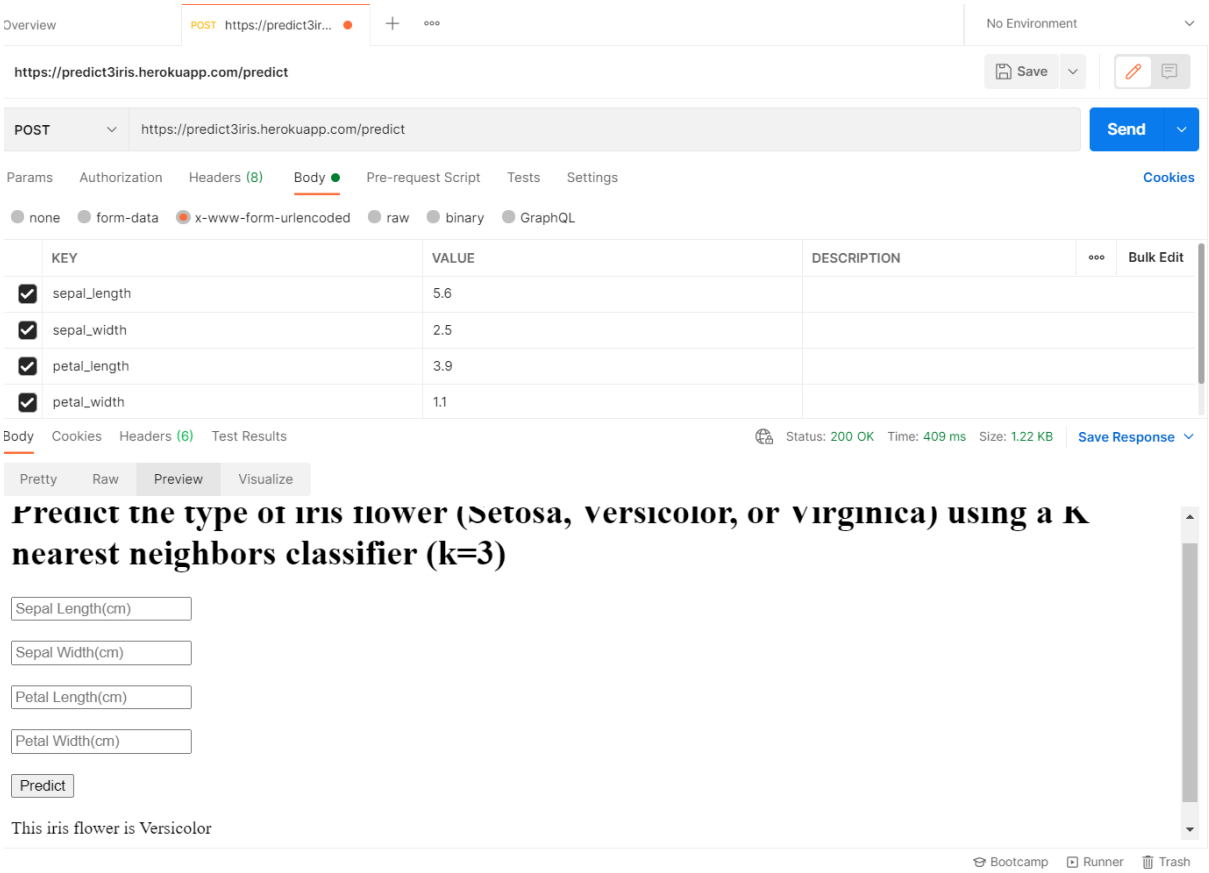


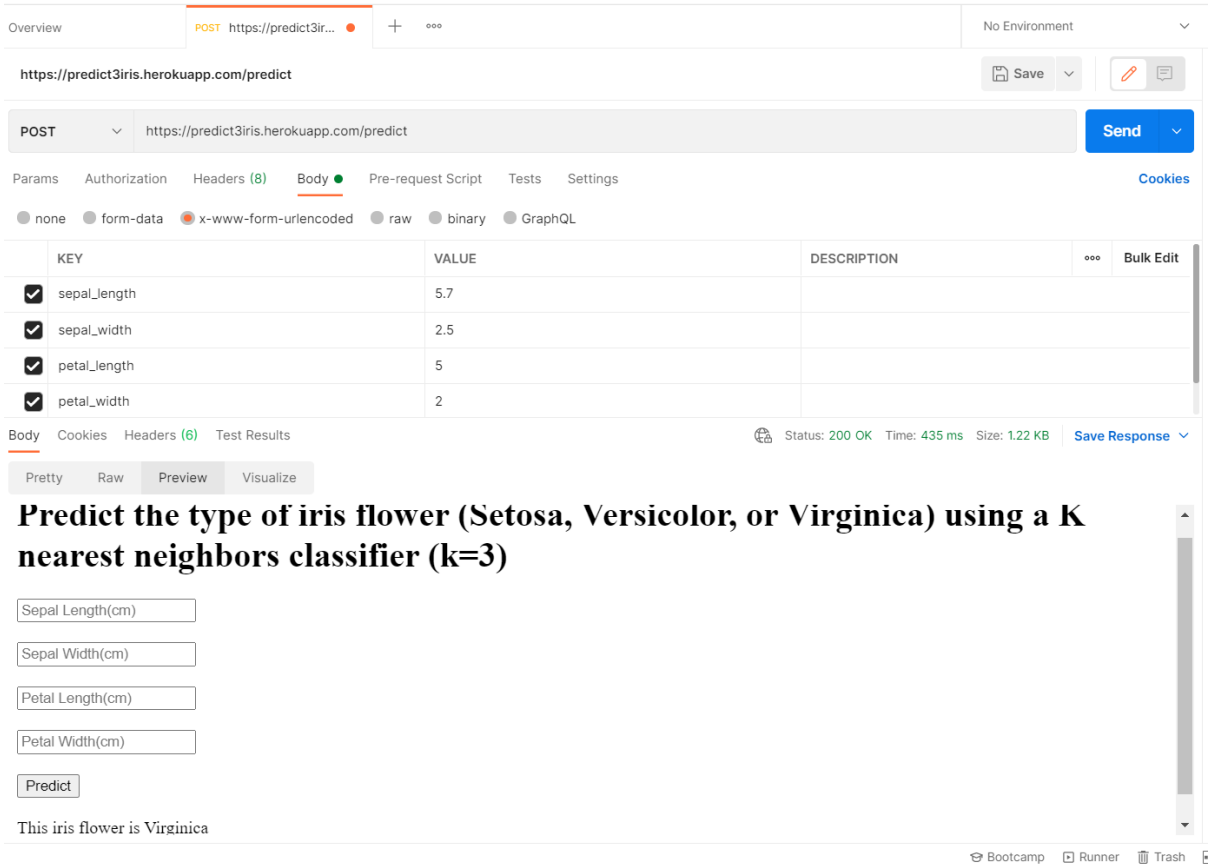


The app is working as intended.

Each type of prediction was tested using Postman:


Overview
POST https://predict3ir...
No Environment
https://predict3iris.herokuapp.com/predict
Save
POST
https://predict3iris.herokuapp.com/predict
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
DESCRIPTION
Bulk Edit
sepal_length
4.5
sepal_width
2.3
petal_length
1.3
petal_width
0.3
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 100 ms
Size: 1.22 KB
Save Response
Pretty
Raw
Preview
Visualize
Predict the type of iris flower (Setosa, Versicolor, or Virginica) using a K nearest neighbors classifier (k=3)
Sepal Length(cm)
Sepal Width(cm)
Petal Length(cm)
Petal Width(cm)
Predict
This iris flower is Setosa
Bootcamp
Runner
Trash


Overview
POST https://predict3ir...
No Environment
https://predict3iris.herokuapp.com/predict
Save
POST
https://predict3iris.herokuapp.com/predict
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
DESCRIPTION
Bulk Edit
sepal_length
5.6
sepal_width
2.5
petal_length
3.9
petal_width
1.1
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 409 ms
Size: 1.22 KB
Save Response
Pretty
Raw
Preview
Visualize
Predict the type of Iris flower (Setosa, Versicolor, or Virginica) using a K nearest neighbors classifier (k=3)
Sepal Length(cm)
Sepal Width(cm)
Petal Length(cm)
Petal Width(cm)
Predict
This iris flower is Versicolor
Bootcamp
Runner
Trash


Overview
POST https://predict3ir...
No Environment
https://predict3iris.herokuapp.com/predict
Save
POST
https://predict3iris.herokuapp.com/predict
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
DESCRIPTION
Bulk Edit
sepal_length
5.7
sepal_width
2.5
petal_length
5
petal_width
2
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 435 ms
Size: 1.22 KB
Save Response
Pretty
Raw
Preview
Visualize
Predict the type of iris flower (Setosa, Versicolor, or Virginica) using a K nearest neighbors classifier (k=3)
Sepal Length(cm)
Sepal Width(cm)
Petal Length(cm)
Petal Width(cm)
Predict
This iris flower is Virginica
Bootcamp
Runner
Trash

The iris types were predicted correctly and everything is working as intended.