Data Glacier

Data Science Intern at Data Glacier

Week 4: Deployment on Flask

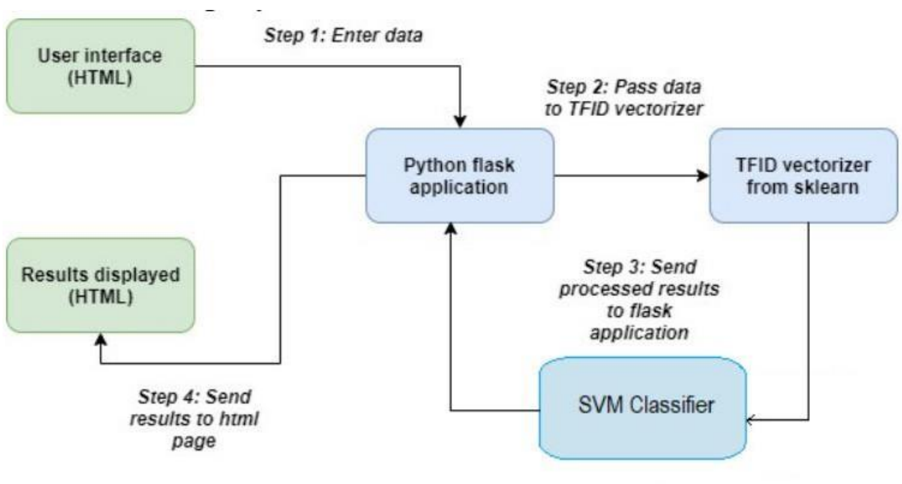
Name: Pravallika sheshabhatter

Batch Code: LISUM21

Date: 28 May 2023

Submitted to: Data Glacier

In this project, we are going to deploying machine learning model using the Flask Framework.



Create an API for the model, using Flask, the Python micro-framework for building web applications. This API allows us to utilize predictive capabilities through HTTP requests.

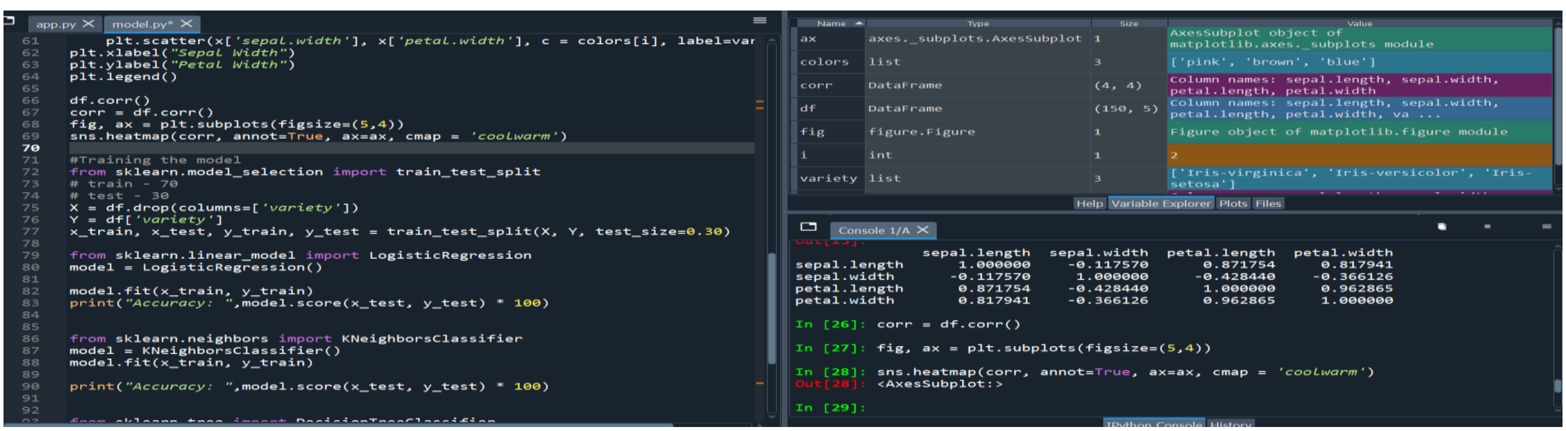
Data Information

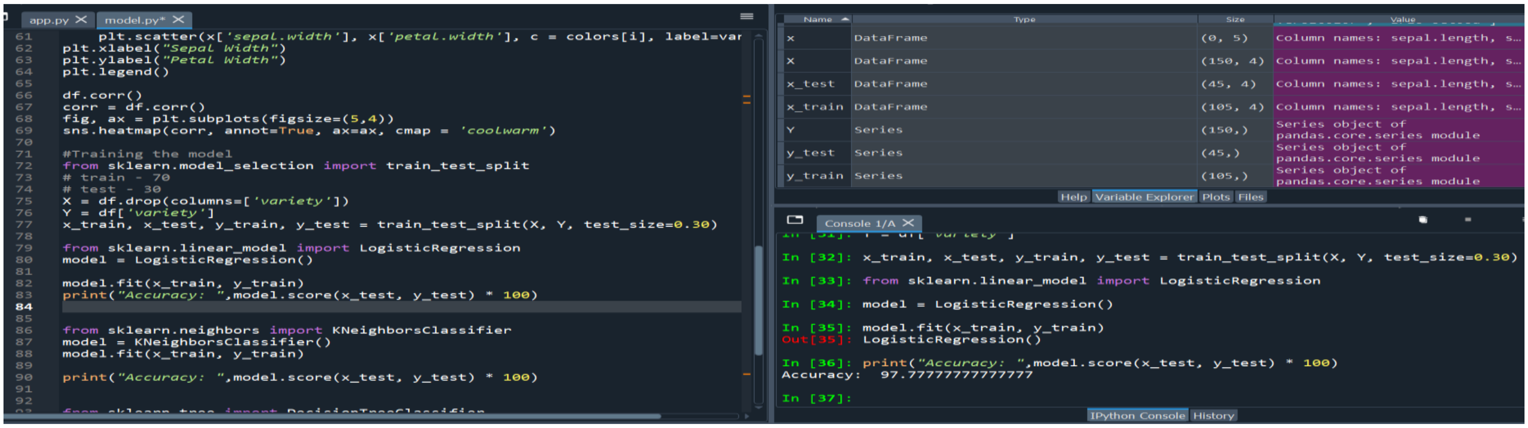
- I have taken the toy data set as mentioned in the assignment i.e., IRIS data set.
- The collection is composed of one CSV file per dataset.

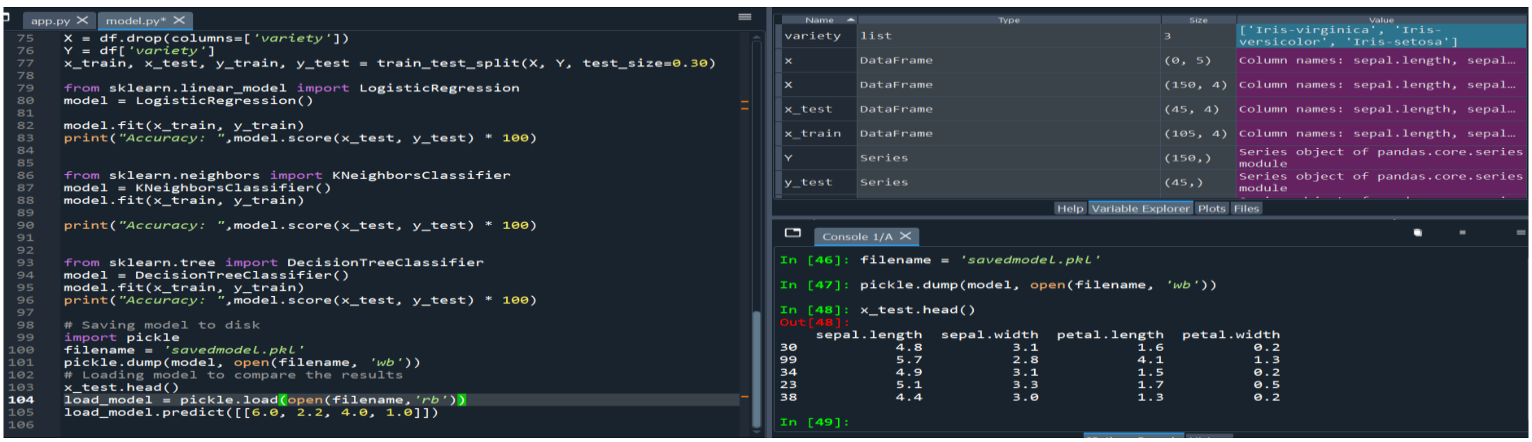
Building a Model

Import Required Libraries and Dataset

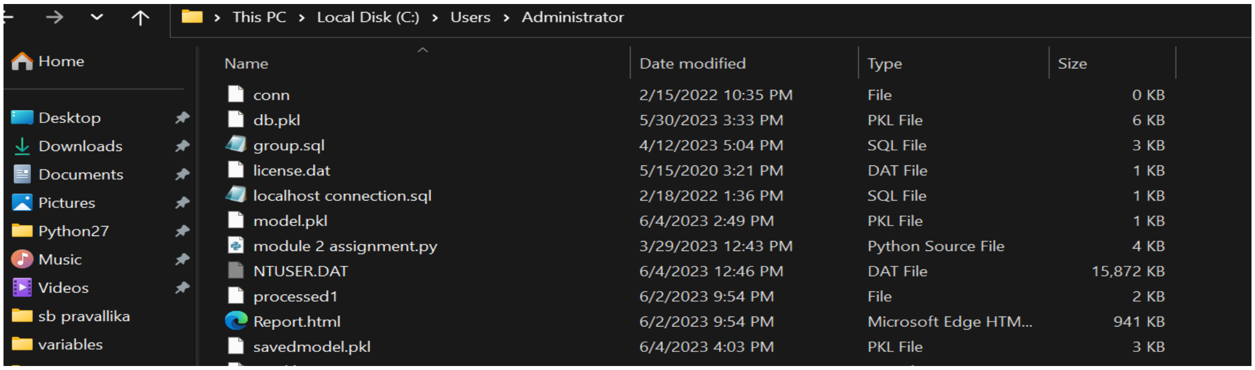
In this part, we import libraires and dataset which contain the information of five most commented video.




app.py
model.py*
plt.scatter(x['sepal.width'], x['petal.width'], c = colors[i], label=var
plt.xlabel("Sepal Width")
plt.ylabel("Petal Width")
plt.legend()
df.corr()
corr = df.corr()
fig, ax = plt.subplots(figsize=(5,4))
sns.heatmap(corr, annot=True, ax=ax, cmap = 'coolwarm')
#Training the model
from sklearn.model_selection import train_test_split
# train - 70
# test - 30
X = df.drop(columns=['variety'])
Y = df['variety']
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.30)
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, y_train)
print("Accuracy: ",model.score(x_test, y_test) * 100)
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
model.fit(x_train, y_train)
print("Accuracy: ",model.score(x_test, y_test) * 100)
Name
Type
Size
Value
x
DataFrame
(0, 5)
Column names: sepal.length, s…
X
DataFrame
(150, 4)
Column names: sepal.length, s…
x_test
DataFrame
(45, 4)
Column names: sepal.length, s…
x_train
DataFrame
(105, 4)
Column names: sepal.length, s…
Y
Series
(150,)
Series object of pandas.core.series module
y_test
Series
(45,)
Series object of pandas.core.series module
y_train
Series
(105,)
Series object of pandas.core.series module
Help
Variable Explorer
Plots
Files
Console 1/A
In [32]: x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.30)
In [33]: from sklearn.linear_model import LogisticRegression
In [34]: model = LogisticRegression()
In [35]: model.fit(x_train, y_train)
Out[35]: LogisticRegression()
In [36]: print("Accuracy: ",model.score(x_test, y_test) * 100)
Accuracy: 97.77777777777777
In [37]:
IPython Console
History

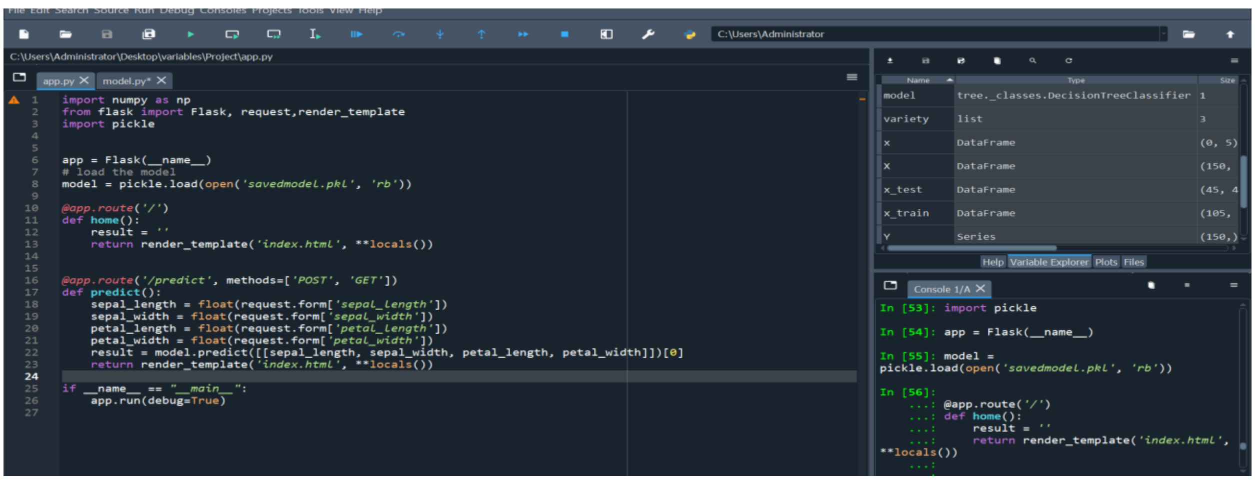

app.py
model.py*
X = df.drop(columns=['variety'])
Y = df['variety']
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.30)
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, y_train)
print("Accuracy: ",model.score(x_test, y_test) * 100)
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
model.fit(x_train, y_train)
print("Accuracy: ",model.score(x_test, y_test) * 100)
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(x_train, y_train)
print("Accuracy: ",model.score(x_test, y_test) * 100)
# Saving model to disk
import pickle
filename = 'savedmodel.pkl'
pickle.dump(model, open(filename, 'wb'))
# Loading model to compare the results
x_test.head()
load_model = pickle.load(open(filename,'rb'))
load_model.predict([[6.0, 2.2, 4.0, 1.0]])
Name
Type
Size
Value
variety
list
3
['Iris-virginica', 'Iris-versicolor', 'Iris-setosa']
x
DataFrame
(0, 5)
Column names: sepal.length, sepal…
X
DataFrame
(150, 4)
Column names: sepal.length, sepal…
x_test
DataFrame
(45, 4)
Column names: sepal.length, sepal…
x_train
DataFrame
(105, 4)
Column names: sepal.length, sepal…
Y
Series
(150,)
Series object of pandas.core.series module
y_test
Series
(45,)
Series object of pandas.core.series module
Help
Variable Explorer
Plots
Files
Console 1/A
In [46]: filename = 'savedmodel.pkl'
In [47]: pickle.dump(model, open(filename, 'wb'))
In [48]: x_test.head()
Out[48]:
sepal.length sepal.width petal.length petal.width
30 4.8 3.1 1.6 0.2
99 5.7 2.8 4.1 1.3
34 4.9 3.1 1.5 0.2
23 5.1 3.3 1.7 0.5
38 4.4 3.0 1.3 0.2
In [49]:

After executing the program in spyder in local disc C the ***savemodel.pkl*** is created .

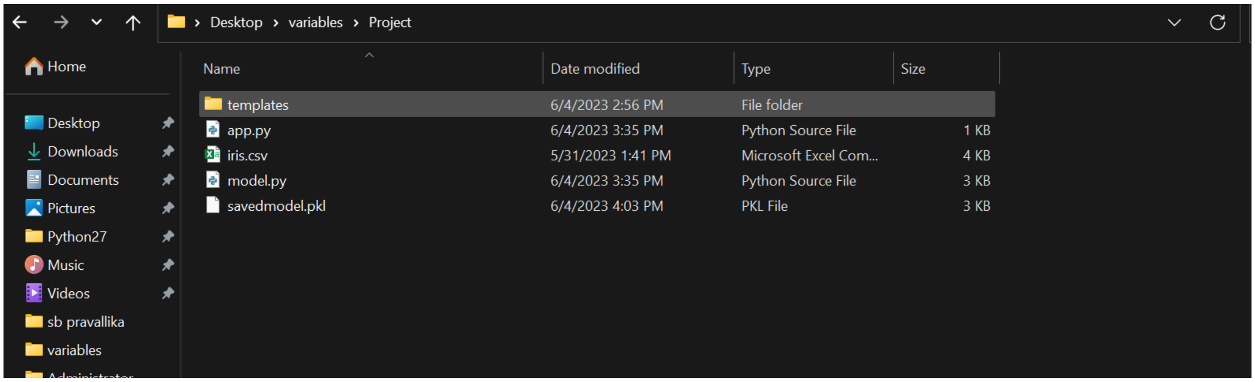


App.py

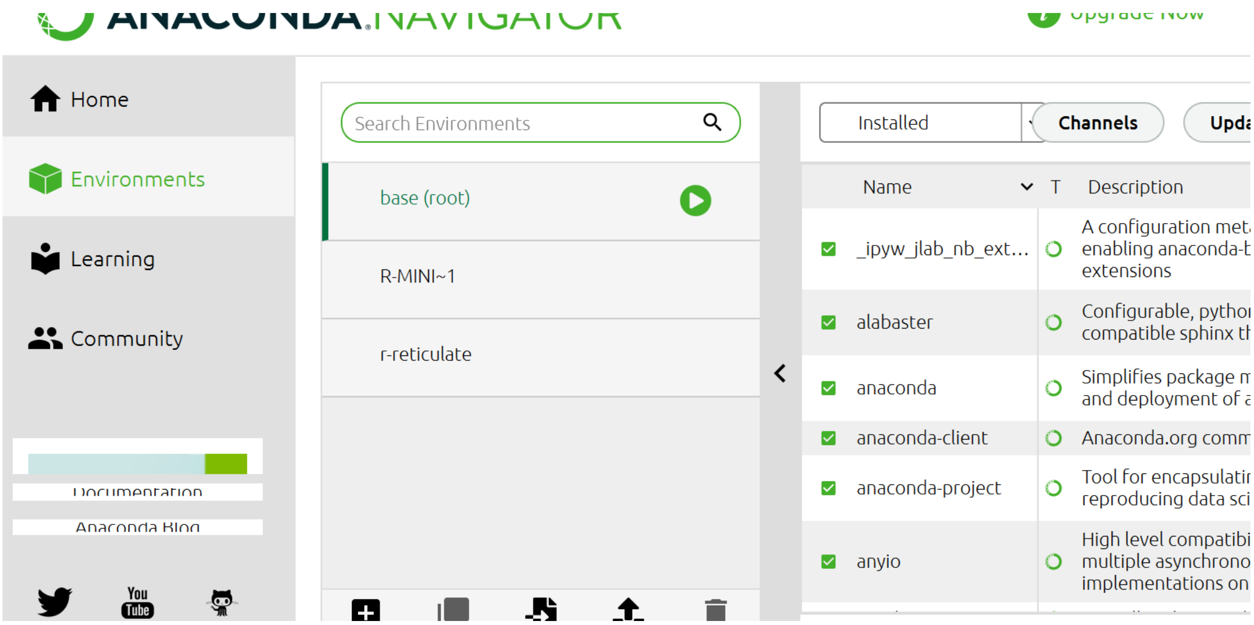
The *app.py* file contains the main code that will be executed by the Python interpreter to run the Flask web application, it included the ML code for classifying SD.



- We ran our application as a single module; thus we initialized a new Flask instance with the argument _*name*_to let Flask know that it can find the HTML template folder (*templates*) in the same directory where it is located.
- Next, we used the route decorator *(@app.route('/'))* to specify the URL that should trigger the execution of the home function.
- Our *home* function simply rendered index.html HTML file, which is located in the *templates* folder.

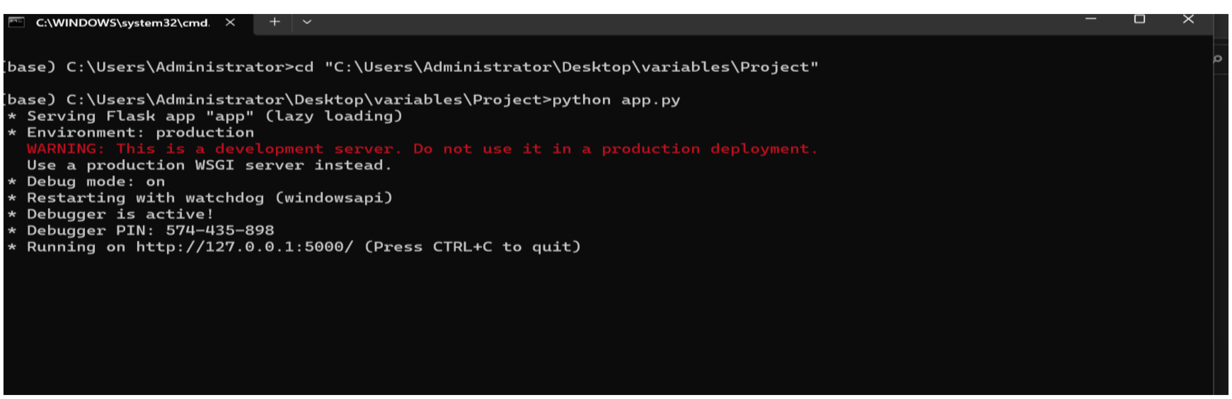


Go to ANACONDA NAVIGATOR and open terminal give the commands:



Running Procedure

Once we have done all of the above, we can start running the API by either double click *app.py*, or executing the command from the Terminal:



Now we could open a web browser and navigate to http://127.0.0.1:5000/ we should see a simple website.

After entering the input click the predict button now, we can the result of our input.