Machine Learning Model Deployment using Flask

This document discusses the various steps involved in deploying a machine learning model using Flask. Refer the data intake report for details about the data used for this project.

Dataset Used: Palmer Penguins

ML Model used: Random Forest Classifier

Step 1: Download the required packages:

```
(base) C:\Users\ggnar>pip install flask
WARNING: Ignoring invalid distribution -ensorflow-gpu (d:\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow-gpu (d:\anaconda\lib\site-packages)
Requirement already satisfied: flask in d:\anaconda\lib\site-packages (1.1.2)
Requirement already satisfied: Jinja2>=2.10.1 in d:\anaconda\lib\site-packages (from flask) (2.11.3)
Requirement already satisfied: itsdangerous>=0.24 in d:\anaconda\lib\site-packages (from flask) (2.0.1)
Requirement already satisfied: Werkzeug>=0.15 in d:\anaconda\lib\site-packages (from flask) (2.0.2)
Requirement already satisfied: click>=5.1 in d:\anaconda\lib\site-packages (from flask) (8.0.3)
Requirement already satisfied: colorama in d:\anaconda\lib\site-packages (from click>=5.1->flask) (0.4.4)
Requirement already satisfied: MarkupSafe>=0.23 in d:\anaconda\lib\site-packages (from Jinja2>=2.10.1->flask) (1.1.1)
WARNING: Ignoring invalid distribution -ensorflow-gpu (d:\anaconda\lib\site-packages)
```

The First step is to install flask using the following command:

>> pip install flask

Other libraries that are required include:

- pickle
- pandas
- scikit learn

Step 2: Writing Python code for building a machine learning model:

Building a Random Forest Classifier for predicting penguin species

```
In [58]: X = penguinsData.drop("species",axis=1)
         y = penguinsData["species"]
```

Creating Dummy variables for island and sex

```
In [60]: X = pd.get_dummies(X,drop_first=True)
```

Splitting model into train and test set

```
In [62]: X_train, X_test, y_train, y_test = train_test_split(X,y,random_state=42,test_size=0.3)
```

Random Forest Model

```
In [63]: rfc = RandomForestClassifier()
         rfc.fit(X_train,y_train)
Out[63]: RandomForestClassifier()
```

Next, we write a python program that creates a random forest classifier for predicting the species of penguin based on a set of features. For full code please refer to the Jupyter Notebook file.

Step 3: Creating a Pickle file:

Creating a Pickle file

```
In [67]: pickle.dump(rfc, open("penguinsRFCModel.pkl","wb"))
```

We then create a pickle file where we store the trained model.

Step 4: Creating HTML file for getting input parameters and displaying the output:

A HTML file is created which is used for providing inputs for the prediction.

Step 5: Creating the Flask App:

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

app = Flask(__name__)

# Load the pre-trained machine learning model
model = pickle.load(open("penguinsRFCModel.pkl", "rb"))

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

@app.route('/predict', methods=['POST'])
def predict():
    # Get the user input from the HTML form
```

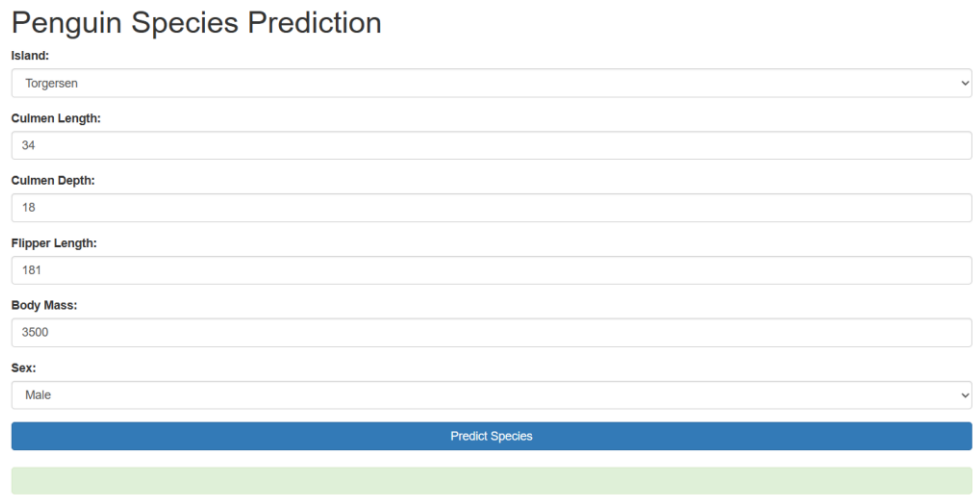
The next step is to create a flask app that uses the trained model stored in the pickle file to predict the penguin species for the input parameters that are input through the webpage.

Step 6: Running the Flask App:

```
(base) C:\Users\ggnar\OneDrive\Desktop\Programs\DG Internship\ML-Model-Deployment>python predictSpecies.py
 * Serving Flask app "predictSpecies" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 266-384-067
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Finally, we run the flask app. This opens the webpage where we enter the input parameters for which we want to predict the penguin species.

Sample input:



On clicking the predict species button we get the following output: