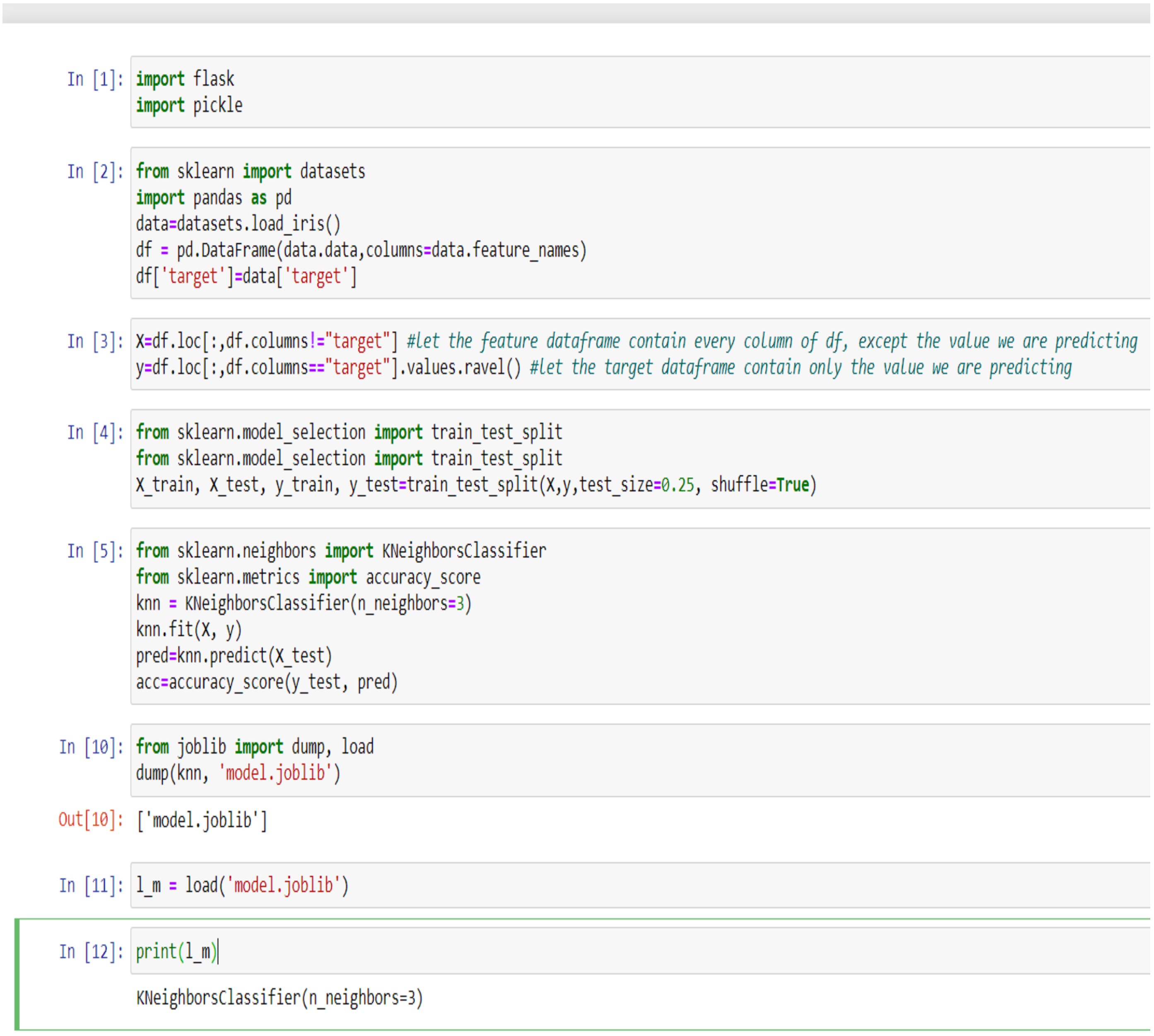
Name: Manoj Nagaraja

Batch code: LISUM19

Submission date:27/03/2023

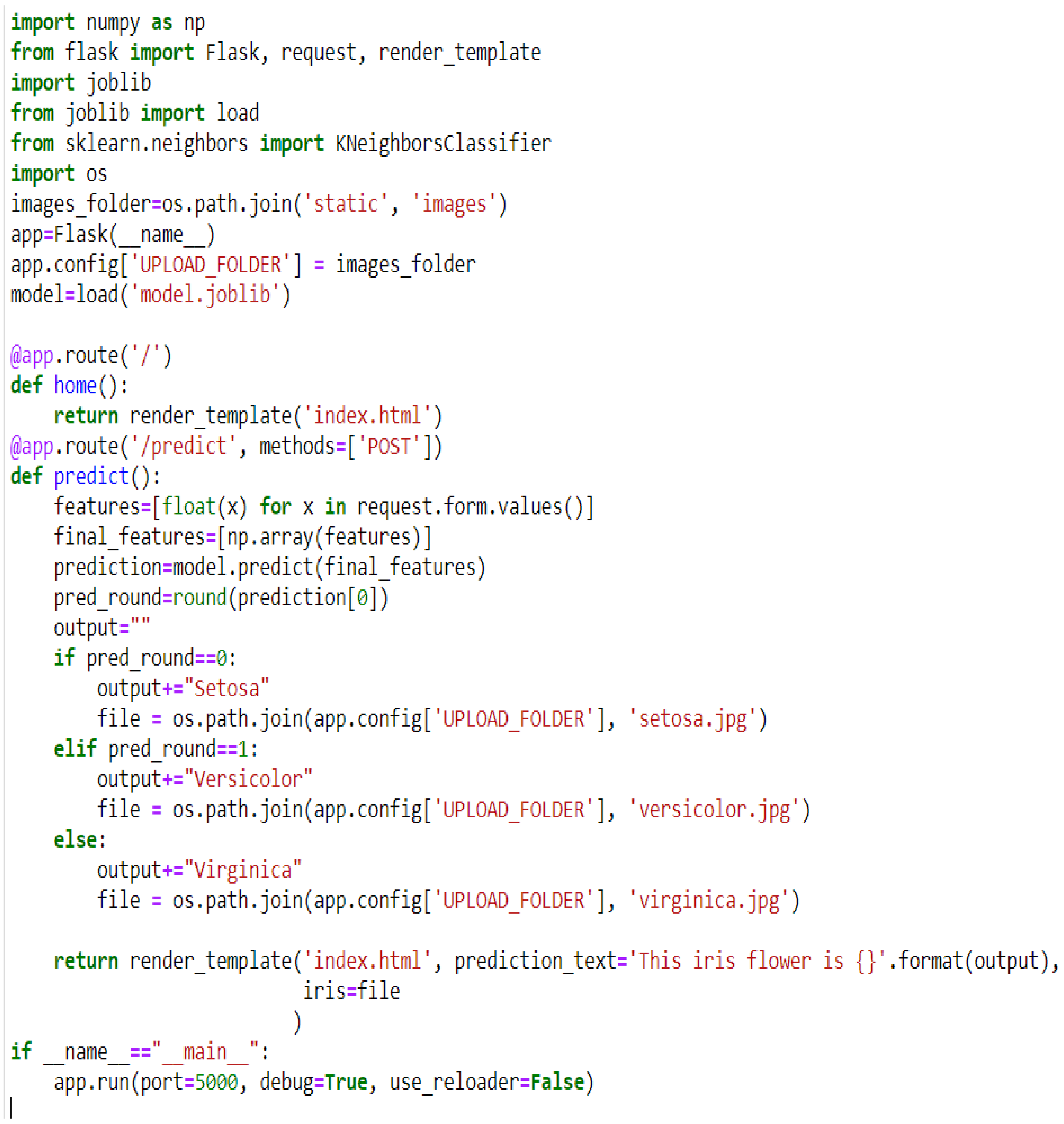
Submitted to: Week 4 Deployment on Flask (on canvas)

Github : https://github.com/ManojN7270/Deployment-on-flask.git



```
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)
```

Use flask so the web app can be deployed locally. Images are included in the app:



```python
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)
```