VIRTUAL DATA INTERNSHIP

NAME: ONIFADE MICHAEL

BATCH CODE: LISUM32

DATE: 29TH APRIL 2024

SUBMITTED TO: DATA GLACIER

Introduction

This project aims at the deployment of a machine learning model for data that expresses the quality of wine using a Flask framework. The proposed workflow is shown in diagram 1.1 while the data information is showcased in table 1.1

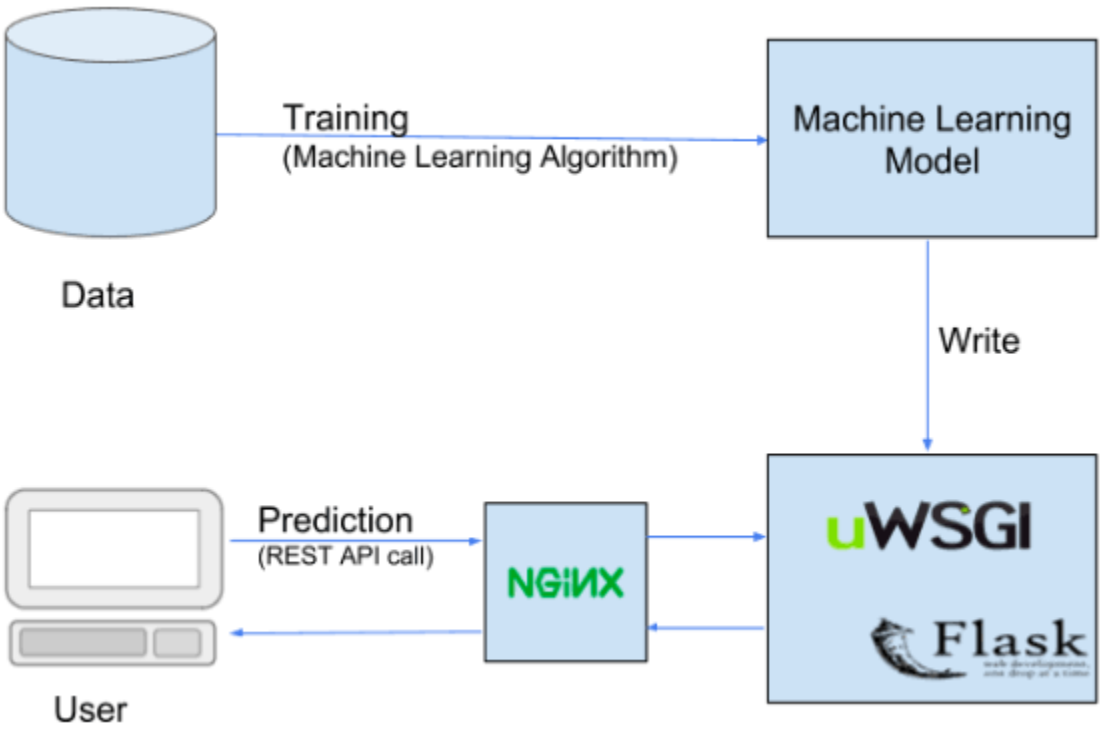


Diagram 1.1 Application Workflow

Table Data Details

Total number of observations	1599
Total number of files	1
Total number of features	12
Base format of the file	CSV
Size of the data	84KB

Table 1.1 Dataset Information

The process for the deployment is summarized as follows

1) Data import

```
import pandas as pd

winequality = pd.read_csv('https://raw.githubusercontent.com/themichaelonifade/DataGlacier_Virtual_Internship/main/Week%204/winequality-red.csv')

winequality
```

	fixedacidity	volatileacidity	citricacid	residualsugar	chlorides	freesulfurdioxide	totalsulfurdioxide	density	ph	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1599 rows x 12 columns

Next steps: Generate code with winequality | View recommended plots

2) Model Building

Build model

```
from sklearn.model_selection import train_test_split
#splitting data into train and test sets

x = winequality.drop(['quality'], axis=1)
y = winequality['quality']


X_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=100)
print(X_train.shape, x_test.shape)
```

```
(1279, 11) (320, 11)
```

Linear Regression

```
[ ] from sklearn.linear_model import LinearRegression

    lr = LinearRegression()
    lr.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
[ ] from sklearn.metrics import mean_squared_error

    predictions = lr.predict(x_test)
    print("Mean Squared Error:", mean_squared_error(y_test, predictions))
```

```
Mean Squared Error: 0.42823367373739396
```

3) Turning Model into web application using flask

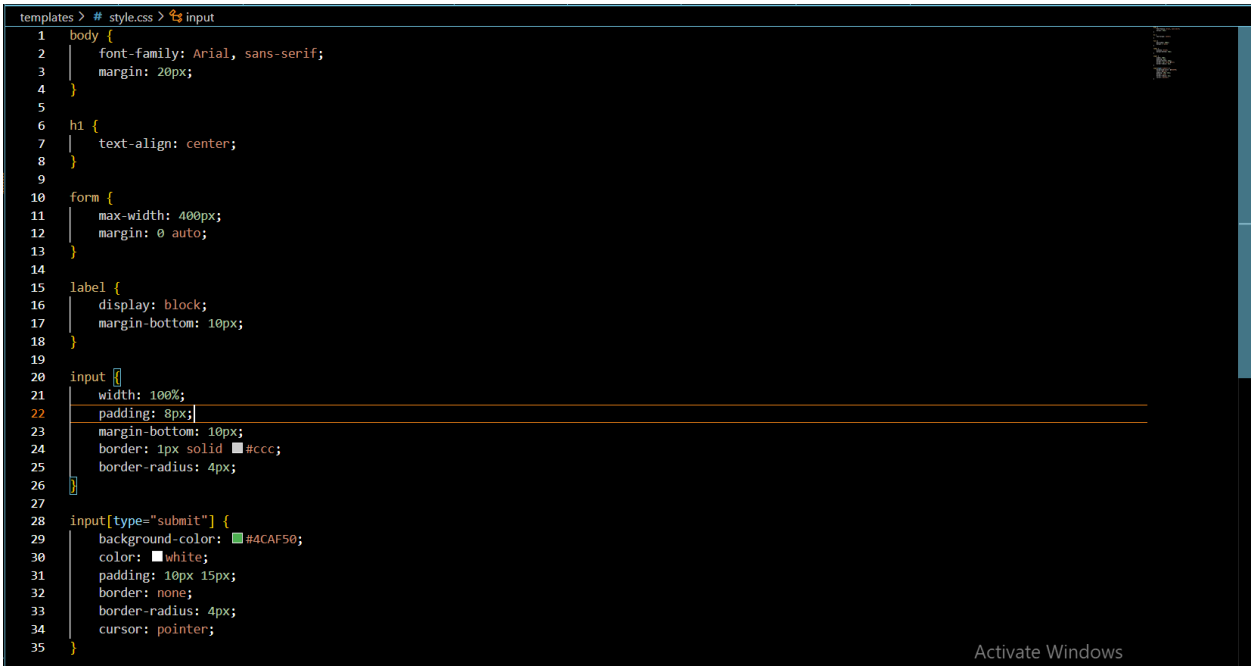
App.py file

app.py > home

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

# Load the trained model
model = pickle.load(open('model.pkl', 'rb'))

app = Flask(__name__)

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

@app.route('/predict', methods=['POST'])
def predict():
  if request.method == 'POST':
    fixedacidity = request.form.get('fixedacidity'),
    volatileacidity = request.form.get('volatileacidity'),
    citricacid = request.form.get('citricacid'),
    residualsugar =request.form.get('residualsugar'),
    chlorides = request.form.get('chlorides'),
    freesulfurdioxide = request.form.get('freesulfurdioxide'),
    totalsulfurdioxide = request.form.get('totalsulfurdioxide'),
    density = request.form.get('density'),
    ph = request.form.get('ph'),
    sulphates = request.form.get('sulphates'),
    alcohol = request.form.get('alcohol')
```

```python

    fixedacidity = float(fixedacidity[0]) if fixedacidity is not None and len(fixedacidity) > 0 else 0
    volatileacidity = float(volatileacidity[0]) if volatileacidity is not None and len(volatileacidity) else 0
    citricacid = float(citricacid[0]) if citricacid is not None and len(citricacid) else 0
    residualsugar = float(residualsugar[0]) if residualsugar is not None and len(residualsugar) else 0
    chlorides = float(chlorides[0]) if chlorides is not None and len(chlorides) else 0
    freesulfurdioxide = int(freesulfurdioxide[0]) if freesulfurdioxide is not None and len(freesulfurdioxide) else 0
    totalsulfurdioxide = int(totalsulfurdioxide[0]) if totalsulfurdioxide is not None and len(totalsulfurdioxide) else 0
    density = float(density[0]) if density is not None and len(density) else 0
    ph = float(ph[0]) if ph is not None and len(ph) else 0
    sulphates = float(sulphates[0]) if sulphates is not None and len(sulphates) else 0
    alcohol = float(alcohol[0]) if alcohol is not None and len(alcohol) else 0

    features_arr = np.array([[fixedacidity, volatileacidity,citricacid, residualsugar,chlorides, freesulfurdioxide, totalsulfurdioxide, density, ph,
    prediction = model.predict(features_arr)
    output = round(prediction[0], 2)

    return render_template('result.html', prediction_text=f'The predicted wine quality is: {output}')

if __name__ == '__main__':
    app.debug=True
    app.run()
```

Style.css



Index.html

```
        <label>Chlorides:
            <input type="number" step="any" name="chlorides">
        </label>
        <br><br>

        <label>Free Sulfur Dioxide:
            <input type="number" step="any" name="freesulfurdioxide">
        </label>
        <br><br>

        <label>Total Sulfur Dioxide:
            <input type="number" step="any" name="totalsulfurdioxide">
        </label>
        <br><br>

        <label>Density:
            <input type="number" step="any" name="density">
        </label>
        <br><br>

        <label>pH:
            <input type="number" step="any" name="ph">
        </label>
        <br><br>

        <label>Alcohol:
            <input type="number" step="any" name="alcohol">
        </label>
        <br><br>

        <input type="submit" value="Predict">
    </form>
</body>
</html>
```

Result.html

templates > result.html > html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Wine Quality</title>
</head>
<body>
    <h1>Wine Quality</h1>
    <p>{{ prediction_text }}</p>
</body>
</html>
```

Final output