Week 4: Deployment on Flask

Name: Darshan Dhanani

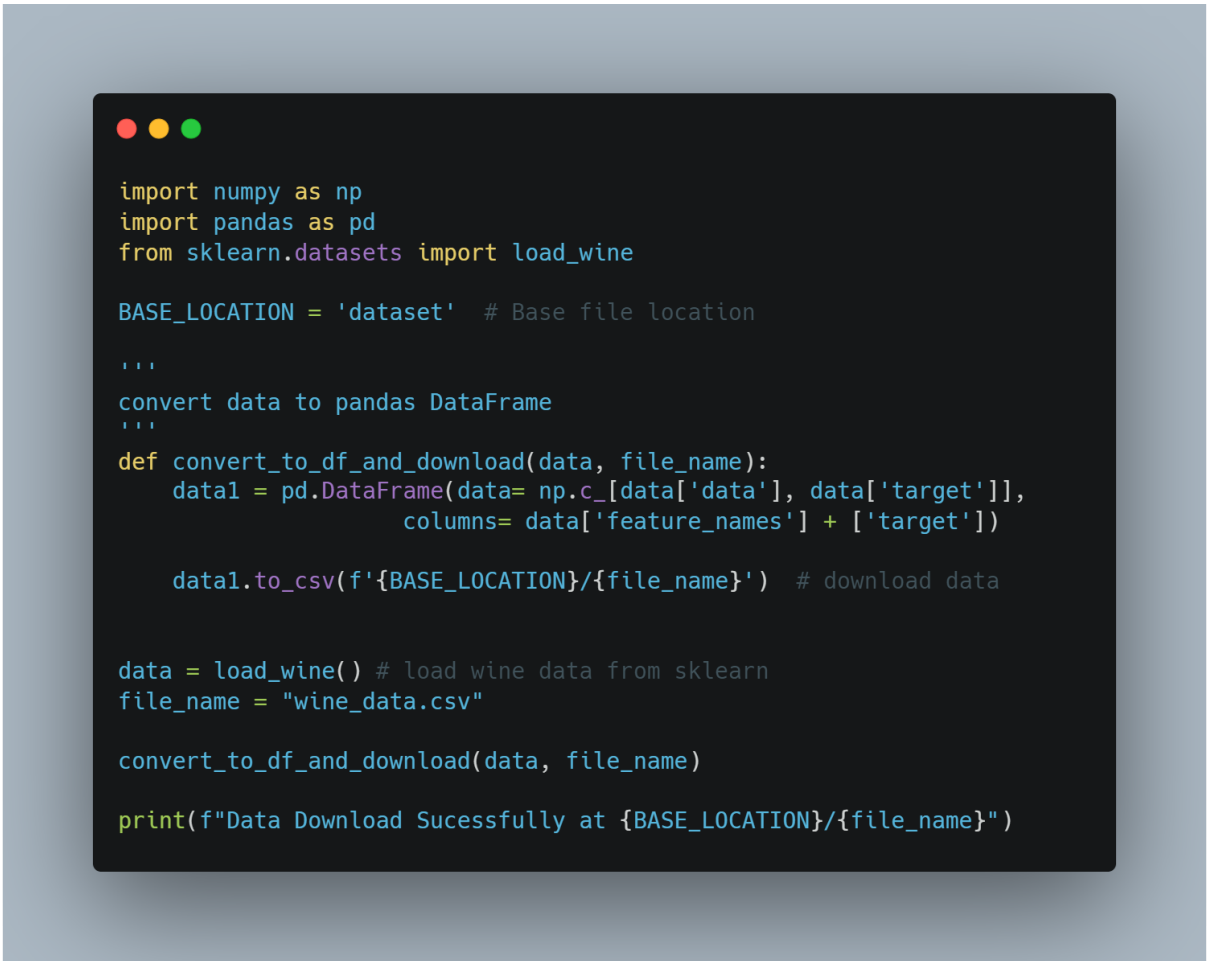
Batch code: LISUM32

Submission date: 28 Apr 2024

Submitted to:

Snapshot of Model Deployment

Step 1: find and download data



```python
import numpy as np
import pandas as pd
from sklearn.datasets import load_wine

BASE_LOCATION = 'dataset'  # Base file location

'''
convert data to pandas DataFrame
'''
def convert_to_df_and_download(data, file_name):
    data1 = pd.DataFrame(data= np.c_[data['data'], data['target']],
                         columns= data['feature_names'] + ['target'])

    data1.to_csv(f'{BASE_LOCATION}/{file_name}')  # download data


data = load_wine() # load wine data from sklearn
file_name = "wine_data.csv"

convert_to_df_and_download(data, file_name)

print(f"Data Download Sucessfully at {BASE_LOCATION}/{file_name}")
```

Step 2: Data Preprocessing

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

dataset = pd.read_csv("../dataset/wine_data.csv")

X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)

# Feature Scaling
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Step 3: Building and Training a Model: Creating a .pkl File for Export

After thorough experimentation with various models, I've chosen to present the K-Neighbors Classifier here.

```python
from sklearn.neighbors import KNeighborsClassifier
from matplotlib.colors import ListedColormap
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
from sklearn.model_selection import cross_val_score
import seaborn as sns
import joblib

def calculate_n_neighbors():
    # Define a range of values for n_neighbors
    neighbors_range = list(range(1, 40))

    # List to store cross-validation scores
    cv_scores = []

    # Perform 10-fold cross-validation for each value of n_neighbors
    for n_neighbors in neighbors_range:
        knn = KNeighborsClassifier(n_neighbors=n_neighbors)
        scores = cross_val_score(knn, X_train, y_train, cv=10, scoring='accuracy')
        cv_scores.append(scores.mean())

    # Find the optimal value of n_neighbors
    optimal_n_neighbors = neighbors_range[np.argmax(cv_scores)]
    return optimal_n_neighbors

# train model
classifier = KNeighborsClassifier(n_neighbors = calculate_n_neighbors(), metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)  # predict data

# create pkl file for external usage of model
joblib.dump(classifier, "pkl/KNeighborsClassifier.pkl")

# accuracy score
accuracy = accuracy_score(y_test, y_pred)

# Calculate precision
precision = precision_score(y_test, y_pred, average='micro')

# Calculate recall
recall = recall_score(y_test, y_pred, average='micro')

print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)

cm = confusion_matrix(y_test, y_pred)  # confusion matrix

 # Labels for classes
class_labels = ['0', '1', '2']

# Plot confusion matrix as heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, cmap='Blues', fmt='g', xticklabels=class_labels, yticklabels=class_labels)
plt.xlabel('Predicted Value')
plt.ylabel('Actual Value')
plt.title('Confusion Matrix(K-Neighbors Classifier)')
plt.show()
```

Step 4: Create and Setup Flask webapp

```python
from flask import Flask, render_template, request
import pandas as pd
import numpy as
import models.predict_signle_row_data_with_multiple_model as pred


app = Flask(__name__, static_url_path="/static", static_folder='static')


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


@app.route("/dataset")
def dataset():
    datas = pd.read_csv('dataset/wine_data.csv').iloc[:, 1:]
    return render_template("dataset.html", datas=datas)


@app.route("/trained_model")
def trained_model():
    return render_template("trained_model.html")


@app.route("/predict_data")
def predict_data():
    return render_template("predict_data.html")


@app.route("/predict", methods=['POST'])
def predict():
    # convert form value into array
    features = [(x) for x in request.form.values()]
    f_features = [np.array(features)]

    # make predication with multiple model
    predicted_data = pred.process_data(f_features)

    return render_template("predict_data.html", datas=[predicted_data, features])

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

Step 5: Create Web Form for Input Data

```
<!doctype html>
<html lang="en" data-bs-theme="auto">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="Darshan Dhanani">
    <title>Predict Data</title>

<link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='dist/css/bootstrap.min.css')
}}">
{% if datas is defined %}
<link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='dist/css/dataTables.bootstrap5.css') }}">
{% endif %}
</head>

<body>
<div class="col-lg-8 mx-auto p-4 py-md-5">
  <header class="d-flex align-items-center pb-3 mb-5 border-bottom">
  <a href="{{ url_for('index') }}"><img src="{{ url_for('static', filename='brand/logo.png') }}"
width="150" height="150" alt="Darshan Dhanani"></a>
  </header>
  <h3 class="text-body-emphasis">Predict Class with provide below details!</h3>
  <br/>

  <!-- form start -->
  <form action = "{{url_for('predict')}}" method ="POST">
  <div class="row">
    <!-- Alcohol -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom01">Alcohol</label>
      <input type="number" class="form-control" id="validationCustom01" placeholder="Alcohol"
name="alcohol" required>
    </div>
    <!-- end Alcohol -->

    <!-- Malic Acid -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom02">Malic Acid</label>
      <input type="number" class="form-control" id="validationCustom02" placeholder="Malic Acid"
name="malic_acid" required>
    </div>
    <!-- end Malic Acid -->

    <!-- Ash -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom03">Ash</label>
        <input type="number" class="form-control" id="validationCustom03" placeholder="Ash" name="ash"
required>
    </div>
    <!-- end Ash -->
  </div>

  <div class="row">
    <!-- Alcalinity of Ash -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom04">Alcalinity of Ash</label>
      <input type="number" class="form-control" id="validationCustom04" placeholder="Alcalinity of Ash"
name="alcalinity_of_ash" required>
    </div>
    <!-- end Alcalinity of Ash -->
```

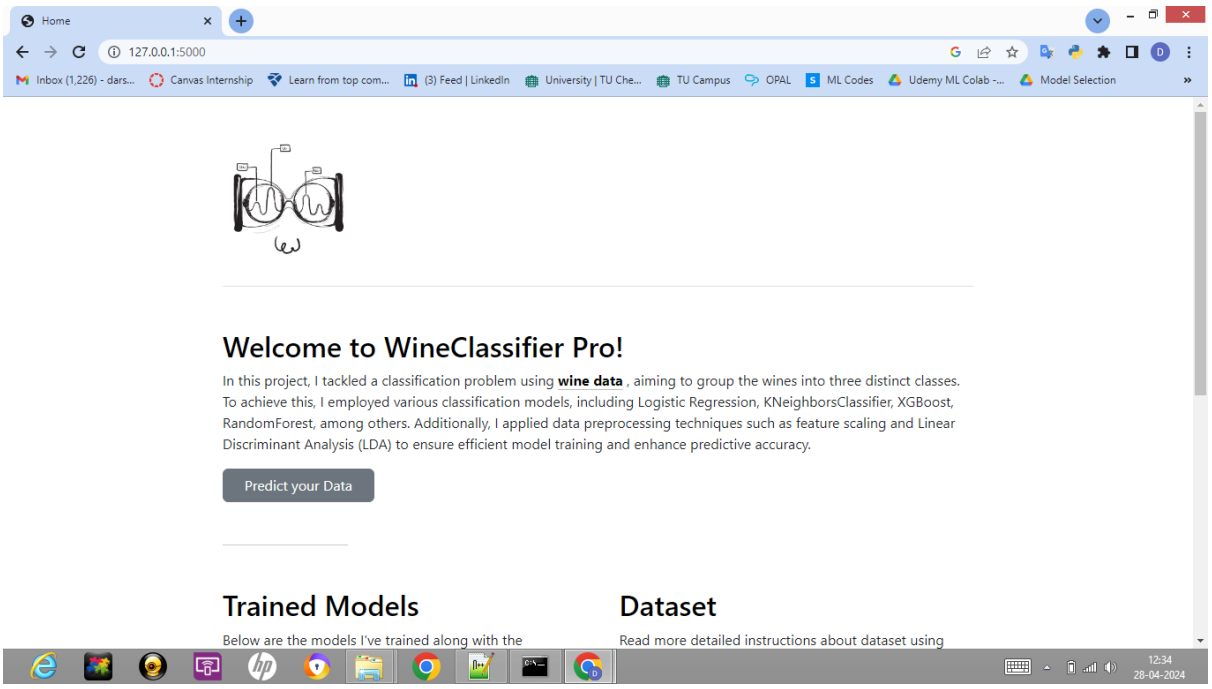
```
    <!-- Magnesium -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom05">Magnesium</label>
      <input type="number" class="form-control" id="validationCustom05" placeholder="Magnesium"
name="magnesium" required>
    </div>
    <!-- end Magnesium -->

    <!-- Total Phenols -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom06">Total Phenols</label>
        <input type="number" class="form-control" id="validationCustom06" placeholder="Total Phenols"
name="total_phenols" required>
    </div>
    <!-- end Total Phenols -->
  </div>

  <div class="row">
    <!-- Flavanoids -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom07">Flavanoids</label>
      <input type="number" class="form-control" id="validationCustom07" placeholder="Flavanoids"
name="flavanoids" required>
    </div>
    <!-- end Alcohol -->

    <!-- Nonflavanoid Phenols -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom08">Nonflavanoid Phenols</label>
      <input type="number" class="form-control" id="validationCustom08" placeholder="Nonflavanoid
Phenols" name="nonflavanoid_phenols" required>
    </div>
    <!-- end Nonflavanoid Phenols -->

    <!-- Proanthocyanins -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom09">Proanthocyanins</label>
        <input type="number" class="form-control" id="validationCustom09" placeholder="Proanthocyanins"
name="proanthocyanins" required>
    </div>
    <!-- end Proanthocyanins -->
  </div>

  <div class="row">
    <!-- Color Intensity -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom10">Color Intensity</label>
      <input type="number" class="form-control" id="validationCustom10" placeholder="Color Intensity"
name="color_intensity" required>
    </div>
    <!-- end Color Intensity -->

    <!-- Hue -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom11">Hue</label>
      <input type="number" class="form-control" id="validationCustom11" placeholder="Hue" name="hue"
required>
    </div>
    <!-- end Hue -->

    <!-- od280/od315 of Diluted Wines -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom12">od280/od315 of Diluted Wines</label>
        <input type="number" class="form-control" id="validationCustom12" placeholder="od280/od315 of
Diluted Wines" name="od280/od315_of_diluted_wines" required>
    </div>
    <!-- end od280/od315 of Diluted Wines -->
  </div>

  <div class="row">
    <!-- Proline -->
    <div class="col-md-4 mb-3">
      <label class="mb-2" for="validationCustom13">Proline</label>
      <input type="number" class="form-control" id="validationCustom13" placeholder="Proline"
name="proline" required>
    </div>
    <!-- end Proline -->

  </div>
  <button class="btn btn-primary" type="submit">Submit form</button>
</form>
<!-- form end -->

</div>
</body>
<script src="{{ url_for('static', filename='/static/dist/js/bootstrap.bundle.min.js') }}"></script>

</html>
```

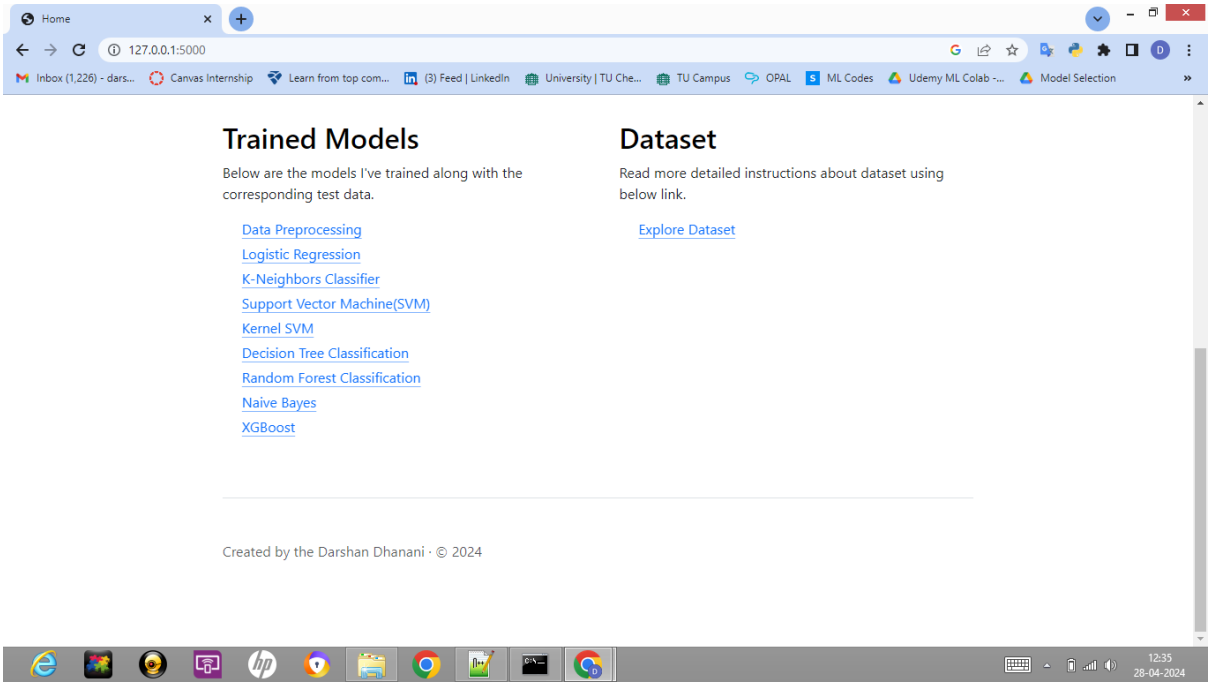
Step 6: Visualize predictions

```
<!-- target result -->
<h2 class="text-body-emphasis">Target result</h2>
        <p>The models are trained to classify wines into three classes: class 0, class 1, and class 2.
You can find below result of predictiong wine class using various pre-trained models.<p>


<table class="table table-sm table-condensed table-striped" style="width: 300px;">
        <tbody>
        {% for i in datas[0] %}
        <tr>
            <td>{{ i[0] }}</td>
            <td>{{ i[1][0] }}</td>
        </tr>
        {% endfor %}
  </tbody>
</table>
<!-- traget result end -->

<hr class="featurette-divider"/>
<br>

<!-- submitted data -->
<h2 class="text-body-emphasis">Submitted Data</h2>

<table id="dataset" class="table table-striped" style="width:100%">
        <thead>
            <tr>
            <th>alcohol</th>
            <th>malic_acid</th>
            <th>ash</th>
            <th>alcalinity_of_ash</th>
            <th>magnesium</th>
            <th>total_phenols</th>
            <th>flavanoids</th>
            <th>nonflavanoid_phenols</th>
            <th>proanthocyanins</th>
            <th>color_intensity</th>
            <th>hue</th>
            <th>od280/od315_of_diluted_wines</th>
            <th>proline</th>
            </tr>
        </thead>
        <tbody>

            <tr>
            {% for data in datas[1] %}
                <td>{{ data }}</td>
                {% endfor %}
            </tr>

            </tbody>
            </table>
<!-- submitted data end -->

<script src="{{ url_for('static', filename='dist/js/jquery-3.7.1.js') }}"></script>
<script src="{{ url_for('static', filename='dist/js/bootstrap.bundle.min.js') }}"></script>
<script src="{{ url_for('static', filename='dist/js/dataTables.js') }}"></script>
<script src="{{ url_for('static', filename='dist/js/dataTables.bootstrap5.js') }}"></script>

<script>
// datatable
new DataTable('#dataset', {
    scrollX: true
});

// scroll down page when load
function Scrolldown() {
     window.scroll(0,800);
}

window.onload = Scrolldown;

</script>
```

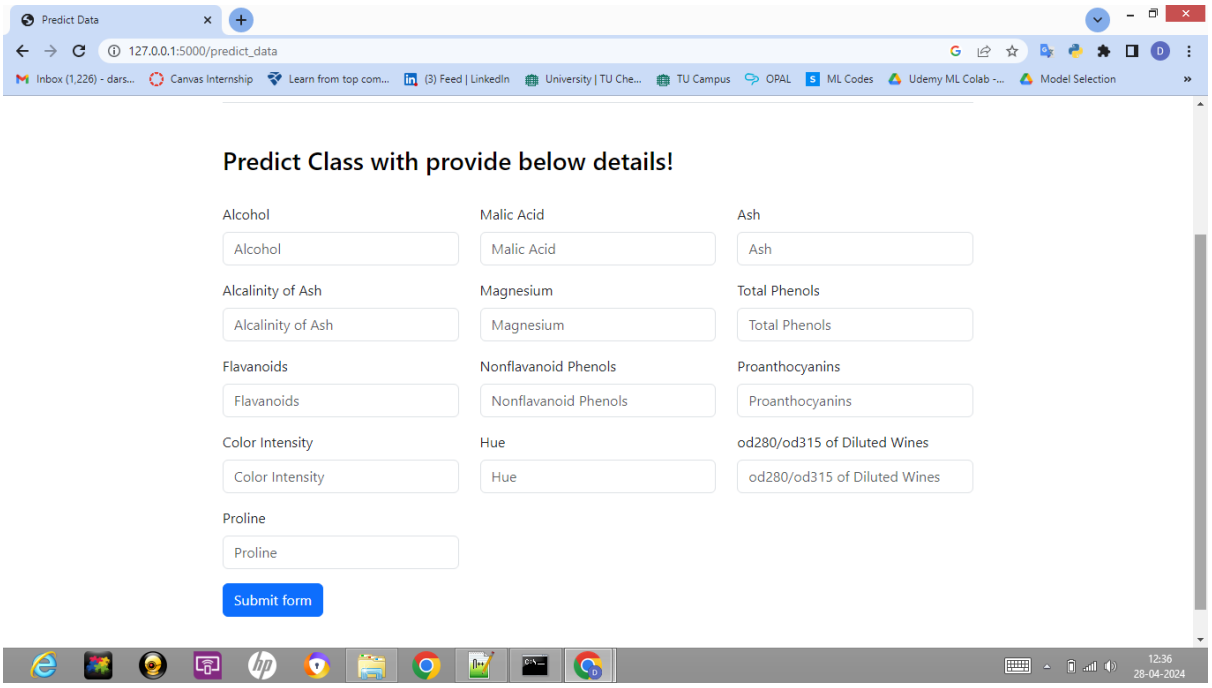
Snapshot of web page

Index Page





Input Form


Predict Data
127.0.0.1:5000/predict_data
Predict Class with provide below details!
Alcohol
Malic Acid
Ash
Alcalinity of Ash
Magnesium
Total Phenols
Flavanoids
Nonflavanoid Phenols
Proanthocyanins
Color Intensity
Hue
od280/od315 of Diluted Wines
Proline
Submit form

Result Prediction