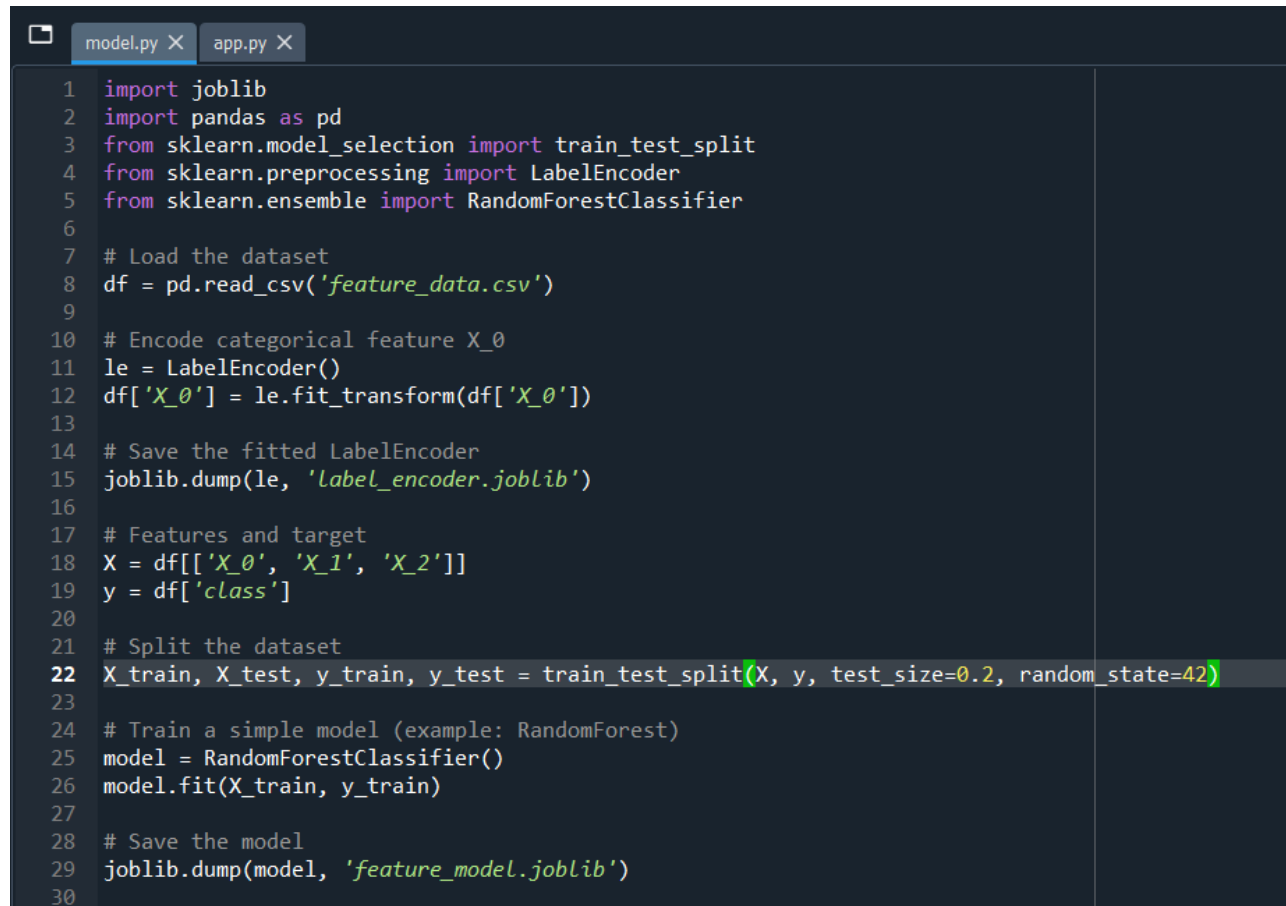


Name: Ayush Jha
Internship Batch: LISUM30
Submitted by: Ayush Jha
Submitted to: Data Glacier

Model Creation:

A screenshot of a code editor with two tabs: 'model.py' and 'app.py'. The 'model.py' tab is active, displaying Python code for model creation. The code includes imports for joblib, pandas, sklearn's train_test_split, LabelEncoder, and RandomForestClassifier. It shows the process of loading a dataset from 'feature_data.csv', encoding a categorical feature 'X_0', saving the LabelEncoder, splitting the dataset into training and testing sets, training a RandomForestClassifier, and finally saving the trained model as 'feature_model.joblib'. Line 22 is highlighted.

```
1 import joblib
2 import pandas as pd
3 from sklearn.model_selection import train_test_split
4 from sklearn.preprocessing import LabelEncoder
5 from sklearn.ensemble import RandomForestClassifier
6
7 # Load the dataset
8 df = pd.read_csv('feature_data.csv')
9
10 # Encode categorical feature X_0
11 le = LabelEncoder()
12 df['X_0'] = le.fit_transform(df['X_0'])
13
14 # Save the fitted LabelEncoder
15 joblib.dump(le, 'label_encoder.joblib')
16
17 # Features and target
18 X = df[['X_0', 'X_1', 'X_2']]
19 y = df['class']
20
21 # Split the dataset
22 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
23
24 # Train a simple model (example: RandomForest)
25 model = RandomForestClassifier()
26 model.fit(X_train, y_train)
27
28 # Save the model
29 joblib.dump(model, 'feature_model.joblib')
30
```

HTML Creation:

```
File Edit View

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Feature Model Prediction</title>
  <style>
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f5f5f5;
      margin: 0;
      padding: 0;
      text-align: center;
    }
    h1 {
      color: #333;
    }
    form {
      max-width: 400px;
      margin: 20px auto;
      background-color: #fff;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    label {
      display: block;
      margin-bottom: 8px;
      color: #555;
    }
    select,
    input {
      width: 100%;
      padding: 10px;
      margin-bottom: 15px;
      box-sizing: border-box;
      border: 1px solid #ccc;
      border-radius: 4px;
      font-size: 16px;
    }
  </style>
</head>
<body>
  <h1>Feature Model Prediction</h1>
  <form>
    <label>Feature</label>
    <input type="text">
    <label>Model</label>
    <input type="text">
    <label>Prediction</label>
    <input type="text">
  </form>
</body>
</html>
```

App creation

```
model.py X app.py X
1 from flask import Flask, render_template, request
2 import joblib
3
4 app = Flask(__name__)
5
6 # Load the saved model
7 model = joblib.load('feature_model.joblib')
8
9 # Load the saved LabelEncoder
10 le = joblib.load('label_encoder.joblib')
11
12 @app.route('/')
13 def home():
14     return render_template('index.htm')
15
16 @app.route('/predict', methods=['POST'])
17 def predict():
18     # Get input from the form
19     x0 = request.form['X_0']
20     x1 = float(request.form['X_1'])
21     x2 = float(request.form['X_2'])
22
23     # Encode X_0 using the saved LabelEncoder
24     x0_encoded = le.transform([x0])[0]
25
26     # Use the model for prediction
27     prediction = model.predict([[x0_encoded, x1, x2]])
28
29     return render_template('result.htm', prediction=prediction[0])
30
31 if __name__ == '__main__':
32     app.run(debug=True)
33
```

Result:

Feature Model Prediction

X_0 (Categorical):
Kind 1

X_1:
11

X_2:
12

Predict

Prediction Result

The predicted label is: 1

[Go back to input](#)