

# Project Report Format

## 1. INTRODUCTION

### 1.1 Project Overview

Startups play a crucial role in driving innovation, employment, and economic growth. However, predicting whether a startup will succeed or fail is a complex task due to multiple influencing factors such as funding history, market conditions, business category, and operational strategy. Traditional evaluation methods rely heavily on intuition and manual analysis, which are often time-consuming and subjective. **Prosperity Prognosticator** is a machine learning–based solution designed to predict startup success using structured startup data. By leveraging classification algorithms such as Random Forest, this project provides an efficient, scalable, and data-driven approach for startup evaluation, helping investors and entrepreneurs make informed decisions.

### 1.2 Purpose

The core purpose of this project is to automate and improve startup success prediction using machine learning techniques.

This project aims to:

- Reduce dependency on manual startup evaluation.
- Improve accuracy and consistency in predicting startup outcomes.
- Enable quick and data-driven decision-making.
- Provide a user-friendly web interface for real-time predictions.

Ultimately, the project bridges artificial intelligence and business analytics to create an intelligent decision-support system that is practical, reliable, and scalable.

## 2. IDEATION PHASE

### 2.1 Problem Statement

### 2.2 Empathy Map Canvas

### 2.3 Brainstorming

## 3. REQUIREMENT ANALYSIS

### 3.1 Solution Requirement

### 3.2 Data Flow Diagram

### 3.3 Technology Stack

## 4. PROJECT DESIGN

### 4.1 Problem Solution Fit

### 4.2 Proposed Solution

### 4.3 Solution Architecture

## 5. PROJECT PLANNING & SCHEDULING

### 5.1 Project Planning

## 6. FUNCTIONAL AND PERFORMANCE TESTING

### 6.1 Performance Testing

#### **Model Results (from your project):**

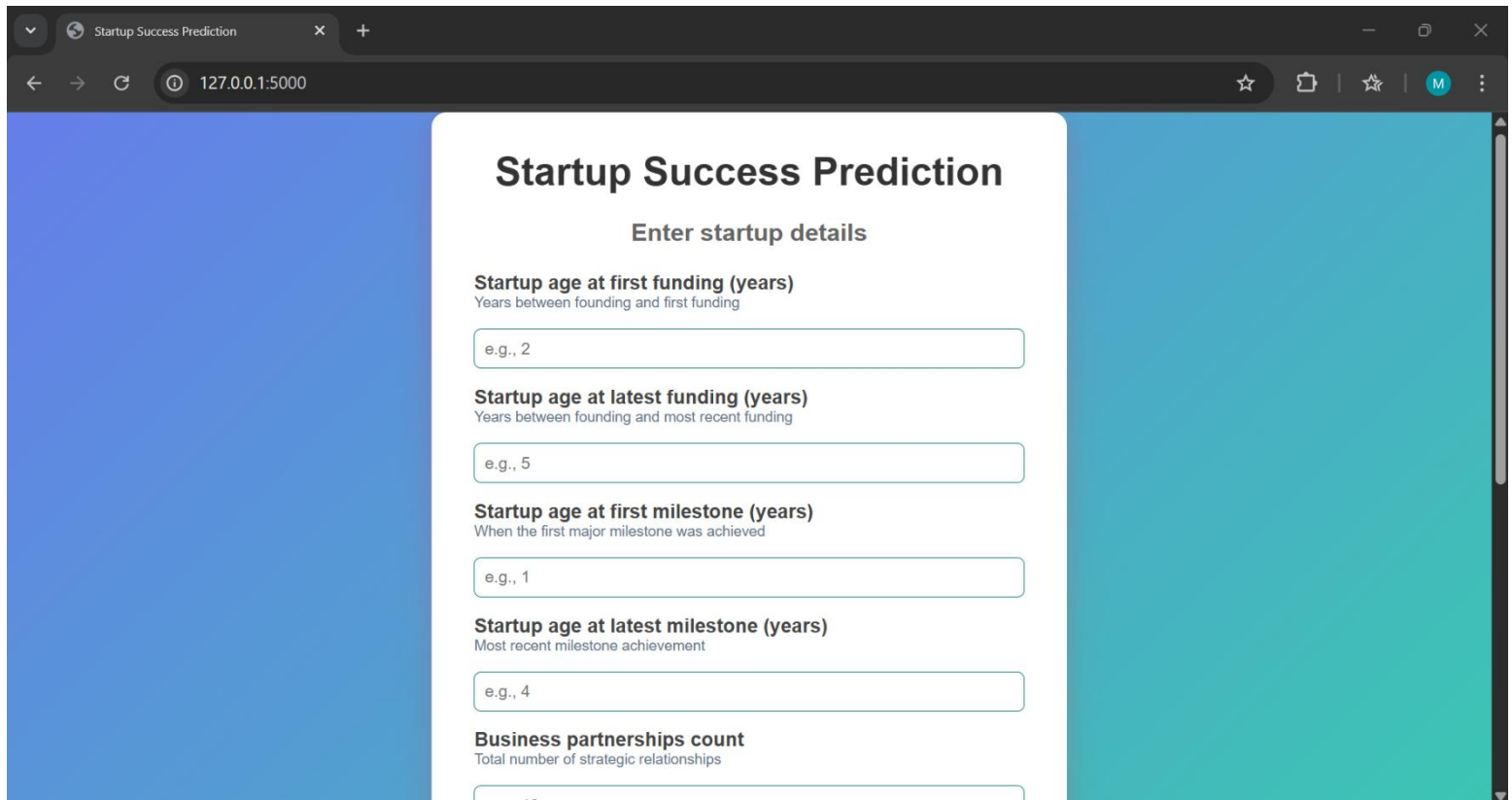
- Training Accuracy: **100%**
- Testing Accuracy: **79%**
- ROC-AUC Score: **0.745**
- Precision-Recall AUC: **0.883**

#### **Classification Summary**

- Strong performance in predicting successful startups.
- Slight overfitting observed due to high training accuracy.

## 7. RESULTS

### 7.1 Output Screenshots



A screenshot of a web browser displaying a form titled "Startup Success Prediction". The browser's address bar shows "127.0.0.1:5000". The form is centered on a white background with blue and teal sidebars. It contains five input fields, each with a label, a description, and a placeholder value. The fields are: "Startup age at first funding (years)" with placeholder "e.g., 2", "Startup age at latest funding (years)" with placeholder "e.g., 5", "Startup age at first milestone (years)" with placeholder "e.g., 1", "Startup age at latest milestone (years)" with placeholder "e.g., 4", and "Business partnerships count" with placeholder "e.g., 10".

**Startup Success Prediction**

Enter startup details

**Startup age at first funding (years)**  
Years between founding and first funding  
e.g., 2

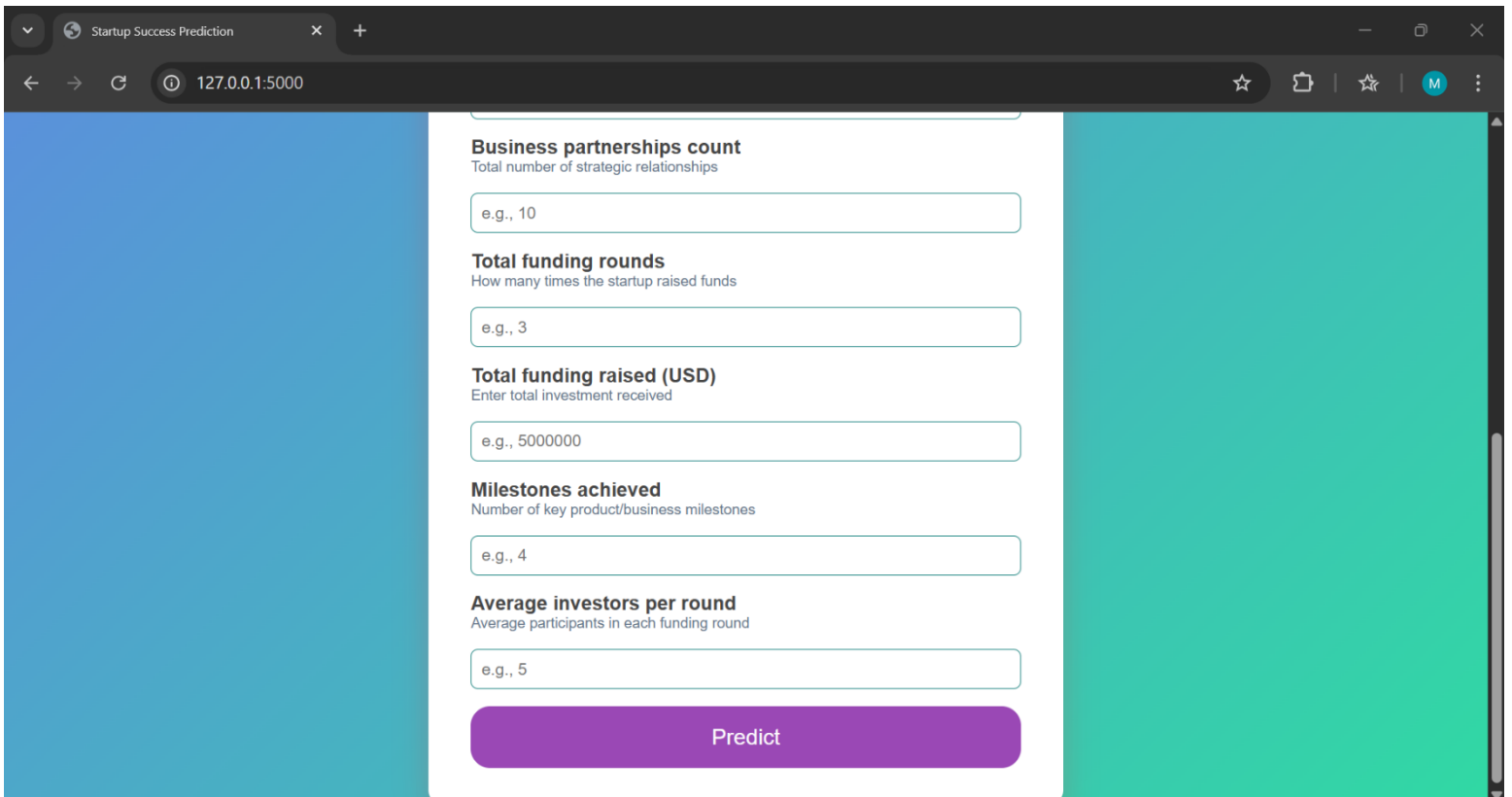
**Startup age at latest funding (years)**  
Years between founding and most recent funding  
e.g., 5

**Startup age at first milestone (years)**  
When the first major milestone was achieved  
e.g., 1

**Startup age at latest milestone (years)**  
Most recent milestone achievement  
e.g., 4

**Business partnerships count**  
Total number of strategic relationships  
e.g., 10

*Figure 7.1*



A screenshot of the same web browser displaying the "Startup Success Prediction" form, showing the bottom section. It contains three input fields and a "Predict" button. The fields are: "Business partnerships count" with placeholder "e.g., 10", "Total funding rounds" with placeholder "e.g., 3", and "Total funding raised (USD)" with placeholder "e.g., 5000000". Below these is a section for "Milestones achieved" with a description "Number of key product/business milestones" and a placeholder "e.g., 4". At the bottom is a section for "Average investors per round" with a description "Average participants in each funding round" and a placeholder "e.g., 5". A large purple "Predict" button is at the very bottom.

**Business partnerships count**  
Total number of strategic relationships  
e.g., 10

**Total funding rounds**  
How many times the startup raised funds  
e.g., 3

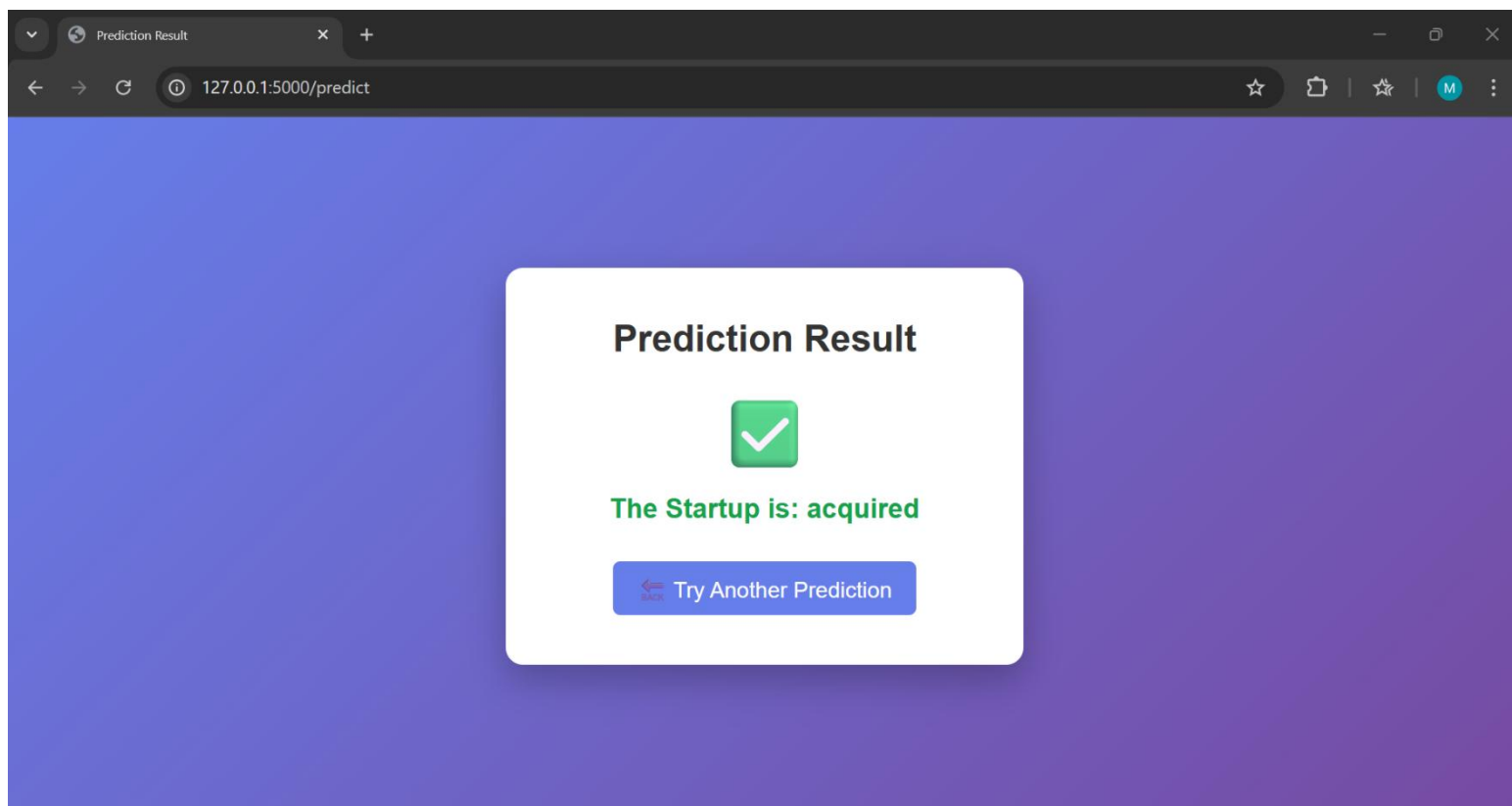
**Total funding raised (USD)**  
Enter total investment received  
e.g., 5000000

**Milestones achieved**  
Number of key product/business milestones  
e.g., 4

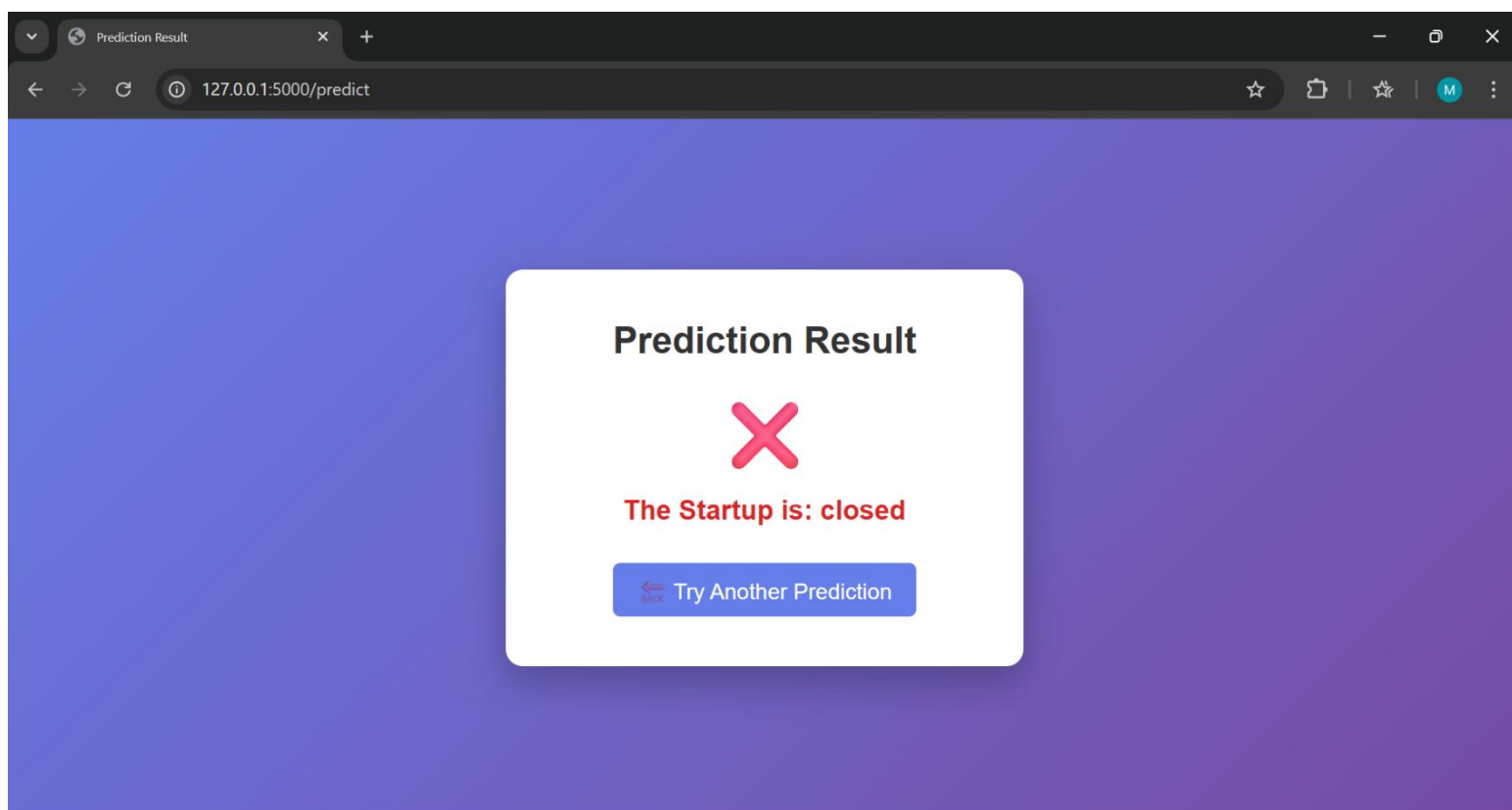
**Average investors per round**  
Average participants in each funding round  
e.g., 5

**Predict**

*Figure 7.2*



*Figure 7.3*



*Figure 7.4*

```

In [ ]: from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()

rf.fit(X_train._get_numeric_data(),y_train)

y_pred_rf = rf.predict(X_test._get_numeric_data())

print("Training Accuracy :", rf.score(X_train._get_numeric_data(), y_train))
print("Testing Accuracy :", rf.score(X_test._get_numeric_data(), y_test))

cm = confusion_matrix(y_test, y_pred_rf)
plt.rcParams['figure.figsize'] = (3, 3)
sns.heatmap(cm, annot = True, cmap = 'YlGnBu', fmt = '.8g')
plt.show()

cr = classification_report(y_test, y_pred_rf)
print(cr)

print("-----")

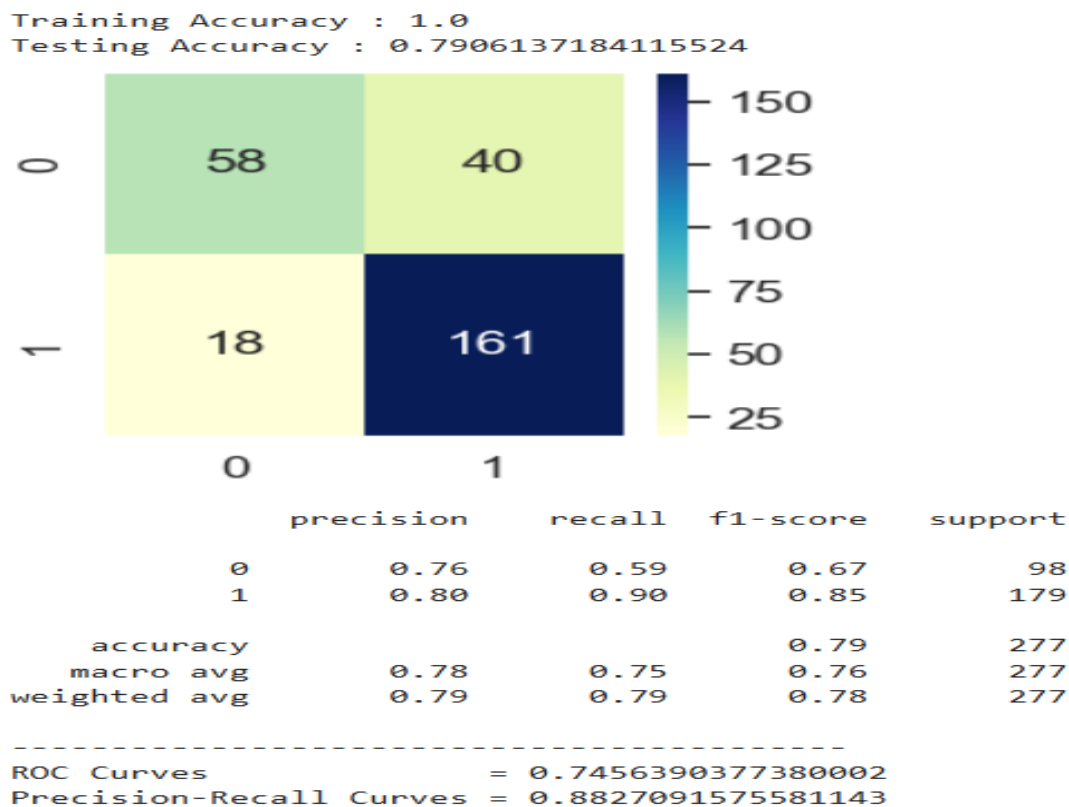
false_positive_rate, true_positive_rate, thresholds = roc_curve(y_test,y_pred_rf)
roc_auc = auc(false_positive_rate, true_positive_rate)
print("ROC Curves          =",roc_auc)

precision, recall, thresholds = precision_recall_curve(y_test, y_pred_rf)
f1 = f1_score(y_test, y_pred_rf)
Precision_Recall_rfs = auc(recall, precision)
print("Precision-Recall Curves =",Precision_Recall_rfs)

```

Training Accuracy : 1.0  
Testing Accuracy : 0.7906137184115524

**Figure 7.5**



**Figure 7.6**

## 8. ADVANTAGES & DISADVANTAGES

### 8.1 Advantages

- Fast and data-driven decision support
- Easy-to-use web interface
- Scalable to larger startup datasets
- Reduces subjective investment decisions

### 8.2 Disadvantages

- Depends on dataset quality
- Slight overfitting observed
- Limited features may affect generalization

## 9. CONCLUSION

The project **Prosperity Prognosticator – Startup Success Prediction** demonstrates how machine learning can effectively support business decision-making. By using Random Forest classification, the system provides reliable predictions for startup success and failure. The solution bridges the gap between traditional analysis and AI-driven insights, enabling faster and smarter decisions for investors and entrepreneurs.

## 10. FUTURE SCOPE

- Integrate larger datasets for better accuracy
- Cloud deployment for public access
- Add user authentication
- Introduce advanced models like XGBoost or Neural Networks
- Provide downloadable analytical reports

## 11. APPENDIX

### Source Code

This appendix contains key parts of the source code used in the project "**Prosperity Prognosticator – Startup Success Prediction**". It includes the backend logic, model training script, and frontend form design.

#### **A. app.py — Flask Backend**

##### **Handles:**

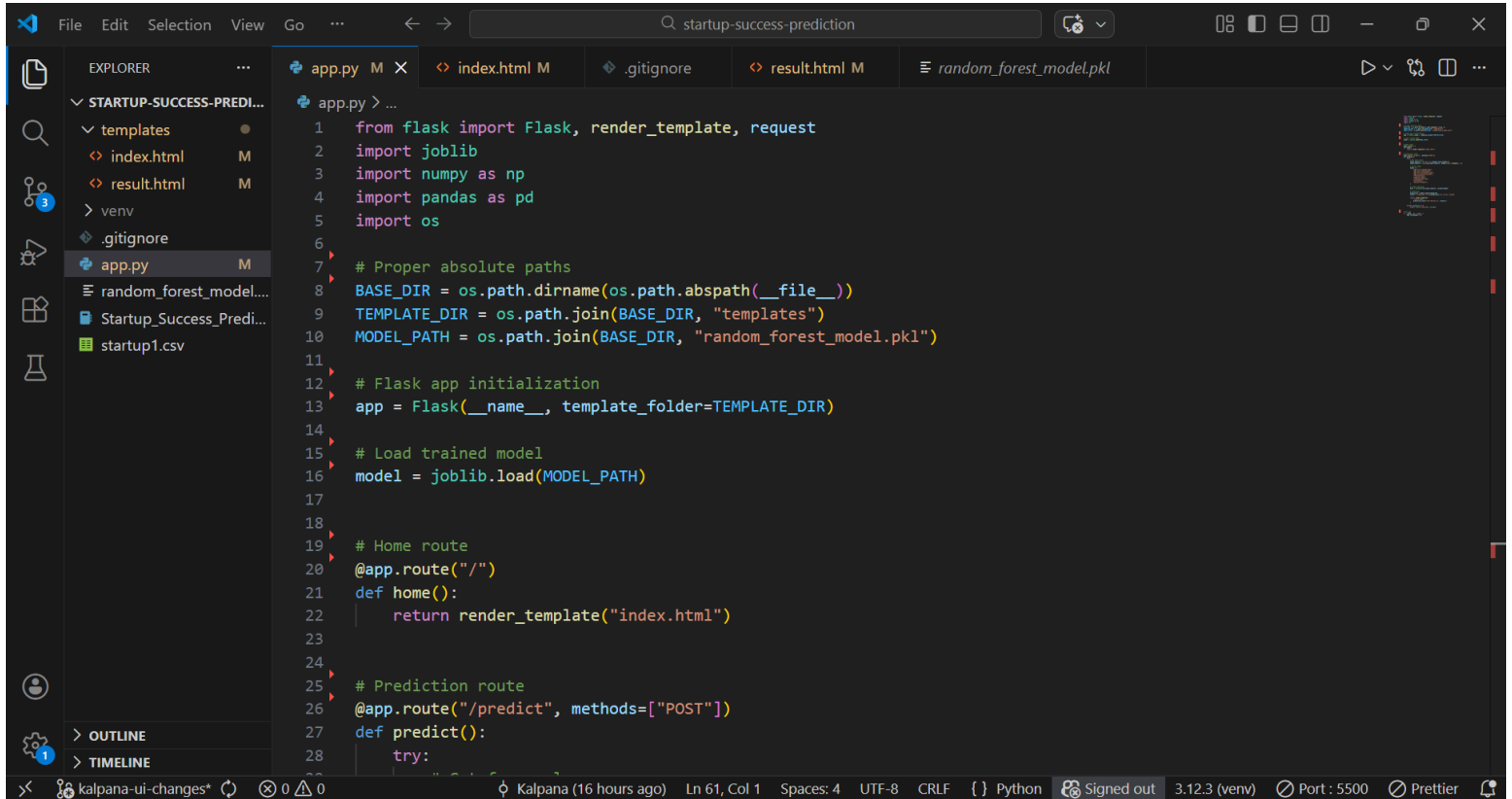
- Route definitions
- Form input
- Model loading
- Prediction and result rendering

#### **B. startup\_model.ipynb — Model Training**

##### **Includes:**

- **Data preprocessing**
- **Model building (Random Forest)**
- **Evaluation metrics**

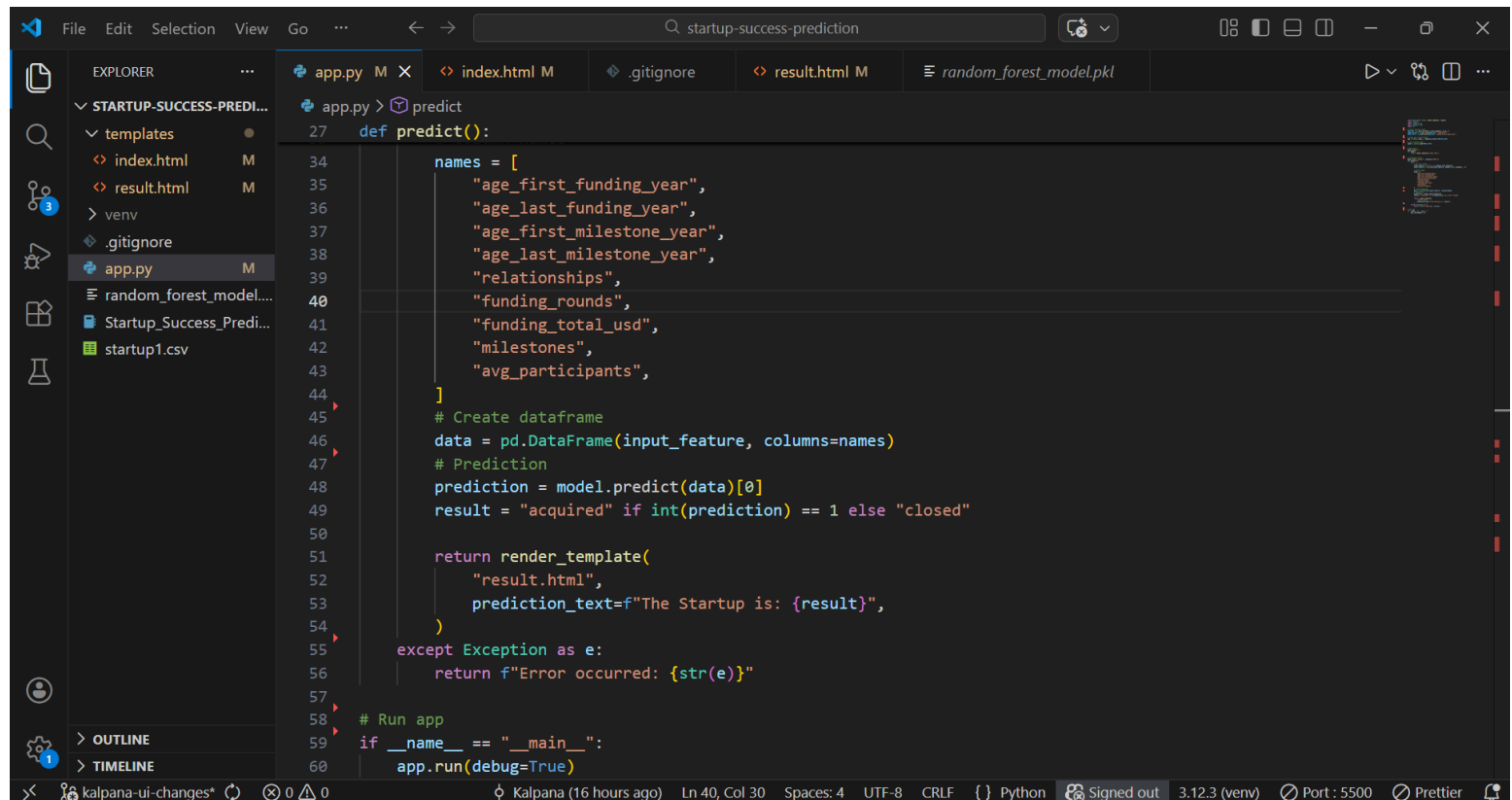
## ○ Model saving



The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows a project named 'STARTUP-SUCCESS-PREDI...' with subfolders 'templates' and 'venv', and files 'index.html', 'result.html', '.gitignore', 'app.py', 'random\_forest\_model.pkl', 'Startup\_Success\_Predi...', and 'startup1.csv'. The 'app.py' file is open in the editor, showing the following code:

```
1 from flask import Flask, render_template, request
2 import joblib
3 import numpy as np
4 import pandas as pd
5 import os
6
7 # Proper absolute paths
8 BASE_DIR = os.path.dirname(os.path.abspath(__file__))
9 TEMPLATE_DIR = os.path.join(BASE_DIR, "templates")
10 MODEL_PATH = os.path.join(BASE_DIR, "random_forest_model.pkl")
11
12 # Flask app initialization
13 app = Flask(__name__, template_folder=TEMPLATE_DIR)
14
15 # Load trained model
16 model = joblib.load(MODEL_PATH)
17
18 # Home route
19 @app.route("/")
20 def home():
21     return render_template("index.html")
22
23
24 # Prediction route
25 @app.route("/predict", methods=["POST"])
26 def predict():
27     try:
```

The status bar at the bottom shows 'Kalpana (16 hours ago)', 'Ln 61, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', '{} Python', 'Signed out', '3.12.3 (venv)', 'Port: 5500', and 'Prettier'.



The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows a project named 'STARTUP-SUCCESS-PREDI...' with subfolders 'templates' and 'venv', and files 'index.html', 'result.html', '.gitignore', 'app.py', 'random\_forest\_model.pkl', 'Startup\_Success\_Predi...', and 'startup1.csv'. The 'app.py' file is open in the editor, showing the following code:

```
27 def predict():
28
29     names = [
30         "age_first_funding_year",
31         "age_last_funding_year",
32         "age_first_milestone_year",
33         "age_last_milestone_year",
34         "relationships",
35         "funding_rounds",
36         "funding_total_usd",
37         "milestones",
38         "avg_participants",
39     ]
40
41     # Create dataframe
42     data = pd.DataFrame(input_feature, columns=names)
43
44     # Prediction
45     prediction = model.predict(data)[0]
46     result = "acquired" if int(prediction) == 1 else "closed"
47
48     return render_template(
49         "result.html",
50         prediction_text=f"The Startup is: {result}",
51     )
52
53 except Exception as e:
54     return f"Error occurred: {str(e)}"
55
56
57 # Run app
58 if __name__ == "__main__":
59     app.run(debug=True)
```

The status bar at the bottom shows 'Kalpana (16 hours ago)', 'Ln 40, Col 30', 'Spaces: 4', 'UTF-8', 'CRLF', '{} Python', 'Signed out', '3.12.3 (venv)', 'Port: 5500', and 'Prettier'.

Dataset Link: [datasetlink](#)

GitHub Link: <https://github.com/Sowmyadiviti/startup-success-prediction>

Project Demo Link: [Videodemolink](#)