Configuration Manual for Supply Chain Optimization Research Study

MSc Research Project
Artificial Intelligence

Sikharam Sai Naga Charan
Student ID: x23141867

School of Computing
National College of Ireland

Supervisor: Prof. Muslim Jameel Syed

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	SIKHARAM SAI NAGA CHARAN
Student ID:	X23141867
Programme:	ARTIFICIAL INTELLIGENCE **Year:** 2024
Module:	Msc RESEARCH PROJECT
Lecturer:	Prof . MUSLIM JAMEEL SYED
Submission Due Date:	11/08/2024
Project Title:	Supply Chain Optimization using Data Analytics: Analysis of Inventory Management, Transportation Logistics, and Procurement Processes.
Word Count:	**632** **Page Count: 08**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: SIKHARAM SAI NAGA CHARAN

Date: 12th AUGUST 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual for Supply Chain Optimization Research Study

Sikharam Sai Naga Charan
Student ID: x23141867

1. Introduction

This configuration manual provides a step-by-step guide to set up and execute the code implementation for the research study titled "Supply Chain Optimization using Data Analytics: Analysis of Inventory Management, Transportation Logistics, and Procurement Processes." The study involves applying various machine learning models to optimize supply chain processes, focusing on predictive modeling to enhance decision-making in areas such as inventory management, transportation logistics, and procurement.

This manual is intended to assist researchers and practitioners in replicating the study, ensuring that the environment is correctly configured, and the code implementation is executed successfully.

2. System Specification

To effectively run the code and process the dataset, the following system specifications are recommended:

- Operating System: Windows 10, macOS, or Linux
- Processor: Intel Core i5 or higher (or equivalent)
- RAM: 8 GB or more
- Storage: At least 20 GB of free disk space
- Graphics Card: Not mandatory, but a dedicated GPU (NVIDIA, AMD) will improve performance in some machine learning tasks.

3. Softwares Used:

The following software tools and libraries are required for executing the code:

- **Python:** Version 3.7 or higher
- **Anaconda Distribution (Optional):** Includes Python and several scientific libraries
- **Jupyter Notebook:** For code execution and analysis
- **Libraries:**
  - **Pandas:** Data manipulation and analysis (pip install pandas)
  - **NumPy:** Numerical computing (pip install numpy)
  - **Matplotlib:** Data visualization (pip install matplotlib)
  - **Seaborn:** Statistical data visualization (pip install seaborn)
  - **Scikit-Learn:** Machine learning library (pip install scikit-learn)
  - **XGBoost:** Gradient boosting library (pip install xgboost)
  - **Plotly:** Interactive visualization library (pip install plotly)

4. Dataset Source

The dataset used in this study is sourced from Kaggle. Follow the steps below to download the dataset:

A. Visit the Kaggle Dataset Page:

- Open your web browser and go to the following link: Transportation and Logistics Tracking Dataset

B. Download the Dataset:

- Sign in to your Kaggle account.
- Click on the "Download" button to download the dataset to your local machine.

C. Extract the Dataset:

- If the dataset is in a compressed format (e.g., ZIP), extract the files to a directory of your choice.

D. Dataset Structure:

- The main dataset file is typically in Excel format (.xlsx). Ensure it is accessible for loading into your analysis environment.

5. Execution of the Code Implementation

Follow the steps below to execute the code implementation:

5.1 Preparing the Environment

A. Open Jupyter Notebook: Launch Jupyter Notebook from Anaconda Navigator or via Command Prompt/Terminal by typing jupyter notebook.

B. Navigate to the Directory: Navigate to the directory where the downloaded dataset is stored.

C. Create a New Notebook: Click on "New" and select "Python 3" to create a new notebook.

5.2 Loading the Dataset

A. Import Necessary Libraries: Begin by importing the required libraries.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go

from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import AdaBoostClassifier
import xgboost as Xgb
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix

import warnings
warnings.filterwarnings('ignore')
```

Python

B. Load the Dataset: Load the dataset into a Pandas DataFrame.

- **Loading the Dataset**

```python
# Load the dataset
# Explicitly specify datetime data type for date/time columns
date_cols = ['BookingID_Date', 'Data_Ping_time', 'Planned_ETA', 'actual_eta', 'trip_start_date', 'trip_end_
data = pd.read_excel("Transportation and Logistics Tracking Dataset.xlsx",parse_dates=date_cols)
```

Python

C. Preprocess the Data: Perform data cleaning, transformation, and feature engineering as needed. Follow the preprocessing steps outlined in the study to ensure data consistency.

```python
# # Drop columns with not relevant for prediction
data.drop(['Driver_Name', 'Driver_MobileNo'], axis=1, inplace=True)
```

Python

```python
# Impute missing Data_Ping_time with median
data['Data_Ping_time'].fillna(data['Data_Ping_time'].median(), inplace=True)
```

Python

```python
# Replace missing Current_Location with 'Unknown'
data['Current_Location'].fillna('Unknown', inplace=True)

# Impute missing actual_eta with Planned_ETA
data['actual_eta'].fillna(data['Planned_ETA'], inplace=True)
```

Python

5.3 Training and Testing the Models

A. Prepare the Data for Modeling: Split the dataset into training and testing sets.

- **Splitting Data for Training and Testing**

```
# Prepare the data into training and testing sets to evaluate the predictive performance of the models.
X = df_cleaned.drop(['ontime', 'delay'], axis=1)  # Features excluding the target variables
y = df_cleaned['ontime']  # or 'delay', depending on what you want to predict

#splitt the dataset into the training and testing for 80:20 ratio
#where 80 percent dataset is reserved for training and 20 percent for testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Python

```
print('Training Features Shape:', X_train.shape)
print('Training Labels Shape:', y_train.shape)
print('Testing Features Shape:', X_test.shape)
print('Testing Labels Shape:', y_test.shape)
```

Python

```
Training Features Shape: (5484, 22)
Training Labels Shape: (5484,)
Testing Features Shape: (1372, 22)
Testing Labels Shape: (1372,)
```

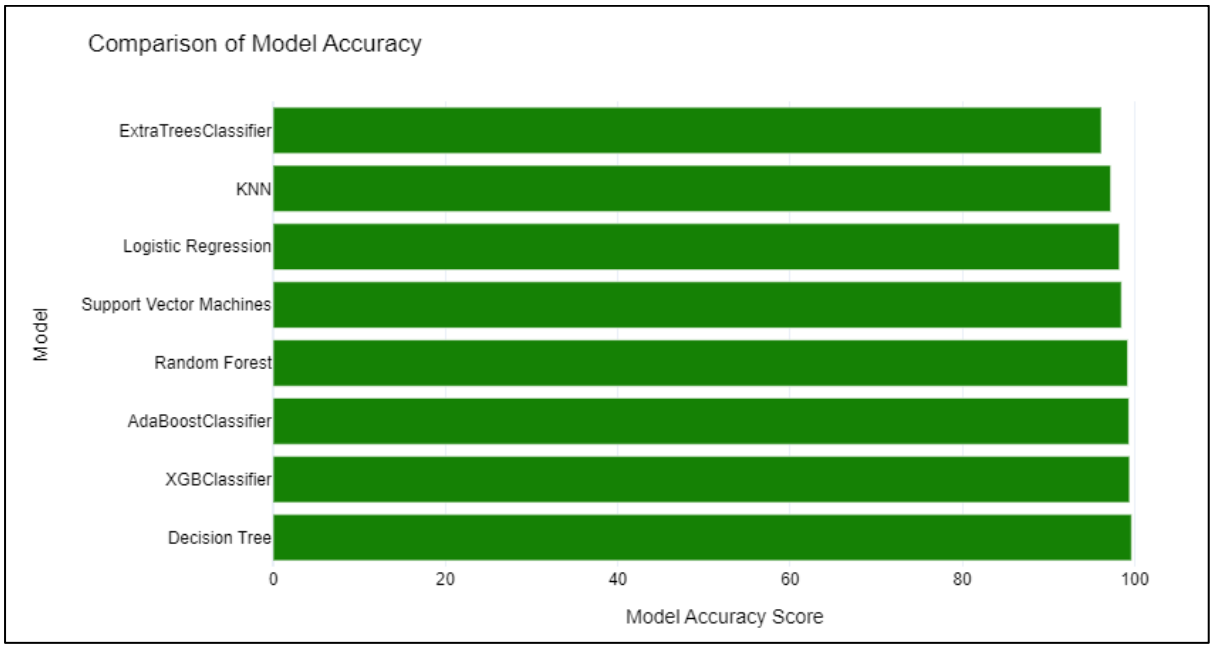
B. Train the Machine Learning Models: Example for training a Random Forest model.

```
# Random Forest
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, y_train)
y_pred_rf = random_forest.predict(X_test)
random_forest.score(X_train, y_train)

random_forest_train = round(random_forest.score(X_train, y_train) * 100, 2)
random_forest_accuracy = round(accuracy_score(y_pred_rf, y_test) * 100, 2)

print("Training Accuracy    :",random_forest_train ,"%")
print("Model Accuracy Score :",random_forest_accuracy ,"%")
print("\033[1m------------------------------------------------------------\033[0m")
print("Classification_Report: \n",classification_report(y_test,y_pred_rf))
print("\033[1m------------------------------------------------------------\033[0m")
# Compute confusion matrix
cm = confusion_matrix(y_test, y_pred_rf)
```

C. Evaluate Model Performance: Evaluate the model's accuracy and generate a classification report.

Model	Training Accuracy	Model Accuracy Score
Decision Tree	100.000000	99.640000
XGBClassifier	100.000000	99.420000
AdaBoostClassifier	99.910000	99.340000
Random Forest	100.000000	99.200000
Support Vector Machines	99.310000	98.470000
Logistic Regression	99.220000	98.250000
KNN	97.760000	97.230000
ExtraTreesClassifier	100.000000	96.140000

D. Visualize Results: Create plots to visualize the model's performance, such as confusion matrices and accuracy charts.



E. Averaging Model Predictions: Implementing an ensemble approach, combine predictions from multiple models to enhance accuracy.

- **Averaging all the Model Predictions**

```
#all predictions
model_predictions = [y_pred_rf,y_pred_svc,y_pred_ada,y_pred_dec,y_pred_et,y_pred_knn,y_pred_lr,y_pred_xgb]
```

Python

```
# Calculate the average of the predicted values
average_predictions = np.mean(model_predictions, axis=0)
```

Python

```
#Average Class Classification Predictions
average_class_predictions = np.round(average_predictions).astype(int)
```

Python

```
average_model_accuracy = round(accuracy_score(average_class_predictions, y_test) * 100, 2)

print("Average Model Accuracy Score :",average_model_accuracy ,"%")
print("\033[1m-----------------------------------------------------\033[0m")
print("Classification_Report: \n",classification_report(y_test,average_class_predictions))
print("\033[1m-----------------------------------------------------\033[0m")
#confusion Matrix
cm = confusion_matrix(y_test, average_class_predictions)

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Greens")
plt.xlabel("Predicted labels")
plt.ylabel("True labels")
plt.title("Confusion Matrix")
plt.show()
```

Python

5.4 Saving and Documenting Results

A. Save Results: Export model performance metrics and visualizations for documentation.

B. Document Findings: Ensure all findings and insights from the model evaluations are well-documented, including any observations about the data or model performance.

This configuration manual provides a comprehensive guide to setting up the environment, downloading the dataset, and executing the code implementation for the research study. By following these steps, you will be able to replicate the study's results and explore the potential of predictive modeling in supply chain optimization.

References

Anaconda: https://docs.anaconda.com/free/anaconda/install/windows/
Kaggle Dataset Source: Transportation and Logistics Tracking Dataset (kaggle.com)