Configuration Manual

MSc Data Analytics
Research Project

Kajal Singh
X23192461

School of Computing
National College of Ireland

Supervisor: Dr. Barry Haycock

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Kajal Singh		
Student ID:	X23192461		
Programme:	MSc Data Analytics	**Year:**	2024
Module:	Research Project		
Supervisor:	**Dr**. Barry Haycock		
Submission Due Date:	12/12/2024		
Project Title:	**Evaluating Machine Learning Models for Defect Rate Prediction and Maintenance Classification in Industrial Systems**		
Word Count:	**xxx**	**Page Count:**	**12**

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:
Date:

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

Kajal Singh
Student ID: X23192461

1 Introduction:

This manual provides detailed instructions for configuring and deploying the phishing URL detection system developed in this research project. The system employs a hybrid model integrating machine learning and deep learning techniques to accurately identify phishing URLs.

2 System Requirements:

To guarantee efficient model processing and to minimize the duration required, it's crucial to be equipped with the necessary hardware and software resources.

1.1. Hardware Requirements:

The implementation is performed on an HP Pavilion; the configuration of the device is as follows.

1. Processor: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz
2. RAM: 8.00 GB (7.88 GB usable)
3. Hard Disk: 1 TB HDD, 256 SSD
4. OS Windows 11

2.2 Software Requirements:

Before beginning the model construction phase, the below mentioned software, libraries, and tools were set up and installed on the system.

Software/Tools	Version	Information
Python		The python coding language has been used because of its vast libraries that can help in developing ML models

Anaconda		A popular open-source distribution that simplifies the management of Python and R environments, including pre-installed libraries for data science, machine learning, and scientific
		computing.
Pandas		A Python library providing data structures like DataFrames for easy manipulation, analysis, and handling of structured data.
Matplotlib		A Python library for creating static, interactive, and animated visualizations, including plots, graphs, and charts.
Sci-kit Learn		A Python library offering tools for machine learning, including classification, regression, clustering, and model evaluation.

3 Package required:

All the requirement packages in the Python environment were installed via pip and conda in Jupyter notebook. Below is the list of the packages that has been installed:

- Pandas
- SMOTE
- Numpy
- Matplotlib
- Seaborn

4 Dataset Description:

There were three dataset which was used for the analysis. All off them were downloaded from Kaggle. Here is the link for each of the datasets:

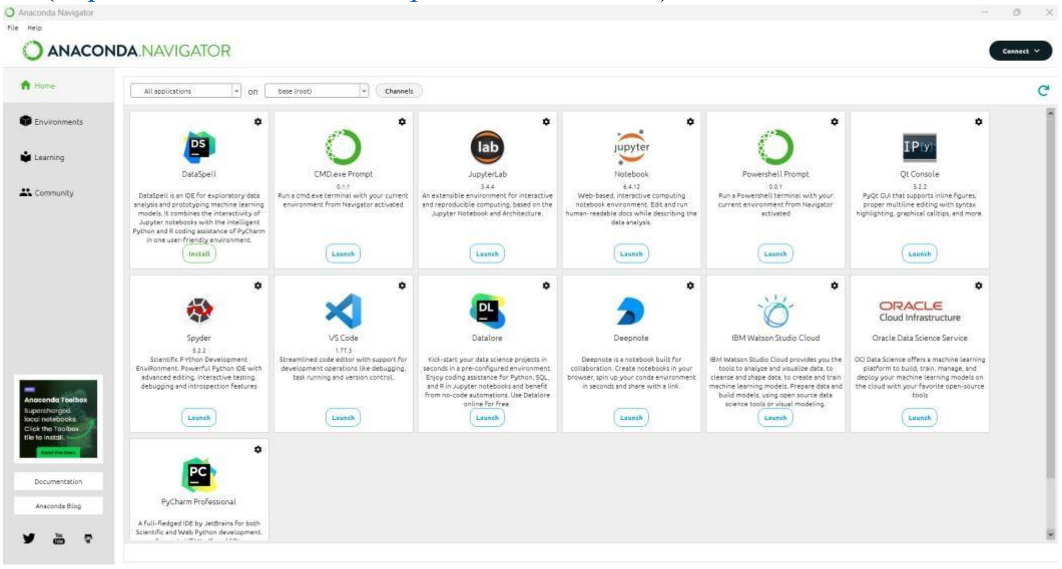
- https://www.kaggle.com/datasets/datasetengineer/logistics-vehicle-maintenancehistory-dataset
- https://www.kaggle.com/datasets/rabieelkharoua/predicting-manufacturing-defectsdataset
- https://www.kaggle.com/datasets/harshsingh2209/supply-chain-analysis

5. Implementation:

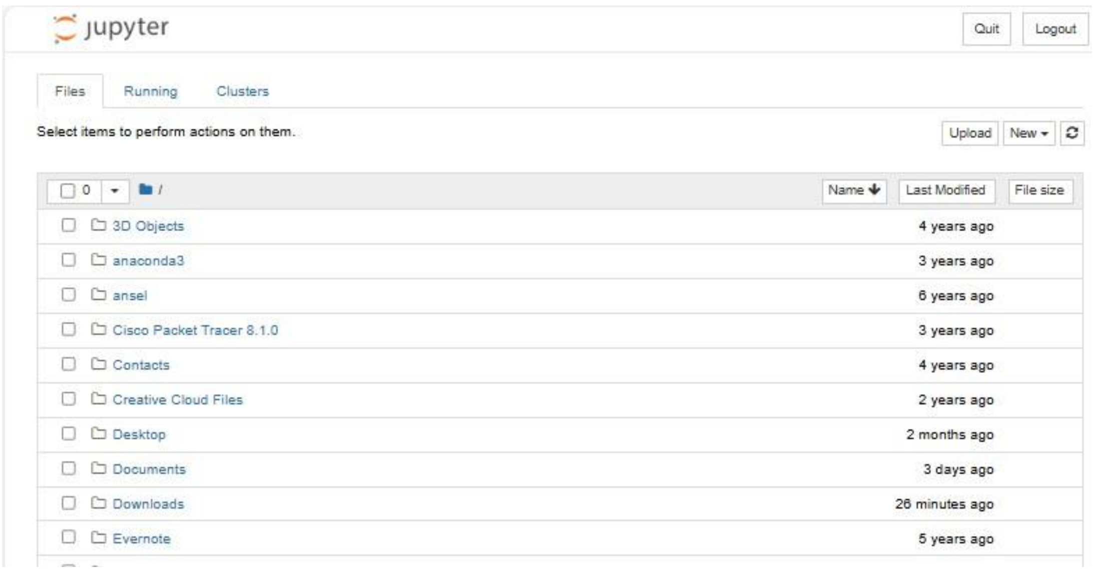
In this section there is a complete guide to run the project in any windows system.

1. Download and Install Anaconda Software in the windows system.

   (https://www.anaconda.com/products/individual)



2. Open the Jupyter Notebook from Anaconda.

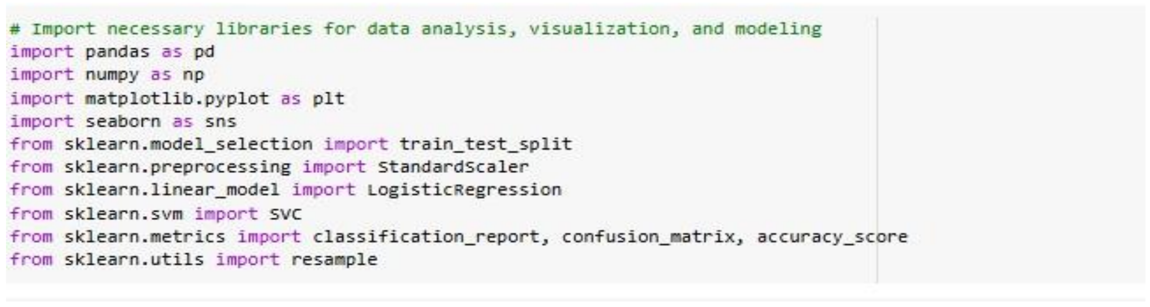


Select items to perform actions on them.

Name	Last Modified	File size
3D Objects	4 years ago	
anaconda3	3 years ago	
ansel	6 years ago	
Cisco Packet Tracer 8.1.0	3 years ago	
Contacts	4 years ago	
Creative Cloud Files	2 years ago	
Desktop	2 months ago	
Documents	3 days ago	
Downloads	26 minutes ago	
Evernote	5 years ago	

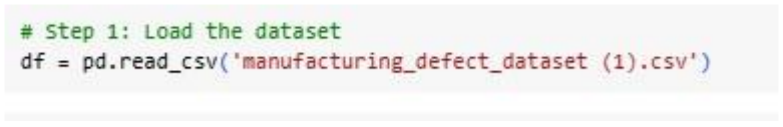
3. After opening jupyter notebook click on the new notebook (python 3).

4. In notebook, Import all the required libraries.



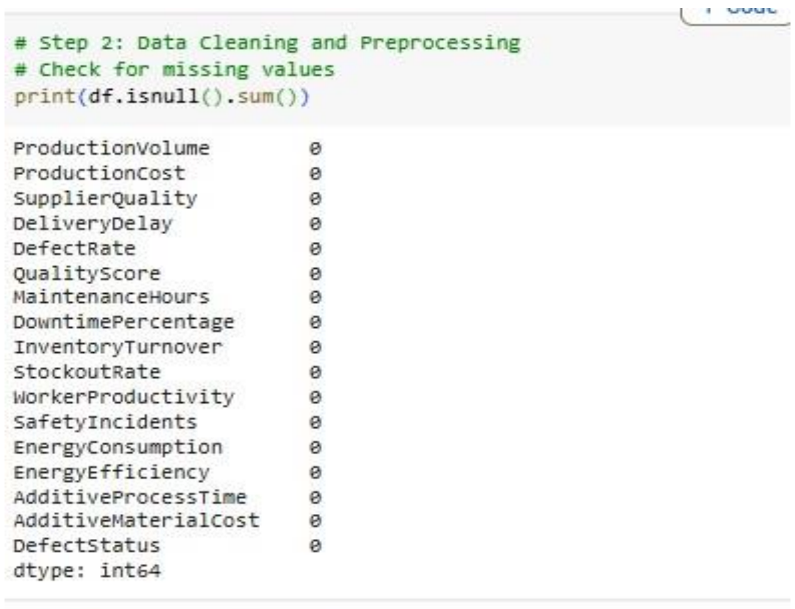
```
# Import necessary libraries for data analysis, visualization, and modeling
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.utils import resample
```

5. Import the Provided “Manufacturing defect dataset”.



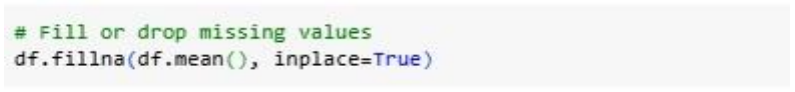
```
# Step 1: Load the dataset
df = pd.read_csv('manufacturing_defect_dataset (1).csv')
```

6. Next Step will be Pre Processing Step will be performed using following Code.

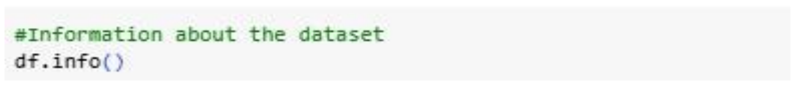


```
# Step 2: Data Cleaning and Preprocessing
# Check for missing values
print(df.isnull().sum())
```

```
ProductionVolume        0
ProductionCost          0
SupplierQuality         0
DeliveryDelay           0
DefectRate              0
QualityScore            0
MaintenanceHours        0
DowntimePercentage      0
InventoryTurnover       0
StockoutRate            0
WorkerProductivity      0
SafetyIncidents         0
EnergyConsumption       0
EnergyEfficiency        0
AdditiveProcessTime     0
AdditiveMaterialCost    0
DefectStatus            0
dtype: int64
```

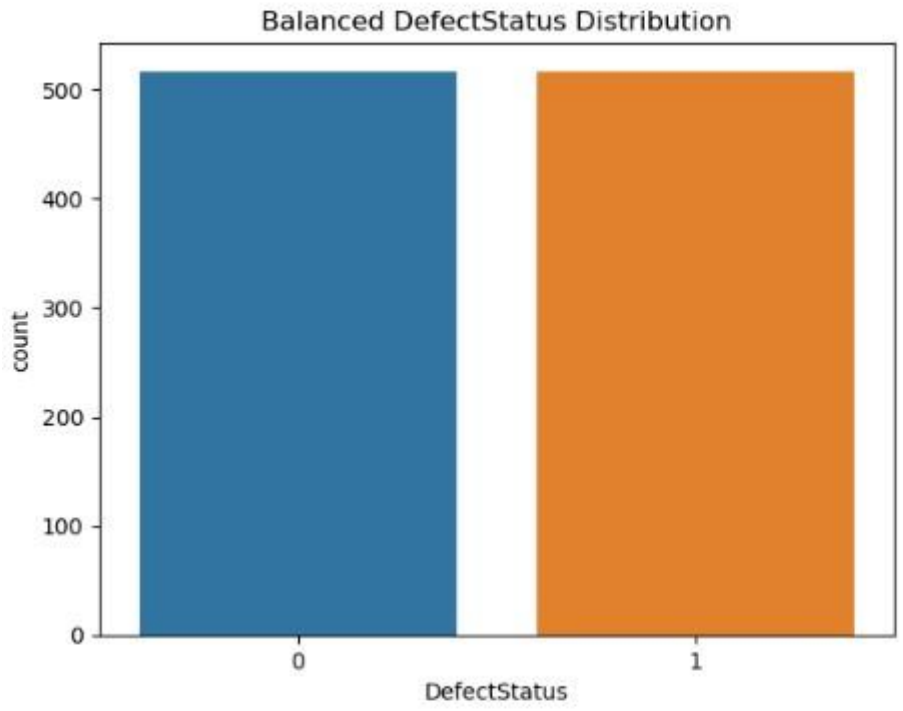


```
# Fill or drop missing values
df.fillna(df.mean(), inplace=True)
```

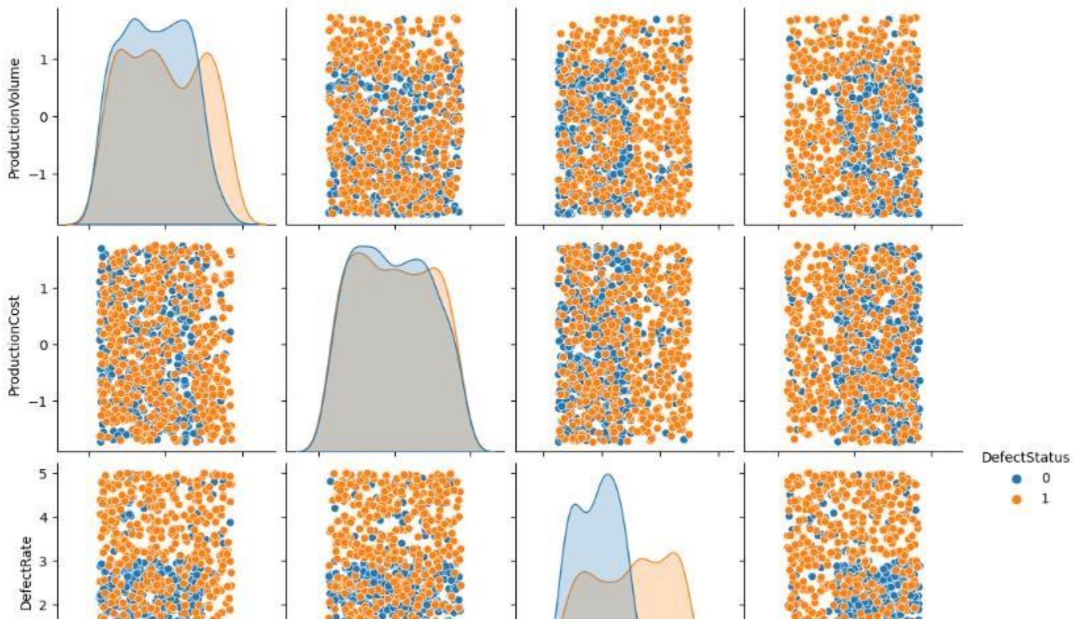


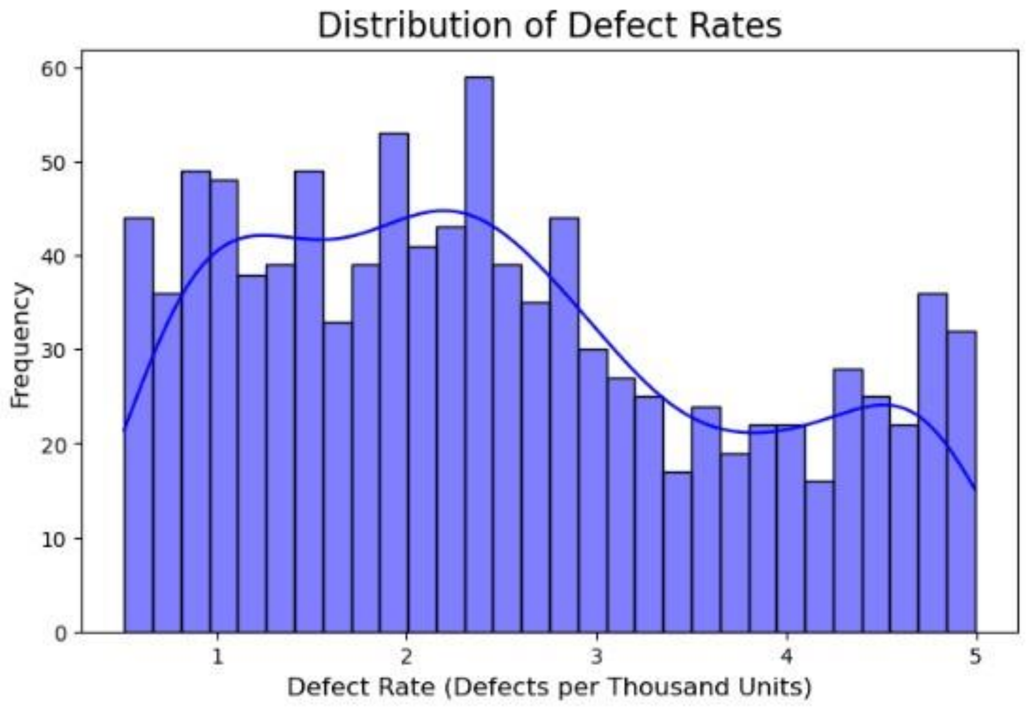
```
#Information about the dataset
df.info()
```

6. Exploratory Data Analysis has been Performed and Visualisation has been done using following Code

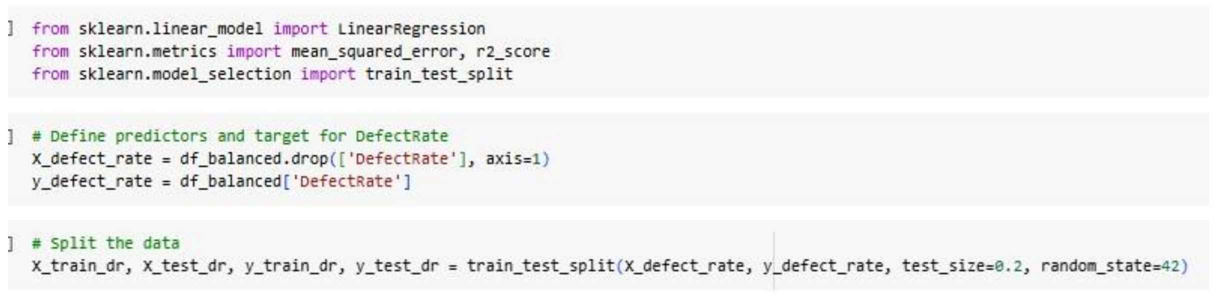


```
/Users/kajalsingh/anaconda3/lib/python3.11/site-packages/seaborn/axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```





7. After Data Pre Processing the Data Splitting is Performed before Building a Model

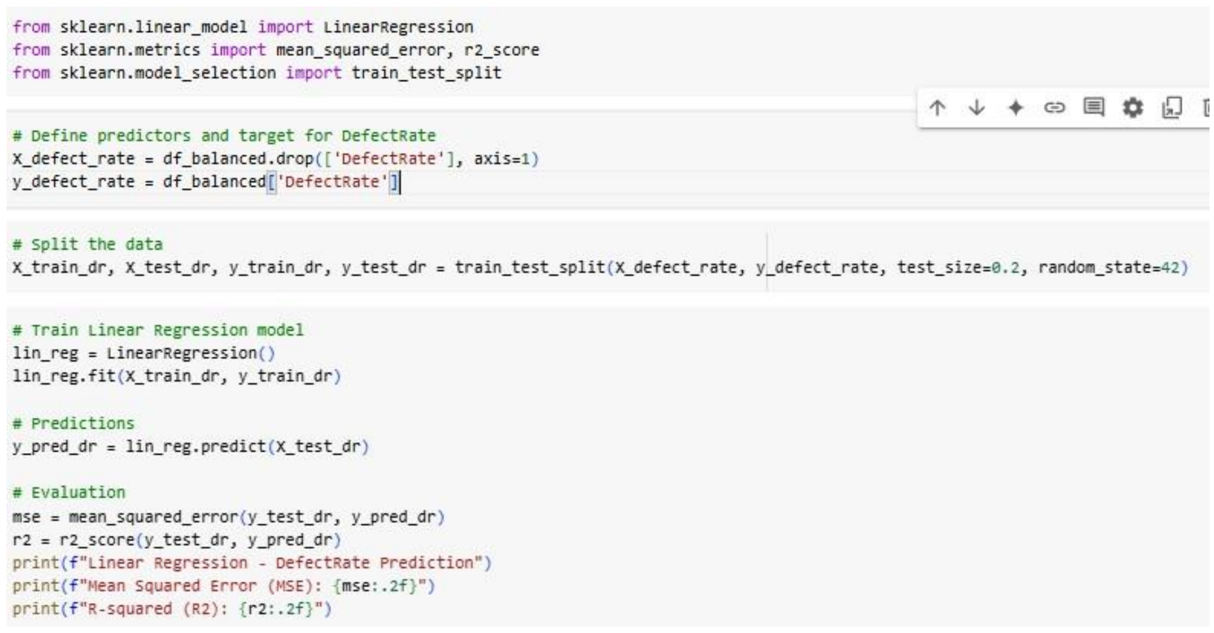


```
] from sklearn.linear_model import LinearRegression
  from sklearn.metrics import mean_squared_error, r2_score
  from sklearn.model_selection import train_test_split
```

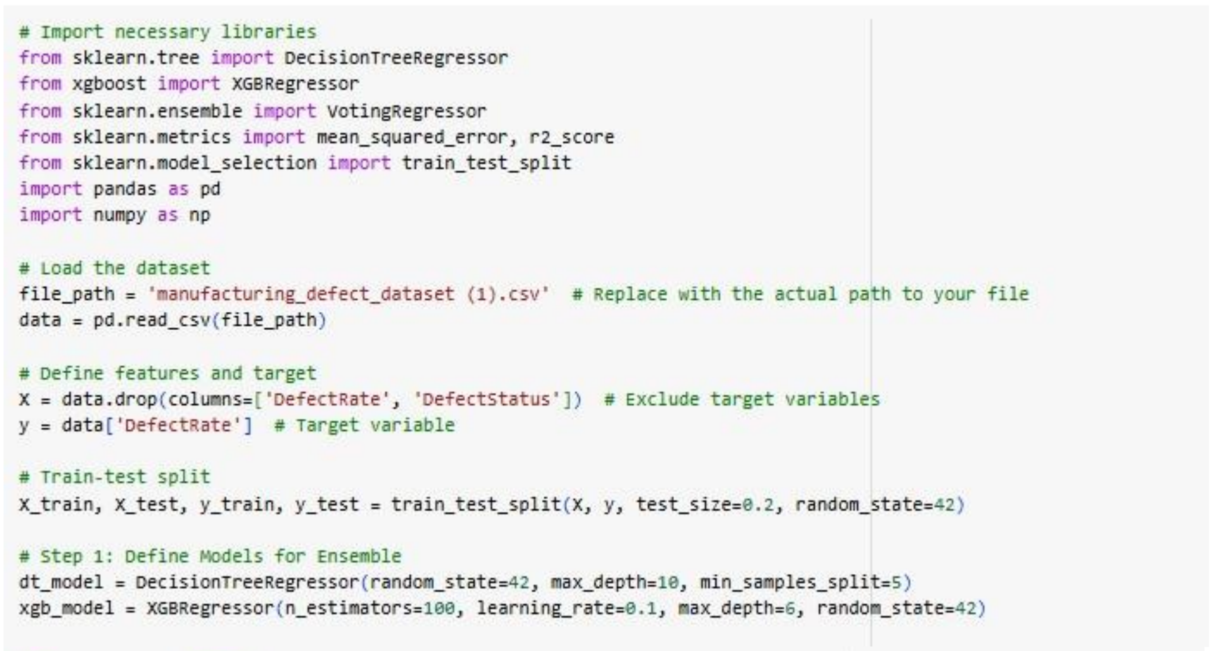
```
] # Define predictors and target for DefectRate
  X_defect_rate = df_balanced.drop(['DefectRate'], axis=1)
  y_defect_rate = df_balanced['DefectRate']
```

```
] # Split the data
  X_train_dr, X_test_dr, y_train_dr, y_test_dr = train_test_split(X_defect_rate, y_defect_rate, test_size=0.2, random_state=42)
```

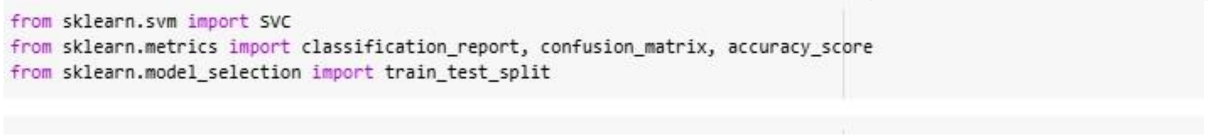
8. Models Implementation has been Performed with the following Code



```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split

# Define predictors and target for DefectRate
X_defect_rate = df_balanced.drop(['DefectRate'], axis=1)
y_defect_rate = df_balanced['DefectRate']

# Split the data
X_train_dr, X_test_dr, y_train_dr, y_test_dr = train_test_split(X_defect_rate, y_defect_rate, test_size=0.2, random_state=42)

# Train Linear Regression model
lin_reg = LinearRegression()
lin_reg.fit(X_train_dr, y_train_dr)

# Predictions
y_pred_dr = lin_reg.predict(X_test_dr)

# Evaluation
mse = mean_squared_error(y_test_dr, y_pred_dr)
r2 = r2_score(y_test_dr, y_pred_dr)
print(f"Linear Regression - DefectRate Prediction")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R-squared (R2): {r2:.2f}")
```

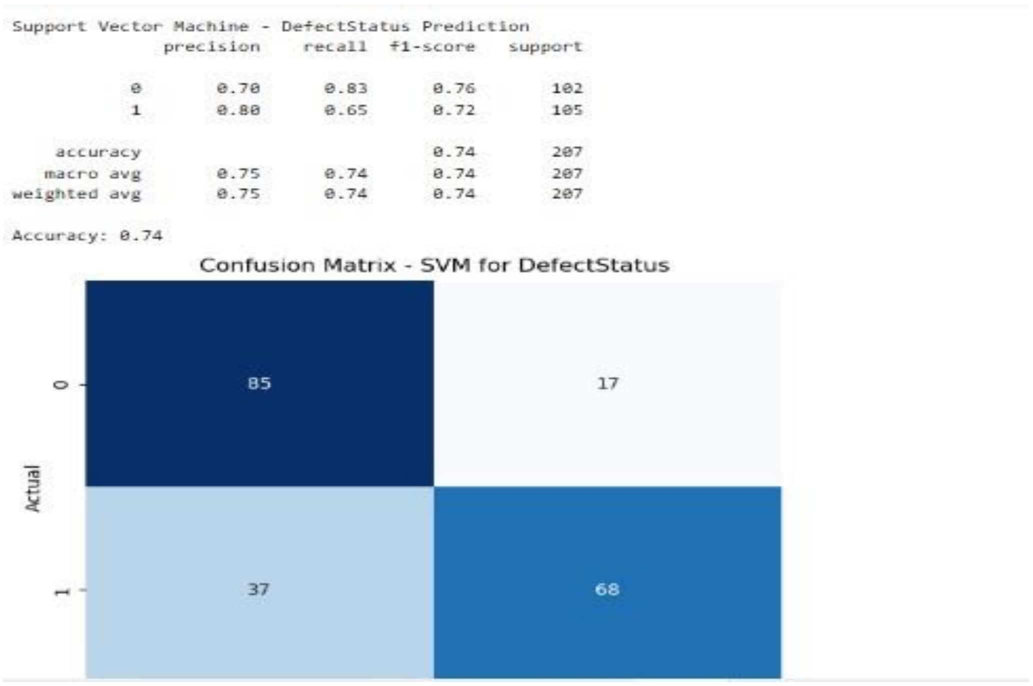


```
# Import necessary libraries
from sklearn.tree import DecisionTreeRegressor
from xgboost import XGBRegressor
from sklearn.ensemble import VotingRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
import pandas as pd
import numpy as np

# Load the dataset
file_path = 'manufacturing_defect_dataset (1).csv'  # Replace with the actual path to your file
data = pd.read_csv(file_path)

# Define features and target
X = data.drop(columns=['DefectRate', 'DefectStatus'])  # Exclude target variables
y = data['DefectRate']  # Target variable

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 1: Define Models for Ensemble
dt_model = DecisionTreeRegressor(random_state=42, max_depth=10, min_samples_split=5)
xgb_model = XGBRegressor(n_estimators=100, learning_rate=0.1, max_depth=6, random_state=42)
```



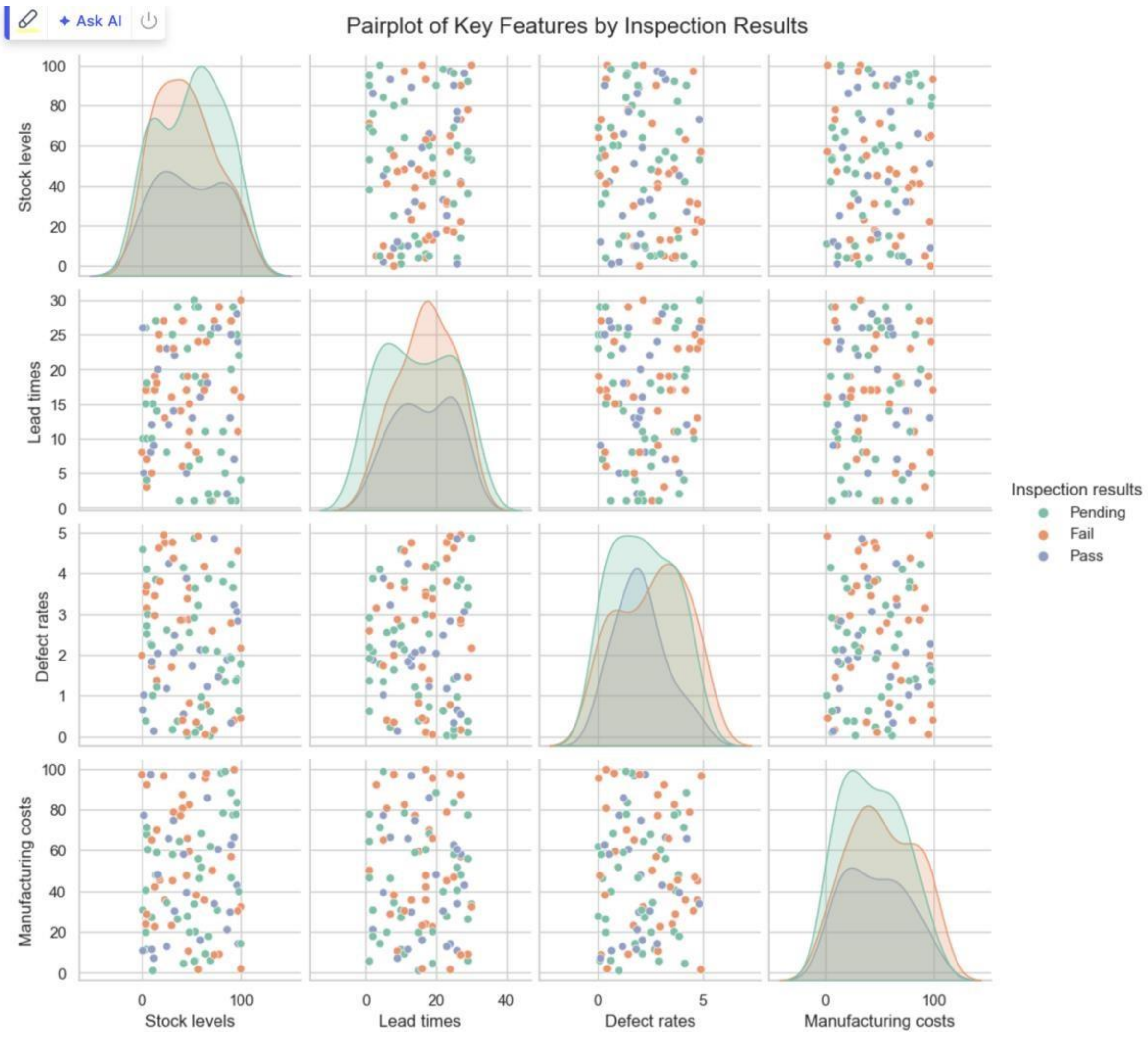
```
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.model_selection import train_test_split
```

9. The Accuracy is considerable as evaluation factor after Model Implementation

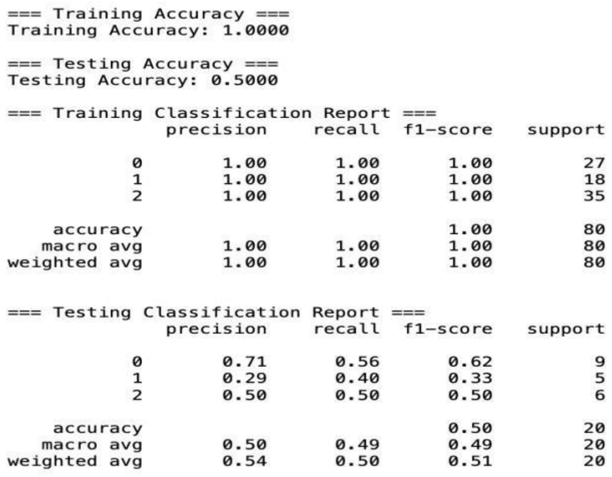


```
Support Vector Machine - DefectStatus Prediction
              precision    recall  f1-score   support

           0       0.70      0.83      0.76       102
           1       0.80      0.65      0.72       105

    accuracy                           0.74       207
   macro avg       0.75      0.74      0.74       207
weighted avg       0.75      0.74      0.74       207

Accuracy: 0.74
```

10. For the other dataset “Supply chain analysis” , the target variable “Inspection result” is then compared with all other features like manufacturing cost, lead times and defect rates.

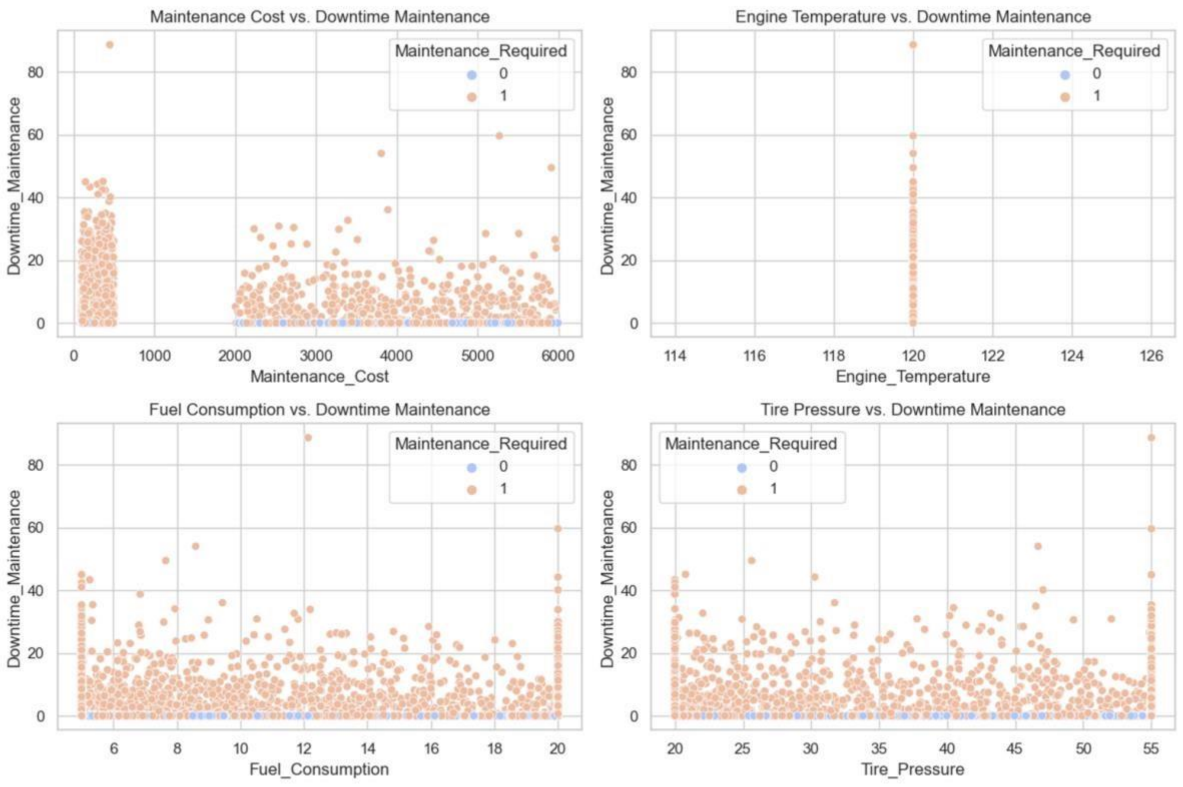


11. When the model XGBost classification was applied on the dataset it gave an precisie result that the model is giving an accuracy of 100% for training but falls off to 50% on test. Showing that the model is overfitting.

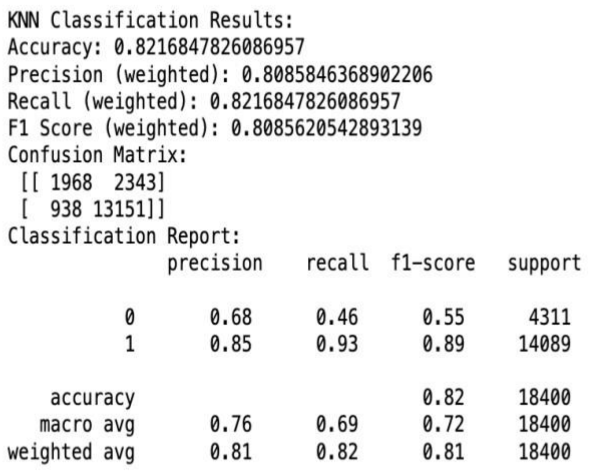


```
=== Training Accuracy ===
Training Accuracy: 1.0000

=== Testing Accuracy ===
Testing Accuracy: 0.5000

=== Training Classification Report ===
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        27
           1       1.00      1.00      1.00        18
           2       1.00      1.00      1.00        35

    accuracy                           1.00        80
   macro avg       1.00      1.00      1.00        80
weighted avg       1.00      1.00      1.00        80


=== Testing Classification Report ===
              precision    recall  f1-score   support

           0       0.71      0.56      0.62         9
           1       0.29      0.40      0.33         5
           2       0.50      0.50      0.50         6

    accuracy                           0.50        20
   macro avg       0.50      0.49      0.49        20
weighted avg       0.54      0.50      0.51        20
```

12. Data pre-processing task was performed in the last dataset which is “Logistics operation” by comparing the target variable which is “Downtime Maintaince” with all other feature.



13. And after the pre processing the KNN classifier model is applied on the dataset which gave a very reliable accuracy with 86%. This implies the model is best suited for the classification.



```
KNN Classification Results:
Accuracy: 0.8216847826086957
Precision (weighted): 0.8085846368902206
Recall (weighted): 0.8216847826086957
F1 Score (weighted): 0.8085620542893139
Confusion Matrix:
 [[ 1968  2343]
 [  938 13151]]
Classification Report:
              precision    recall  f1-score   support

           0       0.68      0.46      0.55      4311
           1       0.85      0.93      0.89     14089

    accuracy                           0.82     18400
   macro avg       0.76      0.69      0.72     18400
weighted avg       0.81      0.82      0.81     18400
```

The concluding code files are include the ipynb file and the csv dataset

References

Anaconda. 2021. Anaconda | The World's Most Popular Data Science Platform. [online] Available at:

.<https://www.anaconda.com/>.
Numpy.org. 2021. NumPy. [online] Available at: <https://numpy.org/>.