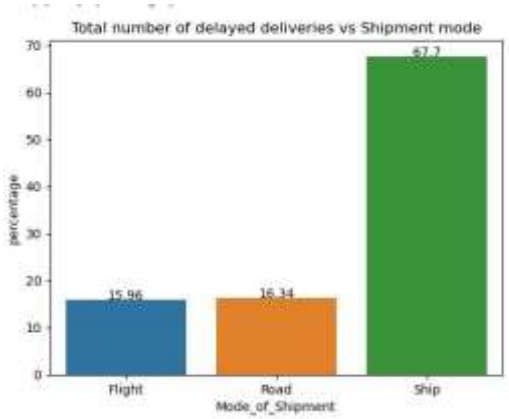
Section	Description
Data Overview	

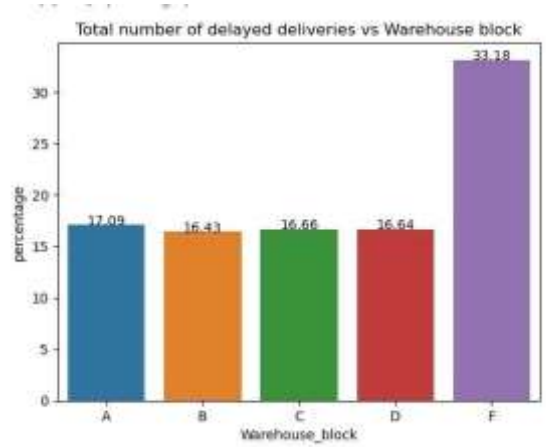
Univariate Analysis	

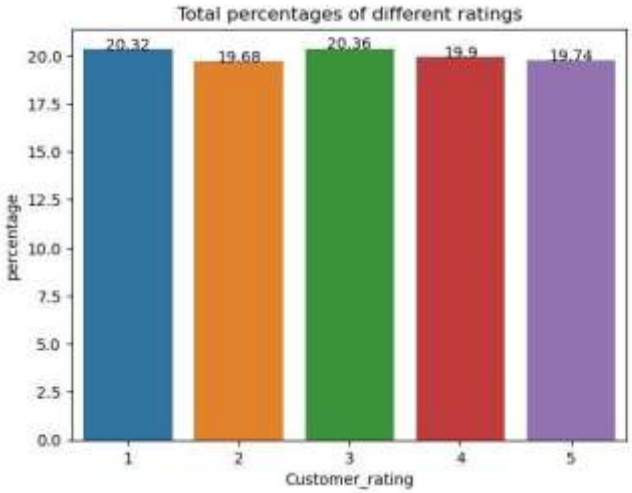
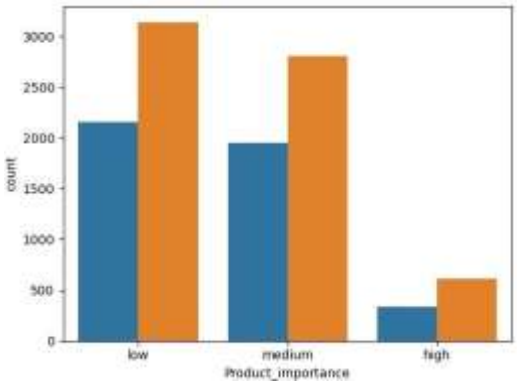
Date	11 JULY 2024
Team ID	SWUID20240034764
Project Title	Predicting Full Load Electrical Power Output of a Base Load Operated Combined Cycle Power Plant Using Machine Learning
Maximum Marks	6 Marks

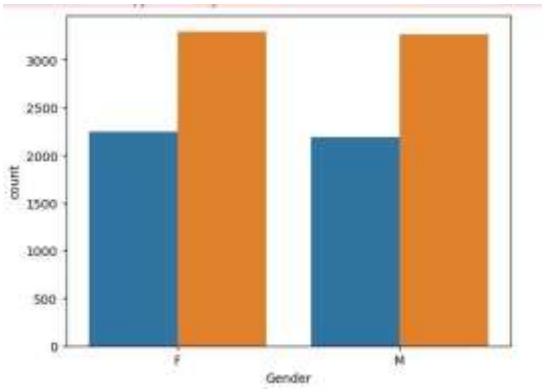
Data Collection and Preprocessing Phase

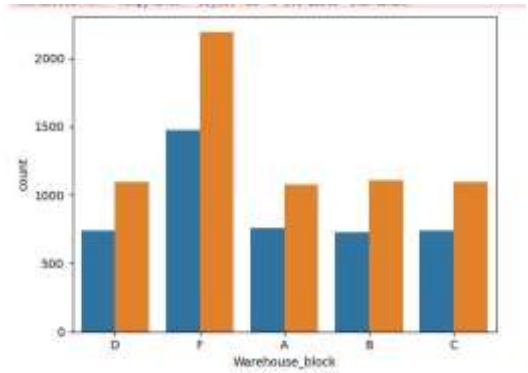
Data Exploration and Preprocessing Template

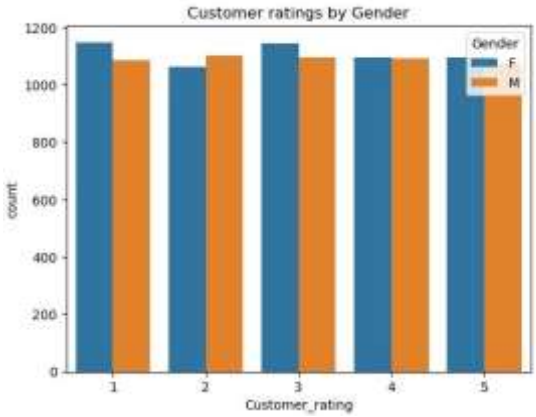
Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

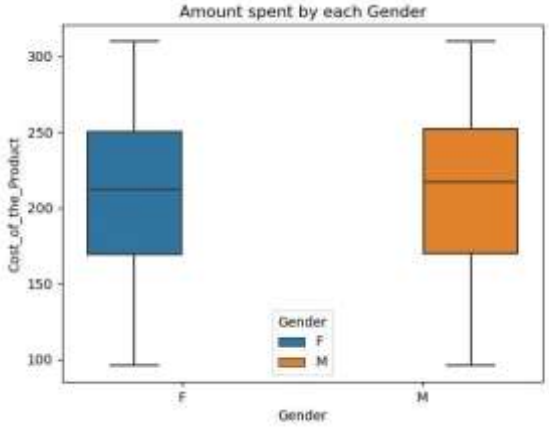
<table>
<tr>
<td></td>
<td> 

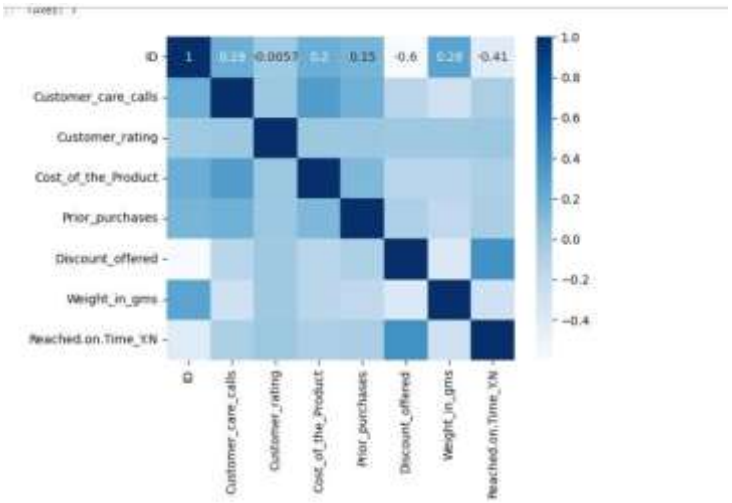
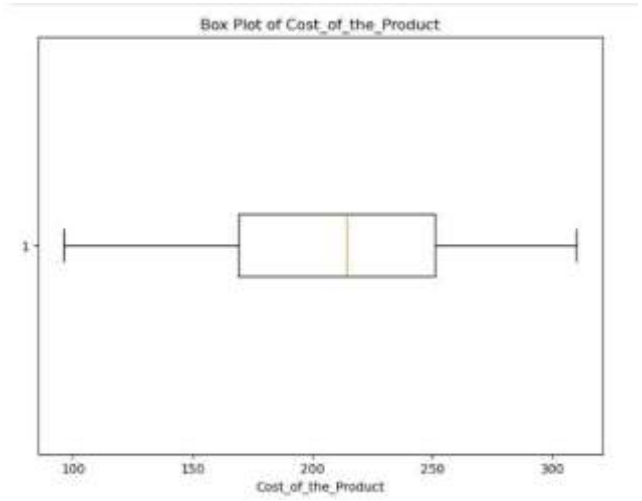
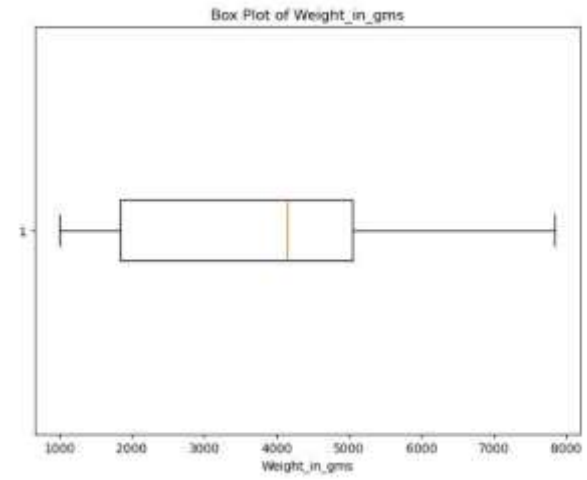

</td>
</tr>
<tr>
<td>Bivariate Analysis</td>
<td> 
</td>
</tr>
</table>


count
Gender
F
M


count
Warehouse_block
D
F
A
B
C


Customer ratings by Gender
count
Customer_rating
Gender
F
M


Amount spent by each Gender
Cost_of_the_Product
Gender
F
M

Multivariate Analysis	
Outliers and Anomalies	 

Data Preprocessing Code Screenshots

Loading Data	import pandas as pd dataset = pd.read_csv('train.csv') dataset
Handling Missing Data	dataset.isnull().sum()
Data Transformation	# to handle data in form of rows and columns import pandas as pd # Numerical libraries import numpy as np # importing ploting libraries import matplotlib.pyplot as plt #importing seaborn for statistical plots import seaborn as sns #implements serialization import pickle data = pd.read_csv("/home/HLO1.csv", header=0 , names = ['AT','V','AP','RH','PE']) data.isnull().sum() data.head() data.describe().T plt.scatter(data['AT'],data['PE']) plt.scatter(data['V'],data['PE']) plt.scatter(data['AP'],data['PE']) plt.scatter(data['RH'],data['PE']) sns.pairplot(data,diag_kind = 'hist') x = data.drop(['PE'], axis=1) y = data['PE'] from sklearn.model_selection import train_test_split xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.2, random_state = 0) xtrain.shape xtest.shape

	from sklearn.linear_model import LinearRegression
	from sklearn.tree import DecisionTreeRegressor
	from sklearn.ensemble import RandomForestRegressor
	# Initializing the models
	LRmodel=LinearRegression()
	DTmodel=DecisionTreeRegressor()
	RFmodel=RandomForestRegressor()
	from sklearn.linear_model import LinearRegression
	LRmodel = LinearRegression()
	LRmodel.fit(xtrain, ytrain)

Feature Engineering	import pickle
	# pickle is used for serializing and de-
	serializing Python object structures
	
	app=Flask(_name_)
	# our flask app
	@app.route('/') # rendering the html
	template
	def home():
	  return
	render_template('home.html')
	@app.route('/predict') # rendering the
	html template
	def index():
	  return
	render_template("index.html")
	@app.route('/data_predict',
	methods=['POST']) # route for our
	prediction
	def predict():
	  at = request.form['at'] # requesting
	for age data
	  v = request.form['v'] # requesting for
	gender data

	ap = request.form['ap'] # requesting for Total_Bilirubin data
	rh = request.form['rh'] # requesting for Direct_Bilirubin data
	
	# coverting data into float format
	data = [[float(at), float(v), float(ap), float(rh)]]
	# Loading model which we saved
	model = pickle.load(open('model.pkl', 'rb'))
	prediction= model.predict(data)[0]
	return render_template('predict.html', prediction=prediction)
	if _name_ == '_main_':

Save Processed Data	dataset.to_csv('my_dataset.csv', index=False)