School of Computer Science and Engineering
Fall Semester-2024-25

Course Code : CBS3007

Course: Data Mining and Analytics

Alan Thomas

21BBS0115

Github link for the datasets and code-

https://github.com/ALANT535/DATA-MINING-RESOURCES/tree/main/DA3

SECTION 1

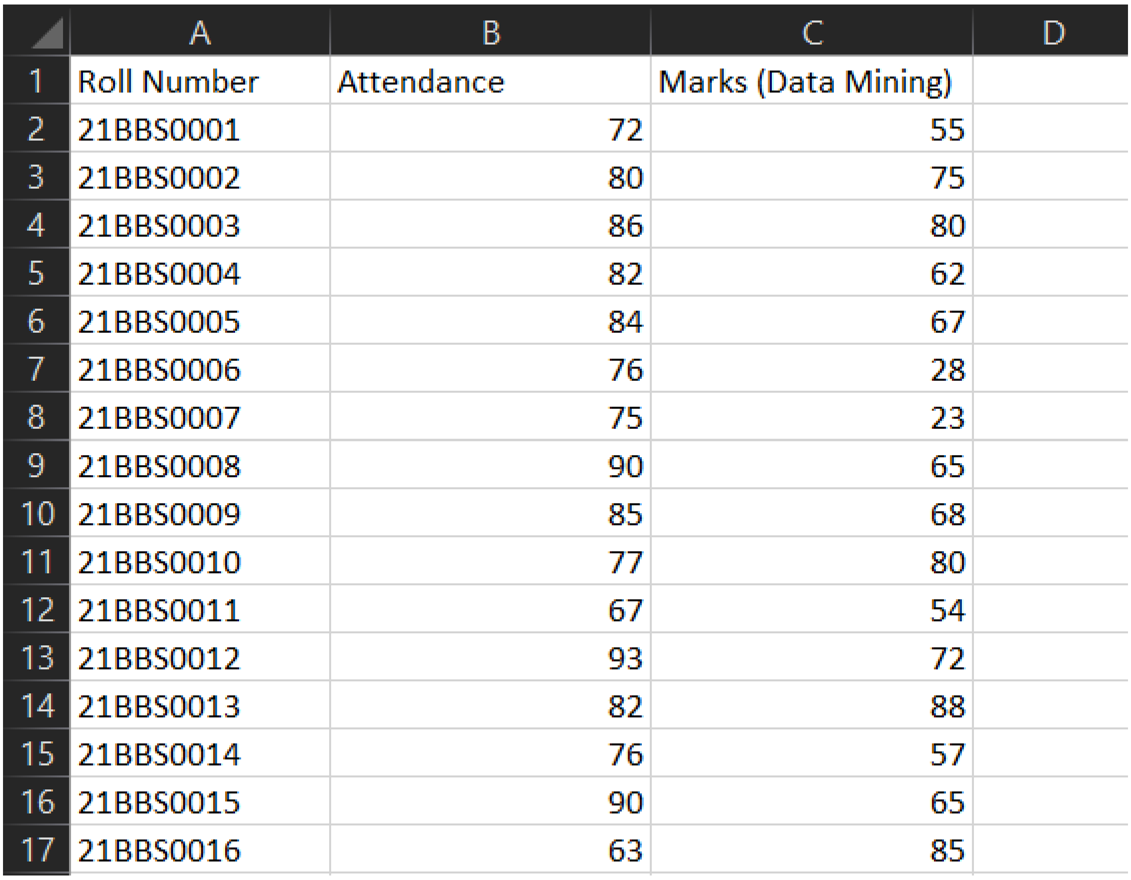
Aim:

Implementation of KNN classification on dataset

Libraries Used : Numpy, Pandas, sklearn, matplotlib, seaborn

Dataset :
https://github.com/ALANT535/DATA-MINING-RESOURCES/tree/main/DA3/Q1

Sample Input

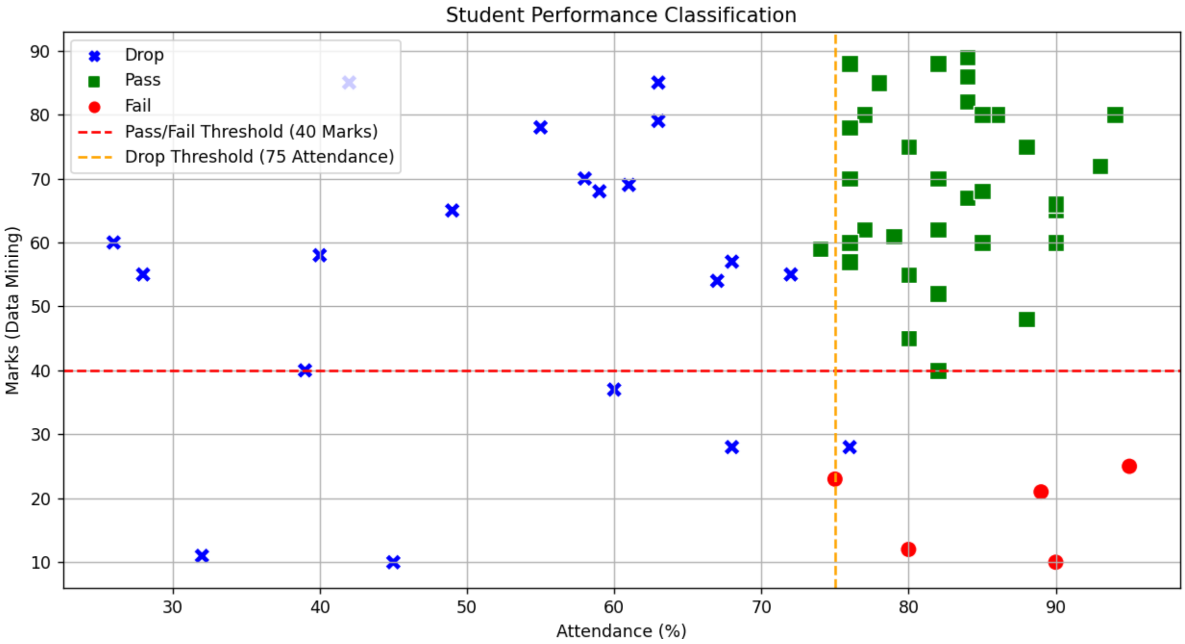


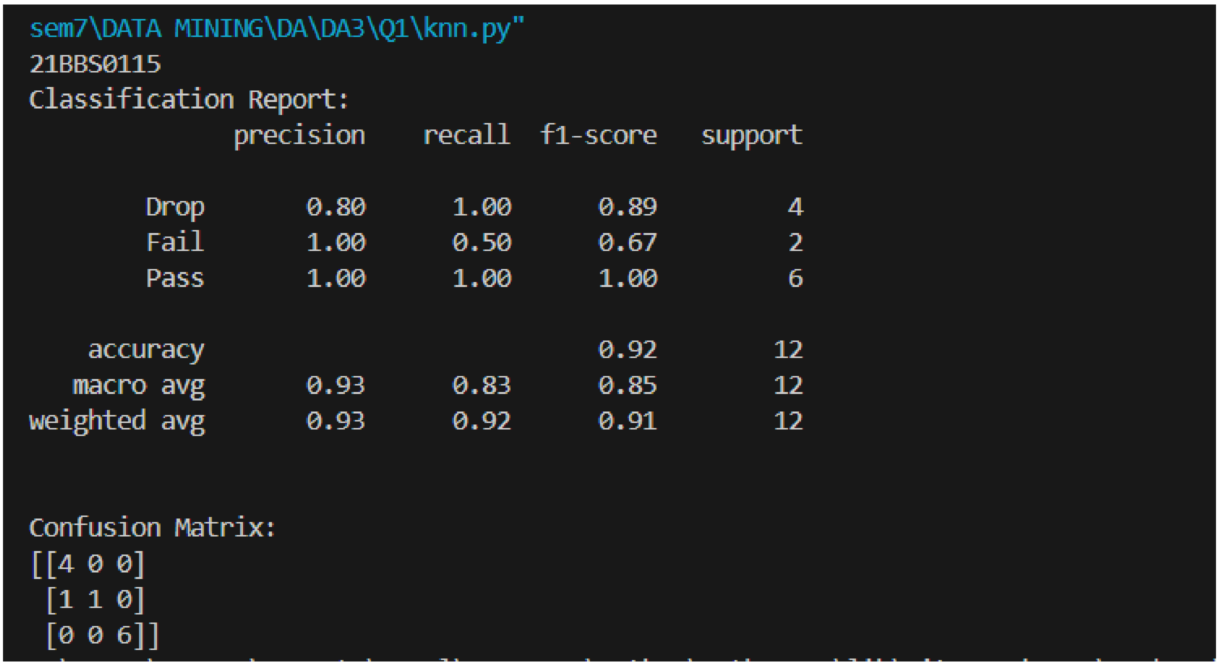
	A	B	C	D
1	Roll Number	Attendance	Marks (Data Mining)	
2	21BBS0001	72	55	
3	21BBS0002	80	75	
4	21BBS0003	86	80	
5	21BBS0004	82	62	
6	21BBS0005	84	67	
7	21BBS0006	76	28	
8	21BBS0007	75	23	
9	21BBS0008	90	65	
10	21BBS0009	85	68	
11	21BBS0010	77	80	
12	21BBS0011	67	54	
13	21BBS0012	93	72	
14	21BBS0013	82	88	
15	21BBS0014	76	57	
16	21BBS0015	90	65	
17	21BBS0016	63	85	

Code

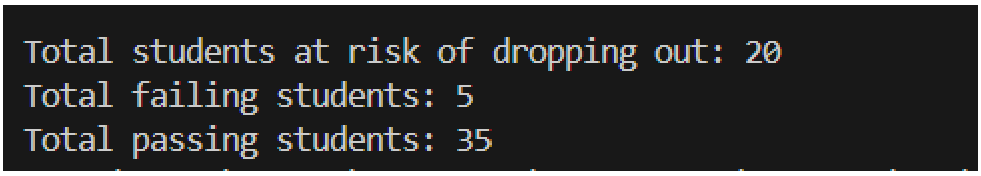
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix

print("21BBS0115")

df=pd.read_csv(r"C:\Users\LENOVO\Documents\Important_documents\VIT\Semesters\sem7\DATA MINING\DA\DA3\Q1\student.csv")

X = df[['Attendance', 'Marks (Data Mining)']]

y = np.where(X['Attendance'] < 75, 'Drop',
         np.where(X['Marks (Data Mining)'] < 40, 'Fail', 'Pass'))

X_train, X_test, y_train, y_test = train_test_split(
   X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train)

y_pred = knn.predict(X_test_scaled)

print("Classification Report:")
print(classification_report(y_test, y_pred))
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

all_predictions = knn.predict(scaler.transform(X))

df['Prediction'] = all_predictions

plt.figure(figsize=(12, 6))

sns.scatterplot(data=df, x='Attendance', y='Marks (Data Mining)',
hue='Prediction',
            palette={'Pass': 'green', 'Fail': 'red', 'Drop': 'blue'},
            style='Prediction', markers={"Drop": "X", "Fail": "o", "Pass": "s"}, s=100)
plt.title("Student Performance Classification")
plt.xlabel("Attendance (%)")
plt.ylabel("Marks (Data Mining)")
plt.axhline(40, color='red', linestyle='--', label='Pass/Fail Threshold (40 Marks)')
plt.axvline(75, color='orange', linestyle='--', label='Drop Threshold (75
Attendance)')
plt.legend()
plt.grid()
plt.show()


drop_count = (df['Prediction'] == 'Drop').sum()
fail_count = (df['Prediction'] == 'Fail').sum()
pass_count = (df['Prediction'] == 'Pass').sum()

print(f"\nTotal students at risk of dropping out: {drop_count}")
print(f"Total failing students: {fail_count}")
print(f"Total passing students: {pass_count}")
```

Output :





```
sem7\DATA MINING\DA\DA3\Q1\knn.py"
21BBS0115
Classification Report:
              precision    recall  f1-score   support

        Drop       0.80      1.00      0.89         4
        Fail       1.00      0.50      0.67         2
        Pass       1.00      1.00      1.00         6

    accuracy                           0.92        12
   macro avg       0.93      0.83      0.85        12
weighted avg       0.93      0.92      0.91        12


Confusion Matrix:
[[4 0 0]
 [1 1 0]
 [0 0 6]]
```



```
Total students at risk of dropping out: 20
Total failing students: 5
Total passing students: 35
```

RESULT:

The KNN was created thereby classifying the students into one of three categories, pass, fail or drop.

SECTION 2

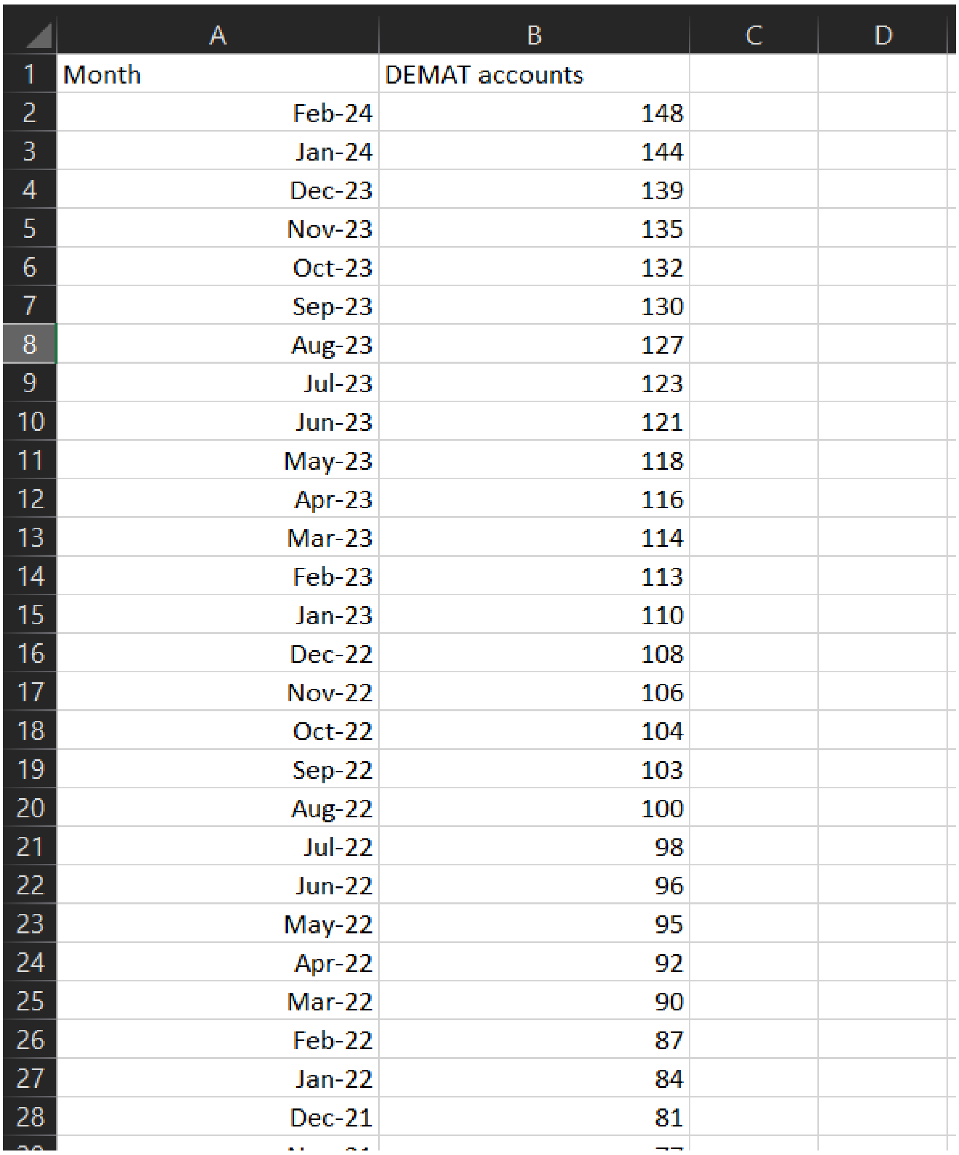
Aim

Implement the Linear regression Technique for prediction of total number of DEMAT accounts for the month of January 2-25. The dataset consists of 60 past months.

LIBRARIES USED: Pandas, Numpy, Matplotlib, Sklearn

Dataset:
https://github.com/ALANT535/DATA-MINING-RESOURCES/tree/main/DA3/Q2

Sample Input



	A	B	C	D
1	Month	DEMAT accounts		
2	Feb-24	148		
3	Jan-24	144		
4	Dec-23	139		
5	Nov-23	135		
6	Oct-23	132		
7	Sep-23	130		
8	Aug-23	127		
9	Jul-23	123		
10	Jun-23	121		
11	May-23	118		
12	Apr-23	116		
13	Mar-23	114		
14	Feb-23	113		
15	Jan-23	110		
16	Dec-22	108		
17	Nov-22	106		
18	Oct-22	104		
19	Sep-22	103		
20	Aug-22	100		
21	Jul-22	98		
22	Jun-22	96		
23	May-22	95		
24	Apr-22	92		
25	Mar-22	90		
26	Feb-22	87		
27	Jan-22	84		
28	Dec-21	81		
29	Nov-21	77		

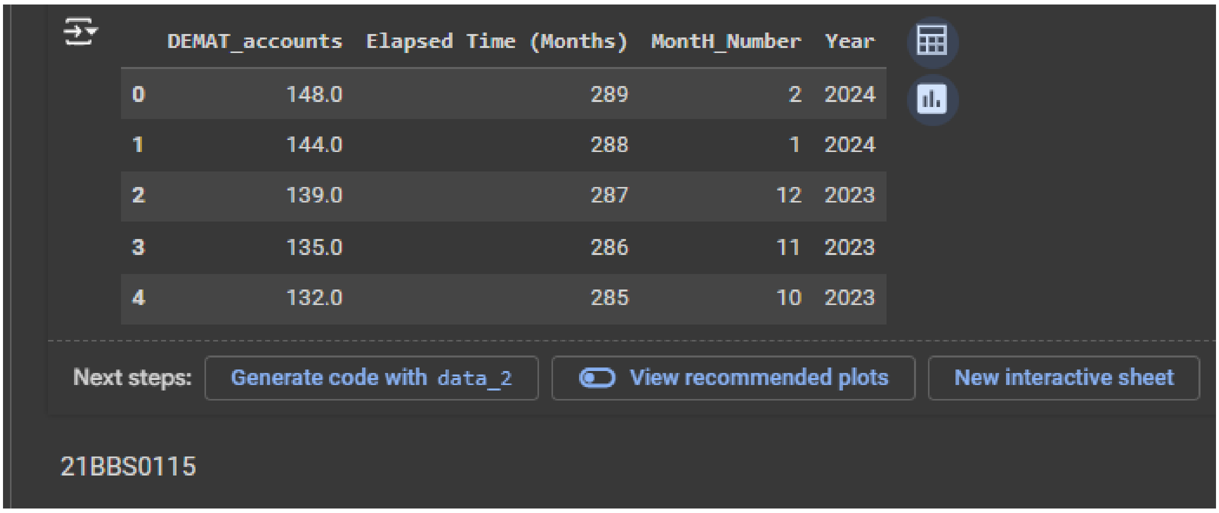
Code

```
import numpy as np
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score


data_df = pd.read_excel('Demat_account_dataset.xlsx')
data_df.columns = ["Month" , "DEMAT_accounts"]

data_df['Date'] = pd.to_datetime(data_df['Month'], format='%B-%Y')
start_date = pd.to_datetime('January-2000', format='%B-%Y')
data_df['Elapsed Time (Months)'] = ((data_df['Date'].dt.year - start_date.year) *
12
                              + data_df['Date'].dt.month - start_date.month)
data_df['MontH_Number'] = data_df['Date'].dt.month
data_df['Year'] = data_df['Date'].dt.year
data_df['Previous_Month_Account'] = data_df['DEMAT_accounts'].shift(-1)

data_2 = data_df.drop(["Month" , "Date" , "Previous_Month_Account"] , axis =
1)
data_2.head()

X = data_2[['Elapsed Time (Months)', 'MontH_Number', 'Year']]
y = data_df['DEMAT_accounts']

X = X.dropna()
y = y[X.index]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

model = LinearRegression()
```

```
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Predictions on test set:", y_pred)
print("Model coefficients:", model.coef_)
print("Model intercept:", model.intercept_)

y_pred = model.predict(X_test)
r2 = r2_score(y_test, y_pred)

print("R-squared (model accuracy):", r2)

input_data = {
    'Elapsed Time (Months)': [300,299,298,297,296,295,294,293],
    'MontH_Number': [1,12,11,10,9,8,7,6],
    'Year': [2025,2024,2024,2024,2024,2024,2024,2024],
}

input_df = pd.DataFrame(input_data)
predicted_demat_accounts = model.predict(input_df)
for i in range(7,-1,-1):
  print("Predicted DEMAT accounts for month " ,
input_data['MontH_Number'][i] ,"2025(in millions):",
predicted_demat_accounts[i])

print("\n\nPredicted DEMAT accounts for January 2025(in millions):",
predicted_demat_accounts[0])
print("21BBS0115")

X = data_df[['Elapsed Time (Months)', 'MontH_Number', 'Year']]
predicted_accounts = (
    model.intercept_ +
    model.coef_[0] * X['Elapsed Time (Months)'] +
    model.coef_[1] * X['MontH_Number'] +
    model.coef_[2] * X['Year']
)
```

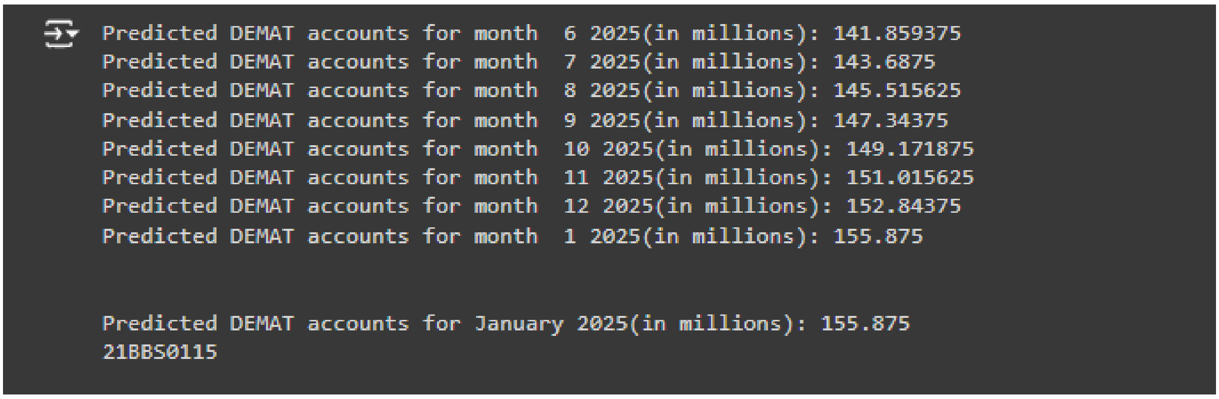
```
plt.figure(figsize=(12, 6))
plt.plot(data_df['Date'], data_df['DEMAT_accounts'], marker='o', label='Actual DEMAT Accounts')
plt.plot(data_df['Date'], predicted_accounts, color='red', label='Predicted Linear Regression Line')
plt.title('Demat Accounts Over Time with Linear Regression Line')
plt.xlabel('Month-Year')
plt.ylabel('Number of Demat Accounts')
plt.xticks(rotation=45)
plt.legend()
plt.grid()
plt.tight_layout()
plt.show()
```

Output

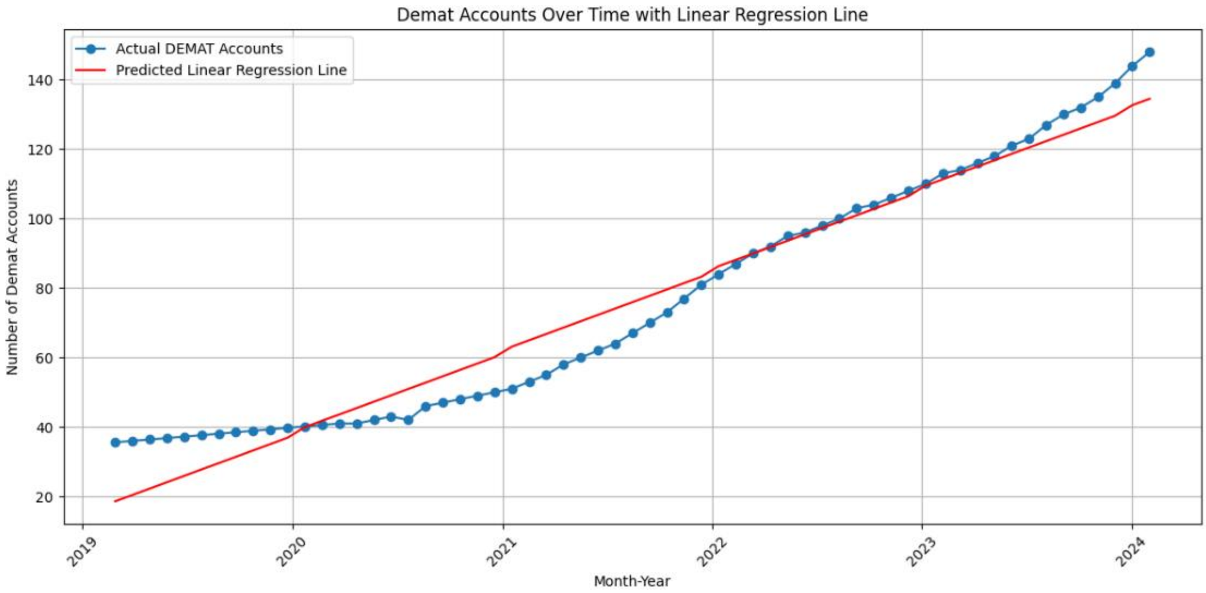


```
   DEMAT_accounts  Elapsed Time (Months)  MontH_Number  Year
0           148.0                    289             2  2024
1           144.0                    288             1  2024
2           139.0                    287            12  2023
3           135.0                    286            11  2023
4           132.0                    285            10  2023
```

```
21BBS0115
```



```
Predicted DEMAT accounts for month  6 2025(in millions): 141.859375
Predicted DEMAT accounts for month  7 2025(in millions): 143.6875
Predicted DEMAT accounts for month  8 2025(in millions): 145.515625
Predicted DEMAT accounts for month  9 2025(in millions): 147.34375
Predicted DEMAT accounts for month  10 2025(in millions): 149.171875
Predicted DEMAT accounts for month  11 2025(in millions): 151.015625
Predicted DEMAT accounts for month  12 2025(in millions): 152.84375
Predicted DEMAT accounts for month  1 2025(in millions): 155.875


Predicted DEMAT accounts for January 2025(in millions): 155.875
21BBS0115
```



Result:

Successfully predicted the number of DEMAT accounts in Januray 2025 using 3 features. Tracked the same using a graph to make it clear how the number of DEMAT accounts is progressing through the years.

SECTION 3

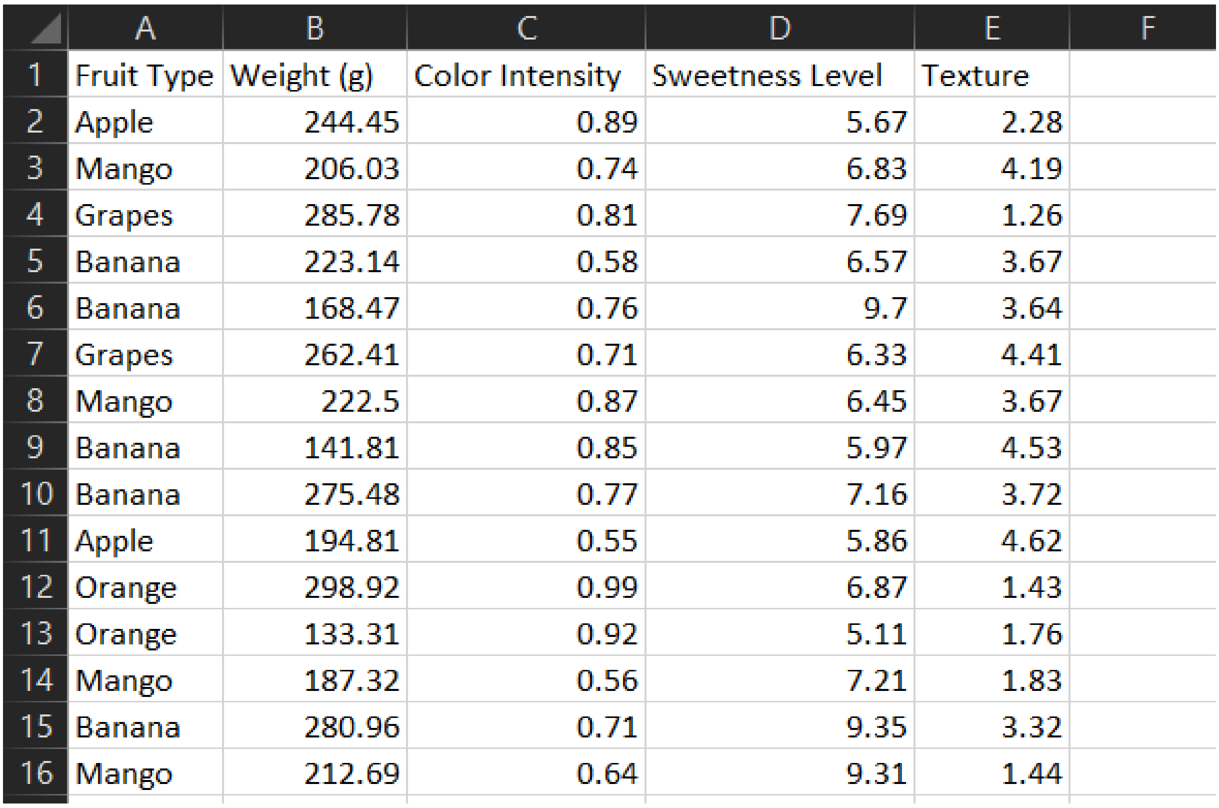
Aim : Implement the Random Forest Supervised Machine Learning Algorithm

Libraries : Numpy, Pandas, sklearn, seaborn

Dataset :

https://github.com/ALANT535/DATA-MINING-RESOURCES/tree/main/DA3/Q3

Sample Input



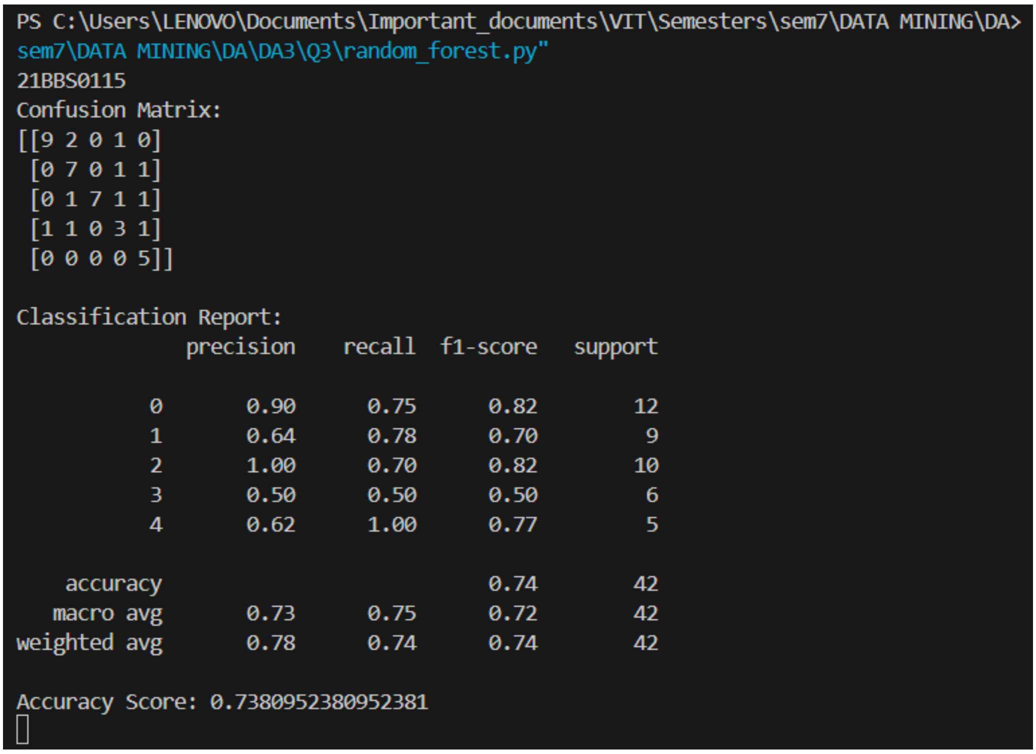
	A	B	C	D	E	F
1	Fruit Type	Weight (g)	Color Intensity	Sweetness Level	Texture	
2	Apple	244.45	0.89	5.67	2.28	
3	Mango	206.03	0.74	6.83	4.19	
4	Grapes	285.78	0.81	7.69	1.26	
5	Banana	223.14	0.58	6.57	3.67	
6	Banana	168.47	0.76	9.7	3.64	
7	Grapes	262.41	0.71	6.33	4.41	
8	Mango	222.5	0.87	6.45	3.67	
9	Banana	141.81	0.85	5.97	4.53	
10	Banana	275.48	0.77	7.16	3.72	
11	Apple	194.81	0.55	5.86	4.62	
12	Orange	298.92	0.99	6.87	1.43	
13	Orange	133.31	0.92	5.11	1.76	
14	Mango	187.32	0.56	7.21	1.83	
15	Banana	280.96	0.71	9.35	3.32	
16	Mango	212.69	0.64	9.31	1.44	

Code

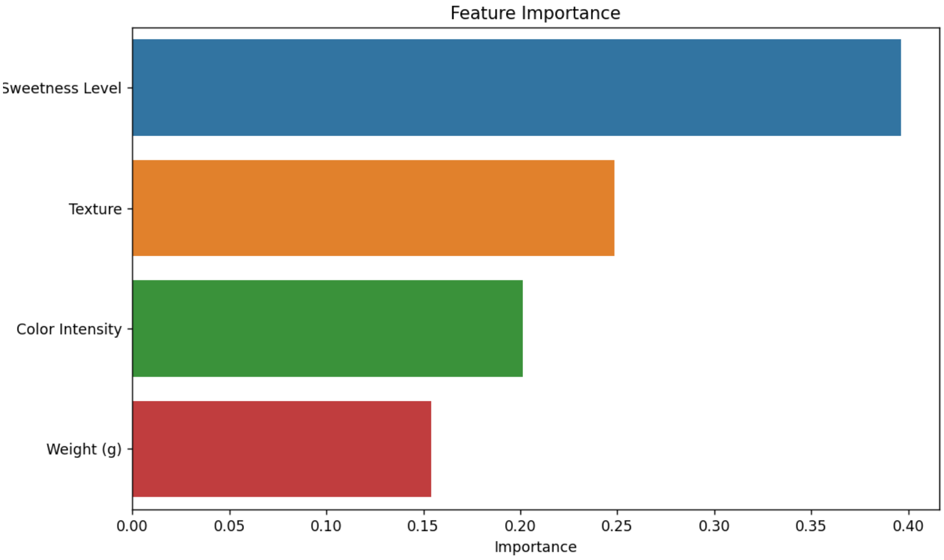
```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

print("21BBS0115")
df=pd.read_csv(r"C:\Users\LENOVO\Documents\Important_documents\VIT\Semesters\sem7\DATA MINING\DA\DA3\Q3\fruits.csv")

df['Fruit Type'] = df['Fruit Type'].astype('category').cat.codes

X = df.drop('Fruit Type', axis=1)
y = df['Fruit Type']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestClassifier(n_estimators=100, random_state=42)

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

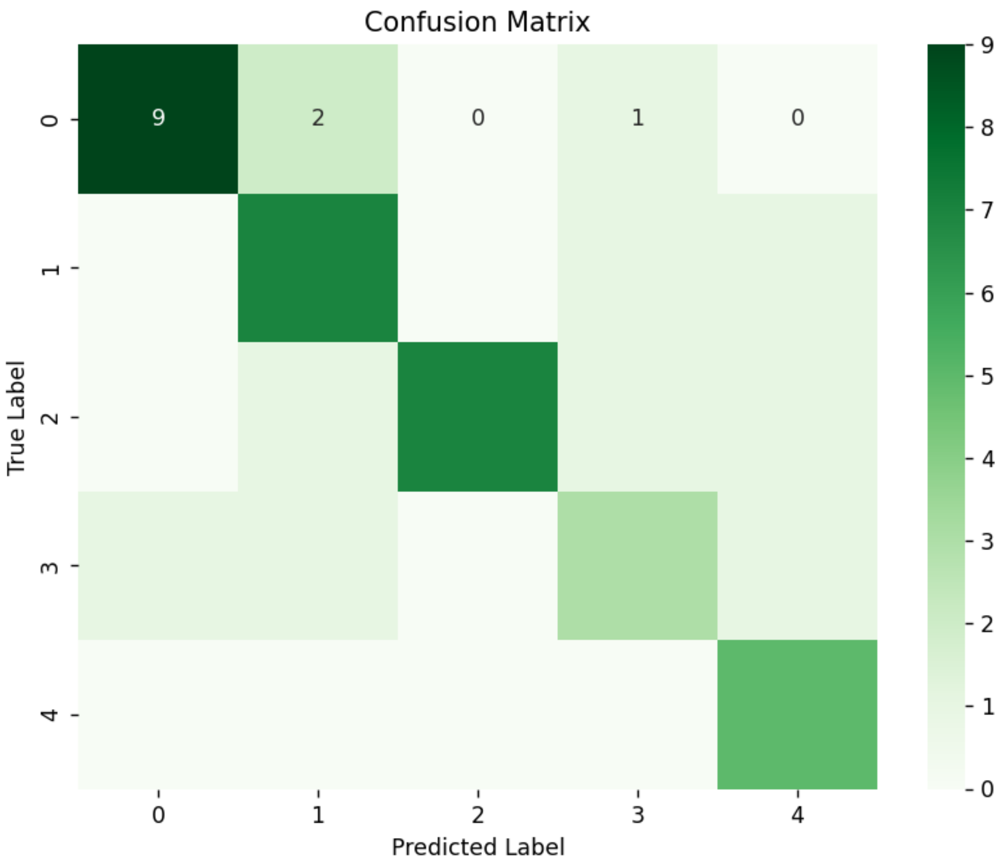
```
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Greens',
xticklabels=df['Fruit Type'].astype('category').cat.categories,
yticklabels=df['Fruit Type'].astype('category').cat.categories)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()


# Plotting Feature Importance
feature_importances = model.feature_importances_
features = X.columns
importance_df = pd.DataFrame({'Feature': features, 'Importance':
feature_importances})
importance_df = importance_df.sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=importance_df)
plt.title('Feature Importance')
plt.show()

plt.figure(figsize=(12, 10))
for i, feature in enumerate(X.columns):
    plt.subplot(2, 2, i + 1)
    sns.histplot(X[feature], bins=15, kde=True)
    plt.title(f'Distribution of {feature}')
plt.tight_layout()
plt.show()
```

Output :



```
PS C:\Users\LENOVO\Documents\Important_documents\VIT\Semesters\sem7\DATA MINING\DA>
sem7\DATA MINING\DA\DA3\Q3\random_forest.py"
21BBS0115
Confusion Matrix:
[[9 2 0 1 0]
 [0 7 0 1 1]
 [0 1 7 1 1]
 [1 1 0 3 1]
 [0 0 0 0 5]]

Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.75      0.82        12
           1       0.64      0.78      0.70         9
           2       1.00      0.70      0.82        10
           3       0.50      0.50      0.50         6
           4       0.62      1.00      0.77         5

    accuracy                           0.74        42
   macro avg       0.73      0.75      0.72        42
weighted avg       0.78      0.74      0.74        42

Accuracy Score: 0.7380952380952381
```





Result:

We have created the Random Forest Algorithm, we were able to assign feature importance to each of the 4 features, and we were also able to classify the fruits into one of 5 categories.

XXXXXXXXXXXXXXXXXXXXXXXXXX