INDEX

Name SHREYA BHARAMANNA PATIL Sub. ML Lab

Std.: VI th Sem Div. E Roll No. 1BM23CS420

Telephone No. E-mail ID.

Blood Group. Birth Day.

Sr.No.	Title	Page No.	Sign./Remarks
1.	To Do Exercise	1-3	
2.	Data Preprocessing	4-5	
3.	Decision Tree : ID3	6-8	
4.	Linear Regression	9-11	
5.	Logistic Regression	12-14	
6.	KNN	15-16	
7.	~~De~~ Random Forest	17	
8.	Random Forest	18	
9.	K-Means Algorithm	21	
10	Ada Boost	19	
11.	PCA	23	

Lab-0

Write a python program to import and export data using Pandas library functions

To Do:

Method 1: Initializing values directly into DataFrame
Insert your know values, five rows of data with column headings as 'USN, Name, Marks'

→
```
import pandas as pd
data = {'USN' : ['1BM22CS001', '1BM22CS002',
                 '1BM22CS003', '1BM22CS004', '1BM22-
                  CS005'],
        'Name' : ['Ankita', 'Aniket', 'Anish', 'Anil',
                  'Anita'],
        'Marks' : [98, 90, 45, 86, 78]}
df = pd.DataFrame(data)
print(df)
```

Method 2: Importing datasets from sklearn.datasets
Loading diabetes datasets sklearn.datasets.load_diabetes

→
```
from sklearn.datasets import load_diabetes
diabetes = load_diabetes()
df = pd.DataFrame(diabetes, columns = diabetes.
                  feature_names)
df['target'] = diabetes.target
print(df)
```

Method-3: Importing Datasets from specific .csv file sample_sales_data.csv

→
```
path = r"/content/sample_sales_data.csv"
df = pd.read_csv(path)
print(df)
```

Method 4 : Downloading datasets from existing dataset repositories like kaggle, UCI, Mendely, KEEL, etc.

→
```
path = r"/content/Dataset of Diabetes.csv"
df = pd.read_csv(path)
print(df)
```

TO DO :

1. HDFC Bank Ltd., ICICI Bank Ltd, Kotak Mahindra Bank Ltd.

```
tickers = ["HDFCBANK.NS", 'ICICIBANK.NS", "KOTAKBANK.NS"]
```

→
```
import yfinance as yf
import matplotlib.pyplot as plt
tickers = ["HDFCBANK.NS", 'ICICIBANK.NS', 'KOTAK
    BANK.NS']
```

2. Start date : 2024-01-01 , End date : 2024-12-30

→
```
data = yf.download(tickers, start = '2024-01-01',
    end = '2024-12-30', group_by = tickers)
print(data)
```

3. Plot the closing price and daily returns for all the three banks mentioned

→
```
# HDFC BANK
HDFC = ~~data[HDFC]~~ data['HOFCBANK.NS']
HDFC['Daily Return'] = HDFC['Close'].pct_change()
plt.figure(figsize = (12,6))
plt.subplot(2,1,1)
HDFC['Close'].plot(title = 'HDFC BANK - Closing Price')
plt.subplot(2,1,2)
HDFC['Daily Return'].plot(title = 'HDFC BANK-
```

```
Daily Return', color='orange')
plt.tight_layout()
plt.show()

# KOTAK BANK
KOTAK = data['KOTAKBANK.NS']
KOTAK['Daily Return'] = KOTAK['Close'].pct_change()
plt.figure(figsize=(12,6))
plt.subplot(2,1,1)
KOTAK['Close'].plot(title='KOTAK BANK - Close
                    Price')
plt.subplot(2,1,2)
KOTAK['Daily Return'].plot(title='KOTAK BANK-
          Daily Return', color='orange')
plt.tight_layout()
plt.show()
```

Lab-1

Demonstrate various data pre-processing techniques for a given dataset

i. To load .csv file into the data frame

→
```
import pandas as pd
import numpy as np

path = r "housing.csv"
df = pd.read_csv(path)
print(df)
```

ii. To display information of all the columns

→
```
print(df.info())
```

iii. To display statistical information of all numerical

→
```
print(df.describe())
```

iv. To display the count of unique labels for "Ocean Proximity" column

→
```
print( df["Ocean Proximity"].value_counts())
```

v. To display which attributes (columns) in a dataset have missing values count greater than zero

→
```
missing_values = df.isnull().sum()
print(missing_values[missing_values>0])
```

1. Which columns in the dataset had missing values ? How did you handle them ?

→ None of column had missing values but If had then numerical column's NaN can be replaced with median and categorial column's

null can be replaced with mode.

2. Which categorical columns did you identify in the dataset? How did you encode them.

→ The diabetes dataset had gender, and class as categorical column and adult dataset had workclass, education, marital-status, occupation, relationship, race, gender, native-country and income as categorical column
Using Ordinal Encoder we can encode categorical columns

3. What is the difference between Min-Max Scaling and Standardization? when would you one over the other?

→ Min-Max scalling, also called normalization, transforms data to fit within a specific range (usually 0 to 1) by scaling based on the min-max values in the dataset.
standardization scales data by subtracting mean and dividing by the standard deviation, resulting in distribution with a mean of 0 and standard deviation of 1.
Use min-max scaling when you need your data need to be in specific range, while standardization when your data is normally distributed and outliers might be present.

05.03.24

Lab-2

Decision Tree : ID3 (Iterative Dichotomizer 3) Algorithm.

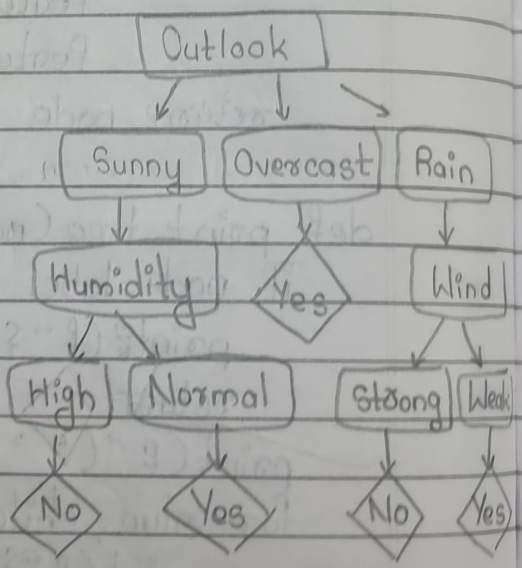
```
import numpy as np
import pandas as pd
from collections import Counter

class Node:
    def __init__(self, feature=None, value=None,
                 label=None):
        self.feature = feature
        self.value = value
        self.label = label
        self.children = {}

def entropy(y):
    counts = np.bincount(y)
    probabilities = counts / len(y)
    return -np.sum([p * np.log2(p) for p in
                    probabilities if p > 0])

def information_gain(X, y, feature):
    total_entropy = entropy(y)
    values, counts = np.unique(X[:, feature],
                               return_counts=True)
    weighted_entropy = sum((counts[i] / sum(counts))
                           * entropy(y[X[:, feature] == v]) for
                           i, v in enumerate(values))
    return total_entropy - weighted_entropy

def best_feature_to_split(X, y):
    gains = [information_gain(X, y, i) fo i in
                              range(X.shape[1])]
    return np.argmax(gains)
```

```
def id3(X, y, features):
    if len(set(y)) == 1:
        return Node(label = y[0])
    if len(features) == 0:
        return Node(label = Counter(y).most_
                common(1)[0][0])
    best_feature = best_feature_to_split(X, y)
    node = Node(feature = features[best_feature])
    feature_values = np.unique(X[:, best_feature])
    for value in feature_values:
        sub_x = X[X[:, best_feature] == value]
        sub_y = y[X[:, best_feature] == value]
        if len(sub_y) == 0:
            node.children[value] = Node(label =
            Counter(y).most_common(1)[0][0])
        else:
            node.children[value] = id3(np.delete(
            sub_X, best_feature, axis=1), sub_y,
            features[:best_feature] + features[best_
            feature+1:])
    return node

def print_tree(node, depth=0):
    if node.label is not None:
        print(f"{' ' * depth}Leaf: {node.label}")
        return
    print(f"{' ' * depth}Feature: {node.feature}")
    for value, child in node.children.items():
        print(f"{' ' * depth} value: {value}")
        print_tree(child, depth + 1)
```

```
# Example dataset
data = pd.DataFrame({
    'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain',
                'Rain', 'Rain', 'Overcast', 'Sunny', 'Rain',
                'Sunny'...],
    'Temperature': ['Hot', 'Hot', 'Hot', 'Mild'...],
    'Humidity': ['High', 'High', 'High', 'Normal'...],
    'Wind': ['Weak', 'Strong', 'Weak', 'Strong'...],
    'PlayTennis': ['No', 'Yes', 'No', 'No'...]})

X = data.iloc[:, :-1].apply(lambda col: pd.factorize
        (col)[0]).to_numpy()
y = pd.factorize(data['PlayTennis'])[0]
features = list(data.columns[:-1])

decision_tree = id3(X, y, features)
print_tree(decision_tree)
```

Output:

```
Feature : Outlook
Value : 0
    Feature : Humidity
    Value : 0
        Leaf : 0
    Value : 1
        Leaf : 1
Value : 1
    Leaf : 1
Value : 2
    feature : Wind
    Value : 0
        Leaf : 1
    Value : 1
        Leaf : 0
```



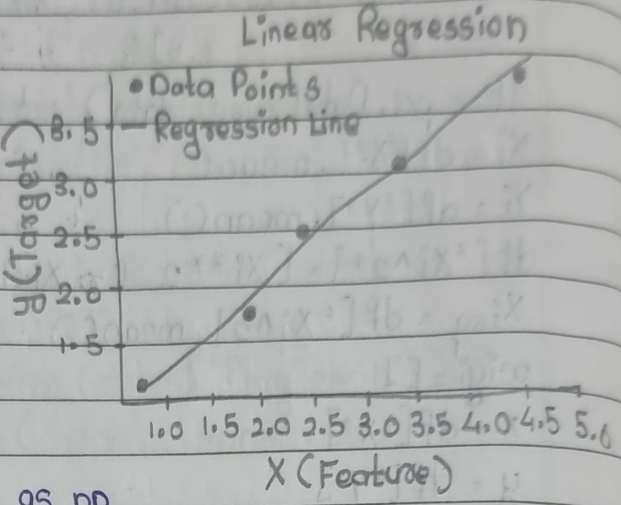
4/12/23

Lab - 3

Linear Regression

```
import pandas as pd
import matplotlib.pyplot as plt

data = {'X': [1,2,3,4,5], 'Y': [1.2,1.8,2.6,3.2,
                                3.8]}
df = pd.DataFrame(data)
Xi = df['X'].mean()
Yi = df['Y'].mean()
df['Xi^2'] = [Xi**2 for Xi in df['X']]
Xisq = df['Xi^2'].mean()
XiYi = []
x = df['X']
y = df['Y']
for i in range(len(x)):
    XiYi.append(x[i]*y[i])
df['XiYi'] = XiYi
XiYi2 = df['XiYi'].mean()

a1 = (df['XiYi'].sum() - len(df)*Xi*Yi)/
    (df['X'].apply(lambda x: x**2).sum()
        - len(df)*Xi**2)
a0 = Yi - a1*Xi

x = int(input())
y = a0+a1*x
print(y)

plt.scatter(df['X'], df['Y'], color = 'blue', label
        = 'Data points')
plt.plot(df['X'], a0+a1*df['X'], color = 'red',
```

```
label = 'Regression Line')
plt.title('Linear Regression')
plt.xlabel('X (Feature)')
plt.ylabel('y (Target)')
plt.legend()
plt.show()
```



Output :

7

5.16

9

6.48

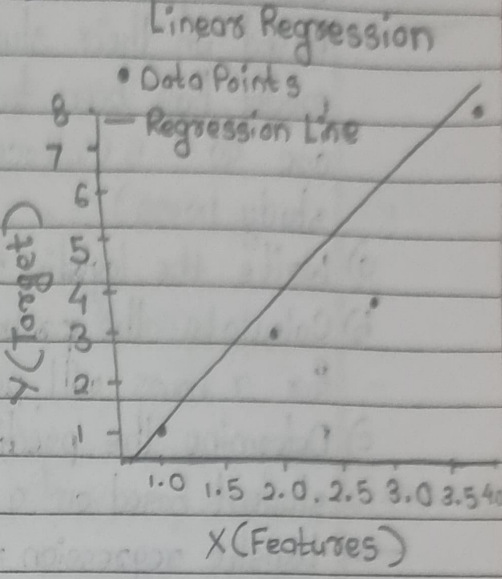
```
import numpy as np
import matplotlib.pyplot as plt

X = np.array([1,2,3,4])
y = np.array([1,3,4,8])

X_matrix = np.c_[np.ones(len(X)), X]
theta = np.linalg.inv(X_matrix.T @ X_matrix)
        @ X_matrix.T @ y

b, m = theta
y_pred = m * X + b

print(f"Slope (m): {m}")
print(f"Intercept (b): {b}")

plt.scatter(X, y, color='blue', label='Data points')
plt.plot(X, y_pred, color='red', label=
```

```
"Regression Line")
plt.title("Linear Regression (Matrix Form)")
plt.xlabel("X (Feature)")
plt.ylabel("Y (Target)")
plt.legend()
plt.show()
```

Output :

Slope (m): 2.20

Intercept (b): -1.5



19/3/2025

Lab 4

Logistic Regression

1. Consider a binary classification problem where we want to predict whether a student will pass or fail based on their study hours. The logistic regression model has been trained and the learned parameters are $a0 = -5$ (intercept) and $a1 = 0.8$ (coefficient for study hours)

a) Write the logistic regression equation for this problem

b) Calculate the probability that a student who studies for 7 hours will pass.

c) Determine the predicted class (pass/fail) for this student based on a threshold of 0.5

→ Logistic regression $= \frac{1}{1 + e^{-(a_0 + a_1 x)}}$

$a_0 = -5 \quad a_1 = 0.8$

$$P(\text{Pass} | x) = \frac{1}{1 + e^{-(-5 + 0.8x)}}$$

$$z = a_0 + a_1 x$$
$$= -5 + (0.8 \times 7)$$
$$= 0.6$$

$$P(\text{Pass} | 7) = \frac{1}{1 + e^{-0.6}}$$
$$= 0.6456 \approx 64.56\%$$

Predicted class is pass as $0.6457 \geq 0.5$

2. Consider $z = [2, 1, 0]$ for three classes. Apply Soft Max function to find the probability values of three classes.

→ $SoftMax(Z_k) = \frac{e^{z_k}}{\sum_{j=1}^{k} e^{z_j}}$

$Z = [2, 1, 0]$

$Softmax(Z_1) = \frac{e^2}{e^2+e^1+e^0} = 0.665$

$softmax(Z_2) = \frac{e^1}{e^2+e^1+e^0} = 0.2447$

$Softmax(Z_3) = \frac{e^0}{e^2+e^1+e^0} = 0.9003$

The probabilities of three classes — 66.5%, 24.47%, 90%

Q. After building the logistic regression models, write the answer for the following questions

1. For dataset file "HR_Comma-Sep.csv"

a) Which variables did you identify as having a direct and clear impact on employee relation? why?

→ Variables impacting employee relation:

- The satisfaction level has a strong direct impact on relation of employee with lower satisfaction are more likely to have.
- Average monthly hours and number of projects also influence relation. Overworked or underutilized employees may leave.
- Salary level and promotion. employees with low salary and no promotions are more likely to leave.

b) What are the accuracy of your logistic regression model? Do you think this is a good accuracy? Why or Why not?

→ The logistic regression model achieved an accuracy of X%.

For accuracy -
- If above 80% → Yes, It's a good model
- If below 70% → It may need improvement
- If around 75% → decent but can be improved

2. For Zoo dataset

a) Did you perform any data preprocessing steps? If yes, what were they, and why were they necessary?

→ • Dropped 'animal_name' → Not useful for classification.
• Split data → 80% training, 20% testing
• Used multinomial Logistic Regression for multiclass classification.

b) Where there any missing or inconsistent values in the dataset? How did you handle them?

→ • No explicit missing values found.
• If present, they could be handled by fitting or dropping rows.

c) What does the confusion matrix tell you about the performance of your model?

→ • It shows how well the model classified each class
• High diagonal values = good predictions,
off-diagonal values = mis classifications

d) Which class type were most frequently misclassified? Why do you think this happend?

→ • Similar feature values across some class
• Possible class imbalance
• Logistic Regression assumes linear decision boundaries, which may not fully capture complex relationship.

Lab 5

Build KNN classification model for a given dataset

Q. Consider the following dataset, for k=3 and test data (x, 35, 100) as (Person, Age, Salary k) solve using KNN classifier model and predict target

Person	Age	Salary k	Target	Distance	Rank
A	18	50	N	52.83	5
B	23	55	N	46.58	4
C	24	70	N	31.96	2
D	41	60	Y	40.44	3
E	43	70	Y	31.06	1
F	38	40	Y	60.08	6
X	35	100			

$$\text{Distance} = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

$A = \sqrt{(35-18)^2+(100-50)^2} = 52.83$

$B = \sqrt{(35-23)^2+(100-55)^2} = 46.58$

$C = \sqrt{(35-24)^2+(100-70)^2} = 31.96$

$D = \sqrt{(35-41)^2+(100-60)^2} = 40.44$

$E = \sqrt{(35-43)^2+(100-70)^2} = 31.06$

$F = \sqrt{(35-38)^2+(100-40)^2} = 60.08$

K = 3 ⟶ X = Yes

- For iris dataset -

How to choose the K value? Demonstrate using accuracy rate and error rate

→ To determine the best K value,

Low K (eg: 1-3): High variance more overfitting

High K (eg: 10+): More bias, risk of underfitting

Best K: Found by plotting accuracy V/S K and error rates

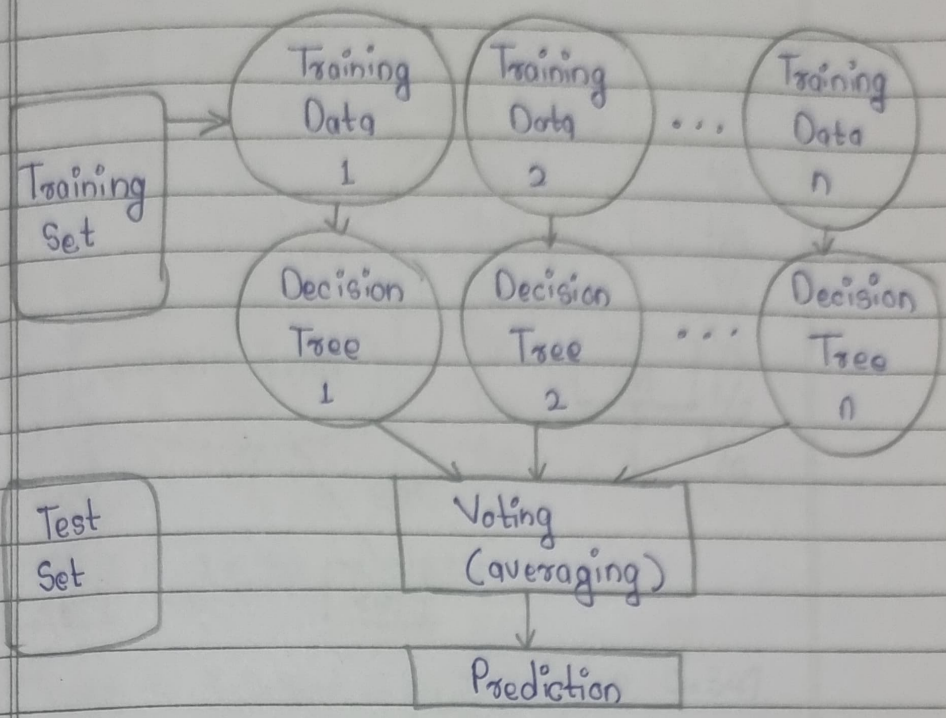
V & K, selecting the k with highest accuracy and lowest error.

- For diabetes dataset
What is the purpose of feature scaling? How to perform it?

→ Features -
- Equal influence of all features
- Faster convergence in training
- Better performance and accuracy

To performs scaling -
Use StandardScaler() in scikit-learn which transform data into a standard normal distributions

Working of Random Forest Algorithm



Steps to explain working Random Forest Algorithm:

Step 1: Select random samples from a given data or training set.

Step 2: This algorithm will construct a decision tree for every training data

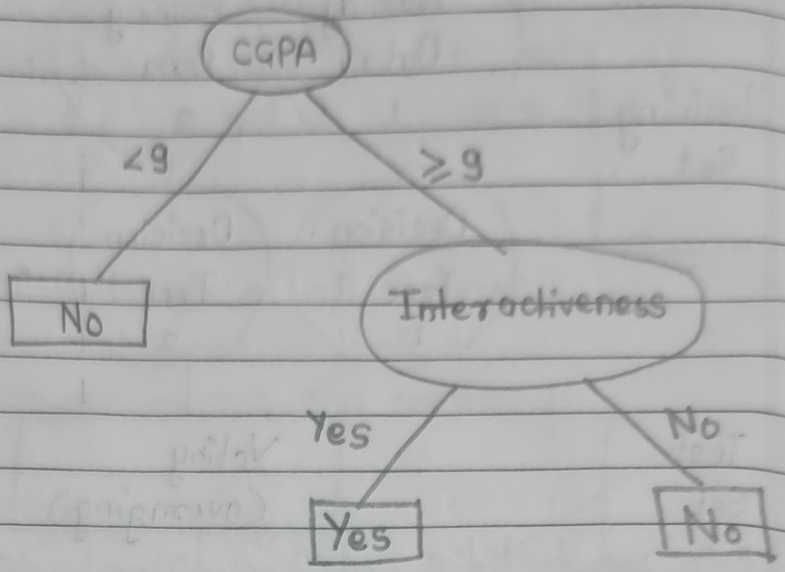
Step 3: Voting will take place by averaging the decision tree

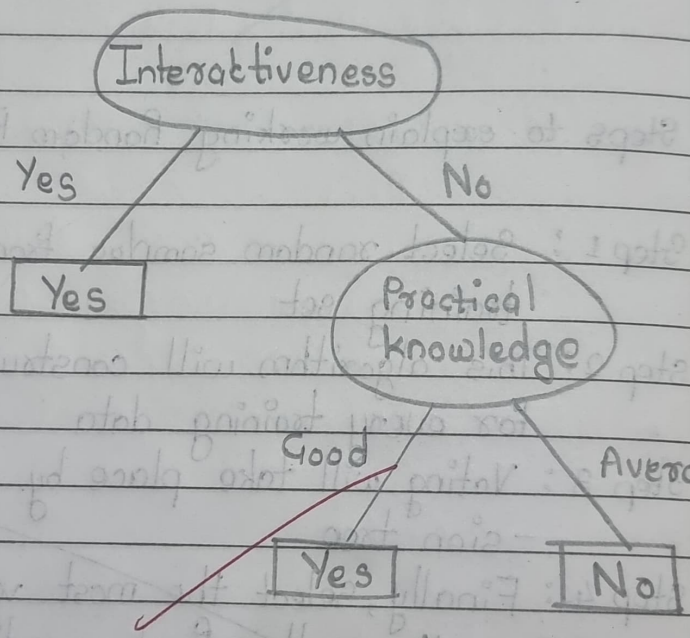
Step 4: Finally, select the most voted prediction result as the final prediction result.

16.04

Lab 7

Random Forest Ensemble





1. For 'iris.csv' dataset, What is the best accuracy score and confusion matrix of the classifier you observed and using how many trees

→ • Best Accuracy Score Observed: 1.0
• Number of Trees (n-estimators): best_n
• Confusion Matrix: [[16 0 0]
[0 14 0]
[0 0 15]]

Lab 8

AdaBoost Algorithm

CGPA	Predicted Job offer	Actual Job Offer	Weight
>=9	Yes	Yes	1/6
<9	No	Yes	1/6
>=9	Yes	No	1/6
<9	No	No	1/6
>=9	Yes	Yes	1/6
>=9	Yes	Yes	1/6

$$\varepsilon_{CGPA} = 2 * 1/6 = 0.333$$

$$\alpha_{CGPA} = \frac{1}{2} \times \ln\left(\frac{1-0.3333}{0.333}\right) = 0.347$$

$$Z_{CGPA} = \frac{1}{6} * 4 * e^{-0.347} + \frac{1}{6} * 2 * e^{0.347}$$

$$= 0.9428$$

$$Wt(d_j)_{i+1} = \frac{1/6 * e^{-0.347}}{0.9428} = 0.1249$$

$$Wt(d_j)_{i+1} = \frac{1/6 * e^{0.347}}{0.9428} = 0.2501$$

CGPA	Predicted Job offer	Actual Job Offer	Weight
>=9	Yes	Yes	0.1249
<9	No	Yes	0.2501
>=9	Yes	No	0.2501
<9	No	No	0.1249
>=9	Yes	Yes	0.1249
>=9	Yes	Yes	0.1249

1. For "income.csv" dataset, What is the best accuracy score and confusion matrix of the classifier you observed and using how many trees?

→ Accuracy with 10 estimators : 0.8277
Confusion Matrix (10 estimators):
[[10722 387]
[2138 1406]]
Best Accuracy : 0.831 with n_estimators = 42

PAGE NO: 21
DATE:

Lab 9

K-Means Algorithm

Iteration 1

Record	Coordinates	Distance to c1	Distance to c2	Assigned Cluster
R1	(1.0, 1.0)	0.0	7.21	C1
R2	(1.5, 2.0)	1.12	6.12	C1
R3	(3.0, 4.0)	3.61	3.61	C1
R4	(5.0, 7.0)	7.21	0.0	C2
R5	(3.5, 5.0)	4.12	2.5	C2
R6	(4.5, 5.0)	5.31	2.06	C2
R7	(3.5, 4.5)	4.30	2.92	C2

Cluster1 {R1, R2, R3} and Cluster2 {R4, R5, R6, R7}
new centroids are:

C1 = (1.0+1.5+3.0)/3 , (1.0+2.0+4.0)/3
= 5.5/3 , 7.0/3
= 1.83, 2.33

C2 = (5.0+3.5+4.5+3.5)/4 , (7+5+5+4.5)/4
= 16.5/4 , 21.5/4
= 4.12, 5.37

Record	Coordinates	Distance to c1	Distance to c2	Assigned Cluster
R1	(1.0, 1.0)	1.57	5.64	C1
R2	(1.5, 2.0)	0.47	4.52	C1
R3	(3.0, 4.0)	2.12	1.63	C2
R4	(5.0, 7.0)	5.57	1.91	C2

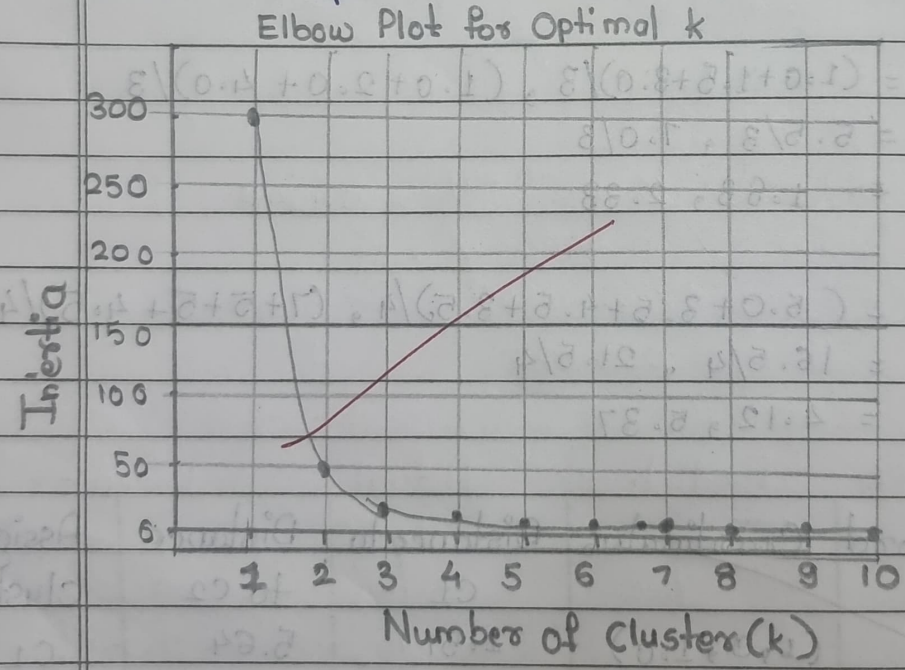
R5	(3.5, 5.0)	3.16	0.72	C2
R6	(4.5, 5.0)	3.58	0.52	C2
R7	(3.5, 4.5)	2.63	1.05	C2

Cluster 1 {R1, R2} and Cluster 2 {R3, R4, R5, R6, R7}
now centroids are

$C_1 = (1.0+1.5)/2, (1.0+2.0)/2$
$= 2.50/2, 3.0/2$
$= 1.25, 1.5$

$C_2 = (3.0+5.0+3.5+4.5+3.5)/5, (4+7+5+5+4.5)/5$
$= 19.5/5, 25.5/5$
$= 3.9, 5.1$

1. For 'iris.csv' dataset, Draw the elbow plot. What was the optimal K value obtained



Elbow point was observed at point 3

Lab 10

Principle Component Analysis

feature	Eg 1	Eg 2	Eg 3	Eg 4
x_1	4	8	13	7
x_2	11	4	5	14

Eigen Values

$\lambda_1 = 30.3849$

$\lambda_2 = 6.6151$

Eigen Vectors

$e_1 = \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}$

$e_2 = \begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$

→ $matrix = \begin{bmatrix} 4 & 8 & 13 & 7 \\ 11 & 4 & 5 & 14 \end{bmatrix}$

mean centre the data

$mean\ x_1 = \frac{4+8+13+7}{4} = 8$

$mean\ x_2 = \frac{11+4+5+14}{4} = 8.5$

$Y_{centered} = \begin{bmatrix} 4-8 & 8-8 & 13-8 & 7-8 \\ 11-8.5 & 4-8.5 & 5-8.5 & 14-8.5 \end{bmatrix} = \begin{bmatrix} -4 & 0 & 5 & -1 \\ 2.5 & -4.5 & -3.5 & 5.5 \end{bmatrix}$

$Z = e_i^T . Y_{centered}$

$Z_1 = (0.5674)(-4) + (-0.8303)(2.5) = -4.30535$

$Z_2 = (0.5574)(0) + (-0.8303)(-4.5) = 3.73635$

$z_3 = (0.5574)(5) + (-0.8303)(-3.5) = 5.69305$

$z_4 = (0.5574)(-1) + (-0.8303)(5.5) = -5.12405$

$z = [-4.30535, 3.73635, 5.69305, -5.1240]$

Model Accuracy without PCA
SVM : 0.8804
Logistic Regression : 0.8533
Random Forest : 0.8859

Model Accuracy with PCA
SVM : 0.8424
Logistic Regression : 0.8641
Random Forest : 0.8533

7/5/2025