INDEX

Name Shreya Nitawe Std _____ Sec 6E

Roll No. _____ Subject ML-LAB School/College _____

School/College Tel. No. _____ Parents Tel. No. _____

Sl. No.	Date	Title	Page No.	Teacher Sign / Remarks
1.	5/3/25	Lab 0 [todo list]	1-2	
2.	5/3/25	Lab 1 [data processing]	3-4	
3.	12/3/25	Lab 2 [ID3]	5-7	
4.	19/3/25	Lab 3 [Liner Reg < Normal Naive]	8-9	
5.	2/4/25	Lab 4 [logistic Reg > Binary Multi]	10-11	
6.	2/4/25	Lab 5 [KNN]	12-13	
7	16/4/25	Lab 6 [Random forest]	13-14	
8.	7/5/25	Lab 7 [Random forest]	15-15	
9.	7/5/25	Lab 8 [Boosting method]	16-17	
10.	7/5/25	Lab 9 [K Means Algo]	18-19	
11.	7/5/25	Lab 10 [dimensionality red. using PCA Method]	20-21	

Date 5/3/25
Page 1

Lab-0

→ Write a python prog to import & export data using Pandas library functions.

To do:

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

→
```
import pandas as pd
data = { 'USN': ['1BM22CS266', '1B--267', '1B--268'],
         'Name': ['Shreya', 'Shr', 'Shree', 'Shri'],
         'Marks': [99, 98, 97, 96] }
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, Mendeley, KSet, etc

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

TODO:

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", "KOTBANK.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 & daily returns for all the three banks mentioned

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

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

Date 5/3/25
Page 3

Lab - 1

Demonstrate various data pre-processing techniques for a given dataset

i) To load .csv file into the dataframe

→
```
import pandas as pd
import numpy as np
path = r"C:\housing.csv"
df = pd.read_csv(path)
print(df)
```

ii) To display info of all the columns

→
```
print(df.info())
```

iii) To display statistical info 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 colum 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.

Lab-1b

2. Which categorial column did you identify in the dataset? How did you encode them?

→ The diabetes dataset had gender, and class as categorial column and adult dataset had workclass, education, marital-status, occup, relt°, race, gender, native-country and income as categorial column. Using Ordinal encoder, we can encode the categorial columns.

3. What is the diff b/w Min-Max scaling & standardization? When would you use one over the other?

→
- Min-Max scaling, 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 to be in specific range, while standardization, when your data is normally distributed and outliers might be present.

5/5

Date 12/3/25
Page 5

Lab-2

ID3

```
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) for i in range
        (X.shape[1])]
    return np.argmax(gains)
```

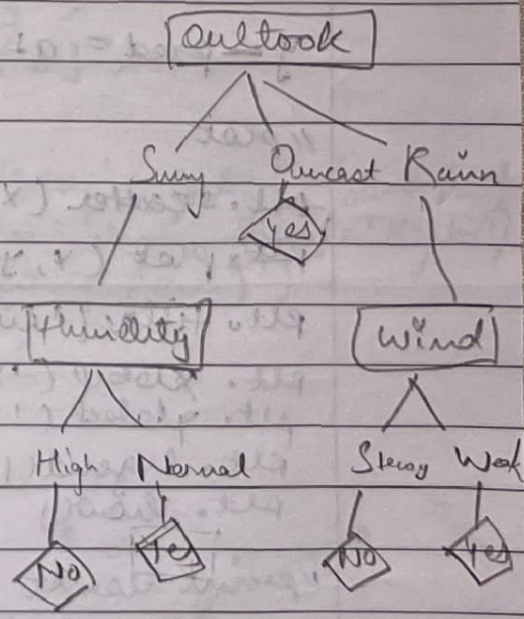
Date __/__/__
Page 6

```
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' : [ 'S', 'S', 'O', 'Rain', 'Sny' ..... ],
        'Temperatm' : [ 'Hot', 'Mild', 'Cold' .... ],
        'Humidity' : [ 'High', 'Normal' .... ],
        'Wind' : [ 'Weak', 'Strong' ..... ],
        'PlayTennis' : [ 'No', 'Yes' .... ]
    } )

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 )
```

O/P

```
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
```



Date 19/3/25
Page 8

Lab - 3

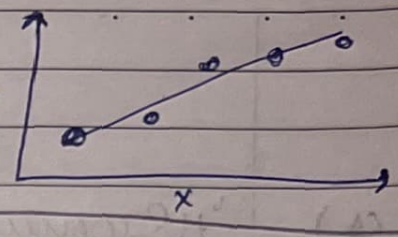
Linear Regression

— using eq° (normal) form:

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

x = np.array ( [1,2,3 4 5])
y = np-array ( [1.2, 1.8, 1.6, 3.2, 3.8])

n = len(x)      # no. of data pairs

sum-x = np.sum (x)
sum-y = np. sum (y)
sum-xy = np.sum ( x*y)
sum-x2 = np.sum ( x**2)

a1 = ( n* sum-xy - sum-x * sum-y) /
        n* sum-x2 - sum-x**2)
a0 = ( sum-y - a1 * sum-x) / n

y-pred = a1 * x + a0

# plot
plt. scatter (x, y, color='blue', label='Datapoints')
plt. plot ( x, y-pred, color='red', label='Regre. line')
plt. title ('Linear Regression (uniformula)')
plt. xlabel ('x')
plt. ylabel ('y')
plt. legend()
plt. show()
# print result
print ( f" slope(a1): {a1} ")
''      ( f" intercept(a0) : {a0} ")
''      ( f" prediction (y = a1 * x + a0) : ")
''      ( f" x = {q}, y-pred = {yi-pred} ")
```

O/P

slope (a_0) = 0.66
intercept (a_1) = 0.54
prediction ($y = a_1 x + a_0$)
$x = 9$, y-pred = 3.84

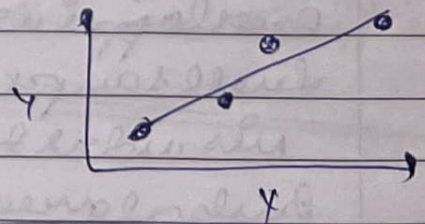


— using matrix form:

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

x = np.array([1 2 3 4 5 ..])
y = np.array([1.2, 2.8, 2.6 ..])

# create matrix [1, x]
X_matrix = np.c_[np.ones(len(x)), x]

# calculate θ (slope & intercept) using normal eqn
theta = np.linalg.inv(X_matrix.T @ X_matrix)
                @ X_matrix.T @ y

# extract intercept & slope
b, m = theta

# make prediction
y_pred = m*x + b

# print result
print(f"Slope (m): {m}")
print(f"intercept (b): {b}")

# plot
plt.scatter(x, y, color='blue', label='Data')
plt.plot(x, y_pred, color='red', label='Reg line')
plt.title('Linear Reg (Matrix form)')
plt.xlabel('x')
plt.ylabel('y')
plt.legend
plt.show
```

O/P

Slope (m): 0.66
intercept (b): 0.54



Date 2/4/25
Page 10

Lab-4
Logistic Regression

(1) "HR_common_sep"

→ which variable did you identify as having a direct & clear impact on employee retention?

- Satisfaction level : low satisf → high attrition
- No. of projects : too few or too many → high attrition
- Avg. monthly hours : extreme hours → high attrition
- Promotion status : ~~promotion~~ in 5 years → high attrition
- Salary : low salary → high attrition

→ What was the accuracy of log. reg model? Is it good accuracy?

- Accuracy : 76 - 80 %

→ it is decent but not perfect, correct prediction 3 out of 4 cases.
eg: employee retention is influenced by many complex factors that not in dataset
⇒ missing external factors (jobs markets...)
⇒ class imbalance may effect prediction
⇒ feature interactions (eg salary + work hours) may not be captured well.

(2) Iris dataset

(iv) which class type most freq misclassified?
Similar species caused confusion.
overlapping features
Small sample size for rare ones leads to mis understanding
Imbalanced classes -

Date __/__/__
Page 11

(2) Zoo. dataset

→ Performed data preprocessing steps? If yes then what & why necessary?

→ Steps:

S ① Converted categorial data into Numerical.
- as ML model works with Numerical data.

S ② Checked for missing values & handled them
- as missing values can cause errors or biased predictions

S ③ Standardized features.
- as some features may have diff scales, thus it ensures equal contribution to model learning.

→ Any missing or inconsistent value? & how to handle
- No, there were NO missing or inconsistent value in this dataset but if any, then
→ filling missing values using mode/mean, median, custom, etc
→ Standardize levels to maintain consistency.

→ Confusion Matrix tells?
- Show how well the model classifies each animal type
- High diagonal values → good prediction
- Off diagonal values → misclassifications

Date 2/4/25
Page 72

Lab-5
KNN

Q Solve for K=3, (X 35, 100)

Person	Age	Salary K	Target	Distance	Rank
A	18	52	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	31.00	1
F	38	40	Y	60.08	6
X	35	100	?		

distance: $\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$

Rank 3 → E → Y, C → N, D → Y } X → Y.

Answer: For X(35, 100) Target = Y (yes)

(1) Iris dataset

→ How to choose k value? use Accuracy Rate & Error Rate

Choose K

Start with diff values of K (eg

Split the dataset into training & testing data

Train the KNN model with each K &

Calculate the accuracy

Use Acc Rate & Error Rate

- Accuracy Rate: measure how well the model classifies the data
- Error Rate: Error Rate = 1 - Acc. Rate
- Plot K vs Accuracy & K vs Error Rate to find optimal K.
- Small K may lead to overfitting, large K may lead to underfitting.

(2) Diabetes Dataset

→ What is purpose & feature scaling? How to perform it?

Purpose:

- ensures all features contribute equally to model
- improves the performance of distance based algo
- Prevent dominance of features with large numerical values.

Perform:

- Standardization (Standard Scale)

$$x' = \frac{x - \mu}{\sigma}$$

{ μ = mean, σ = std. }

DA 2/4/25

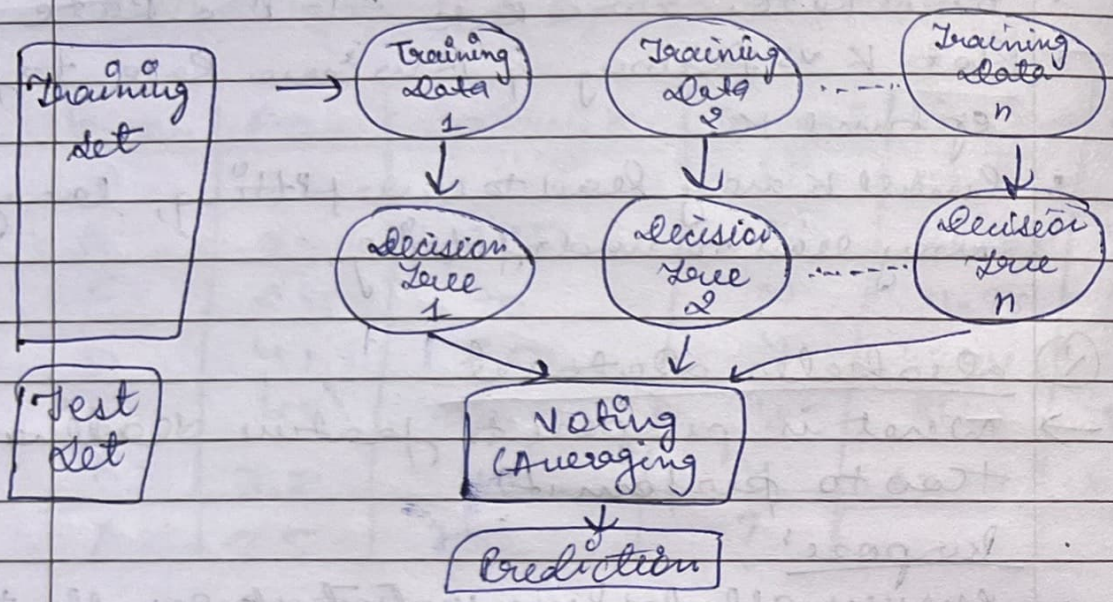
- Normalization (min-max Scaling) (min max Scale)

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Scale data range 0 to 1.

Date 16/4/25
Page 14

Lab 6
Random Forest



→ The following steps are the working of Random Forest Algorithm:

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

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

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

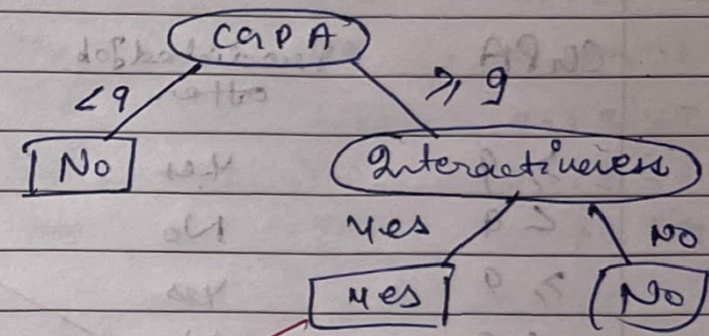
Step 4: Finally, select the ~~most~~ most voted prediction result as the final prediction result.

Date 7/5/25
Page 15

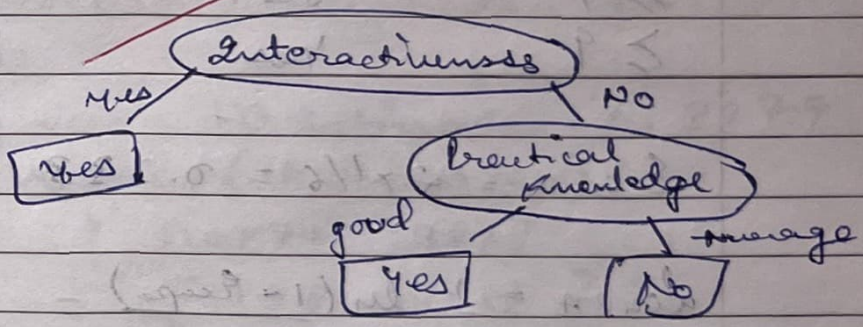
Lab-7

Q Implement random forest ensemble method

For $S_1 \rightarrow$



For $S_2 \rightarrow$



Q Write answers.

(1) what is the best accuracy score & confusion matrix of the classification you observed & using how many trees!

→ Best Accuracy Score Observed = 1.0

→ No. of Trees (n_estimations) = best_n.

→ Confusion Matrix = [[16 0 0] [0 14 0] [0 0 15]]

Date 7/5/25
Page 16

Lab-8

8 Implement Boosting ensemble method

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

$\varepsilon_{equp} = 2 \times 1/6 = 0.333$

$\alpha_{CGPA} = \frac{1}{2} \ln\left(\frac{1-\varepsilon_{equpa}}{\varepsilon_{CGPA}}\right) = \frac{1}{2} \cdot \frac{(1-0.13)}{0.33} = 0.347$

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

$= 0.9428$

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

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

Date / /
Page 17

CGPA	Predicted Job offer	Actual Job offer	weight
≥ 9	Yes	Yes	0.1249
< 9	No	Yes	0.1249
≥ 9	Yes	No	0.2501
< 9	No	No	0.1249
≥ 9	Yes	Yes	0.1249
≥ 9	Yes	Yes	0.1249

Q: Write answer.
1) Best Acc Score & Confusion Matrix ?

→ Accuracy with 10 estimators = 0.8277
confusion Matrix (10 estimators) :

$$\begin{bmatrix} 1072 & 387 \\ 213 & 1408 \end{bmatrix}$$

Best Accuracy : 0.831 with n-estimators = 42

Date 7/5/25
Page 18

Lab = 9

Q Build K means algorithm to cluster

Iteration 1:

Record No.	closest to C1 (1.0, 1.0)	closest to C2 (5.0, 2.0)	Assign to cluster
R_1 (1.0, 1.0)	0.0	7.21	C_1
R_2 (1.5, 2.0)	1.12	6.12	C_2
R_3 (3.0, 4.0)	3.61	3.61	C_1
R_4 (5.0, 7.0)	7.21	0.0	C_2
R_5 (3.5, 5.0)	4.12	2.5	C_2
R_6 (4.5, 5.0)	5.31	2.06	C_2
R_7 (3.5, 4.5)	4.30	2.92	C_2

Cluster 1 {R_1 R_3 R_3} & Cluster 2 {R_4 R_5 R_6 R_7}

Their new centroids are:

$C_1 = (1.0 + 1.5 + 3.0)/3 , (1.0 + 2.0 + 4.0)/3$
$= 5.5/3 = 1.83 , 7.0/3 = 2.33$

$C_2 = (5.0 + 3.5 + 4.5 + 3.5)/4 , (7 + 5 + 5 + 4)/4$
$= 16.5/4 = 4.12 , 21.5/4 = 5.37$

Iteration 2:

Record NO	Close to C_1 (1.83, 2.33)	Close to C_2 (4.12, 5.37)	Assign to cluster
R_1 (1.0, 1.0)	1.57	5.64	C_1
R_2 (1.5, 2.0)	0.47	4.52	C_1
R_3 (3.0, 4.0)	2.12	1.63	C_2
R_4 (5.0, 7.0)	5.57	1.91	C_2
R_5 (3.5, 5.0)	3.16	0.73	C_2
R_6 (4.5, 5.0)	3.58	0.52	C_2
R_7 (3.5, 4.5)	2.63	1.05	C_2

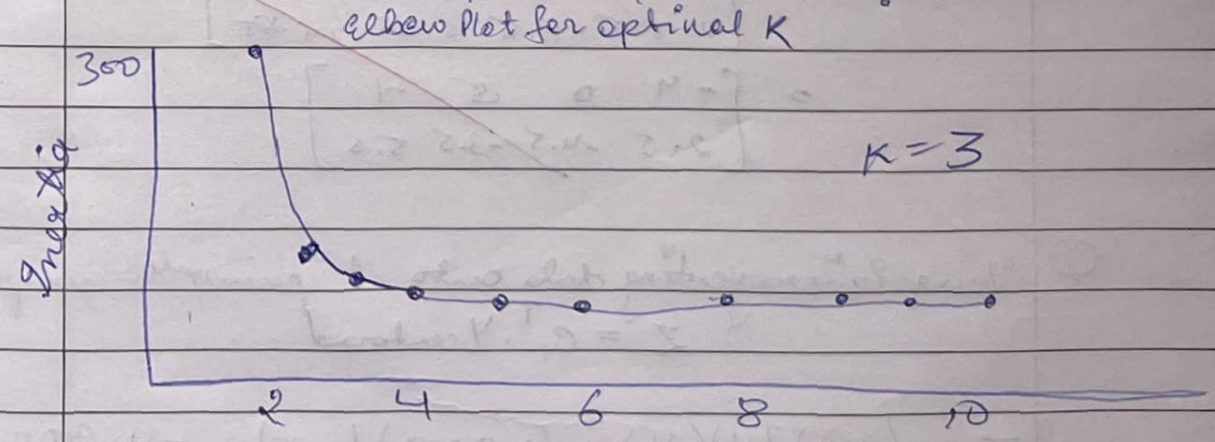
Cluster 1 ($\{R_1, R_2\}$) & cluster 2 $\{R_3, R_4, R_5, R_6, R_7\}$.
their new centroids are -

$C_1 = (1.0+1.5)/2, (1.0+2.0)/2$
$= 2.50/2 = 1.25, 3/2 = 1.5$

$C_2 = (3+5+3.5+4.5+3.5)/5, (4+7+5+5+4.5)/5$
$= 19.5/5 = 3.9, 25.5/5 = 5.1$

Q Write answer:

1) Draw the elbow plot. & K value?



Date 7/5/25
Page 20

Lab -10

Q Implement dimensionality reduction using (PCA) method

feature	Example 1	Example 2	Example 3	Example 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 -
$e_1 = \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}$ $e_2 = \begin{bmatrix} 0.8303 \\ 0.5514 \end{bmatrix}$

A = ① Data matrix = $\begin{bmatrix} 4 & 8 & 13 & 7 \\ 11 & 4 & 5 & 14 \end{bmatrix}$

② Mean center the data =

$$\text{Mean } X_1 = \frac{4+8+13+7}{4} = 8$$

$$\text{Mean } X_2 = \frac{11+4+5+14}{4} = 8.5$$

$Y_{centered} = \begin{bmatrix} 4-8 & 8-8 & 13-8 & 7-8 \\ 11-8.5 & 4-8.5 & 5-8.5 & 14-8.5 \end{bmatrix}$

$= \begin{bmatrix} -4 & 0 & 5 & -1 \\ 2.5 & -4.5 & -3.5 & 5.5 \end{bmatrix}$

③ Using Eq^N projecting data onto 1st principle component -

$$Z = e_1^T \cdot Y_{centered}$$

$Z_1 = (0.5574)(-4) + (-0.8303)(-5) = -4.38535$

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

$Z_3 = (0.5574)(5) + (-0.8303)(-3.7) = 5.69305$

$Z_4 = (0.5574)(-1) + (-0.8303)(5.3) = -5.12405$

$Z = [-4.30535 \quad 3.73635 \quad 5.69305 \quad -5.12405]$

Q Write answers -

1) Report the accuracy of scores before & after PCA

⇒ Model accuracy :

Before :

SVM : 0.8804

Logistic Regression : 0.8533

Random Forest : 0.8859

After :

SVM : 0.8424

Logistic Regression : 0.8641

Random Forest : 0.8533

SA
19-5-2025