05/08

Week-0

TO-DO EXERCISES

(1) Initializing values directly into Data-Frame - Takes names of data with column heading as USN, Name, Marks.

```
import pandas as pd
data = { "USN": ["234", "235", "236", "237",
                 "238"],
         "Name": ["Asha", "Cathy", "Rosa",
                  "Amy", "Gina"],
         "Marks": [85, 90, 78, 92, 88]
       }
df = pd.DataFrame(data)
print(df)
```

df:

	USN	Name	Marks
0	234	Asha	85
1	235	Cathy	90
2	236	Rosa	78
3	237	Amy	92
4	238	Gina	88

(2) Importing Datasets from sklearn.datasets

```
from sklearn.datasets import load_diabetes
import pandas as pd
diabetes = load_diabetes()
df_diabetes = pd.DataFrame(diabetes.data,
              columns = diabetes.feature_names)
print("df - diabetes")
```

	age	sex	bmi	bp	s1	s2	s3	s4	s5	s6
op	0	0.03807	0.50680	0.061696	0.0218	-0.044	-0.034	-0.043	-0.002	0.019
	1	-0.00188	-0.04472	-0.051474						
	2	0.085299	0.050680							
	3									
	4									

(3) Import datasets from a specific .csv files

```
import pandas as pd
df_csv = pd.read_csv("/sample_sales_data.csv")
print(df_csv.head())
```

op

	Product	Quantity	Price	Sales	Region
0	Laptop	5	1000	5000	North
1	Mouse	15	20	300	West
2	Keyboard	10	50	500	East
3	Monitor	8	200	1600	South
4	Laptop	12	950	11400	North

(4) Downloading datasets from existing dataset repositories like kaggle....

```
import pandas as pd
df_kaggle = pd.read_csv("/Dataset of Diabetes.csv")
print(df_kaggle.head())
```

op

	ID	No_Patient	Gender	Age	Urea	Cr
0	502	17975	F	50	4.7	46
1	735	34221	M	26	4.5	62
2	420	47975	F	50	4.7	

	BMI	CLASS
0	24.0	N
1	23.0	N
2	24.0	N
3	24.0	N
4	21.0	N

TO-DO

Stock Market Data Analysis

1. ["HDFCBANK.NS", "ICICIBANK.NS", "KOTAKBANK.NS"]
2. start data and end date
3. Plot the closing price and daily returns for all three banks

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

tickers = ["HDFC.NS", "ICICIBANK.NS",
           "KOTAKBANK.NS"]

data = yf.download(tickers,
    start = "2024-01-01", end =
    "2024-12-30", group_by = "ticker")

print("First 5 rows of the dataset")

print(data.head(5))

print("\n Shape of dataset")

print(data.shape)

print("\n Column names")

print(data.columns)

hdfc_data = data["HDFCBANK.NS"]
```

print ("\n Summary statistics for HDFC Industries")

print (hdfc_data.describe())

hdfc_data['Daily Return'] = hdfc_data['Close'].pct_change()

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

plt.subplot(2,1,1)

hdfc_data['Daily_Return'].plot(title="HDFC Industries Daily Returns", color_range)

plt.tight_layout()

plt.show()

Week-1

To load .csv file into dataframe

```
import pandas as pd
df = pd.read_csv('/content/housing.csv')
```

To display information of all column

```
print (df.columns())
```

	population	households	median income
0	322.0	126.0	322.0
1	2401.0	1138.0	8.304
2	496.0		8.754
3	558.0	219.0	5.6471
4	565.0	219.0	3.8462

To display statistical information of all numerical

```
print (df.info)
```

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

```
print (df.counts ('ocean_proximity'))
```

To display which attributes in a dataset have missing values

```
print (df.isnull())
```

o/p	ocean_proximity
0	NEAR BAY
1	NEAR BAY
2	INLAND

Week-02

103- Using weather dataset

```
import pandas as pd
data = pd.read_csv('tennis.csv')
data
```

	outlook	temp	humidity	windy	play
0	sunny	hot	high	false	no
1	sunny	hot	high	true	no
2	overcast	hot	high	false	yes
3	rainy	mild	high	false	yes
:	:	:	:	:	:

```
from sklearn.preprocessing import
                                LabelEncoder
outlook = LabelEncoder()
temp = LabelEncoder()
humidity = LabelEncoder()
windy = LabelEncoder()
play = LabelEncoder()

data['outlook'] = outlook.fit_transform
                    (data['outlook'])
data['temp'] = outlook.fit_transform
                    (data['temp'])

data['humidity'] = outlook.fit_transform
                    (data['humidity'])
data['windy'] = outlook.fit_transform
                    (data['windy'])
```

data['play'] = outlook.fit_transform(data['play'])

data

	outlook	temp	humidity	windy	play
0	2	1	0	0	0
1	2	1	0	1	0
2	0	1	0	0	1
3	1	2	0	0	1
4	1	0	1	0	1
5	1	0	1	1	0
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

features_cols = ['outlook', 'temp', 'humidity', 'windy']

x = data[features_cols]

y = data.play

print(x)

print()

print(y)

	outlook	temp	humidity	windy
0	2	1	0	0
1	2	1	0	1
2	0	1	0	0
3	1	2	0	0
4	1	0	1	0
5	1	0	1	1
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

0	0
1	0
2	1
3	1
4	1
5	0
...	...
13	0

Name: play, dtype: int64

```
from sklearn.model_selection import
                                train_test_split

x_train, x_test, y_train, y_test =
        train_test_split(x,y,
                         test_size=0.2)

from sklearn.tree import DecisionTree
                                Classifier
classifier= DecisionTreeClassifier
                    (criterion='entropy')

classifier.fit(x_train, y_train)

classifier.predict(x_test)
```

outlook	temp	humidity	windy
9	1	2	1	0
0	2	1	0	0
1	2	1	0	1

y_test

play
1
0
0

classifier.score(x_test, y_test)

1.0

from sklearn import tree
tree.plot_tree(classifier)

clf = DecisionTreeClassifier(criterion = 'entropy')

clf.fit(x,y)

importances = clf.feature_importances_

feature_Importance_df = pd.DataFrame({
    'Feature': x.columns,
    'Information Gain': Importances
})

feature_importance_df = feature_Importance_df.sort_values(by='Information Gain', ascending=False)

print(feature_Importance_df)

	Feature	Information Gain
0	outlook	0.514558
3	windy	0.1279205
2	humidity	0.211237
1	temp	0.000000

19/8/25

Week-03

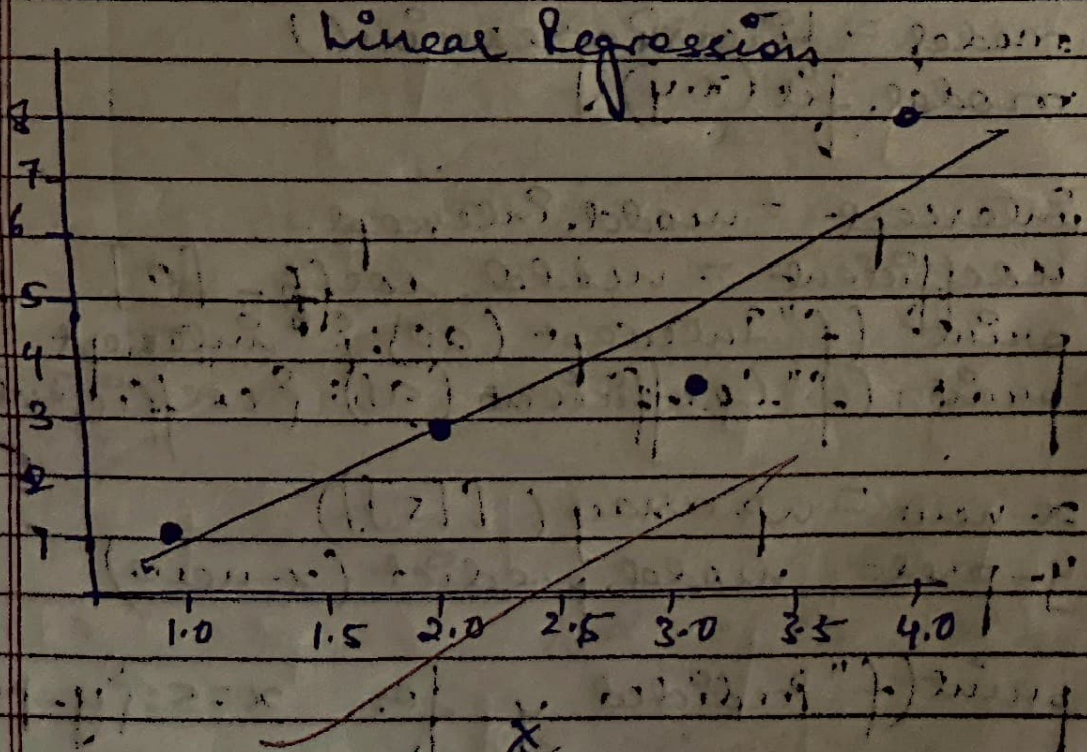
Linear Regression

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import
                        LinearRegression

x = np.array([1,2,3,4]).reshape(-1,1)
y = np.array([1,3,4,8])

model = LinearRegression()
model.fit(x,y)

intercept = model.intercept_
coefficient = model.coef_[0]
print(f"Intercept (a0): {intercept}")
print(f"Coefficient (a1): {coefficient}")

x_new = np.array([[5]])
y_pred = model.predict(x_new)
print(f"Predicted y for x=5: {y_pred[0]}")

plt.scatter(x,y, color='blue', label='Data')

y_line = model.predict(x)

plt.plot(x,y_line, color='red', label=
                        'Regression Line')

plt.xlabel('x')
plt.ylabel('y')
```

plt.title('Linear Regression')
plt.legend()
plt.show()

Output:

Intercept (a0): -1.4999999....1
Coefficient (a1): 2.199999....7
Predicted y for x=5 : 9.5



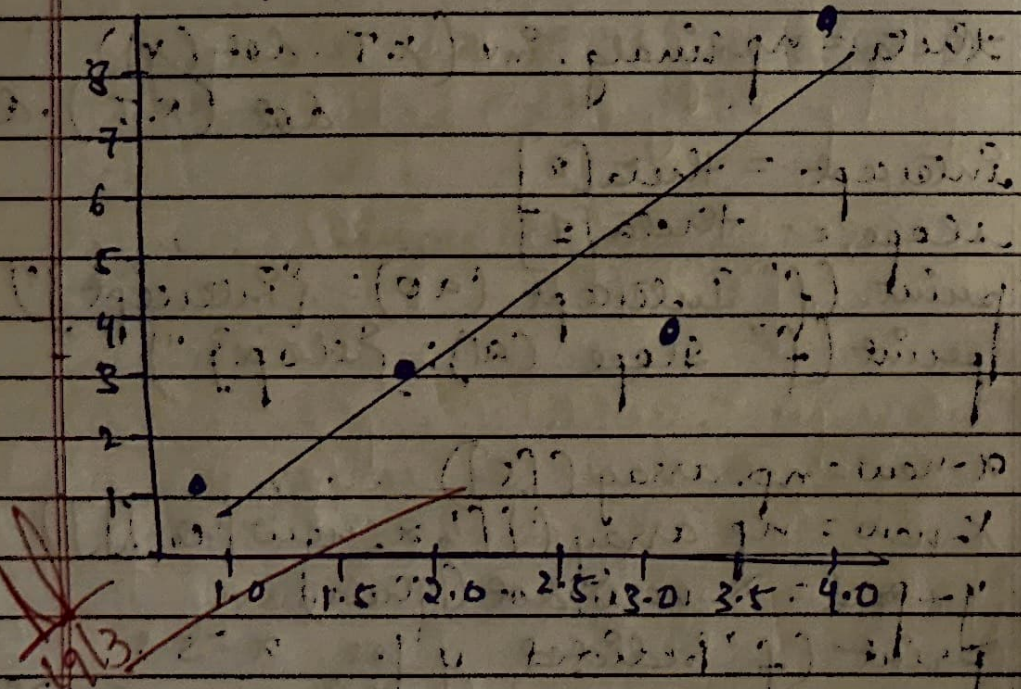
```
import numpy as np
import matplotlib.pyplot as plt

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

x = np.vstack([np.ones(len(x)), x]).T

theta = np.linalg.inv(x.T.dot(x)).
                dot(x.T).dot(y)

Intercept = theta[0]
slope = theta[1]
print(f"Intercept (a0): {Intercept}")
print(f"slope (a1): {slope}")

x_new = np.array([5])
X_new = np.array([[1, x_new[0]]])
y_pred = X_new.dot(theta)
print(f"Predicted y for x=5:
        {y_pred[0]}")

plt.scatter(x[:,1], y, color='blue', label=
                'Data')

y_line = X.dot(theta)
plt.plot(x[:,1], y_line, color='red',
                label='Regression Line')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Different Matrix Method')
plt.legend()

plt.show()
```

Intercept (a0): -1.5

Slope (a1): 2.0000

Predicted y for x=5: 9.5000

Different Matrix Method



2/4/25

Logistic Regression

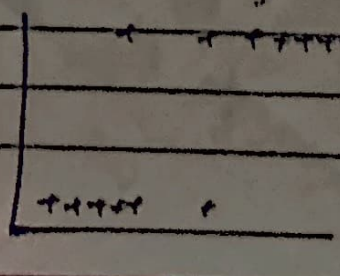
(Insurance dataset)

```
import pandas as pd
from matplotlib import pyplot as plt
df = pd.read_csv("/Insurance_data.csv")
df.head()

plt.scatter(df.age, df.bought_insurance,
            marker='+', color='red')

from sklearn.model_selection import
                        train_test_split
X_train, X_test, y_train, y_test = train_test_
        split(df[['age']],
              df.bought_insurance,
              train_size=0.9, random_
              state=10)
X_train.shape

X_test

from sklearn.linear_model import
                        LogisticRegression
model = LogisticRegression()

model.fit(X_train, y_train)

X_test
y_test
```

```
y_predicted = model.predict(x_test)

y_predicted

model.score(x_test, y_test)

model.predict_proba(x_test)

y_predicted = model.predict([[60]])
y_predicted

model.coef_

model.intercept_

import math
def sigmoid(x):
    return 1/(1+math.exp(-x))

def prediction_function(age):
    z = 0.127 * age - 4.973
    y = sigmoid(z)
    return y

age = 35
prediction_function(age)
```

O/P:

0.3709834



To do:

Q4) 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 $a_0 = -5$ (intercept) and $a_1 = 0.8$ (coeff for study hours)

a) Write the logistic regression eq^n for this.

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

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

2) Consider $z = [2, 1, 0]$ for three classes. Apply SoftMax func^n to find the probability values of three classes.

Soln a) logistic regression equation

$$P(\text{pass}) = \frac{1}{1 + e^{-(a_0 + a_1 x)}}$$

$a_0 = -5, a_1 = 0.8$

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

by x=7

$$P(\text{Pass}) = \frac{1}{1+e^{-(-5+0.8\times7)}}$$

$$= \frac{1}{1+e^{-0.6}} = 0.646$$

∴ probability = 0.646 (64.68%)

c) Predicted class : Pass

2) $z = [2, 1, 0]$

$$P_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Class1 (z=2) = 0.665 (66.52%)

Class2 (z=1) = 0.245 (24.47%)

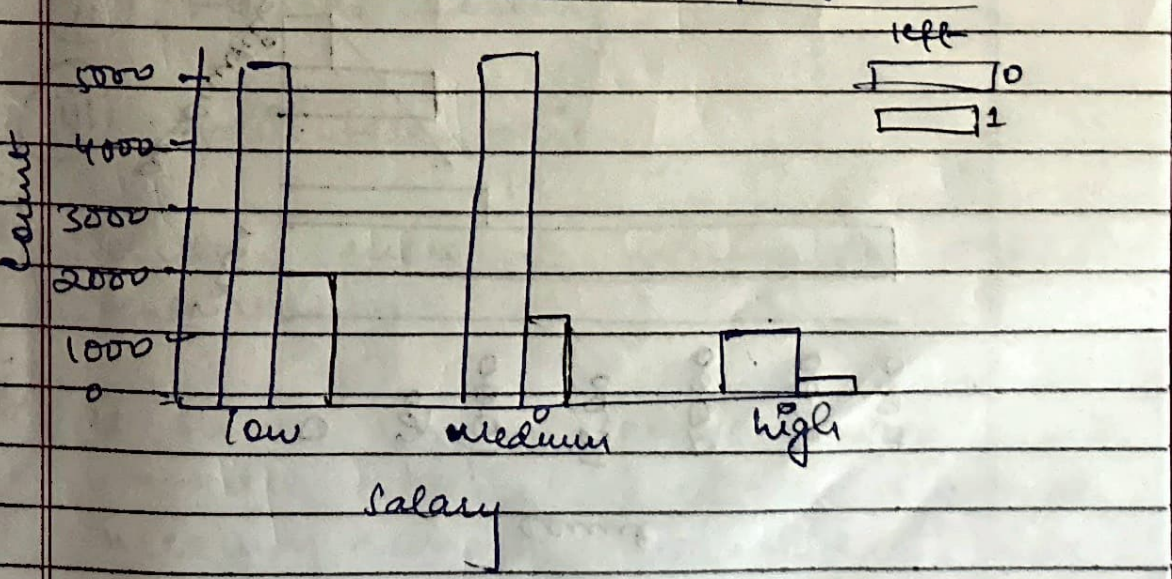
Class3 (z=0) = 0.090 (9.00%)

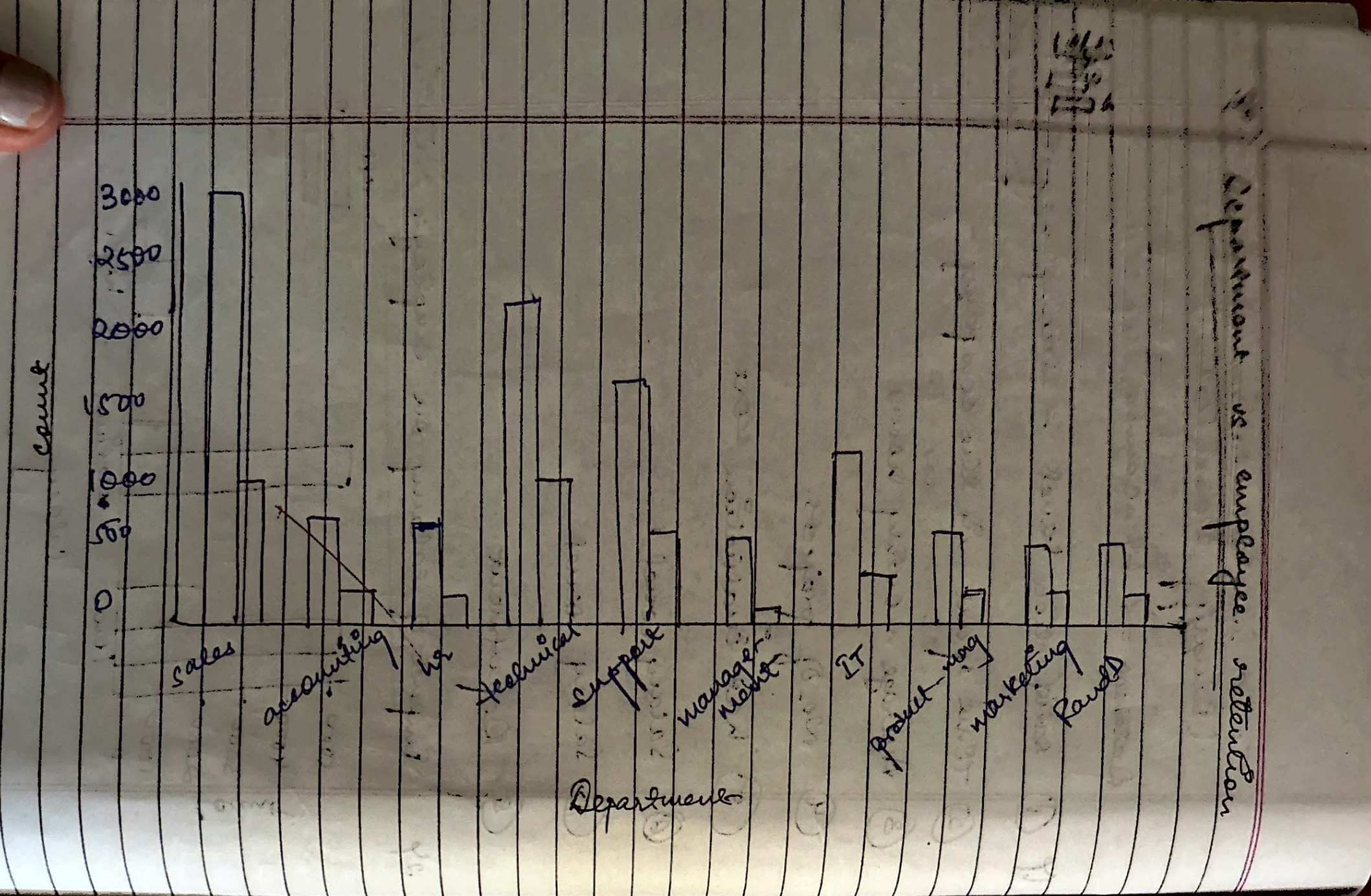
To do: Implementation – Logistic Regression (Binary classification)

Dataset = (HR_comma_sep.csv)

1. satisfaction level – strong negative correlation
2. Time spent in the company –
3. Average monthly hours
4. No_of_projects
5. last evaluation score
6. Salary level
7. Salary level
8. Department

2. Impact of salary on employee retention




Department vs employee retention
count
3000
2500
2000
500
1000
50
0
sales
accounting
hr
technical
support
management
IT
product_mng
marketing
R and D
Department

Model Accuracy = 0.75866.....67

	precision	recall	f1-score	Support
0	0.80	0.92	0.85	2294
1	0.47	0.23	0.31	706

Q) 14 HR_comma_sep.csv :-

(i) employee retention has direct impact on/with satisfaction level

(ii) Accuracy: 0.758
It is good because generally it ranges from 0.75 and 0.85

Q) 27 Zoo dataset :

(i) Data cleaning - removed animal_name column since it was unnecessary

(ii) Feature scaling

(iii) Train_test_split

27 No where were no missing or inconsistent values

37 It shows perfect performance
→ all entries lie on the diagonal, meaning correctly classified into respective classes

4) None. The confusion matrix shows all classes were classified correctly.

KNN

Consider the following dataset for K=3, and test data (X, 35, 100) as (Person, Age, SalaryK) solve using KNN classifier model and predict the target!

Person	Age	SalaryK	Target
A	18	50	N
B	23	55	N
C	24	70	N
D	41	60	Y
E	43	70	Y
F	38	40	Y
X	35	100	?

$$D = \sqrt{(x_{age} - Age)^2 + (x_{salary} - Salary)^2}$$

$A \rightarrow \sqrt{(35-18)^2 + (100-50)^2} = 52.81$

$B \rightarrow \sqrt{(35-23)^2 + (100-55)^2} = 46.56$

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

$D \rightarrow \sqrt{(35-41)^2 + (100-60)^2} = 40.45$

$E \rightarrow \sqrt{(35-43)^2 + (100-70)^2} = 31.05$

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

[C, D, E] → [N, Y, Y]

X(35, 100) → Y

Iris Dataset

→ Here the k value has no impact on the accuracy for k value (1,26) but after that there some fluctuations

K value can be chosen based on hit and trial and based on the size of the dataset.

Diabetes dataset

Purpose -

→ ensures all features contribute equally by bringing them to a similar scale

→ prevents large scale features from dominating distance based algos like KNN

→ Improves convergence speed

How to perform?

1) Standardization (z-score normalization)

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

use StandardScalar() in scikit-learn

2) Normalization (MinMaxScaling)

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

Use MinMaxScaler() in scikit learn.

16/4/25

Decision Tree Algorithm

(1) Input: Dataset D with features F and target variable T

(2)

Random forest Algorithm

(1) Start / Bootstrap sampling

Input: for each tree, randomly select N samples with replacement from the dataset.

(2) Build decision tree:

→ for each tree, use the bootstrap sample to build a decision tree

→ Randomly select k features for each split

(3) Make predictions:

→ Each tree makes a prediction for a new data point

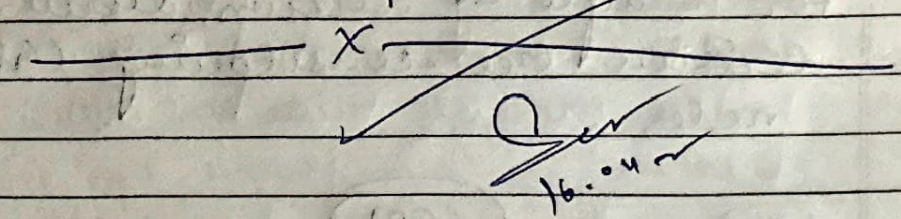
(4) Combine predictions:

→ Classification - Use majority voting to determine the final class

→ Regression - Average the predictions

(5) Out-of-Bag (OOB) Error (Optional):

→ Use out-of-bag samples to estimate the error of the model.

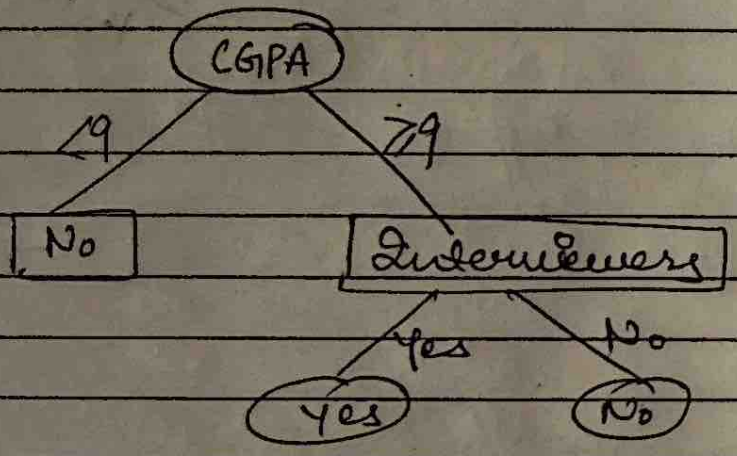


7/5/25

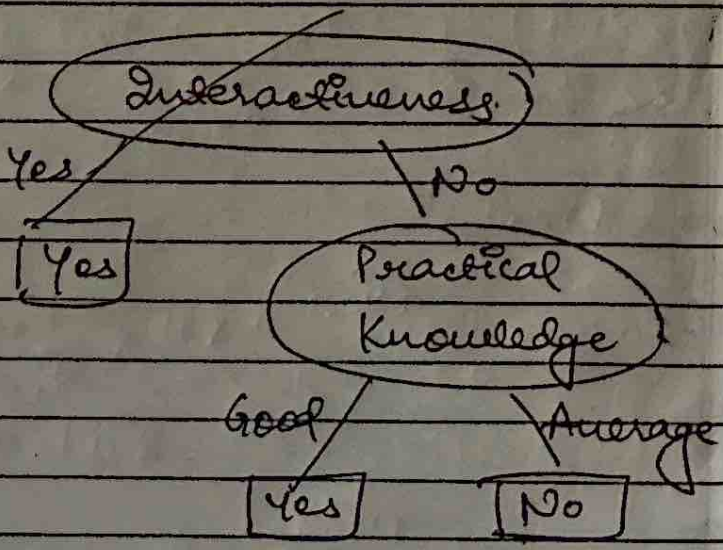
Week-7

① Implement random forest ensemble method on a given dataset.

For sample S1 shown, draw the decision tree considering CGPA as root node.



For sample S2, shown, draw the decision tree considering Interactions as root node



Write the answer for the following question -

(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

⇒ No. of trees (n-estimators)
= best_n

⇒ Confusion Matrix

$$\begin{bmatrix} [16 & 0 & 0] \\ [0 & 14 & 0] \\ [0 & 0 & 15] \end{bmatrix}$$

7/5/25

Lab-8

(8) Implement Boosting ensemble method on a given dataset Considering Ada Boost Algorithm. For the following sample data, show the decision stump calculation steps for the attribute CGPA.

CGPA	Predicted for offer	Actual for 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

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

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

$$= 0.347$$

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

$$= 0.9428$$

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

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

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

Write the answer for the following question-

7) For "income.csv" dataset, what is the best accuracy score and confusion matrix of the classifier you obtained and using how many trees?

⇒ Accuracy with 10 estimates : 0.8277

Confusion matrix (10 estimates):

```
[[10722   387]
 [ 2138  1408]]
```

Best accuracy: 0.831 with n-estimators = 92.

7/5/25

Lab-9

(9) Build K-means algorithm to cluster a set of data stored in a .csv file

Compute two clusters using K-means algorithm for clustering where initial cluster centers are (1.0, 1.0) and (5.0, 7.0). Execute for 2 iterations

→ Iteration 1:

Record Number	Close to C1 (1.0, 1.0)	Close to C2 (5.0, 7.0)	Assign to cluster
R1 (1.0, 1.0)	0.0	7.21	C1
R2 (1.5, 2.0)	1.12	6.12	C2
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

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

Their new centroids are-

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

C2 = (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 C1 (1.83, 2.33)	Close to C2 (4.12, 5.37)	Assign to 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}

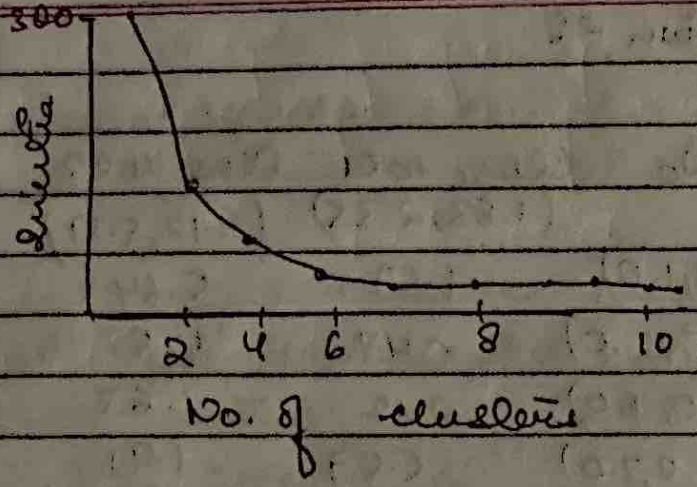
Their new centroids are:

C1 = (1.0 + 1.5)/2 , (1.0 + 2.0)/2
= 2.50/2 = 1.25 , 3/2 = 1.5

C2 = (3 + 5 + 3.5 + 4.5 + 3.5)/5 , (4 + 7 + 5 + 5 + 4.5)/5
= ~~18.5/5~~ = 3.9 , 25.5/5 = 5.1

Q. Write the answers for the following questions:

(1) For "iris.csv" dataset,
Draw the elbow plot. What was the optimal k value obtained?


300
Inertia
2
4
6
8
10
No. of clusters

Lab-10

Q) Implement Dimensionality reduction using Principal Component Analysis (PCA) method.

Given the data in table, reduce the dimensions from 2 to 1 using PCA. Compute the first principal component

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:-

$e_1 = \begin{bmatrix} 0.5594 \\ -0.8303 \end{bmatrix}$ $e_2 = \begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$

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

② Mean centre the data:

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

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

③ 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" projecting data onto first principle component;

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

$$Z_1 = (0.5574)(-4) + (-0.8303)(25) = 4.30535$$

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

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

$$Z_4 = (0.5574)(-4) + (-0.8303)(5.3) = -5.12405$$

$$Z = [-4.30525 \quad 3.73635 \quad 5.69305 \quad 5.12405]$$

Write the answer for the following question:

(V) For "heart.csv" dataset, Report the accuracy scores before and after applying PCA.

⇒ Model accuracy:

Before	After
SVM: 0.8804	= 0.8424
Logistic Regression: 0.8533	= 0.8461
Random forest: 0.8859	= 0.8533

Date
Page