# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



## LAB REPORT on

## **Machine Learning (23CS6PCMAL)**

Submitted by

Anup Vaidya(1BM22CS047)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019

Sep-2024 to Jan-2025

## **B.M.S.** College of Engineering,

**Bull Temple Road, Bangalore 560019** 

(Affiliated To Visvesvaraya Technological University, Belgaum)

### **Department of Computer Science and Engineering**



#### **CERTIFICATE**

This is to certify that the Lab work entitled "Machine Learning (23CS6PCMAL)" carried out by Anup Vaidya(1BM22CS047), who is Bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of a Machine Learning (23CS6PCMAL) work prescribed for the said degree.

Lab Faculty Incharge	
Name: Ms. Saritha A N	Dr. Kavitha Sooda
Assistant Professor	Professor & HOD

Department of CSE, BMSCE

Department of CSE, BMSCE

## Index

Sl. No.	Date	Experiment Title	Page No.
1	21-2-2025	Write a python program to import and export data using Pandas library functions	
2	3-3-2025	Demonstrate various data pre-processing techniques for a given dataset	
3	10-3-2025	-2025 Implement Linear and Multi-Linear Regression algorithm using appropriate dataset	
4	17-3-2025	Build Logistic Regression Model for a given dataset	
5	24-3-2025	Use an appropriate data set for building the decision tree (ID3) and apply this knowledge to classify a new sample	
6	7-4-2025	Build KNN Classification model for a given dataset	
7	21-4-2025	Build Support vector machine model for a given dataset	
8	5-5-2025	Implement Random forest ensemble method on a given dataset	
9	5-5-2025	Implement Boosting ensemble method on a given dataset	
10	12-5-2025	Build k-Means algorithm to cluster a set of data stored in a .CSV file	
11	12-5-2025	Implement Dimensionality reduction using Principal Component Analysis (PCA) method	

Write a python program to import and export data using Panda's library functions

## Screenshot

Date Page
import pandal as pot sulva )
man talks att day Jordan (1)
import pandar as p
df1 = pd. read_csv ('housig - csv')
1 2 1 1 2 1 2 1 2 1 2 1 2 2 2 2 2 2 2 2
Today tolimas ( TAD) while I see it
Index [ longitude, lotitude housing - median-ag,
(total - room 1 'total bedroom ) dtupe
(object)
prins I die column feles ienus () auges)
df1 describe () ward and total soon, lotal-bedroom logagitude latitude housing-media total soon, lotal-bedroom
count 20640 20640 20640 20640 20433
mondo (findada)
cotypical - ai
population households and moone median love
population households median income median-house
"native- country" income ?], other "historial"
, , , ,
ent = len (off 1 · Ocean - proximity · Unique ())
avint (cnt)
print (cnt)
5
print (dti · columns [dti · null () · any ())  Index (l'total - bedrooms 1), oltype = 'object 1)
Index (l'total - bedroom 1) altype = 10bject 1)
dfo = nd > 1 end + 1
df2 = pd. read-usu (Idiabetu · csui)
print (df2. columns [ of f2. isnull(1. any(1])
The CED du - 112 12
Indu ([), doype = 'object')

### Code:

```
import pandas as pd

try:
    df = pd.read_csv('input.csv')
    print("Data imported successfully!\n")
    print(df)

except FileNotFoundError:
    print("The file 'input.csv' was not found.")

df["Processed"] = True

df.to_csv('output.csv', index=False)

print("\nData exported successfully to 'output.csv'.")
```

Demonstrate various data pre-processing techniques for a given dataset

Screenshots:

```
LAB-0
   data = yf. downward ( ticker, start "20
   print ("Daily Return for HDEC in dustries"
   plt. lique (tigsize = (12,61)
    PIL- Subplot (2,111)
   hate data ['close !] plot ( little = " HD FC Indutry ")
     plt, subplot (dil,d)
   hate - wata [ Daily Polon ? ) . Plat ( HHE = HOPE
 pl+, Hight-layoutes
Doily Return for HDFC Industria
2024-01-01
2024 - 01-02
2024 -01-03
2024 -01-03
2024 -01-04
2024_01-05
```

import pandas as pd

import numpy as np

from sklearn.preprocessing import LabelEncoder, StandardScaler, MinMaxScaler

```
data = {
```

'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve', None],

'Age': [25, 30, np.nan, 35, 29, 40],

```
'Department': ['HR', 'IT', 'Finance', 'IT', 'HR', 'Finance'],
  'Salary': [50000, 60000, 58000, 62000, np.nan, 52000]
}
df = pd.DataFrame(data)
print("Original DataFrame:\n", df)
df['Age'].fillna(df['Age'].mean(), inplace=True)
df['Salary'].fillna(df['Salary'].median(), inplace=True)
df['Name'].fillna('Unknown', inplace=True)
le = LabelEncoder()
df['Department_Encoded'] = le.fit_transform(df['Department'])
df.drop_duplicates(inplace=True)
df.rename(columns={'Salary': 'Monthly_Salary'}, inplace=True)
df['Age'] = df['Age'].astype(int)
scaler = MinMaxScaler()
df['Salary_Normalized'] = scaler.fit_transform(df[['Monthly_Salary']])
standard_scaler = StandardScaler()
```

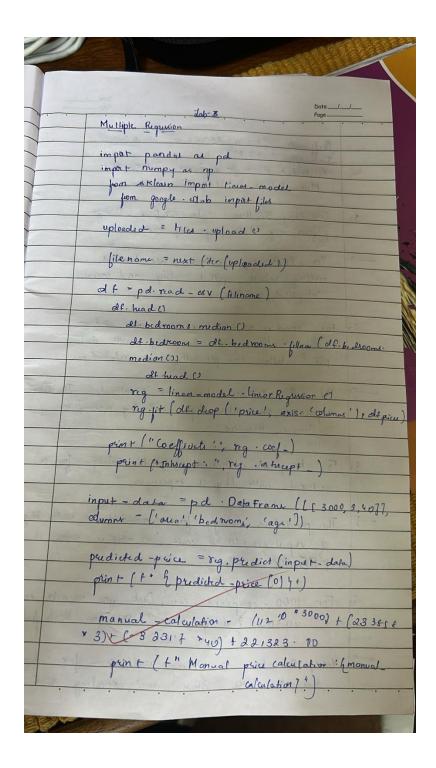
 $df['Age\_Standardized'] = standard\_scaler.fit\_transform(df[['Age']])$ 

 $print("\ \ DataFrame:\ \ \ ",\ df)$ 

## Program 3

Implement Linear and Multi-Linear Regression algorithm using appropriate dataset

Screensho	ts
	Dotte
	dinear Regunion
1	import pandos as pd
	import nump as up
	dinear Regunsion import pandos as pol import num p as up from Sklean linear - model import linear Regunsion
	at ind a confirm
	dt = pd. read ely (1000 companies (44')
	dep-first = true)
	X = ds_encoded drop (Profit ' axis = "columni")
	x = ds-encoded dop (Plaset 'axis="columni") y = ds-encoded [Profit]
	dy predict prof ( Ynd spend, admin spend,
	maskety - Spand, State)!
	mas keting - spund, & tate):  input - data = { col:0 for col in x columns y
-	Inpuidate [] = rada spund
	of the - cot 1 State y state y
	if state - col in input - dofa!
	if state col in input data!  input adata [state col]:
	in put - dt = pd. Data France ([input -data])
	return 1' Predicted Projet \$ f predicted proj
	. 36 3
	- print ( predict-profit (9164 0.48, 1193 1-24,
	(Flouda'))
	Output:
	The state of the s



#### # Linear Regression

import pandas as pd import numpy as np import matplotlib.pyplot as plt from sklearn.datasets import load\_boston from sklearn.linear\_model import LinearRegression from sklearn.model\_selection import train\_test\_split

```
from sklearn.metrics import mean_squared_error, r2_score
```

```
# Load dataset
boston = load_boston()
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df['PRICE'] = boston.target

# Use only one feature for simple linear regression (e.g., RM = average number of rooms)
X = df[['RM']]
y = df['PRICE']

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)

# Train model
lr = LinearRegression()
lr.fit(X_train, y_train)
```

```
# Predict
y_pred = lr.predict(X_test)
print("Linear Regression Results")
print("Coefficients:", lr.coef_)
print("Intercept:", lr.intercept_)
print("MSE:", mean_squared_error(y_test, y_pred))
print("R2 Score:", r2_score(y_test, y_pred))
# Plot
plt.scatter(X_test, y_test, color='blue')
plt.plot(X_test, y_pred, color='red')
plt.xlabel('Average Number of Rooms (RM)')
plt.ylabel('House Price')
plt.title('Simple Linear Regression')
plt.show()
# Multiple Linear Regression
# Use all features
X = df.drop('PRICE', axis=1)
y = df['PRICE']
# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
# Train model
mlr = LinearRegression()
mlr.fit(X_train, y_train)
# Predict
y_pred = mlr.predict(X_test)
# Output
print("\nMultiple Linear Regression Results")
print("Coefficients:", mlr.coef_)
print("Intercept:", mlr.intercept_)
print("MSE:", mean_squared_error(y_test, y_pred))
print("R2 Score:", r2_score(y_test, y_pred))
```

#### Screenshot's

	Lab-5	Dote		
-	Date Page		-	
		- we becoming bateted		
	Logisticis Regression	My wights -= My harring pate & du My bias - My harring pate + db		
		Mil bias - Mil Marring man rab	-	
	import numpy as up		-	-03-
-	CLON DICHAY O -	del presidet coul, v:		KA
-	day interconfiguration of interconstructions of the control of the	4-p- My. sigmoid ( unias motel)		al
	and much carbing-pate = 0.01,	9-p-14-51-151-150-5 USC 0		d
The same of the sa	All learning rate = learning rate	g-p-01=11411030300	-	
	self num-ite num-ite	for iny-p		
	My weights = None			
)+	My bra= None	return np. am ay (4-p-cy)		70
and I am	V NOTA	huwin np. whas cy-r-us	->	2336
any +12	def sigmoist (sub, z):		-	10000
	return 1/(1+np apc-2)		->	N
		ib-пате == " main = ":		0
	def fit (ruf. x, y)			Pn
	n_samples, n. features = x chape	x=np.array([1,2],[2,3],	1	F
	My weights = np. zeros (n-features)	[3,4),(8,6J)	29-10	y
Service Control	All bigles	U- Nh chaul		
0.00		y = np. array ( [0, 0/1.1, 1])		
	400 - in hange (xill , n)?	mmar = 1 and disc floor = 5 co	420	
	linear-model = np. dot (x, mg. myhr)	model = Logistic Regrusione )		
		mose fit (x, y)		
	+ sul bias	The state of the s	2.	SI
	11 mediated = MILL or maid [ None	y-pred = motel · predict(x)		
	y-predicted = My sigmood (linear- model)	/ A Land A MARINER AISA TO A MANAGE M	3.	92
		print ( " predictions: ". y prod.)		-
	dw= (1/n-samples) * np-dot (x.T,	THE PERSON NAMED IN THE PE	4.	10
-	(y-president-y)	autput: [01171]		0
1	db= (1/n-samplus) + np sum(			-
	10- (1712-) and pus) + rep sum(	Lord and Coyno and odb	S.	De
	y-predicted-y)	KIN- Halling - h		1
-				16
-				

#### Code

import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import confusion\_matrix, accuracy\_score, classification\_report

iris = load\_iris()

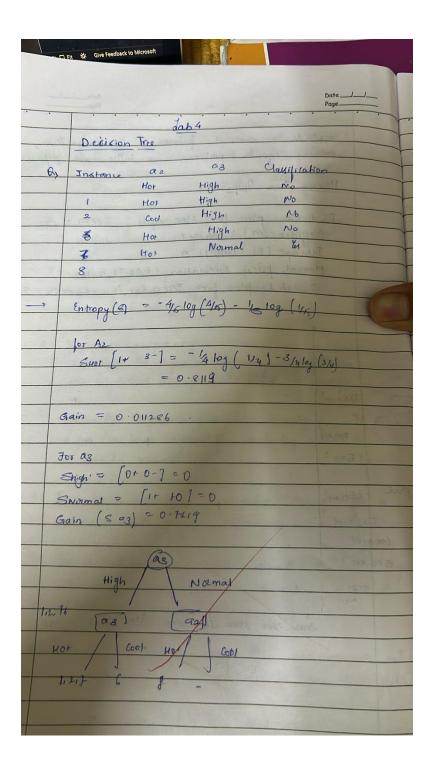
df = pd.DataFrame(iris.data, columns=iris.feature\_names)

df['species'] = iris.target

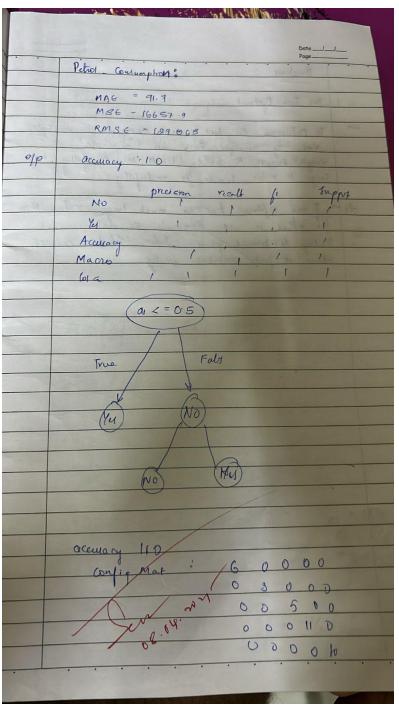
```
df_binary = df[df['species'] != 2] # Remove class 2 (Virginica)
X = df_binary.iloc[:, :-1] # Features
y = df_binary['species']
                         # Target (0 or 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LogisticRegression()
model.fit(X_train, y_train)
# Step 5: Predict and evaluate
y_pred = model.predict(X_test)
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

Use an appropriate data set for building the decision tree (ID3) and apply this knowledge to classify a new sample

## Screenshots



+ 110% D FR # Give Feedback to Microsoft
Date
Code for Dission Treas
import pandas as pel    war extern medel - select import transast   war extern medel - select import transast   sklana true ', Due issuen True Changin   sklana true ', Due issuen True Changin   sklana true ', Due issuen True Changin
dt = pd. read.uv()   x = dt deep ( * speriu*, axis = 1)   y = dt [ speriu*]
X- train - X Hut 14 Ham Y hut
df daujur = Decision Tree (Classified) elf-classifier (x-hair, y-train)
y-prid = dt. day fire predict (x-fist)  accuracy = accuracy, come (y-fist, y-prid)  conf matrix = conf matrix (y-fist, y-prid)
print (" acuracy score ", accuracy)  print (" conf Molin" conf mate)
Int
acu gore :1.0
0 0 b
0 0 1)



Code

import pandas as pd

```
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
data = {
  'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain', 'Overcast',
         'Sunny', 'Sunny', 'Rain', 'Sunny', 'Overcast', 'Overcast', 'Rain'],
  Temperature': ['Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool',
            'Mild', 'Cool', 'Mild', 'Mild', 'Mild', 'Hot', 'Mild'],
  'Humidity': ['High', 'High', 'High', 'Normal', 'Normal', 'Normal',
          'High', 'Normal', 'Normal', 'High', 'Normal', 'High'],
  'Wind': ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong',
       'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Strong'],
  'PlayTennis': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes',
           'No', 'Yes', 'Yes', 'Yes', 'Yes', 'No']
}
df = pd.DataFrame(data)
le = LabelEncoder()
for column in df.columns:
  df[column] = le.fit_transform(df[column])
#Step 3: Separate features and label
X = df.drop('PlayTennis', axis=1)
y = df['PlayTennis']
clf = DecisionTreeClassifier(criterion='entropy') #ID3 uses 'entropy'
```

```
clf = clf.fit(X, y)
print("\nDecision Tree Rules:")
tree_text = tree.export_text(clf, feature_names=X.columns.tolist())
print(tree_text)
#Example: Outlook=Rain, Temperature=Mild, Humidity=High, Wind=Weak
# Encode input sample with same label encoding order used earlier
sample = pd.DataFrame({
  'Outlook': [le.transform(['Rain'])[0]],
  Temperature': [le.transform(['Mild'])[0]],
  'Humidity': [le.transform(['High'])[0]],
  'Wind': [le.transform(['Weak'])[0]]
})
# Predict
prediction = clf.predict(sample)
result = 'Yes' if prediction[0] == 1 else 'No'
print(f"\nPrediction for new sample (Rain, Mild, High, Weak): {result}")
```

Program 6 Build KNN Classification model for a given dataset

	KNN	-	Jab5		Date
di	Person	Age	2		4311
	A	18	Salary k 50	Leight	distance
	В	23	55	N'	50.8
	c	24	70	N	41.67
	D	241	1	N	3.95
	$\epsilon$	43	60	4	40.44
	F	38	70		31.04
	X	85	100	Y	68-07
	(X = 30	5,(00)	1 2 13		2715
	K=3	-13   1	100000	777	
	Di		nk Target		more had to
	31	04 1	Y		284
	31	-95 2	2		2000
	41	0.44 3	Y		Andrew W. C.
		X (35	i (00) target	will bu	. 7
			V	A Chan	
Coo	le!			3 10 17	43.50 20 (1.)
in	port pa	ndal aj	od vor n	J. Berry	TO THE REAL PROPERTY.
he	on st	than mo	del - ule lio	n (n n)	train tel Jph
hu	om ak	ele un	al sur land	, mapa	test Sph
mat		2001 -1	nearics Impor	ricin	ory Scola, confue
Tran	(X				,
d	er = po	1. Yead . y	wis		
x	, = a	2 doct	1. df ["spec	2-1 1	1
					,
X-Tr	ain Xtu	1+ y from	yout = to	oin-Tut	- split (xir,tu

	Data
	Page Page
	Km 1 = KNoghas Classifu (i kurjen = 5)  Km 1 = (x - ton , y train)  y pred = (x napredict (x - tut)  print ("accuracy - State" accuracy sure (y  text , y pred!
	print (" accuracy - Sine accuracy - Sine (4)
	pain ( "conf malie" confusion mater (yeld)
yal-	beine (, Clarit & brito, clan 1)
707	T + Debut
	(0x) met - 10 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	0 10 0 10 0 10 0 10 0 0 0 0 0 0 0 0 0 0
	classification precision real possess hopped
	Atrice
	Validate to the state of the st
	Verginica
	Diabetu Dataset
	decrease age i 0.401
241	acanacy rice: 0.701
	conf mat + 8 Roll
	proficien recelt & Support
7 4	0 0-15 0-30 0-18 100
	1 0.58 2.20 0.86 84

import pandas as pd

 $from\,sklearn.datasets\,import\,load\_iris$ 

 $from \ sklearn.model\_selection \ import \ train\_test\_split$ 

 $from\,sklearn.neighbors\,import\,KNeighborsClassifier$ 

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

#Step 1: Load the Iris dataset

iris = load\_iris()

X = pd.DataFrame(iris.data, columns=iris.feature\_names)

```
y = pd.Series(iris.target)
   # Step 2: Split the data (80% training, 20% testing)
   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
   \# Step 3: Build KNN model (k = 3)
   knn = KNeighborsClassifier(n_neighbors=3)
   knn.fit(X_train, y_train)
   #Step 4: Predict and evaluate
   y_pred = knn.predict(X_test)
   # Results
   print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
   print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

Build Support vector machine model for a given dataset

Screenshots

Jab 7:
Draw an optimal hypuplane using linear sum to  classify points.  {(1,1), (3,1), (1,-1), (3,-1)?} - +ve labelted  {(40), (5,1), (5,1), (6,0)?}ve labelted
-5 -4 -3 -2 -1 0 0 0
$\frac{1}{6!} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \end{bmatrix}$
$\frac{1}{61} = \begin{bmatrix} 2 & \frac{1}{62} \\ \frac{1}{1} & \frac{1}{1} \end{bmatrix} = \begin{bmatrix} 4 \\ \frac{1}{1} & \frac{1}{1} \end{bmatrix}$
α, δι δι + α, δι + α 3 δι δι = +1  α, δι δι δι + α 2 δι βι + α 3 δι δι = +1  α, δι δι δι + α 2 δι βι + α 3 δι δι = -0
$\alpha_{1}(6) + \alpha_{2}(4) + \alpha_{3}(9) = +1$ $\alpha_{1}(4) + \alpha_{2}(9) + \alpha_{3}(9) = +1$ $\alpha_{1}(4) + \alpha_{2}(9) + \alpha_{3}(13) = -1$ $\alpha_{1}(4) + \alpha_{2}(9) + \alpha_{3}(13) = -1$
$\omega = \chi_{1} \frac{1}{2} + \alpha_{2} \frac{1}{2} + \alpha_{3} \frac{1}{3}$ $- \frac{13}{2} \left( \frac{2}{3} \right) + \frac{13}{3} \left( \frac{2}{3} \right) + \frac{7}{2} \left( \frac{4}{3} \right) = \frac{1}{3} \left( \frac{2}{3} \right) + \frac{1}{3} \left( \frac{2}{3} \right$

```
import pandas as pd

from sklearn.datasets import load_iris

from sklearn.model_selection import train_test_split

from sklearn.svm import SVC

from sklearn.metrics import confusion_matrix, classification_report, accuracy_score

# Step 1: Load the Iris dataset

iris = load_iris()

X = pd.DataFrame(iris.data, columns=iris.feature_names)

y = pd.Series(iris.target)

# Step 2: Split the data into train and test sets (80% train, 20% test)
```

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

#Step 3: Build and train the SVM model (linear kernel)

svm_model = SVC(kernel='linear')

svm_model.fit(X_train, y_train)

#Step 4: Predict and evaluate

y_pred = svm_model.predict(X_test)

#Output results

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

print("\nClassification Report:\n", classification_report(y_test, y_pred))

print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

Implement Random Forest ensemble method on a given dataset

Screenshots

		Date
	Lab 8 :-	Page
Parameters	1 Rendom Forus Man	-/10 -
n - cetimate	h no or free t	9
evitamor june	to measure qual	ity of soli
max-dept	Random Fourt Clause of free split min semale	1 1 400
	The state of the s	ag to split set
Je : Train	ing dataset	The Sales
for n trees	1.	
Random	y select samples will	the Toplacement
grow o	y select samples with decision were	- practices
at la	h split chows sul	seek of leafure
spli.	h split chows sul	
	ah no days.	and the
import pana	las as pd	(S 465 LE
from skeld	or ensumble import	Randon Forus
classifu	on ensumble import	Alexan
iris = pa	· rad - ct v ( )	CAVII -
n = iris	[ sys len 1 1	300 12
y = iris	L'sys len 'sup a L'sys len 'sup a L'species 17 Les Ly trois y host tr	and the state of t
2 train y	has by trois y hat he	ain to 1 - 1112
(2, 4)		The state of the s
y-predica	lefer It - of defer	our padict [x hst]
W- med	yout = accuracy	- some (y - ket,
The separate of the separate o		
for h fo	cemory default	top)
	1	( .
=n, random	Randomfo	no (n. utimotou
y- trains)	Random for 8 hore = 42) of	fit ( u-main,
	il prelict ( n-14	
Score =	cludy: 8cre (scre)	yokah; yaprad)
Print	(8 cire).	
ANGERS AND STATE OF THE STATE O		

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
#Step 1: Load Iris dataset
iris = load_iris()
X = pd.DataFrame(iris.data, columns=iris.feature_names)
y = pd.Series(iris.target)
```

```
#Step 2: Split into train and test sets (80% train, 20% test)

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

#Step 3: Train Random Forest model

rf = RandomForestClassifier(n_estimators=100, random_state=42) #100 trees

rf.fit(X_train, y_train)

#Step 4: Predict and evaluate

y_pred = rf.predict(X_test)

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

print("\nClassification Report:\n", classification_report(y_test, y_pred))

print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

Implement Boosting ensemble method on a given dataset

Screenshots

and the		
		Date Page
·	Lab 9	9
	Darthy '-	a
	Compine Multiple weak learnes	8
	heart a strong learner	
	by mainly models against	
	by premient one.	I
	The state of the s	
85	Paro Algorithm	
12	Start with equal with for all train	ing
1	8 amply	1
25	train a weak model.	
33	cot ever y update sample ut	16
2	Add weak model to ensemble	
-	with out based on it accuracy	9
12	Repeat of nextimage	
1	Final pudiction	
	Market Company	

import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.model\_selection import train\_test\_split

 $from\,sklearn.ensemble\,import\,AdaBoostClassifier$ 

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

 $from\,sklearn.tree\,import\,DecisionTreeClassifier$ 

#Step 1: Load the Iris dataset

iris = load\_iris()

```
X = pd.DataFrame(iris.data, columns=iris.feature_names)
y = pd.Series(iris.target)
# Step 2: Split into train and test sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Step 3: Build AdaBoost model with DecisionTreeClassifier as base estimator
base_estimator = DecisionTreeClassifier(max_depth=1)
model = AdaBoostClassifier(base_estimator=base_estimator, n_estimators=50, learning_rate=1.0,
random_state=42)
model.fit(X_train, y_train)
# Step 5: Predict and evaluate
y_pred = model.predict(X_test)
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

Build k-Means algorithm to cluster a set of data stored in a .CSV file

Screenshots

A State of the sta
1000
Date Poge
dab-10 KMeans
11 Write K-means algo
13 charce no of chustus K
3. Assign each point to make t centroid.
4. Recompuse controids of the mean of assigned boilt
So Repeat step 3-4 until central stop
changiy.
In How to delimine not churches
Sinous nethod
SSE SEE Aiff value of K & Calculate
* Plot SSG VK
3 Parameters of K-Means
- Clushy i no of elich his act in the
in method n-int = no of times algo
max ikx max ilyation , my 111.
Mare Corregence.
Ser Man

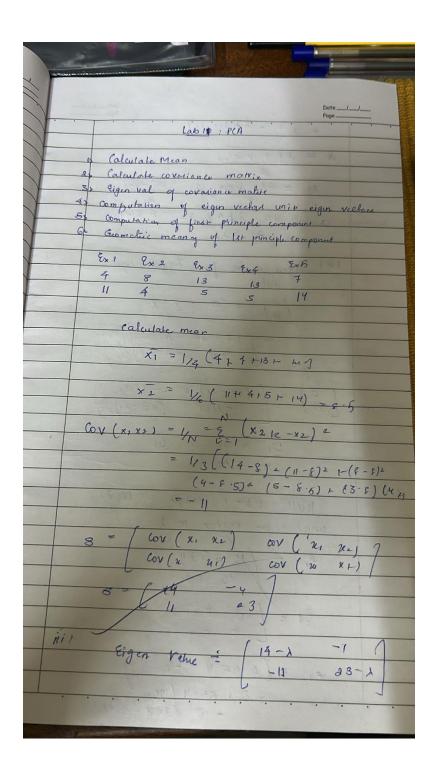
```
import pandas as pd
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

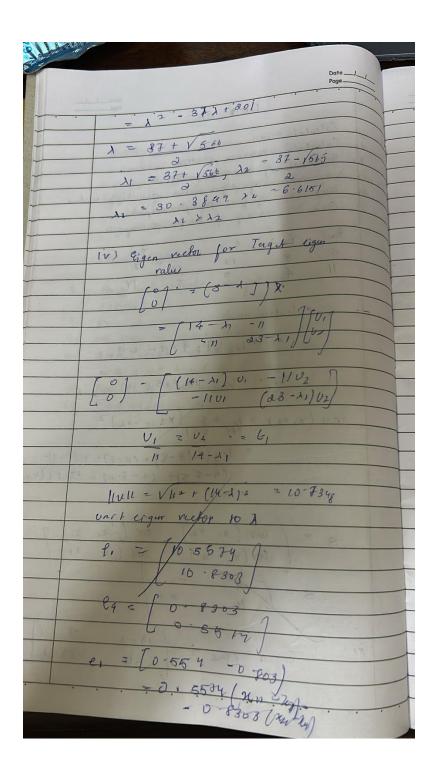
# Step 1: Load dataset from CSV
df = pd.read_csv('your_dataset.csv') # Replace with your file path

# Optional: View first few rows
print("Data Preview:\n", df.head())
```

```
# Step 2: Select relevant numeric columns for clustering
# You can specify specific columns like: df[['column1', 'column2']]
X = df.select_dtypes(include='number')
#Step 3: Scale the data (important for K-Means)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Step 4: Apply K-Means clustering (e.g., 3 clusters)
kmeans = KMeans(n_clusters=3, random_state=42)
df['Cluster'] = kmeans.fit_predict(X_scaled)
#Step 5: Print cluster centers
print("Cluster Centers:\n", kmeans.cluster_centers_)
#Optional Step 6: Visualize (works well for 2D or PCA-reduced data)
if X.shape[1] >= 2:
  plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=df['Cluster'], cmap='viridis')
  plt.title("K-Means Clustering")
  plt.xlabel("Feature 1")
  plt.ylabel("Feature 2")
  plt.show(
```

<u>Program 11</u>
Implement Dimensionality reduction using Principal Component Analysis (PCA) method
Screenshots





import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA import matplotlib.pyplot as plt

# Step 1: Load the Iris dataset

iris = load\_iris()

X = iris.data

```
y = iris.target
   feature names = iris.feature names
   # Step 2: Standardize the data
   scaler = StandardScaler()
   X_scaled = scaler.fit_transform(X)
   # Step 3: Apply PCA (reduce to 2 components for visualization)
   pca = PCA(n_components=2)
   X_pca = pca.fit_transform(X_scaled)
   # Step 4: Plot the 2D PCA result
   plt.figure(figsize=(8, 6))
   scatter = plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y, cmap='viridis', edgecolor='k', s=60)
   plt.xlabel("Principal Component 1")
   plt.ylabel("Principal Component 2")
   plt.title("PCA - Iris Dataset")
   plt.legend(handles=scatter.legend_elements()[0], labels=iris.target_names)
   plt.grid(True)
   plt.show()
   # Explained variance
print("Explained variance ratio:", pca.explained_variance_ratio_)
```