VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



LAB REPORT on

Machine Learning (23CS6PCMAL)

Submitted by

Bhanu Prakash M (1BM22CS067)

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



This is to certify that the Lab work entitled "Machine Learning (23CS6PCMAL)" carried out by Bhanu Prakash M(1BM22CS067), 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 an Machine Learning (23CS6PCMAL) work prescribed for the said degree.

Lab faculty Incharge Name: Megha J

Assistant Professor

Department of CSE, BMSCE

Dr. Kavitha Sooda Professor & HOD

Department of CSE, BMSCE

Index

Sl. No.	Date	Experiment Title	Page No.
1	4-3-2025	Write a python program to import and export data using Pandas library functions	1 - 4
2	11-3-2025	Demonstrate various data pre-processing techniques for a given dataset	5 – 8
3	18-3-2025	Use an appropriate data set for building the decision tree (ID3) and apply this knowledge to classify a new sample.	9 – 12
4	1-4-2025	Implement Linear and Multi-Linear Regression algorithm using appropriate dataset	13 – 16
5	8-4-2025	Build Logistic Regression Model for a given dataset	17 - 20
6	15-4-2025	Build KNN Classification model for a given dataset.	21 – 24
7	15-4-2025	Build Support vector machine model for a given dataset	25 – 28
8	22-4-2025	Implement Random forest ensemble method on a given dataset.	29 – 32
9	22-4-2025	Implement Boosting ensemble method on a given dataset.	33 – 35
10	29-4-2025	Build k-Means algorithm to cluster a set of data stored in a .CSV file.	36 – 38
11	29-4-2025	Implement Dimensionality reduction using Principal Component Analysis (PCA) method.	39 - 40

Github Link: https://github.com/bhanu87777/MachineLearningLab.git

Program 1: Write a python program to import and export data using Pandas library functions

	Porte Trape
	Mochine descring
0	Gedit Costs Good deliction:
-	No of Rouse, 5 8th 800
-1-1-	No 2 Columns 31
	PLU-20 Lower Course Columbia
~	Informal The delexit is likely highly imbalanced with
	your few pardulant konsochone composed to non-
	Informed The delect is likely nightly imbolanced on the reduce few frow delect Konsachone compared to non-frowdent ones. Unful for frood defection among anomaly detection and clarification models
	dekehon and chrispiation models
	Diabeles Predictions
	No of ROWK! 770
	No of (down: 9
-)	Attitude flurous Age, Insulin, BMI
-	person having diabeta using using logiche negleurun,
	porton having diabeta using using logiche negleuran,
	decripentus of neural network
3	Indian diver palmed delated
-	Mod fow' 584
->	I I zamulo) to ON
-1	Att. be : And I I sold sile Aubert
->	Informe: Vice to clary cotron tasks to product lives.
	depose The and crown the cash the
	Chiase in mass correspond capture in moon
	nelaborations in the dataset. If he is alose to I and
	MSE is loss, the model is a good fit. Otherwise, more
	hateres or a non-linear model may be needed.

0	Mushoom Classificohoni
	Not low 8125
DESCRIPTION OF THE PARTY OF THE	Ho of Columns 23
	Alkibiks: Clark Cap-shape cap-color odas
	Infance can be used for decrean the board doughalon
	models teatures emoding is required byon applying
	mortine towning models
	A the second second by the second second
6	Spar dekchan!
-	No of POWN 5573
-	No of blong 12
	Attention VI, U2
	Interesse: could not be loaded due to our erroding were
	needs the preceding proper for the original designation
	needs the mocessing be see further oriotyce
1 6 1 1	Market and the second s
	the state of the state of the state of the
	Bridge Grown in List against the little
	The country of the second seco
	The second secon
	all lands to the trade of the state of the s
200	
	THE RESIDENCE OF THE PARTY OF T

```
import pandas as pd
# Import
iris_df = pd.read_csv('Iris.csv')
print("First 5 rows of the Iris dataset:")
print(iris_df.head())

# Export
output_path = "iris_exported.csv"
iris_df.to_csv(output_path, index=False)
print(f"\nIris dataset has been exported successfully to '{output_path}'")

df = pd.read_csv('iris_exported.csv')
df.head()
```

Program 2: Demonstrate various data pre-processing techniques for a given dataset

1000	Program — V	
	And to And Machine Recogning Project	
	X18 0 X10	
0	Prepare the duta	144
	Select a Model and Train	
9	Visualize	
4	Output.	
2	intext buggs or by	
29	Intest could as ut	
	import malphotlib puplot as pt	
	most gration as sing	
aut	from frontes plothing import scotter making	
	from sklesour model seletion import train text split	
Att 187 ag	cross-val-score, Gridsewah, Randomizahon CV	
	from skleaun encemble import Rombin Focut Claimfron,	
	Kohng clampia	
	from cklean ppeline import Aprine	
-	from skhown metrics import document - scale, confusion matter	
0	Logd Dota Commence .	
100	file_polin - "dialistics coi"	
	of = by read_cro(file_poll)	
130	the second of the second second second second second	
a	Histogram	
	of his (figure - (12,10), bins - 20, edge rolox - "Hork")	
	all charton as an american service of the service o	
	Fix P 27 This ideas - fine	
(3)	Data Cleaning	
	col s-to importe - E'Glowse", "Road Prussie", "Kin Thickness",	
	"Incolin", "ENI")	
	df scds to impole] - If scols to impole] replace to no ron)	
	3 floods to impoke I - officels to impoke filling Cofficels to impoke I.	
AND THE	(Inchem	

1	on forein . 4+ grab columns . (, Ontraus,), columns
01	om-pipeline - Pipeline Cl
	(, whope, combe Introva (except dd ., magion))
2)	The state of the s
- Jul	11 piplins. Column Trongermon (
	Crom, non pipeline nom featous)
	The state of the s
	odel Selection and Training
- 6	odek. <
	"Randon Fered": Parobintared Charifica (n-ahmobu -100)
	. Wagnest bookput. : O ragnest Bockpuck Comilian Co - Expussion - Do
100	Jeaning rate -01, random that -42)
1	The state of the s
1	and the state of the same saley make made
Ser.	name model in models items():
	scales - craw values (model, y-koin, y-koin, cv-5,
	Mind of scounts - wood Accorded Freda wood : AD.)
	and the state of t
3 Report	sporomeles Turing (And Sport and Randomized Smarch)
Soros	0-414.
	"1 ak motor": [50, 100, 200)
	Max depth" [None, 10, 20]
	min-samples glit: [2, 5, 10]
1	min-samples look: [1, 2,4]
	The same of the sa
	To the state of th
A CALL	A PROPERTY AND A STATE OF THE PARTY OF THE P

	Page
7	Evaluate Model Informance
(8)	and Cratical Model A revious accuracy services
	hull - In (Bank when Belog : No, clarifocaper relay (a fat
	Will w (xeenthouses wellow d-bring)
	the state of the same was a superior and a superior
0	Oudput
	Bordon Fered & Mean Anomay : 07720
	Gradual Poschry- Man According 07617
	Final Model According : 0 759740
Vale	111 m 1.89 % 11 10 1066 0 1 221 10
4	Diaheles hedu han
	270 Pows 9 columns
	Attribute Pregnator Hurace, Blood Prusine Star Thickney
	trading KMI , Diaheta Age Outcome
914	6,313126
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
-	Total State of the
-	The second secon
	AND AND ADDRESS OF THE PARTY OF
10000	A STATE OF THE PARTY OF THE PAR
7074	or committee to be a few and the second of t
	and the second s
	Commence of the second

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model selection import train test split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OrdinalEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from scipy import stats
# Diabetes Dataset
df = pd.read csv('diabetes.csv')
# Loading and Inspecting the Dataset
df.head()
df.shape
print(df.info())
print(df.describe())
# Checking for Missing Values
missing values = df.isnull().sum()
print(missing values[missing values > 0])
# Identifying and Encoding Categorical Columns
categorical cols = df.select dtypes(include=['object']).columns
print("Categorical columns identified:", categorical cols)
if len(categorical cols) > 0:
  df = pd.get dummies(df, columns=categorical cols, drop first=True)
  print("\nDataFrame after one-hot encoding:")
  print(df.head())
else:
  print("\nNo categorical columns found in the dataset.")
# Feature Scaling
numerical cols = df.select dtypes(include=['number']).columns
# a. Min-Max Scaling
scaler = MinMaxScaler()
df minmax = df.copy()
df minmax[numerical cols] = scaler.fit transform(df[numerical cols])
# b. Standardization (Z-score Scaling)
```

```
scaler = StandardScaler()
df standard = df.copy()
df standard[numerical cols] = scaler.fit transform(df[numerical cols])
# Printing Scaled Data
print("\nDataFrame after Min-Max Scaling:")
print(df minmax.head())
print("\nDataFrame after Standardization:")
print(df standard.head())
# Adult Income Dataset
df1 = pd.read csv('adult.csv')
# Loading and Inspecting the Dataset
df1.head()
df1.shape
print(dfl.info())
print(dfl.describe())
# Missing Value Detection
missing values = df1.isnull().sum()
print(missing values[missing values > 0])
# Detecting and Encoding Categorical Columns
categorical cols = df1.select dtypes(include=['object']).columns
print("Categorical columns identified:", categorical cols)
if len(categorical cols) > 0:
  df1 = pd.get dummies(df1, columns=categorical cols, drop first=True)
  print("\nDataFrame after one-hot encoding:")
  print(df.head())
else:
  print("\nNo categorical columns found in the dataset.")
# Scaling Numerical Features
numerical cols = df1.select dtypes(include=['number']).columns
# Feature Scaling
numerical cols = df.select dtypes(include=['number']).columns
# a. Min-Max Scaling
scaler = MinMaxScaler()
df minmax = df.copy()
```

```
df_minmax[numerical_cols] = scaler.fit_transform(df[numerical_cols])
# b. Standardization (Z-score Scaling)
scaler = StandardScaler()
df_standard = df.copy()
df_standard[numerical_cols] = scaler.fit_transform(df[numerical_cols])
# Printing Scaled Data
print("\nDataFrame after Min-Max Scaling:")
print(df_minmax.head())

print("\nDataFrame after Standardization:")
print(df_standard.head())
```

Program 3: Use an appropriate data set for building the decision tree (ID3) and apply this knowledge to classify a new sample.

24-3-	Decision Tree- IDS Clarification problem
	101.17-1 1 AL +
200	Roux B 584
	Columns & II
	Atkhola: Age, Gender, Total Riling, Alkaline physphote, Totals
	THE WASTER WASTER THAT BE THE MINE OF THE PARTY OF THE PA
٨	Regin Regin
	TRY
	LOAD the CSV file into addrest
	Kno delaset mensage and show the first few nous of
	CATCH any ones
	Display an eller message and stop execution
-	1) hard martin evicini
	SET target attende to the last column of the balance
	DISPLAY the target atkilote
	1240 Simon
	Define Fundion to comple entropy:
	Inhalpe enkapy-value 60
	to each unique day value in the target afterbute
8 1	Controller the debtall of the large actionic
	Colculate the probability of the days value in the
	opdote entropy value using subtracting Return entropy value.
	aprox entropy salw using subtracting
	Keturn entropy value.

1000	Poge — U
	Define function to compute intermation gain for a given attribute
1	calculate total entrary using a entire dollaret
The state of	To habe expanded entrary to 0
	to such unone value in the attribute:
+	select the subject of the declared when the
	select the subset of the dologet where the attribute
There	equals that value entropy
1	ADD ruseight + subject) to weighted entrupy
4	Return Clotal enterpy-weighted enterpy)
1	nut node to the attribute with the movinum information gain
	Display the selected next node
	solected teature to a specific tratain
1	IF the selected feature is not present in the dateset THEN
	SET selected feature to the previoly detarmined suit-rido
a look	LNO
Out pul t	Target attentite for clampication: Detailet Calculating Information gain to each attentite!
1	Infamation boin to Age: 0.18411
	Information Gender Dur
	Total Bilindain : 0 1529
	aned Bhaubia: 0 1301
	Alkalime phophate: 0.4071
	Albumin: 0.0767
1	Boot Node sleeted boused on highest information gain.
P	Al Ralline_phosphate
11	Decision Thee (Neph.I)
7	Alkaline_Phucphale <= 211.5
	enterpy . 090 enterpy . 0.864 enterpy . 0.606
	samples - 300 K samples - 583 + samples - 293
	dans. I dans dans. I .

```
import pandas as pd
from sklearn.model selection import train test split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy score, confusion matrix, classification report
import matplotlib.pyplot as plt
from sklearn.tree import plot tree
import numpy as np
iris = pd.read csv('Iris.csv')
X = iris.iloc[:, 1:-1]
y = iris.iloc[:, -1]
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
clf iris = DecisionTreeClassifier(criterion='entropy', random state=42)
clf iris.fit(X train, y train)
y pred iris = clf iris.predict(X test)
accuracy iris = accuracy score(y test, y pred iris)
conf matrix iris = confusion matrix(y test, y pred iris)
plt.figure(figsize=(12, 8))
plot tree(clf iris, filled=True, feature names=X.columns, class names=clf iris.classes )
plt.title("Decision Tree for IRIS Dataset")
plt.show()
# New Class
new sample = pd.DataFrame([[5.1, 3.5, 1.4, 0.2]], columns=X.columns)
predicted class = clf iris.predict(new sample)
print("Predicted class for the new sample:", predicted class[0])
```

Program 4: Implement Linear and Multi-Linear Regression algorithm using appropriate dataset

	Dore Page
	Linear Regionson in Machine Loanning
1	Il is a states head method used to model the
	1 1 1 1 hal son a december market out of the
1	more independent yaxianic. If plantide vallagine militari
1	for prediction and data analysis
	This code similates a real-world housing press dataset
4	house price as the independent variable and
4	predick house pich based on the size of the have
10	
4 60	what would as wh
1	import mod plottle poppet as pit
	Rum selegus model selection import to an test split
4	from sklean incor-noted import Lincontequesion from sklean mekius import moun-squared over no scar
	from stream metrics into mon - attack - over, 112200
	p. random seed (42)
4	square feet = 500 + (2500-500)" np. random rand (100,1)
	Trice = 50000 +200 " square feet + np. Nandown. sand n(100, 1)
	X-train, X-fost y-train, y-fost - train-teal-split (requese-freet,
	mu tot six = 02 hordom_stak-42)
	model - Lincon Regulation O
-	model (it CX-kain, y-train)
	y-pred = model, preduct (x-lext)

	mic - mion squand error (y-test, y-pred)
	print(f. wow sommed xxiox; (met.)
	pt scates (x test, y test, color. blue, label. Actual Prices) pt plot (x test, y ped, color. xed, linewidth. 2; label. Regression dine)
	pt. x lahel ('Square Tref') pt. y lahel ('House Bira')
	pt. Little ('House and Prediction vince Kinson Continuer')
	by you?
Outpd)	Pron Squard & Max : 65369951.3716 P-ayroud Scar: 0.9957338020
	House Price Production wing Linear Regions
b.	100000 - Regission Line
HOWAC DAG	30000
	100000
	700 750 1000 1250 1500 1750
	Square Test (18)
	1013/28

```
import pandas as pd
import numpy as np
from sklearn.linear model import LinearRegression
import matplotlib.pyplot as plt
# --- Linear Regression (Single Feature) ---
df1 = pd.read csv('Housing.csv') # columns: area, price
X1 = df1[['area']] # feature
y1 = df1['price'] # target
model1 = LinearRegression()
model1.fit(X1, y1)
# Predict for area = 3300 and 5000
pred 3300 = model1.predict(pd.DataFrame(\{'area':[3300]\}))[0]
pred 5000 = model1.predict(pd.DataFrame(\{'area': [5000]\}))[0]
# Plot for Linear Regression
plt.figure(figsize=(8, 6))
plt.scatter(X1, y1, color='blue', marker='o', label='Data Points')
plt.plot(X1, model1.predict(X1), color='red', label='Best Fit Line')
plt.xlabel('Area (sq ft)')
plt.ylabel('Price')
plt.title('Linear Regression (Area vs Price)')
plt.legend()
plt.grid(True)
plt.savefig('linear regression plot.png') # Save plot as image
plt.show()
# --- Multiple Linear Regression ---
df2 = pd.read csv('Housing.csv') # columns: area, bedrooms, age, price
# Fill missing values in bedrooms if any
df2['bedrooms'] = df2['bedrooms'].fillna(df2['bedrooms'].median())
X2 = df2[['area', 'bedrooms']] # features
y2 = df2['price']
model2 = LinearRegression()
```

```
model2.fit(X2, y2)
# Predict for multiple features
multi pred = model2.predict(pd.DataFrame({'area':[3000], 'bedrooms':[3]}))[0]
# Plot for Multiple Linear Regression:
# Here we visualize predicted price for different bedroom counts with fixed area
bedroom vals = np.arange(int(df2.bedrooms.min()), int(df2.bedrooms.max())+1)
pred prices = model2.predict(pd.DataFrame({'area':[3000]*len(bedroom vals),
'bedrooms':bedroom vals}))
plt.figure(figsize=(8, 6))
plt.plot(bedroom vals, pred prices, marker='o', linestyle='-', color='green')
plt.xlabel('Number of Bedrooms')
plt.ylabel('Predicted Price')
plt.title('Multiple Linear Regression\n(Predicted Price vs Bedrooms for area=3000 sq ft)')
plt.grid(True)
plt.savefig('multiple linear regression plot.png') # Save plot as image
plt.show()
# Print prediction results for clarity
print(f"Linear Regression Predictions:")
print(f"Price for 3300 sq ft: ₹{pred 3300:,.0f}")
print(f"Price for 5000 sq ft: ₹{pred 5000:,.0f}\n")
print(f"Multiple Linear Regression Prediction:")
print(f"Price for 3000 sq ft and 3 bedrooms: ₹{multi_pred:,.0f}")
```

Program 5: Build Logistic Regression Model for a given dataset

21-4-25	dogistic legiterian	
	Trianic Dalosels	-
	no of nows 1 12	
	10 of cole 892	100
	Authorian Camenger, Survival Polon, Noon Sex, Age	1
	lock ton	1
0	Import Required Lithores	1000
9	dood the Dalord	- Sun
6	- Istanic cou	1
	Solved Kebrand features and Target Hardle- Missing Values	4
	- Fill miking Age	MAN
		1000
5	L'mode Cutagorial Smalle	- Marie
6	Propose foatour and datals	- Const
9	Split in Dabid.	
-	1 Ton a de la Decembra Mall	
9	Initialize and main dogs he Register Model	-
6	Make hadrehow on Test 20to	
		-
(0)	trolliate the Model	-
	Maghi	
TATE STATE	Accuracy 0 8100 55	-
		-

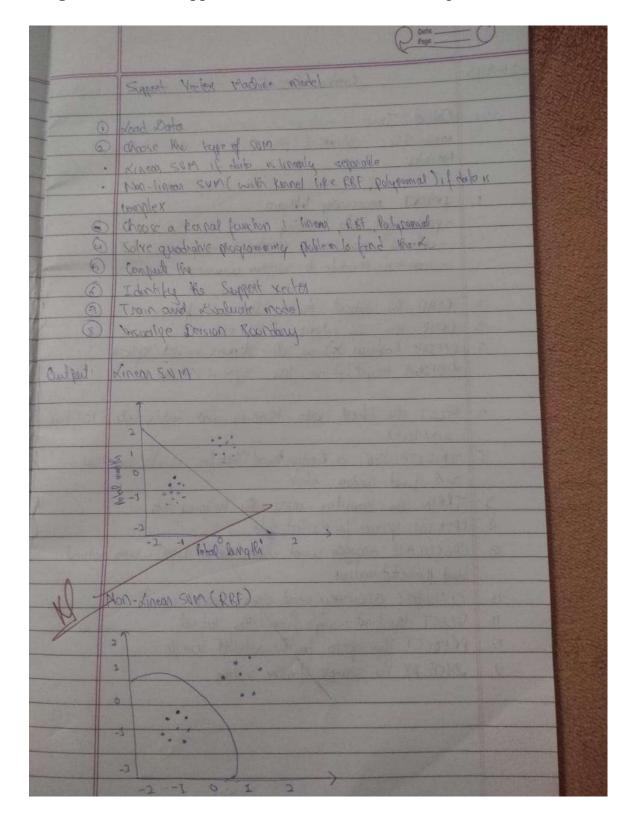
```
import pandas as pd
import numpy as np
from sklearn.linear model import LogisticRegression
from sklearn.model selection import train test split
from sklearn.metrics import accuracy score, confusion matrix, classification report
# Load Titanic dataset
df = pd.read csv('Titanic.csv')
# Select features & target
# Common useful features: Pclass, Sex, Age, SibSp, Parch, Fare, Embarked
# Target: Survived
# Handle missing data
df['Age'] = df['Age'].fillna(df['Age'].median())
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])
# Convert categorical variables to numeric
df['Sex'] = df['Sex'].map(\{'male': 0, 'female': 1\})
df = pd.get dummies(df, columns=['Embarked'], drop first=True)
# Features and target
X = df[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked_Q', 'Embarked_S']]
y = df['Survived']
# Split into train and test sets
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
# Build Logistic Regression model
model = LogisticRegression(max iter=1000)
model.fit(X train, y train)
# Predict on test data
y pred = model.predict(X test)
# Evaluation
print("Accuracy:", accuracy score(y test, y pred))
print("\nConfusion Matrix:\n", confusion matrix(y test, y pred))
print("\nClassification Report:\n", classification report(y test, y pred))
```

Program 6: Build KNN Classification model for a given dataset.

74	25 Rould KNN Classification model and SAM model
200	Talk street one party little states and
	Com Constrated model
	The time of the party and the control of the last and
	Catal Ixis 1 had 11 had
7.60	Nows 150 columns: 6
	Attributes : Sepal langth Sepal width, petal langth,
	petal midle Species
	trans at the state of the state
	KNIN as modelis and half paint
0	Regna value to K
6	Calculate the first of the state of
	Calculate the xection dictance blue now date only
333	and all other existing duta entires. Assumge thour in
11-16	Find the K nownest to be now entry housed on the
Star Ste	coloriolet grepared to no wood hours horses ou live
9	Peters the age of the sale & the sale of t
	reasest neighbors
	The state of the s
	trouble para tile of a comment of
	Input 1000 dub - \$1 56, 32, 14, 0 00
	July & Courage of END model 7
	and a superior. T
	Prediction for now Lot point Inis-sotion
	1998 3000 199
9	The state of the s
	are the first Colors on many shall the a 1 1 h
	Authority with the
	and the state of t
	The same as a second value of the

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy score, confusion matrix, classification report
import seaborn as sns
import matplotlib.pyplot as plt
# Load iris dataset
iris df = pd.read csv("Iris.csv")
# Features and target
scaler = StandardScaler()
X train = scaler.fit transform(X train)
X \text{ test} = \text{scaler.transform}(X \text{ test})
# Split data
X train, X test, y train, y test = train test split(X iris, y iris, test size=0.2, random state=42)
# Train KNN
knn iris = KNeighborsClassifier(n neighbors=3)
knn iris.fit(X train, y train)
# Predict
y pred = knn iris.predict(X test)
# Evaluate
print("Iris Dataset - Accuracy:", accuracy score(y test, y pred))
print("Confusion Matrix:\n", confusion matrix(y test, y pred))
print("Classification Report:\n", classification report(y test, y pred))
sns.heatmap(confusion matrix(y test, y pred), annot=True, cmap="Blues", fmt='g')
plt.title("Confusion Matrix - Iris KNN")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

Program 7: Build Support vector machine model for a given dataset

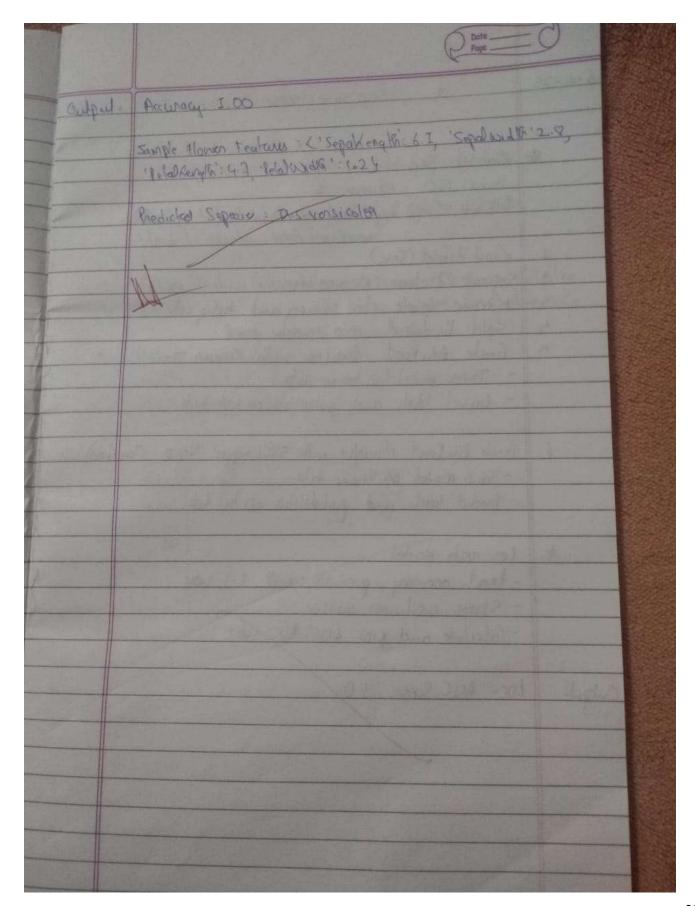


```
import pandas as pd
from sklearn.model selection import train test split
from sklearn.svm import SVC
from sklearn.metrics import accuracy score, confusion matrix
import matplotlib.pyplot as plt
import seaborn as sns
# Load the IRIS dataset
iris df = pd.read csv("Iris.csv")
# Split into features and target
X = iris_df.drop("Species", axis=1)
y = iris df["Species"]
# Split into training and testing (80/20)
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
# --- SVM with Linear Kernel ---
svm linear = SVC(kernel='linear')
svm linear.fit(X train, y train)
y pred linear = svm linear.predict(X test)
# Accuracy and Confusion Matrix
acc linear = accuracy score(y test, y pred linear)
cm linear = confusion matrix(y test, y pred linear)
print("Linear Kernel:")
print("Accuracy:", acc linear)
print("Confusion Matrix:\n", cm linear)
# Plot Confusion Matrix for Linear Kernel
plt.figure(figsize=(6, 4))
sns.heatmap(cm_linear, annot=True, fmt="d", cmap="Blues",
       xticklabels=svm linear.classes, yticklabels=svm linear.classes)
plt.title("Confusion Matrix - Linear Kernel")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight layout()
plt.show()
# --- SVM with RBF Kernel ---
svm rbf = SVC(kernel='rbf')
```

```
svm_rbf.fit(X_train, y_train)
y_pred_rbf = svm_rbf.predict(X_test)
# Accuracy and Confusion Matrix
acc rbf = accuracy score(y test, y pred rbf)
cm rbf = confusion matrix(y test, y pred rbf)
print("\nRBF Kernel:")
print("Accuracy:", acc_rbf)
print("Confusion Matrix:\n", cm rbf)
# Plot Confusion Matrix for RBF Kernel
plt.figure(figsize=(6, 4))
sns.heatmap(cm_rbf, annot=True, fmt="d", cmap="Greens",
       xticklabels=svm_rbf.classes_, yticklabels=svm_rbf.classes_)
plt.title("Confusion Matrix - RBF Kernel")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight layout()
plt.show()
```

Program 8: Implement Random forest ensemble method on a given dataset

21-4-25	Pandom Ferrat	
	peabed: Ins	4
	NOWS ITO COLUMNS ! 6	~
	Attibula:	M
1	INPORT necessary labourer:	
	- parties for dulo hardling	
	- extrain modula for machini temning	
	- womings module to supplies mouning	4
	and the first of places to	14
2.	READ to dateset from 'I'm cov' file	1
3.	OPOP Her Id colome if it exter	-
4 1	DIFINO fratum (x) as all columns except special	1
0	OFFINE larget (4) on the "species column"	4
	21-2 10 11 4	1
3.	PLIT the dolars into training and taking sets (901-trains	1
		1
6- 0	INTITALIZE a london forest Champion with two tras	
1 7	and fixed random state	
9 00	RASIN the committee using the terroring but	10
2. 12	INTET species for the fast data	
9. CH	d producted values	
16. 10	CHI ATT OFFICE A DELLE	
11. (41)	COLATE accuracy and algoritisation report	
12. 98	BCT 180 first sample from the total	
15 0	TOTCT Ku spreno for he reload som ple	
J DI	SPLAX W sample of alow values	
CHARLES (I.E.)		



```
import pandas as pd
from sklearn.model selection import train test split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy score, confusion matrix, classification report
import matplotlib.pyplot as plt
import seaborn as sns
# Load the dataset
df = pd.read csv("Iris.csv") # Adjust filename if needed
# Prepare data
X = df.drop(columns=["Id", "Species"]) # Drop non-informative columns
y = df["Species"]
# Split dataset with stratified sampling
X train, X test, y train, y test = train test split(
  X, y, test size=0.2, random state=42, stratify=y
)
# Default Random Forest with 10 trees
rf default = RandomForestClassifier(n estimators=10, random state=42)
rf default.fit(X train, y train)
y pred default = rf default.predict(X test)
acc default = accuracy score(y test, y pred default)
conf matrix default = confusion matrix(y test, y pred default)
print(f"Default RF (10 trees) Accuracy: {acc default:.4f}")
print("Confusion Matrix:\n", conf matrix default)
print("\nClassification Report for Default Model:")
print(classification report(y test, y pred default))
# Try different numbers of trees to find the best
best acc = 0
best n = 10
acc list = []
for n in range(1, 101):
  rf = RandomForestClassifier(n estimators=n, random state=42)
  rf.fit(X train, y train)
  y_pred = rf.predict(X_test)
  acc = accuracy score(y test, y pred)
  acc list.append((n, acc))
```

```
if acc > best acc:
     best acc = acc
     best n = n
     best conf matrix = confusion matrix(y test, y pred)
     best model = rf # Save the best model
print(f"\nBest Accuracy: {best acc:.4f} using {best n} trees")
print("Best Confusion Matrix:\n", best conf matrix)
# Plot accuracy vs number of trees
x vals, y vals = zip(*acc list)
plt.plot(x vals, y vals, marker='o')
plt.title("Accuracy vs Number of Trees")
plt.xlabel("Number of Trees")
plt.ylabel("Accuracy")
plt.ylim(0.8, 1.05) # Optional axis limit
plt.grid(True)
plt.axvline(best n, color='r', linestyle='--', label=f'Best: {best n} trees')
plt.legend()
plt.show()
# Evaluate best model
y pred best = best model.predict(X test)
print("\nClassification Report for Best Model:")
print(classification report(y test, y pred best))
# Plot feature importances
importances = best model.feature importances
features = X.columns
sns.barplot(x=importances, y=features)
plt.title("Feature Importances from Best Random Forest")
plt.xlabel("Importance")
plt.ylabel("Feature")
plt.tight layout()
plt.show()
```

Program 9: Implement Boosting ensemble method on a given dataset.

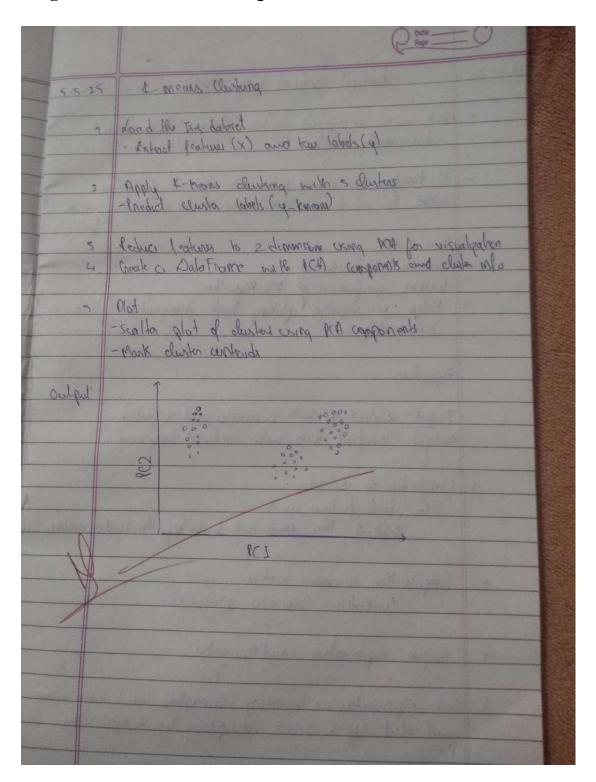
3-6-26	Boating Knseenble
Ger Ger	Dobsd Ins
	none 150 rolumers: 6 Atkibet: Sopal langle, Sopal width, pelol langle, pelolusidits
1	Lood datased (Zeis)
2 3.	Separate features (X) and labels(4) Knode labels who toming and testing sets Split Knodabid into numeric formal
5.	Geale Ada Boost dawifier with Derision The
	- Train model on training data - Inedict labels and probablities on test data
6.	Greate Ada Roost clausefrer with SVC(support Vector Claustin)
	- Thous model on toining dato - Product labels and probabilities on lest date
AND PRODUCTION OF THE PERSON O	tor roch wodel
	Show conduction making
	Colculate and xint 400-AUC sine
Outgul: KO	X- AUC Sierce 10

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.model selection import train test split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy score, classification report
# Load dataset
df = pd.read csv("adult.csv")
# Drop rows with missing values
df.dropna(inplace=True)
# Encode categorical columns
label encoders = {}
for column in df.select dtypes(include=['object']).columns:
  le = LabelEncoder()
  df[column] = le.fit transform(df[column])
  label encoders[column] = le
# Separate features and target
X = df.drop(columns=['income'], errors='ignore', axis=1)
y = df['income']
# Split into training and testing sets
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
# Define base estimator
base est = DecisionTreeClassifier(max depth=3)
# AdaBoost with 10 estimators using SAMME algorithm
model 10 = AdaBoostClassifier(estimator=base est, n estimators=10, random state=42,
algorithm='SAMME')
model 10.fit(X train, y train)
y pred 10 = model 10.predict(X test)
score 10 = accuracy score(y test, y pred 10)
print(f"Accuracy with 10 estimators: {score 10:.4f}")
print("Classification Report (10 Estimators):\n", classification report(y test, y pred 10))
```

```
# Initialize variables for fine-tuning
estimators range = list(range(10, 201, 10))
scores = []
best score = 0
best n = 0
# Fine-tune number of estimators
for n in estimators range:
  model = AdaBoostClassifier(estimator=base est, n estimators=n, random state=42,
algorithm='SAMME')
  model.fit(X train, y train)
  y pred = model.predict(X test)
  score = accuracy_score(y_test, y_pred)
  scores.append(score)
  print(f"n estimators={n}, Accuracy={score:.4f}")
  if score > best score:
     best score = score
     best n = n
     best model = model # Save the best model
     best_y_pred = y_pred
print(f"\nBest Accuracy: {best score:.4f} using {best n} estimators")
print("Classification Report (Best Estimator):\n", classification report(y test, best y pred))
# Plot accuracy vs number of estimators
plt.figure(figsize=(7, 4))
plt.plot(estimators range, scores, marker='o', linestyle='-', color='blue')
plt.title("Accuracy vs Number of Estimators (AdaBoost)")
plt.xlabel("Number of Estimators (Trees)")
plt.ylabel("Accuracy")
plt.grid(True)
plt.xticks(estimators range)
plt.tight layout()
plt.show()
# Visualize feature importances for best model
importances = best model.feature importances
features = X.columns
plt.figure(figsize=(8, 5))
sns.barplot(x=importances, y=features)
plt.title("Feature Importances (AdaBoost)")
plt.xlabel("Importance")
plt.ylabel("Feature")
```

plt.tight_layout()
plt.show()

Program 10: Build k-Means algorithm to cluster a set of data stored in a .CSV file.



```
import os
os.environ["OMP NUM THREADS"] = "1" # Avoid Windows MKL memory leak warnings
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load iris
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import confusion matrix, ConfusionMatrixDisplay
def load data(csv path='Iris.csv'):
  try:
     df = pd.read csv(csv path)
     df.columns = [c.strip().replace('', '').replace('(', ").replace('), ").replace('.', ") for c in
df.columns]
     if 'species' not in df.columns and 'Species' not in df.columns:
       # Try to find a species column case-insensitive
       species cols = [c for c in df.columns if 'species' in c.lower()]
       if species cols:
          df['Species'] = df[species cols[0]]
       else:
          raise ValueError("Species column not found in CSV")
     if 'Species' not in df.columns and 'species' in df.columns:
       df['Species'] = df['species']
  except Exception:
     # fallback to sklearn iris dataset
     iris = load iris()
     df = pd.DataFrame(
       data=np.c [iris['data'], iris['target']],
       columns=iris['feature names'] + ['target']
     )
     df.columns = [c.strip().replace('(cm)', ").replace('', '') for c in df.columns]
     df['Species'] = [iris['target names'][int(t)] for t in df['target']]
  return df
def preprocess(df):
  # Use PetalLengthCm and PetalWidthCm if present, else fall back to sklearn column names
  if 'PetalLengthCm' in df.columns and 'PetalWidthCm' in df.columns:
     X = df[[PetalLengthCm', PetalWidthCm']].values
  else:
```

```
# Fall back to iris feature names without (cm)
     col pl = next((c for c in df.columns if 'petal length' in c.lower()), None)
     col pw = next((c for c in df.columns if 'petal width' in c.lower()), None)
     if col pl and col pw:
       X = df[[col pl, col pw]].values
     else:
       raise ValueError("Cannot find petal length and width columns.")
  scaler = StandardScaler()
  X scaled = scaler.fit transform(X)
  return X scaled, scaler
def plot_elbow(X_scaled, max k=10):
  inertias = []
  ks = range(1, max k + 1)
  for k in ks:
     km = KMeans(n clusters=k, random state=42)
     km.fit(X scaled)
     inertias.append(km.inertia)
  plt.figure(figsize=(6, 4))
  plt.plot(ks, inertias, 'o-', linewidth=2)
  plt.xlabel('Number of clusters (k)')
  plt.ylabel('Inertia')
  plt.title('Elbow Method for Optimal k')
  plt.xticks(ks)
  plt.grid(True, linestyle='--', alpha=0.5)
  plt.tight layout()
  plt.show()
  return inertias
def run kmeans(X scaled, k):
  km = KMeans(n clusters=k, random state=42)
  labels = km.fit predict(X scaled)
  return km, labels
def plot confusion(df, labels, k):
  species names = df['Species'].unique()
  species to num = {name: idx for idx, name in enumerate(species names)}
  true nums = df['Species'].map(species to num)
  cm = confusion matrix(true nums, labels)
  disp = ConfusionMatrixDisplay(confusion matrix=cm,
                     display labels=[f"Cluster {i}" for i in range(k)])
  fig, ax = plt.subplots(figsize=(6, 6))
  disp.plot(ax=ax, cmap='Blues', colorbar=True)
  ax.set xlabel('Predicted Cluster')
```

```
ax.set ylabel('True Species')
  plt.title('K-Means Clustering Confusion Matrix')
  plt.tight layout()
  plt.show()
  cm df = pd.DataFrame(cm,
               index=[f"True: {name}" for name in species names],
               columns=[f"Cluster {i}" for i in range(k)])
  print("\nConfusion Matrix (counts):")
  print(cm df)
def main():
  df = load data('Iris.csv')
  if 'Species' not in df.columns:
     print("Error: 'Species' column not found in the data.")
     return
  X scaled, scaler = preprocess(df)
  print("Generating elbow plot to find optimal k...")
  plot elbow(X scaled, max k=10)
  optimal k = 3
  print(f''Choosing k = \{optimal k\} based on elbow plot.")
  km model, labels = run kmeans(X scaled, optimal k)
  df['cluster'] = labels
  plt.figure(figsize=(6, 4))
  plt.scatter(X scaled[:, 0], X scaled[:, 1], c=labels, cmap='viridis', edgecolor='k', s=50)
  centroids = km model.cluster centers
  plt.scatter(centroids[:, 0], centroids[:, 1], marker='X', c='red', s=200, label='Centroids')
  plt.xlabel('Scaled Petal Length')
  plt.ylabel('Scaled Petal Width')
  plt.title(f'K-Means Clusters (k={optimal k})')
  plt.legend()
  plt.grid(True, linestyle='--', alpha=0.5)
  plt.tight layout()
  plt.show()
  plot confusion(df, labels, optimal k)
if __name__ == "__main__":
  main()
```

Program 11: Implement Dimensionality reduction using Principal Component Analysis (PCA) method.

6	
18-5-26	Aurapal Component Analysis (RCA)
	dimensionally neduction
	of the form the original professes into a new set of features and the original features.
-	Oxa of RA Reduces redundancy
	there is recalled plotting high dimensional doctrin
	20 or 30 recalle plotting high dimensional dute in
H	godin
J.	put: Dabiet x with a samples and infraform April Transpored dato with reduced dissertional
CONTRACT OF THE PARTY OF THE PA	En each feature in K:
	Subject the mean and divide by standard demaken
2. Com	pule 160 co variona makin'
	Controleix: Coronomo mateix(x)
5. Irmi	whe cigen values and Eponymotors
4 Sort and	eigenvector by disconding expressables, cont eigenvalues

	Page	
6.	Transform the original duty	
	Transform the original duter. Reduced Date . X > PC	
	nekan Reduced Date	
	Mexan regard road	
1-		爱
1		1000
		1
2		
1		
		SEA PRO

```
import numpy as np
from sklearn.decomposition import PCA
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
# 1. Load sample data (Iris dataset)
data = load_iris()
X = data.data # features
y = data.target # labels (optional, for visualization)
# 2. Initialize PCA and reduce to 2 components
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
# 3. Show explained variance ratio by each component
print("Explained variance ratio:", pca.explained_variance_ratio_)
# 4. Plot the 2D transformed data
plt.figure(figsize=(8,6))
for target in np.unique(y):
  plt.scatter(
    X_pca[y == target, 0],
    X_pca[y == target, 1],
    label=data.target_names[target]
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.title("PCA - Iris dataset")
plt.legend()
plt.show()
```