VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



Machine Learning (23CS6PCMAL)

Submitted by

Nandini A T (1BM23CS411)

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 Nandini A T (1BM23CS411), who is a 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.

Dr. Pallavi G B	Dr. Kavitha Sooda
Assistant Professor	Professor & HOD
Department of CSE,	Department of CSE, BMSCE
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	4
2	3-3-2025	Demonstrate various data pre-processing techniques for a given dataset	7
3	10-3-2025	Implement Linear and Multi-Linear Regression algorithm using appropriate dataset	11
4	17-3-2025	Build Logistic Regression Model for a given dataset	15
5	24-3-2025	Use an appropriate data set for building the decision tree (ID3) and apply this knowledge to classify a new sample.	18
6	7-4-2025	Build KNN Classification model for a given dataset.	21
7	21-4-2025	Build Support vector machine model for a given dataset	24
8	5-5-2025	Implement Random forest ensemble method on a given dataset.	27
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.	31
11	12-5-2025	Implement Dimensionality reduction using Principal Component Analysis (PCA) method.	33

Github Link:

https://github.com/NandiniAT/ML.git

Program 1

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

	DATE: 03 03 25
	Lab- 01
	all and the second of the seco
	Demonstrate various data pre-prousing techniques for a given dataset
	Python code:
27	To look as Cl at a
201	To load csv file into the data fame
11)	To desplay information of all columns
W	10 display statistical information of all
	nymerical
	To display information of all columns To display statistical information of all numerical To display the count of unique labels for "ocean proximity" column
	tor "ocean proximity" column
1)	10 coping which atthoris (coloms) III a
-	dataset have missing values count
.01	greater than zero.
	5×9
1	Housing, Cay
	Housing. Cav
	import pandos as pd
0	import numpy as np
	and the second of gallery
	from google colab import files oploaded = files. upload ()
	Tort going : total
/	opposed = 11105. Opposed
	ex 1 = pd. read_csv('housing.csv')
+	The second secon
)#	exi
	The contract has been the second
1)	ex1. describely
	The same of the party division to the graph of the
	Ocean-proximity = exil ocean-proximity I valve con
	Ocean-proximity

100	PAGE NO: 9_DATE:
۷)	missing-values = ext. isnull. sum() Colums-missing = missing-values [missing-values >0].
25	Diabetis. CSV
75	which columns in the dataset had missing values? How did you handle them? The file "Diabetes Cev" has no missing
	values. If there are any we handle st through fillow keyword, exa. isnull(). sum()
9.>	Which categorical columns did you identify in dataset? How did you encode them? -> exa-pd. get-dumnies (ex2, drop first = True) exa
	The Columns Gencles & class has Palentified as categorical columns.
	What is the difference between Min-Max Scaling & standardization? When would you use one over the other?
	Min-Max - Scales the data by subtracting the to a fixed range which is usually [0,1] standardization - centres the data by subtracting the mean & Scales it by dividing the
6/3/15	Standard deviation. Ne use min-max when bounded range is required while standardy zation is used when data is normally distributed.

Housing

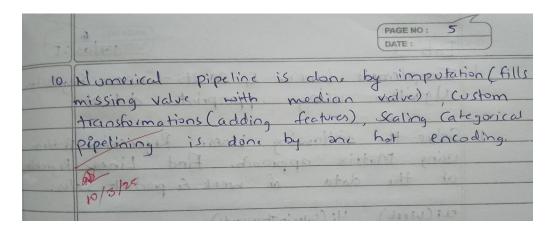
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model selection import train test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OrdinalEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from scipy import stats
from google.colab import files
uploaded=files.upload()
ex1=pd.read_csv('housing.csv')
ex1
ex1.describe()
Ocean_proximity=ex1['ocean_proximity'].value_counts()
Ocean_proximity

```
missing values = ex1.isnull().sum()
   columns missing values = missing values [missing values > 0]
   columns missing values
   Diabetes
   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
   from google.colab import files
   uploaded=files.upload()
   ex2=pd.read csv('Dataset of Diabetes .csv')
   ex2
   ex2.describe()
# Handling missing values
   ex2.isnull().sum()
#Handling categorical data
   ex2 = pd.get dummies(ex2, drop first=True)
   ex2
#Handling Outliers using IQR method (only for numerical columns)
   print("Handling Outliers...")
   numeric cols = ex2.select dtypes(include=[np.number]).columns
   Q1 = ex2[numeric cols].quantile(0.25)
   O1
   Q3 = ex2[numeric cols].quantile(0.75)
   O3
   IQR = Q3 - Q1
   IOR
   lower bound = Q1 - 1.5 * IQR
   lower bound
   upper bound = O3 + 1.5 * IOR
   upper bound
   ex2 = ex2[\sim ((ex2[numeric cols] < lower bound) | (ex2[numeric cols] > upper bound)).any(axis=1)]
   ex2
   ex2.isnull().sum()
   ex2.boxplot("AGE")
#Min-Max Scaling (Normalization)
 print("Applying Min-Max Scaling...")
   scaler = MinMaxScaler()
   scaler
   df scaled = pd.DataFrame(scaler.fit transform(ex2[numeric cols]), columns=numeric cols)
   df scaled
#Standard Scaling
 print("Applying Standard Scaling...")
 scaler = StandardScaler()
   scaler
   print("\nCleaned Data:")
   (ex2.head())
   print("\nNormalized Data:")
   print(df scaled.head())
```

Demonstrate various data pre-processing techniques for a given dataset

	Jah 2 PAGE NO: 3 DATE: 10 03 25
	Demonstrate the steps to build a Mi-model that predicts the median bousing price Using the given dataset
1.	print (df. describe())
	print x label as 'columns' print y label as 'frequency' plot histogram for each plot interpretation: median income shows median income
Shell	values of dataset indicates high law income house median age Shows distribution of median house age in different blocks.
it ass	test set is created by splitting data into training & testing set by random Sampling. (or) stratified sampling. random sampling: Samply datapoints randomly without considering distribution. Shatified sampling distribution. Shatified sampling distribution. Shatified sampling distribution. Shatified sampling distribution.
4,	"Ocean-proximity" indicated geographical feature in dataset.
5,	median income feature Correlates to maximum graph indicates positive correlation compred house valvey dataspread & few outliers.
	As median income increases median hove valve

	DATE:
	also irreases the correlation there are a few orthing
	also increases to
6.	Features that could be combined to improve correlation could be:
	rooms for
	· bedrooms per room - population I household
	even after comming
7 8	Features that ped to be cleaned include total bed 9100 ms, ocean-proximity,
4	median house value cleaning
7	done by cheeling missing values encoding categorical values, features scaling, outlier
	handling.
Я	conting and business in the last the
23	The ocean proximity column is Categorical that can be converted to numerical data by
	that can be convained to numerical data by using one hot encoding to convert where the
998	dato is reshaped, transformed Either Converted
90.	Ecoture scaling is highly important due to various factors such as
1	* improved model performance
	* HOSTIV LONVEIGENCE
	* Enhanced interpretability * prevention of numerical issues
	Tromman scaling dechniques
	* Standard 2 ation
	# robust scaling



from google.colab import files
diabetes=files.upload()
from google.colab import files
adult_income=files.upload()
dfl=pd.read_csv("Dataset of Diabetes .csv")
dfl.head()
df2=pd.read_csv("adult.csv")
df2.head()
df1.info()
df2.info()

```
df1.describe()
df2.describe()
missing values1=df1.isnull().sum()
print(missing values1)
missing values2 = df2.isnull().sum()
print(missing values2)
df1['Gender']=df1['Gender'].replace('f', 'F')
ordinal encoder = Ordinal Encoder (categories = [["F",
df1["Gender Encoded"] = ordinal encoder.fit transform(df1[["Gender"]])
onehot encoder = OneHotEncoder()
encoded data = onehot encoder.fit transform(df1[["CLASS"]])
encoded array = encoded data.toarray()
encoded df=pd.DataFrame(encoded array, columns=onehot encoder.get feature names out(["CLASS"]))
df encoded = pd.concat([df1, encoded df], axis=1)
df1 = pd.concat([df1, encoded df], axis=1)
df1.drop("CLASS", axis=1, inplace=True)
df1.drop("Gender", axis=1, inplace=True)
print(df2.head())
from sklearn.preprocessing import OrdinalEncoder, OneHotEncoder
df copy2 = df2
ordinal encoder = OrdinalEncoder(categories=[["Male","Female"]])
df copy2["Gender Encoded"] = ordinal encoder.fit transform(df copy2[["gender"]])
print(df copy2[["gender","Gender Encoded"]])
onehot encoder = OneHotEncoder()
encoded data =
onehot encoder.fit transform(df2[["occupation","workclass","education","marital-status","relationship","race","n
ative-country", "income"]])
encoded array = encoded data.toarray()
encoded df = pd.DataFrame(encoded array,
columns=onehot encoder.get feature names out(["occupation","workclass","education","marital-status","relatio
nship", "race", "native-country", "income"]))
df encoded = pd.concat([df copy2, encoded df], axis=1)
df encoded.drop("gender", axis=1, inplace=True)
df encoded.drop("occupation", axis=1, inplace=True)
df encoded.drop("workclass", axis=1, inplace=True)
df encoded.drop("education", axis=1, inplace=True)
df encoded.drop("marital-status", axis=1, inplace=True)
df encoded.drop("relationship", axis=1, inplace=True)
df encoded.drop("race", axis=1, inplace=True)
df encoded.drop("native-country", axis=1, inplace=True)
df encoded.drop("income", axis=1, inplace=True)
print(df encoded. head())
normalizer = MinMaxScaler()
df_encoded[["fnlwgt","educational-num","capital-gain","capital-loss","hours-per-week"]] =
normalizer.fit transform(df encoded[["fnlwgt","educational-num","capital-gain","capital-loss","hours-per-week"]
1)
df encoded.head()
```

```
\label{eq:continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous_continuous
```

Program 3

Implement Linear and Multi-Linear Regression algorithm using appropriate dataset

	PAGE NO: 6 DATE: 17 03 25
1.	Inplement Linear & Multi-Linear Regression algorithm using appropriate dataset
1. 5	Solve the following Linear Regression Problem Using Matrix approach. Find Linear Regression of the data of week Exproduct sales.
	di (Week) Yi (Sales in thousands)
	1 2
	2 4
	3 5
	4 9 (Y1) (1 X1) (E)
	$\beta = (\alpha T \times)^{-1} \times T \times Y$ $\beta = (\alpha T \times)^{-1} \times T \times Y$ $\gamma = (\alpha T \times)^{-1} \times T \times Y$ $\gamma = (\alpha T \times)^{-1} \times T \times Y$ $\gamma = (\alpha T \times)^{-1} \times T \times Y$ $\gamma = (\alpha T \times)^{-1} \times T \times Y$ $\gamma = (\alpha T \times)^{-1} \times T \times Y$
-> -	$X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{bmatrix}$ $Y = \begin{bmatrix} 2 \\ 4 \\ 9 \end{bmatrix}$
	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
*	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$

3	PAGE NO: DATE:
Toverse of XTX	CEDERAL COM
	- Electronic de la constitución
(xTx)-1 - (1.5 -0.5)	(C) (S)
[6.7]	1 17 - 17
-> compute B = (xTX)-1 xT)	Y
= [1.5 -0.5] -0.5 0.2]	20]
-0.5 0.2	61
	32 34
- [-0.5]	
Ullian acisto (9.21) soible	ACtor hu
Regression Line Equation y=	-0.5+2.22.
	0 101
2. Simple Linear Regression	
Drameter(X) Price(Y) Pre	dict the price of
8, 10, 20	inch pizze using
10 13 +	he data given
12	1 Constitution of the
X Y X1 X1 X1 X1 Y1 80	of all
x x x1.x1 x1.x1	
8 10 64 80	meillannist -
10 13 100	ents ask
12 16 144 192	11.1.5 -5
7 - 30/3=10 13 102.67 134	La Caraca
The state of a state of the sta	water by might
a:=(xy)-(x)(y)	In water at
(X1) x (X)	
134-(10)(13) _ 13	04-130 -4 - 1.49
(602-67)-102	2.67 2.44
in administration and desired tour	4,21.7

19 19	DATE:
	(ao=(y)-a, xx
	= 13 - 1.5 × 10
	= 13 -2
	4 = -2+1.57
	Victor (Max)
	The prediction of price of 20 is
	115 - 2 + 1.5(20)
	y = -2 + 1.5(20) y = 28
	120.7.
	After huilding the recression models
	After building the regression models, write the following questions
1.	Data Preprocessing Steps:
	*Missing Valves: Handled by imputing or dropping missing data
30 11	dropping missing data
0000	& Scaling: Applied to features with different
	* Scaling Applied to features with different Scales to ensure uniformity *Encoding: Categorical variables were encoded
	tronoung lategorical variables were encoded
	to convert them to numeric
9.	Regression line for courts much it is
	Yes the Regression line was placed
	Regression line for Canada perscapita income der Yes, the Regression line was platted
3.	Predicted Salary for Hiring Dateurch
	tura candidate with 12 years of experience a
	to in test sore & to in interview the model predicted
	o salary of \$64629.625
15	(8) \$ (18)
4.	Encoding: State was encoded using one but encoding
4	The frest of flatures were scaled to ensure consistent range & prevent larger flatures from dominating model
	range a provent larger flatures from dominating model

```
Code:
```

```
from google.colab import files
per capita income=files.upload()
from google.colab import files
salary=files.upload()
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
from sklearn import linear model
dfl=pd.read csv("canada per capita income.csv")
df1.head()
df2=pd.read csv("salary.csv")
df2.head()
df2.YearsExperience.median()
df2.YearsExperience = df2.YearsExperience.fillna(df2.YearsExperience.median())
plt.xlabel("year")
plt.ylabel("per capita income (US$)")
plt.scatter(df1.year, df1['per capita income (US$)'])
plt.xlabel("YearsExperience")
plt.ylabel("Salary")
plt.scatter(df2.YearsExperience, df2.Salary)
reg1 = linear model.LinearRegression()
reg1.intercept
reg1.predict([[2020]])
reg2 = linear model.LinearRegression()
reg2.fit(df2.drop('Salary', axis='columns'), df2['Salary'])
reg2.coef
reg2.intercept
reg2.predict([[12]])
from google.colab import files
hiring=files.upload()
from google.colab import files
companies=files.upload()
df3=pd.read csv("hiring.csv")
df3.head()
df4=pd.read csv("1000 Companies.csv")
df4.head()
```

```
df3.isnull().sum()
df4.isnull().sum()
df3 copy = df3.copy()
experience mapping = {'two': 2, 'three': 3, 'five': 5, 'seven': 7, 'ten': 10, 'eleven': 11}
df3 copy['experience'] = df3 copy['experience'].map(experience mapping)
median experience = df3 copy['experience'].median()
df3 copy['experience'] = df3 copy['experience'].fillna(median experience)
df3 copy
df3 copy['test score(out of 10)'] = df3 copy['test score(out of 10)'].fillna(df3 copy['test score(out of
10)'].mean())
reg3 = linear model.LinearRegression()
reg3.fit(df3 copy.drop('salary($)', axis='columns'), df3 copy['salary($)'])
reg3.coef
reg3.intercept
reg3.predict([[2,9,6]])
reg3.predict([[12,10,10]])
ohe = OneHotEncoder(sparse output=False, handle unknown='ignore')
state encoded = ohe.fit transform(df4[['State']])
state encoded df = pd.DataFrame(state encoded, columns=ohe.get feature names out(['State']))
df4 = pd.concat([df4, state encoded df], axis=1).drop(columns=['State'])
print(df4)
reg4 = linear model.LinearRegression()
reg4.fit(df4.drop('Profit',axis='columns'),df4.Profit)
print(reg4.coef )
print(reg4.intercept )
reg4.predict([[91694.48, 515841.3, 11931.24,0,1,0]])
```

Build Logistic Regression Model for a given dataset

Screenshot:

Lab-Off Logistic Regression PAGENO: 91 DATE: 94/03/25	
Consider a primary of the control of	
Consider a binary classification problem where	
predict whether a student will	3 De
pass or fail based on their study hours.	dh
The togistic regression model has been	
trained & the learned parameters are	
as -5 lintercept) & a. O. 8 (coefficient box	P
Study hours)	
1. Write logistic regression equation.	11 0
	40.6
P(passia).	308
1+ e-(a0+a,7)	Sol
Girco: 90: -5, 91:08	
+ e - (a ₀ + a ₁ x) Giren: a ₀ = -5 a ₁ = 0.8 p(pass x) = 1	
1+e=(-5+0.8x)	1.
Southern Industry with our grant &	
Calculate probability for x= 7 hours	2, 9
D(Dass 3)	
1+e-(-5+0.8x 7)	3- D
Stral : Compute linear Combination	Sc
Z=90+917 !	- 0
Z=-5+0.8 x7=-5+5.6=0.6	SOF
Stro2: Plus Z into logistic form	- 0.
P(nacs 7) - 1	- Soft
1+e-0.6	
1+e.0.6 Step3: Compute e-0.6 e-0.6 ~ 0.5488	> For
P-06 & 0.5488	
Clark: Calculate Const 1:12to	ISMY
Step - Callotte Propositing	direc
11251 112 - 1.5488	to The
Step 4: Carlevlate probability P(pass) 73. 1 140.5488	reter
so, the probability that student will pass approximately is 64.57%	160
approximately is 64.24%	Mine

ermine the predicted dass using a exhold of 0.5 P(pass|n) > 0.5, predict pan P(pass|n) < 0.5, predict fail x= 7 hours (pass/7) % 0, 645 7 0.5 ness, the predict class is pass. sider 2=[2,1.0] for 3 classes. Apply mon fun to find prob of 3 dance 50ft Max (21) = e21 Z = [2,1,0] exponentiate Pach value e2 = [e2, e1, e0] = [7.3891, 2.7183,1] me up the exponentialed values: Sum = 7.3891+ 2.7183+1 = 11.1074 ide each exponediated value by com: tmax (20) . 7.3841 & 0.6652 dono 11.1074 nam (21) _ 2.7183 = 0.2447 (lan) 11. 4074 9x (2x) = 1 0.0901 clans 111,1074 dataset file "HR-Romma - up. (50) variables did you identify as him a & clear propert on employee retention? why ky variable impact the employees are attitochen level lower Satisfaction intreases cent in company - employees 5t years tend to leave

Arealary. low entaries lead to higher tomover

All of projects a average monthly hours very

high or low values affect retention.

ii) What was the accountry of your logistic regression model? To you think this is a good accoracy! Why or why not?

The accoracy of logistic regression model is 7840%. This accoracy is overall good.

But the model still much an improvement.

```
Code:
from google.colab import files
hr=files.upload()
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
from sklearn import linear model
import seaborn as sns
from sklearn.model selection import train test split
from sklearn.linear model import LogisticRegression
from sklearn.metrics import accuracy score
df1=pd.read csv("HR comma sep.csv")
df1.head()
dfl.isnull().sum()
plt.figure(figsize=(12, 6))
sns.barplot(x='Department', y='left', data=df1)
plt.title('Employee Retention Rate by Department')
plt.xlabel('Department')
plt.ylabel('Proportion of Employees Left')
plt.xticks(rotation=45, ha='right')
plt.show()
ohe = OneHotEncoder(handle unknown='ignore', sparse output=False)
department encoded = ohe.fit transform(df1[['Department']])
department encoded df = pd.DataFrame(department encoded,
columns=ohe.get feature names out(['Department']))
df1 = pd.concat([df1, department encoded df], axis=1)
df1 = df1.drop('Department', axis=1)
ordinal encoder = OrdinalEncoder(categories=[['low', 'medium', 'high']], dtype=np.int64)
salary encoded = ordinal encoder.fit transform(df1[['salary']])
df1['salary encoded'] = salary encoded
df1 = df1.drop('salary', axis=1)
df1.head()
correlation matrix = df1.corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix of Features')
plt.show()
plt.figure(figsize=(8, 6))
sns.barplot(x='salary encoded', y='left', data=df1)
plt.title('Impact of Employee Salary on Retention')
```

plt.xlabel('Salary Level (Encoded)')
plt.ylabel('Proportion of Employees Left')

plt.show()

```
df copy = df1[['number project', 'average montly hours', 'time spend company', 'left', 'salary encoded',
'satisfaction level','Work accident']]
df copy.head()
X = df copy.drop('left', axis=1)
y = df copy['left']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
model = LogisticRegression(max iter=1000)
model.fit(X train, y train)
y pred = model.predict(X test)
accuracy = accuracy score(y test, y pred)
print(f"Accuracy of the Logistic Regression model: {accuracy}")
from google.colab import files
zoodata=files.upload()
zootype=files.upload()
zoo data
            =
                 pd.read csv('zoo-data.csv')
zoo class = pd.read csv('zoo-class-type.csv')
merged data = pd.merge(zoo data, zoo class, left on='class type', right on='Class Number')
merged data = merged data.drop(['Animal Names', 'Number Of Animal Species In Class',
'Class Number', 'class type', 'animal name'], axis=1)
X = merged data.drop('Class Type', axis=1)
y = merged data['Class Type']
print(merged data.head())
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
model = LogisticRegression(max iter=1000)
model.fit(X train, y train)
y pred = model.predict(X test)
accuracy = accuracy score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
cm = confusion matrix(y test, y pred)
disp = ConfusionMatrixDisplay(confusion matrix=cm, display labels=np.unique(y test))
disp.plot(cmap="Blues", values format="d")
plt.title("Confusion Matrix")
plt.show()
```

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

Screenshot:

Lab 604 Decission tres Chosider the Following Addaset, calculate entropy 8 information gain part to tanget voricity 8 information gain part to tanget voricity 8 information gain part to tanget voricity 1 the thirty whether the splitting cade About the as or or otherwise. About the as or or otherwise. Present a or or or or otherwise. Present a or or or or or otherwise. Present a or or or or otherwise. Present a or		11 11 11 11	18 20 18	al Print	PAGE NO: 19	
Coosider the Estouring dataset, calculate entropy a information gain with to tanget vorious, a information gain whether the splitting rade thould be as or go attribute. About the as of classification the High No that this No that this No that this No factor (s, a) = 0.6079 - max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.1249 The false 1,2,6,7,8 3,4,5,9,10 Attribute (a) Values (a) = that, (o) S a: = [1+, 4] Entropy (so,) = -1/5 log 1/5 - 4/5 log 4/5 Cool = [1+, 3-]: Entropy = 1/4 log 1/4 = 3/4 log 3/4 = 0 cool = [0+,1-] Entropy = 0.0 ain (s, a) = Entropy (s) . If Sul Entropy (so) = -1/5 Sul Entropy (so) = -1/				was	DAIE:	-
Coosider the Estouring dataset, calculate entropy a information gain with to tanget vorious, a information gain whether the splitting rade thould be as or go attribute. About the as of classification the High No that this No that this No that this No factor (s, a) = 0.6079 - max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.1249 The false 1,2,6,7,8 3,4,5,9,10 Attribute (a) Values (a) = that, (o) S a: = [1+, 4] Entropy (so,) = -1/5 log 1/5 - 4/5 log 4/5 Cool = [1+, 3-]: Entropy = 1/4 log 1/4 = 3/4 log 3/4 = 0 cool = [0+,1-] Entropy = 0.0 ain (s, a) = Entropy (s) . If Sul Entropy (so) = -1/5 Sul Entropy (so) = -1/	JLob €	504 D	ecission	11		1
Expression gain The thing and the splitting and the splitting and the should be as or as attribute. The stank as as as classification The thingh No Gain (s, as) = 0.6079 - mex gain Gain (s, as) = 0.1245 Gain (s, as) = 0.4197 The false 1,2,6,7,8 3,4,5,7,10 The thingh Ca Valves (as) = that, (00) So = [1+,4,] Entropy (so,) = "/s logs "/s - 4/5 logs 4/5 Cool = [1+,3-]: Entropy = "/4 logs 1/4 - 3/4 logs 3/4 - 6 Cool = [0+:1-) Entropy = 0.0 Cool = [0+:1-) Entropy = 0.0					t, calculate pa	1
Expression gain The thing and the splitting and the splitting and the should be as or as attribute. The stank as as as classification The thingh No Gain (s, as) = 0.6079 - mex gain Gain (s, as) = 0.1245 Gain (s, as) = 0.4197 The false 1,2,6,7,8 3,4,5,7,10 The thingh Ca Valves (as) = that, (00) So = [1+,4,] Entropy (so,) = "/s logs "/s - 4/5 logs 4/5 Cool = [1+,3-]: Entropy = "/4 logs 1/4 - 3/4 logs 3/4 - 6 Cool = [0+:1-) Entropy = 0.0 Cool = [0+:1-) Entropy = 0.0	Consi	der the	tollow ic	- xit to	tanget Vorich	Per
(a) (S, a) = Entropy (S) . (Cassiscation) 1 Hot High No 2 Hot High No 6 Coal High No 7 Hot High No 8 Hot Name Yes Gain (S, a) = 0.6079 - max gain (Gain (S, a) = 0.1245 (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a) 3/4 1 a) (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a) 3/4 1 a) (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a)	E 10 F	mation	gain	1-70 ether	the splitting	1
(a) (S, a) = Entropy (S) . (Cassiscation) 1 Hot High No 2 Hot High No 6 Coal High No 7 Hot High No 8 Hot Name Yes Gain (S, a) = 0.6079 - max gain (Gain (S, a) = 0.1245 (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a) 3/4 1 a) (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a) 3/4 1 a) (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a)	'daysi	ication', S	Identity -	attribo	te.	ade
(a) (S, a) = Entropy (S) . (Cassiscation) 1 Hot High No 2 Hot High No 6 Coal High No 7 Hot High No 8 Hot Name Yes Gain (S, a) = 0.6079 - max gain (Gain (S, a) = 0.1245 (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a) 3/4 1 a) (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a) 3/4 1 a) (Gain (S, a) = Entropy (S) . (La lay 14 1 a) 3/4 1 a)	Should	be as	57 05			
1 Hot High No 2 Hot High No 6 Coal High No 7 Hot High No 8 Hot Name Yes Gain (s, a) = 0.6079 - max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.1245 Gain (s, a) = 0.4199 (a) False 1,2,6,7.8 3,4,5,7,10 Attribute (a) = Hot, (00) S a = [1+, 4, .] Entropy (so,) = -1/5 log 1/5 - 4/5 log 4/5 Cool a [0+:1] Entropy = 1/4 log 1/4 = 3/4 log 3/4 = 0. Cool a [0+:1] Entropy = 0.0 Ain (s, a) = Entropy (s) - 4/5 log 1/4 = 3/4 log 3/4 = 0.0			-	classificat.	an	-
1 Hot trigh No 2 Hot High No 6 Coal High No 7 Hot High No 8 Hot Normal Yes Gain (S, a) = 0.6079 - mex gain Gain (S, a) = 0.1245 Gain (S, a) = 0.4197 False 1,2,6,7,8 3,4,5,7,10 Atholic Cla Valves (a) = Hot, (00) S o = [1+,4,.] Entropy (So.) - "/S log, "/S - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-] Entropy = "/4 log, 1/4 3/4 109, 3/4 00 coal a [0+:1-] Entropy = 0.0	Postan	e ar	03	No		
6 (col High No. 7 Hot High No. 8 Hot Normal Yes Gain (s, a) = 0.6079 - max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.4199 (a) False 1,2,6,7,8 3,4,5,7,10 Atholic cla Valves (a2) = Hot, (00) S o = [1+,4] Entropy (so,) = -1/5 log, 1/5 - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-]: Entropy = -1/4 log, 1/4 - 3/4 log, 3/4 - 0. Cool = [0+:1-] Entropy = 0.0	1	HOL	Hish			
Gain (s, a) = 0.6079- max gain Gain (s, a) = 0.6079- max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.4199 (a) Falue 1,2,4,7,8 3,4,5,9,10 At tribula (12 Value (a2) = Hot, (00) S a = [1+, 4] Entropy (sa) = -1/5 lag 1/5 - 4/5 lag 4/5 = 0.7217 SHot = [1+,3-]: Entropy = 1/4 lag 1/4 - 3/4 lag 3/4 = 0.00 Gain (s, a) = Entropy (s) - 4/5 lag 1/4 - 3/4 lag 3/4 = 0.00		Hot	thigh	,100	114 10 51	-
Gain (s, a) = 0.6079- max gain Gain (s, a) = 0.6079- max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.4199 (a) Falue 1,2,4,7,8 3,4,5,9,10 At tribula (12 Value (a2) = Hot, (00) S a = [1+, 4] Entropy (sa) = -1/5 lag 1/5 - 4/5 lag 4/5 = 0.7217 SHot = [1+,3-]: Entropy = 1/4 lag 1/4 - 3/4 lag 3/4 = 0.00 Gain (s, a) = Entropy (s) - 4/5 lag 1/4 - 3/4 lag 3/4 = 0.00		Cool	High	No		
Gain (s, a) = 0.6077- max gain Gain (s, a) = 0.1245 Gain (s, a) = 0.4197 Gain (s, a) = 0.4197 (a) Falue 1,2,6,7,8 3,4,5,7,10 Athibut (12 Value (a2) = Hot, (00) S a = [1+,4] Entropy (so,) = -1/5 log, 1/5 - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-]: Entropy = -1/4 log, 1/4 - 3/4 log, 3/4 = 0.00 Gain (s, a) = Entropy (s) = -1/5 log, 1/5 Entropy (s)		404	High	Lis	1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 -	-
Gain (5, 92) = 0.1245 Gain (5, 92) = 0.4199 The fall 1,2,6,7,8 3,4,5,7,10 at thibuta (22 Volvey (92) = Hot, (00) S 91 = [1+, 4] Entropy (50) = -1/5 lag 1/5 - 4/5 log 4/5 = 0.7219 SHot = [1+,3-]: Entropy = -1/4 lag 1/4 - 3/4 lag 3/4 - 6. Gain (5, 92) = Entropy (5) - 4/5 log 1/4 lag 1/4 - 3/4 lag 3/4 - 6.	8	Hot	Mornay	, 0,		_
Gain (5, 92) = 0.1245 Gain (5, 92) = 0.4199 The fall 1,2,6,7,8 3,4,5,7,10 at thibuta (22 Volvey (92) = Hot, (00) S 91 = [1+, 4] Entropy (50) = -1/5 lag 1/5 - 4/5 log 4/5 = 0.7219 SHot = [1+,3-]: Entropy = -1/4 lag 1/4 - 3/4 lag 3/4 - 6. Gain (5, 92) = Entropy (5) - 4/5 log 1/4 lag 1/4 - 3/4 lag 3/4 - 6.						
Gain (5,93) = 0.4/99 (a) Falle 1,2,4,7,8 3,4,5,9,10 At this bit Cl2 Values (92) = Hot, (00) S 01 = [1+, 4] Entropy (50) = -1/5 log, 1/5 - 4/5 log, 4/5 = 0.7219 SHot = [1+,3-]: Entropy = -1/4 log, 1/4 - 3/4 log, 3/4 = 0. Cool (5,90) = Entropy (5) - 4/5 log, 1/4 - 3/4 log, 3/4 = 0.	Gain (5,91)=	0.6079-	max gui	<u> </u>	-
Falm 1,2,6,7,8 3,4,5,7,10 at this but as Valves (a2) = Hot, (00) So = [1+,4,] Entropy (50,) = "/5 log, "/5 - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-]: Entropy = "/4 log, 1/4 = 3/4 log, 3/4 = 0. cool = [0+:1-] Entropy = 0.0 ain (5, a2) = Entropy (5) = [5v] Entropy (5)	Gain C	, 02) =	0.1245			
Falm 1,2,6,7,8 3,4,5,7,10 at this but as Valves (a2) = Hot, (00) So = [1+,4,] Entropy (50,) = "/5 log, "/5 - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-]: Entropy = "/4 log, 1/4 = 3/4 log, 3/4 = 0. cool = [0+:1-] Entropy = 0.0 ain (5, a2) = Entropy (5) = [5v] Entropy (5)	Gain (5,03) =	0.4199	and the same of	The Table of the same of the s	
Falm 1,2,6,7,8 3,4,5,9,10 at tribula (22 Valves (92) = Hot, (00) So = (1+,4,) Entropy (50) = -1/5 log, 1/5 - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-]: Entropy = -1/4 log, 1/4 - 3/4 log, 3/4 = 0. cool = (0+;1-) Entropy = 0.0 Jain (5,90) = Entropy (5) - 4/5 JSVI Entropy (5)						
1,2,6,7,8 3,4,5,7,10 at tribute (12 Values (22) = Hot, (00) S a: = [1+, 4] Entropy (50,) = -1/5 log, 1/5 - 4/5 log, 4/5 = 0.7217 SHot = [1+,3-]: Entropy = 1/4 log, 1/4 = 3/4 log, 3/4 = 0.0 cool = [0+:1-] Entropy = 0.0 dain (5, 92) = Entropy (5). # [5v] Entropy (5)						
Solution (2) = Hot, (00) Solution (2) = Hot, (00) Solution (3) = [1+, 4,] Entropy (50.) = -1/5 log, 1/5 - 4/5 log, 4/5 = 0.7210 Stot = [1+, 3-]: Entropy = 1/4 log, 1/4 = 3/4 log, 3/4 = 0.0 Cool = [0+:1-] Entropy = 0.0 Cool = [0+:1-] Entropy (5) = [5vl Entropy (5)]		ane/	False			
Value (a2) = Hot, (00) S 01 = [1+, 4] Entropy (50) = -1/5 log 1/5 - 4/5 log 4/5 = 0.7211 SHOT = [1+,3-]: Entropy= 1/4 log 1/4 23/4 log 3/4 0. Cool = [0+;1-] Entropy= 0.0 Gain (5, a2) = Entropy (5) = [50] Entropy (5)	1,2	,6,7,8	3,4,5,	9,10		
Value (a2) = Hot, (00) S 01 = [1+, 4] Entropy (50) = -1/5 log 1/5 - 4/5 log 4/5 = 0.7211 SHOT = [1+,3-]: Entropy= 1/4 log 1/4 23/4 log 3/4 0. Cool = [0+;1-] Entropy= 0.0 Gain (5, a2) = Entropy (5) = [50] Entropy (5)						
Entropy (50.) ~ = 1/5 log, 1/5 - 4/5 log, 4/5 = 0.72/1 Stot = [1+,3-]: Entropy = 1/4 log, 1/4 - 3/4 log, 3/4 = 0.00 (co) = [0+:1-] Entropy = 0.0 (ain (5, 9.) = Entropy (5) - #	at tribut	i a2				
Entropy (50.) ~ = 1/5 log, 1/5 - 4/5 log, 4/5 = 0.72/1 Stot = [1+,3-]: Entropy = 1/4 log, 1/4 - 3/4 log, 3/4 = 0.00 (co) = [0+:1-] Entropy = 0.0 (ain (5, 9.) = Entropy (5) - #	Value	(a2)=	Hot , C	100		
(cool (C, qo) = Entropy (s) . If						
(cool (C, qo) = Entropy (s) . If	Entrop	(50.) =	-1/5 1-	11-	4/2/09 4/5	
(cool (C, qo) = Entropy (s) . If			= 0.721	92 (5	13 39 1/3	
ain (s, a.) = Entropy (s) . If I sul Entropy (s)	SHOT 2	11+ 3-	1. 6-1	11 .	11 31 31	-0
ain (s, a.) = Entropy (s) . If I sul Entropy (s	cont 6 5	1+:1-76	COTTO	Py - 14 12	92 14 = 14 139 14	
		1 26	Mopy =	0.0		-
	iain Cc		1 -	/		-
12/ E1003, 10HZ 3V	Can (?)	3 = (ep				1 (5
				VE SHOT , C	121 8100	-

iv .	PAGE NO: 13
Gain (s, a2) = Entropy (s) - 4/5	Entropy (SHOt) -1/5
Bain (5, a) = 0.9709 - 415 = 0.3219	X0.8112-1/5 X0.0
- 0.3714	Lower Land
Attribute . 43	A
value (a3) = High, Norma	5-7- 115129 115-
Sai - Cl+, H-) totropy C	109, 415 = 0.7219
SHigh ~ Cot, 4, 7) Entrepy	(SHigh) = 0.0
spormal = [1+,0-] Entrop	y (normal)=0.0
Gain (S,a3) = Entropy (S) -	5 ISVI Forting (Sr)
Gain (5,93) = Entropy (5) -4/0	Entropy (SHigh) - 115
Goin (S, 93) = 0.9709 x - 415	x0.0-1/5 x0.0= 0.7219
Althora To as to believe	A STATE OF THE STA
1,2,7,6,8 3,4,5	ne
1,2,7,6,8 3,4,5	5,9,10
(03)	Yes
pomod yeth	
Hish Domal 1,2,6,7, NO Yes	and marriaged
	milicants the
For "iris.csv" dataset	
what was the account	racy Store
enthe according store of	the mode was 1.00
(1000) meaning the mode	persectly classified

(b) the confessor mative shows that the model mode no errors as all rediction model the archael class.

[Cro 00]

[O 0 11]

[D 0 0 11]

[D 0 0 11]

[D No misselvesification occurred since all of clagonal values in the confusion matrix are recy species were classified correctly without confusion.

[For petrol consumption, csv datasel
[O) The Regression tree shuther splits date to minimize mix important features for predicting the guy petrol consumption.

[D) The most important features for predicting petrol consumption are retail to so population priver license.

[O) The Regression tree predicts continue value while a classifier predicts categories. The regression tree minimizes variance while minimizes impurity.

```
from google.colab import files
iris=files.upload()
df1=pd.read csv("iris.csv")
dfl.head()
df1.isnull().sum()
X = df1.drop('species', axis=1)
y = df1['species']
X train, X test, y train, y test = train test split(X, y, test_size=0.2, random_state=42)
clf = DecisionTreeClassifier(criterion='entropy')
clf.fit(X train, y train)
y pred = clf.predict(X test)
accuracy = accuracy score(y test, y pred)
print(f'Accuracy: {accuracy:.2f}')
print(classification report(y test, y pred))
plt.figure(figsize=(12, 8))
plot tree(clf, filled=True, feature names=X.columns, class names=y.unique())
plt.show()
cm = confusion matrix(y test, y pred)
disp = ConfusionMatrixDisplay(confusion matrix=cm, display labels=clf.classes)
cmap = plt.cm.get cmap('PuBuGn')
disp.plot(cmap=cmap)
plt.show()
drug=files.upload()
df2=pd.read csv("drug.csv")
df2.head()
df2.isnull().sum()
label encoders = \{\}
for column in df2.columns:
  le = LabelEncoder()
  df2[column] = le.fit transform(df2[column])
  label encoders[column] = le
X = df2.drop('Drug', axis=1)
y = df2['Drug']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
clf = DecisionTreeClassifier(criterion='entropy')
clf.fit(X train, y train)
y pred = clf.predict(X test)
accuracy = accuracy score(y test, y pred)
print(f'Accuracy: {accuracy:.2f}')
print(classification report(y test, y pred))
plt.figure(figsize=(12, 8))
plot tree(clf, filled=True, feature names=X.columns, class_names=[str(c) for c in y.unique()])
plt.show()
cm = confusion matrix(y test, y pred)
disp = ConfusionMatrixDisplay(confusion matrix=cm, display labels=clf.classes)
```

```
cmap = plt.cm.Blues
disp.plot(cmap=cmap)
plt.show()
pc=files.upload()
df3=pd.read csv("petrol consumption.csv")
df3.head()
df3.isnull().sum()
X = df3.drop('Petrol Consumption', axis=1)
y = df3['Petrol Consumption']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
regressor = DecisionTreeRegressor(random state=42)
regressor.fit(X train, y train)
y pred = regressor.predict(X test)
mse = mean squared error(y test, y pred)
rmse = sqrt(mse)
mae = mean absolute error(y test, y pred)
r2 = r2 score(y test, y pred)
print(f'Mean Squared Error: {mse:.2f}')
print(f'Root Mean Squared Error: {rmse:.2f}')
print(f'Mean Absolute Error: {mae:.2f}')
print(f'R-squared: {r2:.2f}')
plt.figure(figsize=(30, 30))
plot tree(regressor, filled=True, feature names=X.columns, fontsize=10)
plt.show()
```

Build KNN Classification model for a given dataset.

Lab -as	chossific	ation V	1011	PE STED TO STA	3
KMIN			1001		
C-0634.	the fo	Mowing	dataset.	2-0 × -0	
Coriside	+ Cx . 3	5 1000	as (Per	son Aar	5
Colve .	icion KN	on das	sifier me	del & de	enary K)
5014-	target				
the	3				
Person	Age	Salaryk	Target		
A	18	50	N	the water	
B	23	55	N	and a stand	100
C	24	70	N	naint.	
D	201	60	4.	- Complete Same	
E	43	70	~	2015	
F	38	LIO	7	21 0 . 10	es l
	35	100	3		ry l
×					
				rank	
distance	0 (19	E9 81	5	
1(35-18	12+ Cloc	,-50)°=	52.6	1,	
135-23	D2 + (100	-55)2	40.51	2	
1(35-21	1)2 + (100	5-70)2 -	31.95	3	
1(35-41)	12+(100	$-60)^2 =$ $-70)^2 =$	40-43	1	
J(35-4	3)2+ (100	-70)2 =	31.05		
· [C35 - 2	8)2+(100	-40)2 = (51.07	6	
185/-35	X2				
		7			
KEI	Jarge				
	target	- N			
K = 3		= 4			
tange	t for	Person 7	(=)		
for ici	s dat	110-			
	Call				

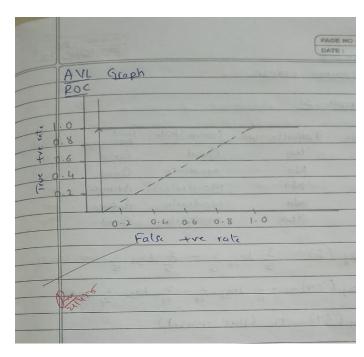
	PAGE NO: 16 DATE:
*	How to choose the k value? Demonstrate
2010,71	Best k volve in iris data set
Maria	and error rate the k value of 3 was
	choosen because of high accuracy &
	Shows douglos soft access
2.	For diabetes dataset of feature scaling?
	how to partim it
3310	Executive scaling in diabetes to ensure that all the features like glucose age insulin are on the same scale to that KNN does
	not get biased by large numerical values.
	dere de la constante de la con
	- Cart
	6 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

```
from google.colab import files
iris=files.upload()
dfl=pd.read csv("iris (2).csv")
df1.head()
dfl.isnull().sum()
X = dfl.drop('species', axis=1)
y = df1['species']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
best k = 1
best accuracy = 0
for k in range(1, 11):
  knn = KNeighborsClassifier(n neighbors=k)
  knn.fit(X train, y train)
  y pred = knn.predict(X test)
  accuracy = accuracy score(y test, y pred)
  if accuracy > best accuracy:
    best accuracy = accuracy
    best k = k
print(f"Best k value: {best k}")
knn = KNeighborsClassifier(n neighbors=3)
knn.fit(X train, y train)
y pred = knn.predict(X test)
print("Accuracy Score:", accuracy score(y test, y pred))
print("\nConfusion Matrix:")
cm = confusion matrix(y test, y pred)
print(cm)
print("\nClassification Report:")
print(classification report(y_test, y_pred))
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
       xticklabels=knn.classes , yticklabels=knn.classes )
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
diabetes=files.upload()
df2=pd.read csv("diabetes.csv")
df2.head()
df2.isnull().sum()
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X scaled = scaler.fit transform(df2.drop('Outcome', axis=1))
X train, X test, y train, y test = train test split(X scaled, df2['Outcome'], test size=0.2, random state=42)
best k = 1
best accuracy = 0
for k in range(1, 11):
  knn = KNeighborsClassifier(n neighbors=k)
  knn.fit(X train, y train)
  y pred = knn.predict(X test)
  accuracy = accuracy score(y test, y pred)
  print(f"Accuracy for k={k}: {accuracy}")
  if accuracy > best accuracy:
```

```
best accuracy = accuracy
     best k = k
print(f"Best k value: {best k}")
knn = KNeighborsClassifier(n neighbors=best k)
knn.fit(X train, y train)
y pred = knn.predict(X test)
accuracy = accuracy score(y test, y pred)
print("Accuracy:", accuracy)
cm = confusion matrix(y test, y pred)
print("Confusion Matrix:")
print(cm)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
print("\nClassification Report:")
print(classification report(y test, y pred))
heart=files.upload()
df3=pd.read csv("heart.csv")
df3.head()
df3.isnull().sum()
X = df3.drop('target', axis=1)
y = df3['target']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
best k = 1
best accuracy = 0
for k in range(1, 11):
  knn = KNeighborsClassifier(n neighbors=k)
  knn.fit(X train, y train)
  y pred = knn.predict(X test)
  accuracy = accuracy score(y test, y pred)
  print(f''Accuracy for k=\{k\}: {accuracy}, Error Rate for k=\{k\}: {1-accuracy}'')
  if accuracy > best accuracy:
     best accuracy = accuracy
     best k = k
print(f"Best k value: {best k}")
knn = KNeighborsClassifier(n neighbors=optimal k)
knn.fit(X train, y train)
y pred = knn.predict(X test)
print("Accuracy Score:", accuracy score(y test, y pred))
print("\nConfusion Matrix:")
cm = confusion matrix(y test, y pred)
print(cm)
print("\nClassification Report:")
print(classification report(y_test, y_pred))
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
       xticklabels=knn.classes , yticklabels=knn.classes )
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

Build Support vector machine model for a given dataset

PAGE NO : 17 DATE : 310 4	PAGE NO: 18
Build SVM model for a given dataset	
Build SVM more pro given acrosses	
2 1 2 2 10 2 11 by des	In ivis dataset: both is 100%
Points (4,1) (4,-1) & (6,0) belong to the doss	1. The way for both is 100%
6 points (1,0) (0,1) & (0,-1) belong to -ve	To ivis dataset: 100%
class. Draw an optimal hyperplane	180th personer be choosen
A	+ As a gusalt one than both give,
3	ats a rusult one cannot give , over the other as they both give ,
2 2 action of the E	accoracy.
(0,1)	
0 (11)	a Letter-recognition. (SV!
1-6-5-4-3-2-1 1 2 3 4 5 6	YES K & P, S & Z, V& W, N & Y, K & Y a
(0,-1) (4,-1)	some of examples where in the model ?
11 mans day dayll -3 25 library 120 b	ame of champion with
	getting confused
47th regative class is clustered around the	carea under the curve = 0.99 70.95
	of the model is excellent at seperating to
origin there is arouped towards the	Classes across thresholds in it's it is 100%
origin (left side) of the positive class is grouped towards the right side of the plane as 3 separates	Will accordy for letter-recognition the according
gright side of the parts	is 93% (or) 0,4305 of Reformed better on in
the classes clearly	dataset that it did on the other if the
Finding optimal hyperplane	letter-recognition dataset.
	and the second
support vectors one: closest positive point: (4,1) & (4,1)	Carpison motion
	and and desired the second of
closest positive points: (1,0).	
the hiperolan	accused & RBE Kernel
the margin is meximized when the hyperplace	accuracy 1:0 accuracy 1:0
26 canidistant from	Contrain matrix
actifive side -	
negative side 2:1	[0 9 0]
Ticanic July	(o o u) (o d o)
2=(4+1) 2	
7:2.5	A Subject of the Subj
1.11	nembic side



```
from google.colab import files
iris=files.upload()
df1=pd.read csv("iris (1).csv")
df1.head()
X = df1.drop('species', axis=1)
y = df1['species']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
rbf svm = SVC(kernel='rbf')
rbf svm.fit(X train, y train)
rbf y pred=rbf svm.predict(X test)
print("RBF Kernel SVM:")
print("Accuracy:", accuracy score(y test, rbf y pred))
cm = confusion matrix(y test, rbf y pred)
sns.heatmap(cm, annot=True, fmt='d',cmap="Blues")
plt.title('Confusion Matrix for RBF Kernel SVM')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
print(classification report(y test, rbf y pred))
linear svm = SVC(kernel='linear')
linear svm.fit(X train, y train)
linear y pred = linear svm.predict(X test)
print("\nLinear Kernel SVM:")
print("Accuracy:", accuracy score(y test, linear y pred))
cm = confusion matrix(y test, linear y pred)
sns.heatmap(cm, annot=True, fmt='d',cmap="Blues")
plt.title('Confusion Matrix for Linear Kernel SVM')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
print(classification report(y test, linear y pred))
letter=files.upload()
df2=pd.read csv("letter-recognition.csv")
df2.head()
X = df2.drop('letter', axis=1)
y = df2['letter']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
svm classifier = SVC(kernel='linear', probability=True)
svm classifier.fit(X train, y train)
y pred = svm classifier.predict(X test)
print("Accuracy:", accuracy score(y test, y pred))
print(classification report(y test, y pred)
cm = confusion matrix(y test, y pred)
plt.figure(figsize=(10,10))
sns.heatmap(cm, annot=True, fmt='d', cmap="Blues")
plt.title('Confusion Matrix for SVM')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
lb = LabelBinarizer()
lb.fit(y test)
y test lb = lb.transform(y test)
```

```
y_pred_prob = svm_classifier.predict_proba(X_test)
fpr = \{\}
tpr = \{\}
thresh = \{\}
roc auc = dict()
n class = y test lb.shape[1]
for i in range(n class):
   fpr[i], tpr[i], thresh[i] = roc_curve(y_test_lb[:,i], y_pred_prob[:,i])
  roc auc[i] = auc(fpr[i], tpr[i])
plt.plot(fpr[0], tpr[0], linestyle='--',color='orange', label='SVM (AUC = %0.2f)' % roc_auc[0])
plt.title('ROC Curve for Class 0')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive rate')
plt.legend(loc='best')
plt.show()
print(f"AUC score for class 0: {roc auc[0]}")
```

Program 8 Implement Random forest ensemble method on a given dataset

Screenshot:

Screensnot:			
100000	PAGE NO: 90 DATE: 05 05 25	15)	PAGE NO: 21 CATE:
Lab- o F	100121 31/6		2. Sande Si
Random Forest 1) Sample SI		-	SNO CAPA THEOCHERS Comm Hacker Johns , K9 No Moduate Good Yes
The CGPA Interactiveness comm. St. 1 29 Yes Good 2 29 No Moderat 3 29 No Moderat 4 29 No Moderat	Good Yes e Good Yes a Average No	Selection of the select	2 29 No Moderate Arg No 3 29 No Moderate Arg No 4 29 Yes Moderate Good Yes 5 29 Yes Moderate Good Yes
5 29 Yes Modera Entropy (si) = -3 log 3 - 5 92 5	2 log, 2 = 0.971 5 5	71	Entropy (5,1)3 log 3 - 2 log 2 2 2 0 171 5 5 5 5 5 Entropy (5 Attactiveneu)1 log 1 - 2 log 2 2 3 3 3 3 3
Entropy (29) 2 log, 2 - 4 Entropy (29) - 0 (pure Nos) TG (GPA) - 0.971 - (4)	et)	0.8	7 G (Interchycoum) = 0.971 - (2 x0 + 3 x 0.918).
= 0.171	5 /		Enteractiveners Yes
Desision Tree (GPA? 29 (Joboffin: Yu)	79 Jobolfis: Yes		Practical Good Knowlege Tob offer: Yes Doboffer: Yes
3	Heractivenen: No Yes: Yes Joboffes: No	150.1	What is the best ace Apre & confusion motion of classifier you observed & viring how many trees' Best accuracy: 1.000 with 1 tree
1000			Confusion Matrix: (10 8 0)

```
from google.colab import files
iris=files.upload()
dfl=pd.read_csv("iris (4).csv")
dfl.head()
X = dfl.drop('species', axis=1)
y = dfl['species']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
rf_classifier = RandomForestClassifier(random_state=0)
rf_classifier.fit(X_train, y_train)
y_pred = rf_classifier.predict(X_test)
default_accuracy = accuracy_score(y_test, y_pred)
```

```
print(f"Accuracy with default n estimators: {default accuracy}")
best accuracy = 0
best n estimators = 0
for n estimators in range(1, 101):
  rf_classifier = RandomForestClassifier(n_estimators=n_estimators, random_state=0)
  rf classifier.fit(X train, y train)
  y pred = rf classifier.predict(X test)
  accuracy = accuracy_score(y_test, y_pred)
  if accuracy > best accuracy:
    best accuracy = accuracy
    best n estimators = n estimators
print(f"\nBest accuracy: {best accuracy} achieved with n estimators = {best n estimators}")
cm = confusion matrix(y test, y pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
       xticklabels=np.unique(y test), yticklabels=np.unique(y test))
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

Implement Boosting ensemble method on a given dataset

Lab-	oost Algor	iltm		
_			Almostha	For the
Consid	loring Had	7.40	Show	the Decision
tollow	Calculation	Alens	for the	attribut.
stump	OA CATCO IATIONY	, , , ,	-	
	IPA .			
CGPA	Interactivener	> Practical	Skills	Job Profile
29	Yes	Good	Good	40
29	No	Good	Moderate	Yes
29	No	Avg	Moderate	de
∠9 ≥9	No	Aug	Good	20
	Yes	Good	Moderate	Yes
29	Yes	Good	Moderate	Yes
Car by	server el.	TT TVO -	La bank Hard	100 P 17
Toitial	weight (Ji) - 1		
100:00	as almas a	CGPA		
	GPA 7= 9 -			
i6 (GPA <9 ->	No		
	ted essor		ation	
	12+W3 - 1			33
				The state of the s
	ting model		(alpha)	
d-	1 10 (1- 8		
			>	THE STREET
	- 12 10	0.667		
No. of the last of	BERTHAND OF THE		March 18	danie -
-	7 101	2)	Section would	V317-12-10
	2 0.3466		A Commence of the	D 40 301 100
	~ 0.3466		Master and	or will be a second

00 1000 1000	PAGE NO: DATE:
2CGPA - 1 x4xe-0.3466+	1 x 2x e0.3466
= 0.9428	
2. Best accuracy score & Co & how many trees? Best accuracy score = 0. with N estimation = 80 confusion matrix = [7136]	3

```
from google.colab import files
income=files.upload()
dfl=pd.read csv("income.csv")
df1.head()
X = dfl.drop('income level', axis=1)
y = df1['income level']
X = pd.get dummies(X)
X train, X test, y train, y test = train test split(X, y, test size=0.3, random state=42)
abc = AdaBoostClassifier(n estimators=10, random state=42)
abc.fit(X train, y train)
y_pred = abc.predict(X test)
accuracy = accuracy score(y test, y pred)
print(f"Initial AdaBoost accuracy (10 trees): {accuracy}")
param grid = {'n estimators': [50, 100, 150, 200]}
grid search = GridSearchCV(AdaBoostClassifier(random state=42), param grid, cv=5, scoring='accuracy')
grid search.fit(X train, y train)
print(f"Best parameters: {grid search.best params }")
print(f"Best cross-validation score: {grid search.best score }")
best abc = grid search.best estimator
y pred best = best abc.predict(X test)
best accuracy = accuracy score(y test, y pred best)
print(f"Accuracy of the best model on the test set: {best accuracy}")
cm = confusion matrix(y test, y pred best)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
       xticklabels=['<=50K', '>50K'], yticklabels=['<=50K', '>50K'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

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

DATE: 12 5/25	DATE:
lab-02	Now dustro are:
Build & means algorithm to cluster a set of	(1 (R1, P2) & (2(P3, P4, P5, R6, R7)
Build k means algorithm to contile.	The state of the same of the s
data stored in a territoria	new controids are:
To the wine kinear also	$\frac{(1-2.5)}{2}$ $\frac{3}{2}$ $\frac{(2-19.5)}{5}$ $\frac{25.5}{5}$
Tor clustering initial clusters centre as	
(1,1) ELST) Execut for 2 Herations	C1 = 1.25, 1.5 C2 - 3.9, 5.1
(1,0 48.)	
Tetration:	1. For "ins. d csv" dataset:
record non clusteriles clusteriles assign to	The ellow plat (interior 45k) shows a
	sharp "ellor" at k=3, indicating that
R.(1,1) 0 7.21 C1	clusters ?8 the optimal choice for the
Ro (1.5.2) 1.12 6.12 C1	ins dataset.
23(3.4) 3.61 (1	S. P. C.
Pu(5,7) 7.21 0.0 (1	
Rs (3.5,5) 4.12 9.5 (2	
R6 (4.5.5) 3.31	The second second second
R7 (35.45) 4.30 2.92 C2	
Week Williams	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
New antroids are:	10-20-4122-012-4
C1 - 5.5 7.0 C2 = (6.5 21.5	
	10 00 00 00
C1 - 1.83, 2.33 C2 = 4.12, 5.37	1
[Heration2: (1.83, 2.33) (4.12, 5.31)	
record num Closeto (1 close to (2 assignt a cluster	1 (1 ()) (1 () () () () () ()
R. R. \$ 1.57 5.37 CI	
R2 0.47 4.27 (1	
R3 2.04 1.77 C2	The state of the s
Ru 5.64 1.85 C2	
Rs 3-15 0.72 C2	
RC 3.78 0.52 (2	
R7 1.74 1.07 (2	

```
Code:
from google.colab import files
iris=files.upload()
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model selection import train test split
from scipy import stats
import seaborn as sns
from sklearn.model selection import train test split
from sklearn.metrics import accuracy score
from sklearn.metrics import classification report, confusion matrix, accuracy score
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
df1=pd.read csv("iris (4).csv")
```

```
df1.head()
df = df1.drop(['sepal length','sepal width','species'],axis=1)
```

```
scaler = StandardScaler()
scaled df = scaler.fit transform(df)
wcss = []
for i in range(1, 11):
  kmeans = KMeans(n clusters=i, init='k-means++', max iter=300, n init=10, random state=0)
  kmeans.fit(scaled df)
  wcss.append(kmeans.inertia)
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
kmeans = KMeans(n clusters=3, init='k-means++', max iter=300, n init=10, random state=0)
pred y = kmeans.fit predict(scaled df)
df['cluster'] = pred y
plt.scatter(df['petal length'], df['petal width'], c=df['cluster'])
plt.title('Clusters of Iris Flowers')
plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
plt.show()
```

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

1ah-10	
Principle Component Analysis (PCA)	2- (-4.305) 3.736
	5.693
- Reduce the dimension from 2 to 1 very	-5.124
PLA compute it principle component	[-5.(1-1)
Ocda magir!	1. For "heart Csv" dateset:
X1 4 8 13 7 X2 11 4 5 14	1. 40 Mast. SV amese
	& According before PCA:
1 30.3849 e1 - (0.5574) e2 - (0.8303)	logistic regression: 0.9016
1 = ((15) -0.8303) (0.5574)	SVM: 0.8525
	Random Forest: 0.8361
Mean of X1 = 4+8+13+7-80	
Mean of XI = u	* According after PLA:
	* Accoracy after PLA: logistic regression: 0.8689 (VM: 0.8689
Mean of x2 - 11+4+5+14 - 8.5	(VM: 0.862)
	Random Forest: 0.8852
Xuntred - X - Mean4 0 5 -1 2 5 -4-5 -3.5 5.5	
2.5 -4.5 -3.5 5.5)	
Since 1: is larger, P. is 1st principle comparet	
e, = [0.5574, -0.8503]	
let zi = ei - 2i	
21 -4, 8.5) = 0.5574 (-4) + (-0.8 303) (2-5)	
-4 20555	
72(0,-4.5) = 0.5574(0) + (-0.8303) (-4.5)	
- 3.7 3635	
23(5, -3.5) = 0.5574(5) + (-0.7363)(-3.5)	
- 5.69305	
Z4(-1,5.5)=0.5574(-1)+(-0.303)(5.5)	
= -5.12405	

Code: from google.colab import files heart=files.upload() import pandas as pd import numpy as np import matplotlib.pyplot as plt from sklearn.model_selection import train_test_split from scipy import stats

import seaborn as sns from sklearn.preprocessing import LabelEncoder, OneHotEncoder

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy score

from sklearn.metrics import classification report, confusion matrix, accuracy score

from sklearn.preprocessing import StandardScaler

from sklearn.svm import SVC

from sklearn.linear model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.decomposition import PCA

```
dfl=pd.read csv("heart (1).csv")
dfl.head()
text cols = df1.select dtypes(include=['object']).columns
label encoder = LabelEncoder()
for col in text cols:
  df1[col] = label encoder.fit transform(df1[col])
print(df1.head())
X = dfl.drop('HeartDisease', axis=1)
y = df1['HeartDisease']
X train, X test, y train, y test = train test split(X, y, test size=0.2, random state=42)
scaler = StandardScaler()
X train = scaler.fit transform(X train)
X \text{ test} = \text{scaler.transform}(X \text{ test})
# Support Vector Machine
svm model = SVC(kernel='linear', random state=42)
svm model.fit(X train, y train)
svm predictions = svm model.predict(X test)
svm accuracy = accuracy score(y test, svm predictions)
print(f"SVM Accuracy: {svm accuracy}")
# Logistic Regression
lr model = LogisticRegression(random state=42)
```

```
lr model.fit(X train, y train)
lr predictions = lr model.predict(X test)
lr accuracy = accuracy score(y test, lr predictions)
print(f"Logistic Regression Accuracy: {lr accuracy}")
# Random Forest
rf model = RandomForestClassifier(random state=42)
rf model.fit(X train, y train)
rf predictions = rf model.predict(X test)
rf accuracy = accuracy score(y test, rf predictions)
print(f"Random Forest Accuracy: {rf accuracy}")
models = {
  "SVM": svm accuracy,
  "Logistic Regression": lr accuracy,
  "Random Forest": rf accuracy
best model = max(models, key=models.get)
print(f"\nBest Model: {best model} with accuracy {models[best model]}")
pca = PCA(n components=0.95)
X train pca = pca.fit transform(X train)
X test pca = pca.transform(X test)
sym model pca = SVC(kernel='linear', random state=42)
svm model pca.fit(X train pca, y train)
svm_predictions_pca = svm_model_pca.predict(X test pca)
svm accuracy pca = accuracy score(y test, svm predictions pca)
print(f"SVM Accuracy (with PCA): {svm accuracy pca}")
lr model pca = LogisticRegression(random state=42)
lr model pca.fit(X train pca, y train)
lr predictions pca = lr model pca.predict(X test pca)
lr accuracy pca = accuracy score(y test, lr predictions pca)
print(f"Logistic Regression Accuracy (with PCA): {lr accuracy pca}")
rf model pca = RandomForestClassifier(random state=42)
rf model pca.fit(X train pca, y train)
rf predictions pca = rf model pca.predict(X test pca)
rf accuracy pca = accuracy score(y test, rf predictions pca)
print(f"Random Forest Accuracy (with PCA): {rf accuracy pca}")
models pca = {
  "SVM": svm accuracy pca,
  "Logistic Regression": lr accuracy pca,
  "Random Forest": rf accuracy pca
}
best model pca = max(models pca, key=models pca.get)
print(f"\nBest Model (with PCA): {best model pca} with accuracy {models pca[best model pca]}")
```