Practical No: 01

* Aim :-

Familiarization with Numpy, Panda, and matplotlib by loading dataset in Python.

* Theory :-

1] Numpy Library :-

- Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with arrays. It is the fundamental package for scientific computing with python.
- one of the most popular open-source python packages. Numpy focuses on scientific and mathematical computation. It makes it easy to work with large matrices & multidimensional data.

2] Pandas Library :-

- It is a powerful and versatile library that simplifiers the tasks of data manipulation in python.
- Pandas is an open-source library authorized under BSD. This well-known library utilized in the field of data science.

Date

- Pandas is well suited for working with tabular data, such as spreed sheets or SQL tables.

3] Matplotlib library :-

- This library is responsible for the plotting of numerical data. It is utilized in data analysis for this reason.
- Matplotlib is a plotting library for python that provides a comprehensive set of that provides a tool for creating high quality 2D & 3D plots.

4] Scipy library :-

- Scipy is a library for scientific computing in python.
- This program demonstrates linear regression using 'linregress()' function.
- It provides a function for scientific and engineering application.
- It is for scientific computing, information processing, and high-level computing are primary uses for this open source library.

* Program:-

Code 1:- Numpy :

```
import numpy as np

arr = np.array ([20,30,40])
print ("Array with Rank 1: \n", arr)

arr = np.array ([[10,20,30],
                 [40,50,60]])
print ("Array with Rank 2: \n", arr)

arr = np.array ((10,30,20))
print ("\n Array created using"
       " passed tuple : \n", arr)
```

output:-

```
Array with Rank 1:
[20 30 40]
Array with Rank 2:
[[10 20 30]
 [40 50 60]]
Array created using passed tuple:
[10 30 20]
```

Code 2:- panda

```
import pandas as pd
import numpy as np.
ser = pd.Series ()
```

```
print ("Pandas Series:", ser)
data = np.array (['g', 'e', 'e', 'k', 's'])
ser = pd.series (data)
print ("Pandas Series:\n", ser)
```

output :- Pandas series : Series ([], dtype: float64)

Pandas series :

0	g
1	e
2	e
3	k
4	s

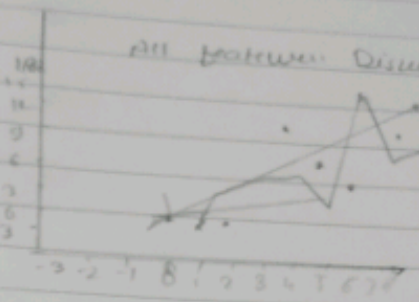
dtype : object

Code 3 :- matplotlib

```
import matplotlib.pyplot as plt
a = [1, 2, 3, 4, 5]
b = [0, 0.6, 0.2, 15, 10, 8, 16, 20]
plt.plot (a)
plt.plot (b, "Or")
plt.plot (list (range (0, 22, 3)))
plt.xlabel ('Day ->')
plt.ylabel ('Temp ->')
c = [4, 2, 6, 8, 3, 20, 13, 15]
plt.plot (c, label = '4th Rep')
ax = plt.gca ()
ax.legend (['1st Rep', '2nd Rep', '3rd Rep'])
```

```
plt.annotate('Temperature v/s Day',)
plt.title('All features Discussed')
plt.show()
```

O/P:-



Code 4:- Scipy

```
from scipy import sparse
A = sparse.lil_matrix((1000,1000))
print(A)
A[0,:100] = np.random.rand(100)
A[1,100:200] = A[0,:100]
A.setdiagr(np.random.rand(1000))
print A
```

O/P:-

```
(1,130)    0.6315
(1,131)    0.44
(1,132)    0.57
(1,133)    0.93
(1,134)    0.22
(1,135)    0.79
```

Result: Hence, In python, the implementation of Numpy, panda, matplotlib by Loading dataset are executed successfully.

Verified 5/7/24 AP

Date

Practical No. 2

* Aim :- Implement and demonstrate find-s Algorithm Read training data from a CSV file.

* Theory :-

- find-s Algorithm :

→ It is simple machine learning algorithm used for finding most specific hypothesis that fits training data.

→ It works with binary classification problems where each instance is either a positive or negative example.

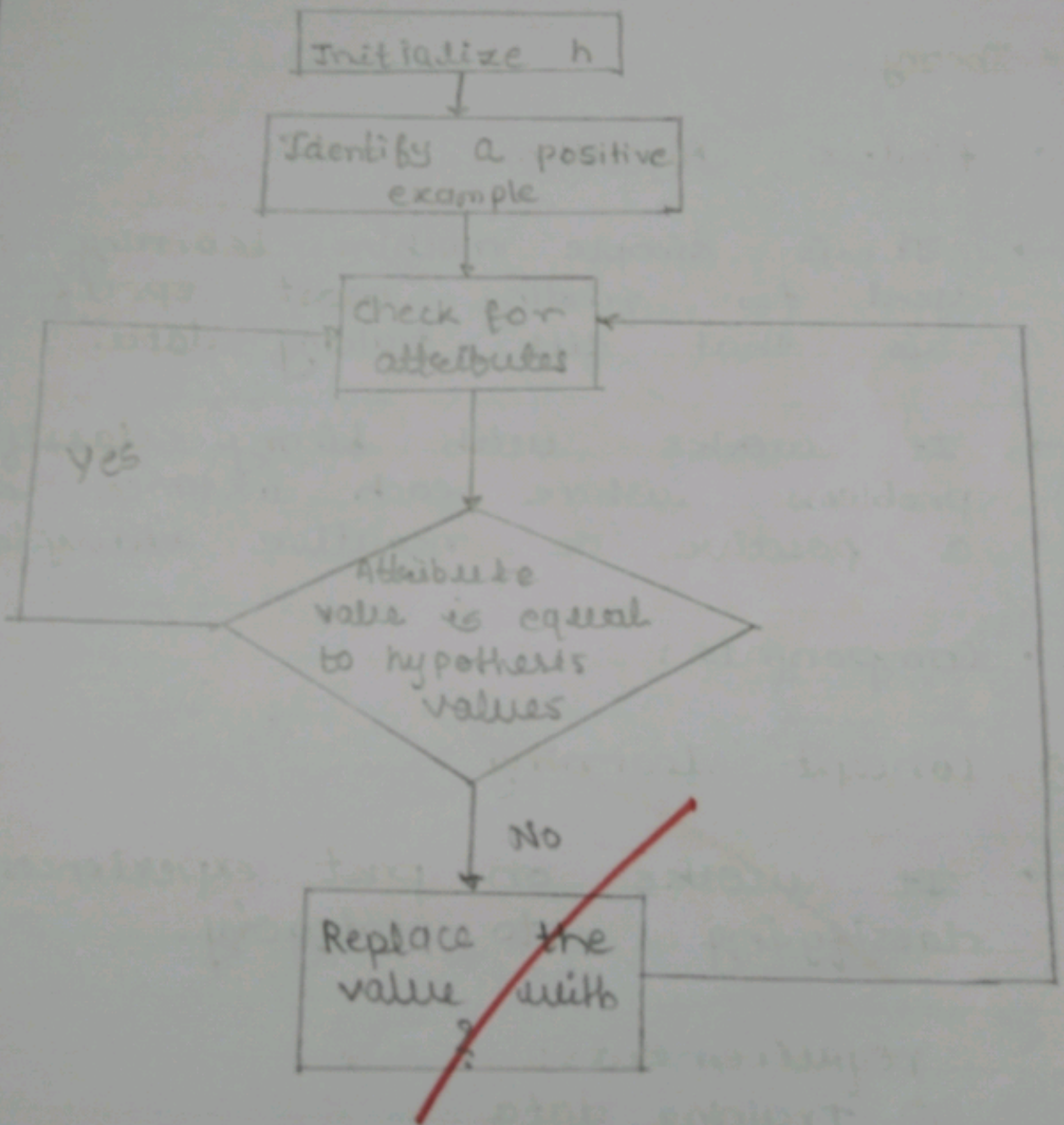
- Components :-

① Concept learning :

→ It works on past experiences and classifying into category.

requirements :-
① Training data
② Target concept
③ Actual data objects

* Flowchart of finds Algorithm :



Date :

② General Hypothesis :

→ It is an explanation for something It states relation ship betⁿ major concepts.

③ Specific Hypothesis :

→ If It fills in all the important details about variables given in general hypothesis

$S = \{ \phi, \phi, \phi, \phi, \ldots\ldots \phi \}$

* Algorithm :-

① Initialize Hypothesis :

→ start with most specific hypothesis, e.g, ' $h = (?, ?, \ldots\ldots, ?)$, where ? represents any h value.

② Iterate through examples :

→ for each positive example (x, y) :
  → Update ' h ' to match positive example by replacing '?' in 'h' with corresponding attribute value from 'x'.

→ Negative examples are ignored.

Page No. 59

⑤ Output: The final hypotheses 'h' represents most specific hypothesis consistent with all positive examples.

* Output:- Data frame:

	Name	branch	result	mood	beh.	color	mat.	above 18
0	rohit	AI	pass	happy	good	fit		yes
1	komal	CS	pass	happy	good	fit		yes
2	Palak	EE	fail	unhappy	bad	unfit		no
3	dhan.	ETC	pass	unhappy	good	fit		yes
4	vinay	CS	pass	happy	bad	unfit		no
5	aditya	CI	fail	unhappy	bad	unfit		yes
6	harshal	AI	pass	unhappy	good	fit		no
7	piyush	EE	fail	happy	bad	unfit		yes
8	tushar	ETC	pass	happy	good	unfit		no
9	aditya	AI	fail	unhappy	good	fit		yes
10	nikhil	ETC	pass	happy	bad	fit		no

Attribute:-

[['komal' 'CS' 'pass' 'happy' 'good' 'white' 'fit']
['Palak' 'CI' 'fail' 'unhappy' 'bad' 'black' 'unfit']
['aditya' 'EE' 'pass' 'unhappy' 'good' 'white' 'unfit']
['dhananjay' 'ETC' 'Fail' 'happy' 'bad' 'black' 'fit']
['vinay' 'CS' 'pass' 'unhappy' 'bad' 'balek' 'Fit']
['aditya' 'CI' 'fail' 'unhappy' 'good' 'white' 'unfit']
['harshal' 'AI' 'fail' 'happy' 'bad' 'White' 'unfit']
['piyush' 'EE' 'pass' 'unhappy' 'good' 'black' 'until']
['nikhil' 'ETC' 'pass' 'happy' 'bad' 'White' 'fit']]

The target is ['yes' 'no' 'yes' 'no' 'yes' 'no' 'yes' 'no']

* Result:- Hence, the Find-S algorithm implemented and demonstrated successfully

Page No.

Date

Practical no: 08

*Aim:- Demonstrate various data pre-processing techniques and plot density plot, box plot using suitable datasets.

* Theory:-

① Creating database:

```
import pandas as pd
stu = ["abc", "xyz", "rst", "cmn"]
yr = ["1", "1", "3", "3"]
dict = {'student': stu, "year": yr}

df = pd.DataFrame(dict)
print(df)
```

O/P:-

	student	year
0	abc	1
1	xyz	1
2	rst	3
3	cmn	3

② Connecting into CSV files:-

→ To connect a database frames into a CSV file using python's pandas library, we can use to to.CSV method.

Command: df.to_csv('pandas.csv')

Date :

③ Loading a database :-

→ Loading a database into a pandas data-frame typically involves using a data-base connector and the pandas library

```
df = pd.read_csv ('Pandas csv')
print (df.head ())
```

Unnamed : 0	0	student	year
0	0	abc	1
1	1	xyz	1
2	2	rst	3
3	3	lmn	3

④ Getting information of database in detail :-

→ It includes using a single command in python i.e. 'info'.

```
df.info()
<class 'pandas.core.frames.DataFrame'>
Range index : 4 entities, 0 to 3
Data columns (Total 3 columns)
```

*	column	not-null count	Dtype
0	unnamed : 0	4 - not null	int 64
1	student	4 - not null	object
2	year	4 not null	int 64.

Date

dtypes : int 64 (2), object (1)
memory usage: 224.04 bytes

⑤ To get summary of the database :-

→ Knowing the detailed description of the database including using a sing command "describe()."

dt.describe().

	Unnamed : 0	year
count	4.00000	4.00000
mean	1.50000	2.00000
std	1.290994	1.154701
min	0.000000	1.00000
25%	0.7500000	1.00000
50%	1.50000	2.00000
75%	2.250000	3.00000
max	3.00000	3.00000

⑥ To know about null values :-

→ It present in the database includes using the command is "null().sum()".

command : df.isnull().sum().

Page No. 14

Date :

df. is null(). sum()

Unnamed : 0	0
student	0
year	0

d type : int 64

④ To Know about training & testing data

→ Segragating data gives specifically the data present in the training part and the amount of data present in testing part following is the method.

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.05, random_state = 0)

ratio = 0.75
total_rows = df.shape[0]
train_size = int(total_rows * ratio)
train = df[0 : train_size]
test = df[train_size :]

print(train)
```

	Unnamed : 0	student	year
0	0	abc	1
1	1	xyz	1

Date :

2

2 rst 3

print (test)

unnamed : 0 student year
3 3 Unn 3

* **Result** :- Hence, various data pre-processing techniques & plot density plot in Python is done successfully.

Verified

Date

Practical No. 4.

* Aim :- Demonstrate Dimensionality reduction using principle component analysis (PCA) method.

* Theory :

* Dimensionality Reduction :

- As a number of dimensions increased, the volume of the feature space grows exponentially, leading to sparce data, which makes it difficult for models to generalize.

- Dimensionality reduction is the transformation of data from a high-dimensional space into a low-dimensional space.

* Principle Component Analysis (PCA):
- principle component analysis is a dimensionality reduction method that is often used to reduce the dimensionality of large dataset.

* Step-by-step Explanation of PCA:

Step 1 :- Standerdization.

The aim of this step is to standerdized the range of continous initial variables so that each one of them contributes equally to analysis.

Mathematically, $z = \frac{value - mean}{S.d.}$

Page No. 18

Step2:- Covariance Matrix computation.
— The aim of this step is to understand how the variables of the input data set are very from mean with respect to each other.

$$Co(x, y) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$$

Step3 :- Calculate Eigenvalues & Eigenvectors: one you have the covariance matrix, the next step is to calculate the eigenvalues & eigenvectors. Eigenvectors represent directions (principle components) of the new feature space.

Step4:- Sort Eigenvalues & Eigenvectors:

- Sort the eigenvalues in descending order and rearrange the corresponding eigenvectors. The eigenvectors with highest eigenvector are principle components that capture the most variance in the data.

Step5:- Select principle components.

- Decide the numbers of principle components you want to keep. Usually, the number of components is chosen based on the cumulative explained variance. which indicate how much of total variance is retained by selected components.

* Cumulative Explained variance : The proportion of the total variance captured by the first k principal components.

Step 6 :- Transform the data :
- finally tranform the original dataset into the new space defined by the selected principal components. This step involves multiplying the original data by the matrix of selected eigenvectors.

$$Z = X \cdot W$$

where,
Z is the transformed dataset
X is the original standardized data.
W is matrix of selected eigenvectors.

Step 7 :- Interpret the Result :

- The resulting tranformed dataset consists of new features that are uncorrelated & ordered by the amount of variance they explain.

* Result :- Program has been executed successfully.

Date :

Practical No. 5.

* Aim:- Implement the multiple linear regressions algorithm in order to fit data points.

* Theory:

• Multiple linear regression :

In this, previous topic, simple linear regression, where a single independent / Predictor (x) variable is used to modes the response variable is used to model the response variable (Y). But there are may be various cases in which the response variable is affected by more than one predictor variable.

We assume that y is linearly dependent on the factors according to.

$$Y = \beta 0 + \beta 1 x 1 + \beta 2 x 2 + \cdots + \beta k x k + \varepsilon$$

- The variable y i is dependent or predicted
- The slope of y depends on the y-intercepts.
that is, when x_i & x_2 are both zero.
y will be 10.

* Multiple Regression working process :

1. Data Preparation : Gather and preprocess the data, including handling missing values,

encoding categories variables, and scaling the features if necessary.

2. Splitting the Dataset: Divide the dataset into a training set & a test set to evaluate the model's performance on unseen data.

3. fitting the model: Use the training set to fit the multiple linear regression model by finding the best coefficients $B_0, B_1, B_2, \ldots, B_n$ that minimize the sum of squared residuals (errors)

4. Making Predictions: Use the fitted model to predict the dependent variable on the test set.

5. Evaluating the model: Assess the model's performance using matrices such as R-squared, Mean Absolute Error (MAE), Mean squared Errors (MSE), and Root Mean squared Error (RMSE).

* Result: Hence, a implementation of multiple linear regression algorithm is successfully.

Model A

Date :

Practical No. 6.

* Aim :- Apply K-nearest Neighbor algorithm to classify the data set.

* Theory :- K- Nearest Neighbor (KNN) Algorithm for machine learning.

- K- Nearest Neighbor is one of the simplest Machine learning algorithms based on Supervised learning technique.

- K-NN algorithm assumes the similarity between the new case / data into the category that is most similar to the available categories.

* How does K-NN work :

- The K-NN working can be explained on the basis of the below algorithm.

- step 1 : Select the number K of the neighbors.

- step-3 : Take the k nearest neighbours as per the calculated Euclidean distance.

- step-4 : Among these k neighbours, count the numbers of the data points in each category.

Page No. es

Date

- step 6 : Assign the new data points to that category for which the no. of the neighbor is maximum.
- step 6 : Our model is ready.

* How to select value of K.

- A very low values for K such as K=1 or K=2, can be noisy & lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.

* Distance Matrics in KNN

1. Euclidean Distance : The most commonly used distance matrix that measures the "straight line" distance between two points in Euclidean space.

$$d(x, y) = \sqrt{\sum (x_i - y_i)^2}$$

* Key features of KNN :

- Non-Parametric : KNN does not make any assumptions about the underlying data distribution, making it flexible for various type of datasets.

Date

- lezy learning : KNN is a lazy algorithm because it does not learn a discriminative memorizes the teaching dataset. Predictions are made only when the query sample is provided.

* Result : Hence the implementation of k-nearest neighbours algorithm successfully.

Practical A

Date

Practical No. 7.

* Aim:- Demonstrate the working of the decision tree based ID3 algorithm.

* Theory:-

- The ID3 algorithm (Iterative Dichotomiser) is a popular decision tree algorithm used for classification tasks.

- The ID3 of builds the decision tree by recurrively partiontioning the dataset based on the attribute that results in the maximum information gain.

* Steps in ID3 Algorithm :

1. Entropy calculation:
   - Entropy measures the impurity or uncertainty in the dataset.

formula:-

$$\text{Entropy}(s) = -\sum P_i \log_2(P_i)$$

- lower entropy means less randomness, and higher entropy means are disorder in the data.

2. Information Gain:
   - Information Gain (IG) helps select the attribute that best splits the data by reducing entropy the most.

Date

$IG(S,A) = Entropy(S) - \sum \left(\frac{|S_v|}{|S|} * Entropy(S_v)\right)$

- The attribute with the highest information gain is selected as the root or interested node.

3. Tree construction:

- Recursively partition the dataset based on the attribute with the highest information gain until:
  - All instances belong to the same class.
  - There are no more attributes to split.

4. Stopping Criteria:

- when no. more attributes are available, or when all instances in the subset have the same label.

* Application:
  - used in classification problems such as medical diagnosis, customer segmentation, and more.

Result :- Hence, a program executed successfully.

Page No. 30

Date

Practical No. 08

* Aims- Implement a Bayesian network algorithm Considering medical data. weather data.

* Theory :-

- Naive Bayes algorithm

It is the one of the simple and most effective classification algorithm which help in building fast model that can make swift predictions. it is the probabilistic classifier which mean it predict on the basis of probability of an object.

- Naive of is called Naive because it assume that the occurrence of certain feature is independed of occurence of other features.

- Bayes It is called Bayes because : It depends on principle of Bayes theorem.

Formula :- $$P(A|B) = \frac{P(B/A) \cdot P(A)}{P(B)}$$

- P(A) is priory of 'A' : a probability of event before evidence seen.

- P(B/A) is probability the likelihood that hypothesis will come to based on evidence.

Date

- P(A/B) is Posterior probability the likelihood and that a hypothesis will come to based on outcomes.
- Types :-
  1) Gaussian Naive Bayes.
  2) Multinomial Naive Bayes.
  3) Bernoulis Naive Bayes.

* Advantages :-
  - Simple & fast.
  - Works well with high-dimensional data.
  - Efficive for that classification.

* Disadvantages.
  - Independent Assumtion.
  - Zero probability problem.

* Results - Hence, we perform practical successfully.

Practical A

Date :

Practical No. 9.

* Aim:- Implement & demonstrate the K-means algorithm.

* Theory:-

K-mean clustering is an algorithm that is used to solve the clustering problems in machine learning or data science. It is an unsupervised learning algorithm, which groups the unlabeled dataset into different clusters.

Here K defines the no. of pre-defined clusters that need to be created in the process, as if K = 2, there will be two clusters, & for K = 3, there will be three clusters, & so on.

The algorithm takes the unlabeled dataset as input, divides the dataset into k-no. of clusters, & repeats the process until it does not find the best clusters.

- The K-means clustering algorithm mainly performs two tasks :-

(i) Determines the best value for K centroid points or centroids by an iterative process.

(ii) Assigns each data point to its closest K-center. Those data points which are near to the particular K-center, create a cluster.

Date

* How does the K-means algorithm work ?

step1 :- Select no. K to decide no. of clusters.

step2 :- Select random K points or centroids.

step3 :- Assign each data point to their closest centroid, which will form the predefined K. clusters.

step4 :- Calculate the variance and place a new centroid of each cluster.

step5 :- Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

step6 :- If any reassignment occurs, then go to step-4 else go to FINISH.

step7 :- The model is ready.

Its objective is to divide data into clusters, making similar data points part of the same group

* Result :- Hence, the implementing the K-means algorithm successfully.

Marks A

Page No. 26

Date

Practical No. 10.

* Aim:- Write a python program for to check weather email is spam or not.

* Theory:-

- Spam detection is the process of identifying & filtering unsolicited and potentially harmful emails, typically referred to as "spam". Machine learning, natural language processing (NLS). and statistical techniques are commonly used to classify emails as "spam" or "not spam" based on patterns in the email content, metadata, or subject line.

- In a spam detection system, we use a dataset of labeled emails either spam or not) to train a machine learning model. The model learns patterns into words, phrases, and structures of these email to predict the probability that new, unseen emails are spam.

- Key techniques include text processing removing stop words, punctuation, tokenization converting text to numerical format using method like bag of words or TF IPf). and classification algorithms. like naive Bayes, support vector machine (SVM) or neural networks.

Date :

* Key components :

① Data Processing : Text is converted to lower case. Punctuation is removed.

② Vectorization :
- CountVectorizer converts the text into a matrix of token counts (Bag of words).

③ Model training :
- The multinomial NB classifier is trained on the vectorized email data.

④ Prediction :
- The model is used to predict whether new emails are spam or not.

⑤ Evaluation :
- The program calculates the accuracy of the model and provides a confusion matrix to evaluates the performance.

* Result :- Hence, a progeram executed successfully.

Result A