INDEX

Name Gunjal Kothari Sub.

Std.: Machine Learning Lab Div. Roll No.

Telephone No. E-mail ID.

Blood Group. Birth Day.

Sr.No.	Title	Page No.	Sign./Remarks
(Q1)	Write a python program to import & export.		
(Q2)	Demonstrate data processing techniques		
(Q3)	ID3 algorithm for decision Tree		
(Q4)	Linear Regression & Multiple Regression.		
(Q5)	Logistic Regression Model.		
(Q6)	SVM for a given dataset.		
(Q7)	K-Means algorithm to cluster data.		
(Q8)	Dimensionality Reduction using PCA.		
(Q9)	Neural Network with back propagation.		
(Q10)	Random Forest Ensemble		
(Q11)	Implement AdaBoost.		

WEEK - 1
LAB - 1

PAGE NO:
DATE: 5/4/24

(Q1) Write a python program to import & export data using Pandas library function.

```
import pandas as pd
#Read the CSV file
airbnb_data = pd.read_csv(r"/content/austinHousingData.csv")
#View the first 5 rows.
airbnb_data.head()
```

Output:

Index	zpid	city	streetAdd	zipcode	TaxRate	hasCooling
0	11389	pfluger-ville	14424 Lake Victoria	780	1.98	true
1	12401	pfluger-ville	1104 Dr. Strickling	313	2.30	true
2	11340	pfluger-ville	1408 Ford Dessau Road	490	1.45	false
3	10390	pfluger-ville	1025 Donna Jane Loop	910	0.90	true
4	18340	pfluger-ville	1380 Street Jane	830	0.85	true

PAGE NO :
DATE :

(Q2) Demonstrate various data preprocessing techniques for a given dataset.

Algorithm:

Import the data set using read_csv()
Identify and handle the missing data values
dataset.isnull().sum()

The gives the no. of null values in each column.

* Solution to handle null values :
1) Use Dropna is drop columns having high no. of null values
2) Use Fillna to replace a NULL values with a specified value.

Encoding categorical data using pd.get_dummies() which converts categorical data into dummy or indicator variables.

PAGE NO :
DATE :

(Q3) Use an appropriate data set for building the decision (03) and apply this to classify a new sample.

ID3 (iterative autocomiser 3) as a classic algorithm for building decision trees.

Input : Training dataset 'D' with 'm' examples and set of attributes A.

O/P : A decision tree T.

* PROCEDURE :

- If all the examples in D belong to the same class C, return a leaf node labelled with class C.
- If A is empty, return a leaf node labelled with a majority class in D.
- Otherwise choose the best attribute Abest from A to split D.
- Create a decision tree N using A abest.
- For each possible value V of abest

PAGE NO :
DATE :

1) Partition D into subset, D_V such that examples with value V from Abest are in subset D_V.

2) If D_V is empty add a leaf node to N labelled with majority in D.

3) Otherwise, add a subtree to N by recursively calling the ID3 algorithm with D_V and A - Abest.

4) Return the ~~iteration~~ decision tree :

Output :

(i) Highest info gain = 00.246 = outlook.

(ii) Highest Info gain = 0.971 = rainy

(iii) best attribute is windy.

PAGE NO:
DATE:

* 3KNN Classification

```
def get_user input ():
num_sample = int (input ("Enter the number
        of samples in your dataset:"))
num_features = int (input("Enter the number
of features in your dataset : "))

x = []
y = []

for (i in range (num_samples):
features = list (map ( float, input ( " Enter
features for sample {i+1} : " ) . split()));
x . append (features)
y . append ( label)

k = int (input ( " Enter the value of k
        for KNN :"))

knn = KNeighbours Classifier (n_neighbours = k)
knn. fit (X_train, y_train)

accuracy = accuracy_score( y_test, y_pred)
print ( "Accuracy of KNN model:
        {accuracy} ").
```

Output:

```
Enter the no. of samples : 2
Enter the no. of features : 2
Enter the values for features & labels:
Enter features for sample 1:
 1.00  2.00
Enter label for sample 1: 0
```

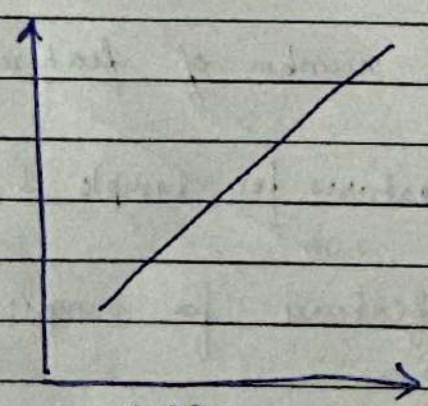
PAGE NO:
DATE:

(8) Linear Regression & Multiple regression.

Generate dataset:

```
np.random.seed(0)
X = 2 * np.random.rand (100,1)
y = 4 + 3 * np.random.randn (100,1)

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

model = LinearRegression()
model.fit ( X_train, Y_train)

y_pred = model.predict (X_test)

plt.scatter ( xtest, y_test, color='blue')
plt.title ( "Linear Regression")
plt.xlabel ('X')
plt.ylabel ('y')
plt.show

mse = mean_squared_error ( y_test, y_pred)
print ('Mean Square Error')
```

Output:



MSE : 0.91775324

PAGE NO:
DATE:

(2 *) Logistic Regression Model

```
def get_user_input():
num_samples = int(input("Enter the number of samples in your dataset :"))
num_features = int(input("Enter the number of features in your dataset:"))
x = []
y = []
print("Enter the features for the dataset & labels")
for (i in the range (num_samples):
features = list(map(float, input(f"Enter features for sample {i+1} : ") .split()))
label = int(input(f" Enter label for sample {1+i} :"))

x.append(features)
y.append(label)
return np.array(x), np.array(y)

accuracy = accuracy_score (y_test, y_pred)
print (f" Accuracy of the Logistic Regression Model : {accuracy} ")
```

* Output :

Enter the number of samples in your dataset : 3

Enter the number of features in your dataset : 5.

Enter the features for sample 1 : 10.00 2.00 6.00 4.00 1.00

Enter the features for sample 2 : 5.00 8.00 4.00 12.00 4.00.

PAGE NO :
DATE :

Enter the features for sample 3: 3.00 9.00 6.00 4.00
Enter the label: 0

Accuracy of the Logistic Regression Model: 1.0

PAGE NO:
DATE:

(Q) Build support vector machine for the following dataset.

Steps:

(i) Import the dataset to be trained (Iris dataset)

(ii) Convert the dataset into dataframes. Add the column (largest) to the dataframe.

(iii) Print the first few rows of dataframe to input the data. Create a scatter plot to visualize the relationship b/w sepal length and sepal width.

(iv) Split the dataset into training and testing sets. Use 70% for training 10% for testing.

(v) Predict the test labels. Use trained data classifier to predict labels for the test set.

Output:

Accuracy of the SVM classifier is 1.0.

y-pred: array([1, 0, 2, 1, 1, 0, 2, 1, 1, 2, 0, 0, 0, 1, 2, 1, 1, 0, 2, 2, 2, 2, 2, 0, 0,

10/10

PAGE NO :
DATE :

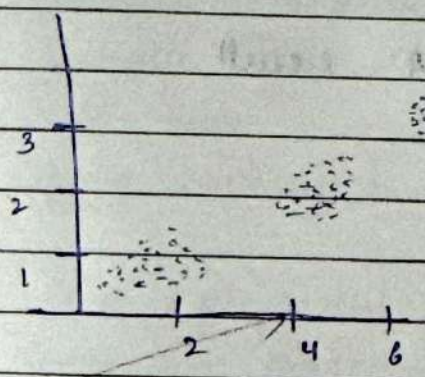
(Q) Build K-Means algorithm to cluster set of data :

Algorithm

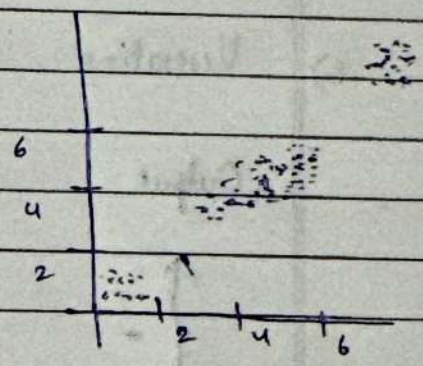
(i) Load the iris dataset.
(ii) Initialize and fit k-means model.
(iii) Visualize the clustering results.
(iv) Plot Original classification
(v) Plot K-means clustering results.

Output :

Real Cluster :



K-Means Cluster



PAGE NO :
DATE :

(Q) Implement dimensionality reduction using principle component analysis method:

Algorithm:

1) Import necessary libraries for data handling standardization, PCA and plotting:

2) Load the iris dataset.

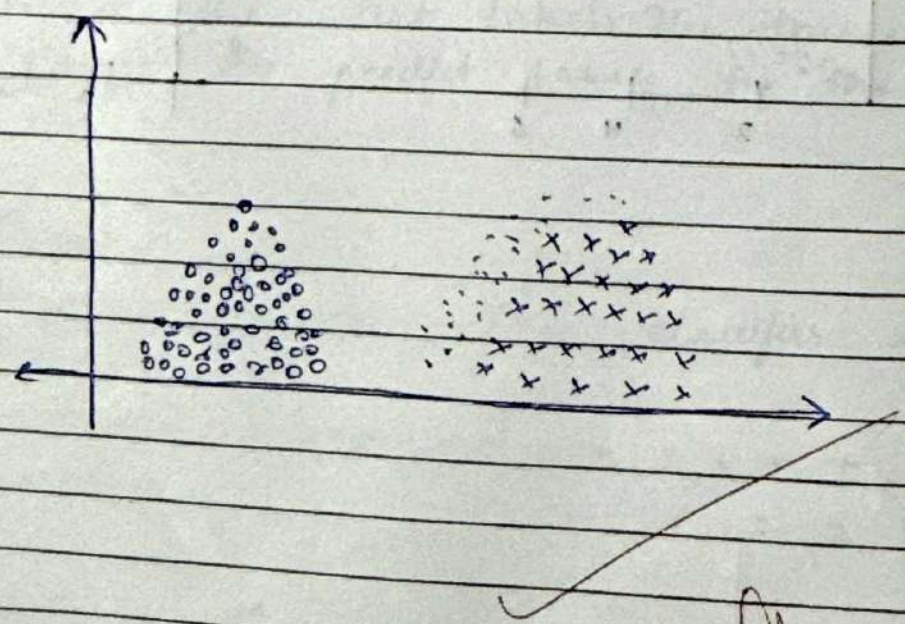
3) Standardize the data.

4) Apply PCA using the model.

5) Convert PCA result to dataframe.

6) Visualize the PCA result.

Output:



PAGE NO
DATE

(10) Build an artificial neural network model with back propagation

Algorithm

1) Initialize parameters:
- Normalize input feature matrix 'n'
- Normalize the output 'y'.
- Set hyperparameter : 'no. of epochs', no. of neurons'

2) Define activation functions:
- Sigmoid function adjustments

3) Training the network.
Forward approach
→ Compute I/P to hidden layer.
→ Add bias
→ Apply activation function

4) Back propagation :
→ Compute error
→ Compute gradient
→ Compute delta.

5) Update weights and biases.

Output : I/P [[0.6667 1]
[0.333 0.556]
[0.1 0.667]]

Actual Output : [[0.92] [0.86] [0.89]]

Predicted Output : [[0.3005687]
0.7939 38]
0.30112347]

PAGE NO:
DATE:

(Q2b) Implement Random Forest Ensemble Method:

Algorithm:

1) Import necessary libraries.
2) Load and inspect data.
3) Pre-process the data as in separating features and strength.
4) Split the data to training and test samples. Use 0.4 to allocate 40% of data to testing and use rest for training.
5) Initialize random first classifier and train it using 'fit' method.
6) Make predictions on test sample using method predict.
7) Evaluate the model

Output: Accuracy = 0.98

Confusion Matrix : $\begin{bmatrix} 23 & 0 & 0 \\ 0 & 19 & 0 \\ 0 & 1 & 17 \end{bmatrix}$

PAGE NO :
DATE :

(Q2b) Implement Boosting method

1. Import libraries
2. Load the dataset
3. Data preprocessing involves separation of features and dataset.
4. Split the dataset to train and test samples.
5. Intialize the dataset classifier with specified number of estimates and base estimators.
6. Train the model using the training dataset.
7. Make predictions for the test.
8. Evaluate the model.

Output:-

Metrics accuracy score: 0.9833