Student ID: 700747817

Name: Manikanta Venkata Kondasai Kalyan Badeti Parameswara

Machine Learning - Assignment IV

**Video Link:** <https://drive.google.com/file/d/1c-pqJ9F4OPYLaIBe80h48O5ZSi0Qr97o/view?usp=sharing>

GitHub Link: <https://github.com/Kalyansai6/ML---Assignment-4>

1. Pandas
2. Read the provided CSV file ‘data.csv’.

import pandas as pd

df = pd.read\_csv('data.csv') #reading csv file

df

Graphical user interface, text, email

Description automatically generated

**Description:** Using import keyword, I imported pandas module. read\_csv() reads CSV files from the system.

1. Show the basic statistical description about the data.

df.describe()

Graphical user interface, text

Description automatically generated with medium confidence

**Description:** With describe() function from pandas module we get the statistical description of data which is present in data frame. Statistical description contains min, max, count, 1st quantile, mean, median, standard deviation values of columns.

1. Check if the data has null values.

df.isnull().any()

Graphical user interface, text, application

Description automatically generated

**Description:** To check any null values present in data frame we need to use isnull() function which results a Boolean value. If null values present return true otherwise false. In data frame we imported contains null values only in ‘Calories’ column.

1. Replace the null values with the mean.

mean=df['Calories'].mean()

df['Calories'].fillna(value=mean, inplace=True)

Graphical user interface, text, application, email

Description automatically generated

**Description:** With fillna() function we can replace null values in a data frame. Null values present in only calories column, so we need to replace those null values with calories column mean value. Mean() function gives mean value. Using fillna() we can replace null values. After replacing null values with mean of column, we can see that there are no null values in our data frame.

1. Select at least two columns and aggregate the data using: min, max, count, mean.

df.agg({'Pulse' : ['min', 'max', 'count', 'mean'], 'Maxpulse' : ['min', 'max', 'count', 'mean'],

'Calories' : ['min', 'max', 'count', 'mean'] })

Graphical user interface, application, Word

Description automatically generated

**Description:** Using agg() method we can apply certain operation on data. Here I applied aggregate functions on three columns like Pulse, Maxpulse and Calories.

1. Filter the dataframe to select the rows with calories values between 500 and 1000.

df[(df['Calories'] > 500) & (df['Calories'] < 1000)]

Graphical user interface

Description automatically generated with medium confidence

**Description:** Using ‘& ‘operator we can filter the data based on the conditions given. Here I applied ‘&‘ operator on calories column whose values are between 500 and 1000.

1. Filter the dataframe to select the rows with calories values > 500 and pulse < 100.

df[(df['Calories'] > 500) & (df['Pulse'] < 100)]

Graphical user interface, text, application, email

Description automatically generated

**Description:** Using ‘& ‘operator we can filter the data based on the conditions given. Here I applied ‘&‘ operator on calories column whose values are greater than 500 and in pulse column whose values are less than 100.

1. Create a new “df\_modified” dataframe that contains all the columns from df except for “Maxpulse”.

df\_modified = df[['Duration', 'Pulse', 'Calories']].copy()

df\_modified

Graphical user interface, application, Word

Description automatically generated

**Description:** Using copy() method we can copy the data from the original data frame to another data frame. Here I copied the data excluding the data of Maxpulse column.

1. Delete the “Maxpulse” column from the main df dataframe

df.pop('Maxpulse')

df

Graphical user interface, text, application, email

Description automatically generated

**Description:** Pop() method can be used to remove a particular column from the data frame. Here pop() is applied on the original data frame to remove Maxpulse column.

1. Convert the datatype of Calories column to int datatype.

df['Calories'] = df['Calories'].astype(int)

df.dtypes

Graphical user interface, text, application

Description automatically generated with medium confidence

**Description:** astype() method to convert one data type to other. Here we can see Calories is of float type and it is being converted to int data type using astype() function.

1. Using pandas create a scatter plot for the two columns (Duration and Calories).

df.plot.scatter(x='Duration', y='Calories')

Chart, scatter chart

Description automatically generated

**Description:** pandas module contains functions to represent the data in visual format. Plot.scatter() method to represent data in scatter plot where duration values lie on x-axis and Calories values on yaxis.

1. (Titanic Dataset)

Using Python NumPy and Pandas libraries, I imported test and train data and combined them into a single dataset.

import numpy as np

import pandas as pd

test\_df = pd.read\_csv("test.csv")

train\_df = pd.read\_csv("train.csv")

combine = [train\_df, test\_df]

Graphical user interface, text, application

Description automatically generated

1. Find the correlation between ‘survived’ (target column) and ‘sex’ column for the Titanic use case in class.

train\_df['Sex'].str.get\_dummies().corrwith(train\_df['Survived']/train\_df['Survived'].max())

A picture containing text

Description automatically generated

**Description:** As sex column contains string object so we cannot find the correlation between sex column and survived column. So, first we need to convert into type of objects with which we are comparing and find the correlation with survived column.

1. Do you think we should keep this feature?

As correlation results shows that males were strongly negatively correlated, and females were Strongly positively correlated with their survival. Males are inversely proportional, and females are directly proportional to their survival. So, we need this feature to analysis.

1. Do at least two visualizations to describe or show correlations.

import seaborn as sns #For Visualisation import seaborn library

import matplotlib.pyplot as plt

sns.barplot(x = train\_df['Sex'], y = train\_df['Survived'])

Chart, box and whisker chart

Description automatically generated

sns.regplot(x = train\_df['Age'], y = train\_df['Survived']) #Regression Plot

\Graphical user interface, chart

Description automatically generated

g = sns.FacetGrid(train\_df, col='Survived')

g.map(plt.hist, 'Sex', bins=20) #Multi-plot grids

Chart

Description automatically generated

**Description:** Seaborn library is used to visually show the correlations between the columns data. Here, I am representing correlation between Sex and Survived column using bar plot where Sex on x-axis and survived on y-axis. Similarly, Regression plot for Age and survived columns and Multi plot grids for Survived and sex columns.

1. Implement Naive Bayes method using scikit-learn library and report the accuracy.

train\_df = train\_df.drop(['Ticket', 'Cabin','Parch','SibSp', 'Name', 'PassengerId'], axis=1)

test\_df = test\_df.drop(['Ticket', 'Cabin','Parch','SibSp', 'Name'], axis=1)

combine = [train\_df, test\_df]

for dataset in combine:

dataset['Sex'] = dataset['Sex'].map( {'female': 1, 'male': 0} ).astype(int) #Converting Categorical Feature

print(train\_df.isnull().sum()) #Checking any Null values present in the dataset

Graphical user interface, text, application

Description automatically generated

train\_df['Embarked'].describe()

Graphical user interface, text, application, email

Description automatically generated

common\_value = 'S'

data = [train\_df, test\_df]

for dataset in data:

dataset['Embarked'] = dataset['Embarked'].fillna(common\_value)

ports = {"S": 0, "C": 1, "Q": 2}

data = [train\_df, test\_df]

for dataset in data:

dataset['Embarked'] = dataset['Embarked'].map(ports)

meanAge = int(train\_df.Age.dropna().mean())

print('Mean Age = ', meanAge)

Text

Description automatically generated

**Description:** Before applying machine learning algorithms on data, first we need to preprocess the data to replace null values and to remove any inconsistency present in data. Here, we converted sex columns features into integers i.e., replacing female with 1 and male as 0. In the age column, there are few missing values and those are replaced with age columns mean value. Similarly, in the fare column missing values are replaced with median value. In embarked column, I replaced null values with ‘S’ and those features are converted into integers i.e., S with 0, C with 1, and Q with 2.After completing the preprocessing then we need to apply machine learning algorithms over the data.

#Implementing Naive Bayes on Preprocessed data

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

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

from sklearn.naive\_bayes import GaussianNB

X\_train = train\_df.drop('Survived', axis=1)

Y\_train = train\_df['Survived']

X\_test = test\_df.drop('PassengerId', axis=1).copy()

gaussian = GaussianNB()

gaussian.fit(X\_train, Y\_train)

Y\_pred = gaussian.predict(X\_test)

gaussian.score(X\_train, Y\_train)

Text

Description automatically generated

**Description:** Here, I applied Naive Bayes Algorithm on the preprocessed data using sklearn library. Python sklearn library contains many machine learning algorithms to analyze the data. In the given data, there are no labels present in the test data set to compare with our predicted data. So, we need to use the training data set to compare with our predicted data set using Naive bayes algorithm. Using accuracy, we can compare with other machine learning algorithms to find which method is performing better on this data set.

1. (Glass Dataset)
2. Implement Naive Bayes method using scikit-learn library.
3. Use the glass dataset available in Link also provided in your assignment.

Using read\_csv method from pandas module I imported glass data set.

import numpy as np

import pandas as pd

glass = pd.read\_csv("glass.csv")

glass

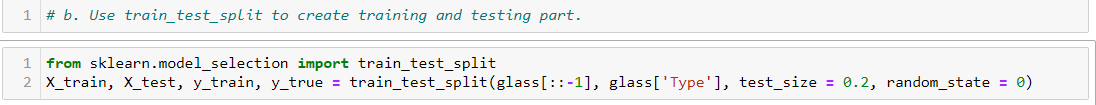
Graphical user interface, text, application

Description automatically generated

1. Use train\_test\_split to create the training and testing part.

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

X\_train, X\_test, y\_train, y\_true = train\_test\_split(glass[::-1], glass['Type'], test\_size = 0.2, random\_state = 0)



Description: sklearn module contains train\_test\_split method to split our data set into training and testing data sets. In this data set Type column can be used for labels. In this method, test\_size defines how much proportion of data to be in the test data set. When we test\_size value whole analysis results will change.

1. Evaluate the model on testing part using score and classification\_report(y\_true, y\_pred)

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

# Gaussian Naive Bayes

from sklearn.naive\_bayes import GaussianNB

classifier = GaussianNB()

classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

# Summary of the predictions made by the classifier

print(classification\_report(y\_true, y\_pred))

print(confusion\_matrix(y\_true, y\_pred))

# Accuracy score

from sklearn.metrics import accuracy\_score

print('accuracy is',accuracy\_score(y\_pred,y\_true))

Graphical user interface, text

Description automatically generated with medium confidence

Description: In the given data, there are no missing values present in it. So, we can directly apply the machine learning algorithms on the data. sklearn module is imported to analyze the data using different algorithms. Classification\_report and confusion\_matrix methods to result the summary of the predictions made using the specific algorithm. These summaries can be used to compare with another algorithms to define which algorithm is better. Naive bayes on this data set results 77% of accuracy.

1. Implement linear SVM method using scikit library
2. Use the glass dataset available in Link also provided in your assignment.

Using read\_csv method from pandas module I imported glass data set.

import numpy as np

import pandas as pd

glass = pd.read\_csv("glass.csv")

glass

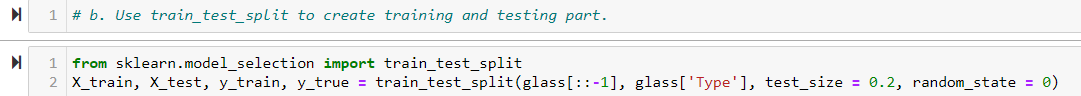
Table

Description automatically generated with low confidence

1. Use train\_test\_split to create the training and testing part.

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

X\_train, X\_test, y\_train, y\_true = train\_test\_split(glass[::-1], glass['Type'], test\_size = 0.2, random\_state = 0)



Description: sklearn module contains train\_test\_split method to split our data set into training and testing data sets. In this data set Type column can be used for labels. In this method, test\_size defines how much proportion of data to be in the test data set. When we test\_size value whole analysis results will change.

1. Evaluate the model on testing part using score and classification\_report(y\_true, y\_pred)

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

# Support Vector Machine's

from sklearn.svm import SVC

classifier = SVC()

classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

# Summary of the predictions made by the classifier

print(classification\_report(y\_true, y\_pred))

print(confusion\_matrix(y\_true, y\_pred))

# Accuracy score

from sklearn.metrics import accuracy\_score

print('accuracy is',accuracy\_score(y\_pred,y\_true))

Graphical user interface, text, application

Description automatically generated

Description: Support vector machine algorithm is applied to this data set using sklearn module. We got an accuracy of 21% using SVM.

Do at least two visualizations to describe or show correlations in the Glass Dataset.

import seaborn as sns #For Visualisation import seaborn library

import matplotlib.pyplot as plt

sns.barplot(x = glass['Type'], y = glass['Ca'])

Chart, bar chart

Description automatically generated

sns.catplot(data=glass, x="Type", y="K")

Chart, scatter chart

Description automatically generated

sns.regplot(x="Type", y="Fe", data=glass);

Chart, scatter chart

Description automatically generated

Description: Seaborn library is used to visually show the correlations between the columns data. Here, I am representing correlation between Type and Ca column using bar plot where Type on x-axis and Ca on y-axis. Similarly, Regression plot for Type and Fe columns and categorized plot for Type and K columns

Which algorithm got better accuracy? Can you justify why?

Among Naive Bayes and Support vector machine algorithms, naive bayes got better accuracy than the SVM. Naive Bayes gives better results than SVM for this data set. we may get better results using SVM than naive bayes when we work with another data set. In this glass data set, types of glass are independent predictors. When there are any independent predictors present in the data set naive bayes perform better than other models.