MACHINE LEARNING
ASSIGNMENT - 2

Name: Yukthi Sravani T

Student ID: 700746296

Git hub Link :- https://github.com/yukthi16/Machine_Learning_Assignment2

Video Link: -

https://drive.google.com/file/d/1leeQyCmlPXVN-T7aYqnOuFpVEZ5xOnVk/view?usp=sharing

1. Pandas

1. Read the provided CSV file 'data.csv'.

https://drive.google.com/drive/folders/1h8C3mLsso-R-sIOLsvoYwPLzy2fJ4IOF?usp=sharing

```
#Read the provided CSV file 'data.csv'.

import pandas as pd
df = pd.read_csv('data.csv')      #reading csv file
df
```

2. Show the basic statistical description about the data.

```
#Show the basic statistical description about the data.

df.describe()   #describe() results statistical description of data in data frame
```

3. Check if the data has null values. a. Replace the null values with the mean

```
#Replace the null values with the mean

mean=df['Calories'].mean()
df['Calories'].fillna(value=mean, inplace=True)  #replacing Nan values with
particular columns mean value
```

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

```
#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'] })
#agg method to aggreate operation on the dataframe
```

5. Filter the dataframe to select the rows with calories values between 500 and1000. 6. Filter the dataframe to select the rows with calories values > 500 and pulse <100.

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

```
df[(df['Calories'] > 500) & (df['Calories'] < 1000)]   #'&' operator to filter the dataframe
```

7. Create a new “df_modified” dataframe that contains all the columns from df except for “Maxpulse”.

```
#Create a new “df_modified” dataframe that contains all the columns from df except for “Maxpulse”.

df_modified = df[['Duration', 'Pulse', 'Calories']].copy()  #copy method to create an another data frome with specified columns from the original dataframe.
df_modified
```

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

```
# Delete the “Maxpulse” column from the main df dataframe

df.pop('Maxpulse')   #pop method to remove a column from the data frame
df
```

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

```
#Convert the datatype of Calories column to int datatype.

df['Calories'] = df['Calories'].astype(int)  #astype function converts one data type into another
df.dtypes
```

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

```
#Using pandas create a scatter plot for the two columns (Duration and Calories).
df.plot.scatter(x='Duration', y='Calories')
```

2. Scikit-learn

1. Implement Naïve Bayes method using scikit-learnlibrary.

a. Use the glass dataset available in Link also provided in your assignment. b. Use train_test_split to create 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)
```

2. Evaluate the model on testing part using score and

```
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))
```

1. Implement linear SVM method using scikit library
a. Use the glass dataset available in Link also provided in your assignment. b. Use train_test_split to create 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)
```

2. Evaluate the model on testing part using score and

```
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))
```

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'])
```

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

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