the assignment. The results and the models can be put into further research with even more data and additional independent variables to increase the accuracy of the models and to know more about consumer behaviours towards telemarketing and fixed deposit schemes.

TASK – II

TITLE

Application of clustering algorithms in the medical records of diabetes patient

1. Introduction

Clustering algorithms is an unsupervised learning algorithms. It helps us to make a clustered or a group with data points. (George,2018) When it comes to evaluating the outcomes of the cluster it is not straightforward as in the case of supervised learning since clustering does not contain any truth labels. Theoretically data points which belong to the same group should share some similarity in their features or have identical properties. Clustering is extensively used in data science to gain future information on the data points by investigating what group each data points fall on. In day-to-day life clustering algorithms are used in various fields such as fraud detection, filtering out spam, sales, marketing and grouping traffic on networks.

In this task, we have taken a dataset which contains the medical information of patients who has diabetes. The objective of this clustering algorithm is to find how many potential groups are there in the patients at different levels of BMI and Glucose. The reason behind choosing these variables is (health essentials, 2021) people with higher BMI levels than normal tend to be at more risk of increased glucose levels in the body which can lead to diabetes. By applying the clustering algorithm to these data, we will be able to extract different groups of patients with various levels of BMI and glucose and with a relation between both values. These data groups have be helpful in finding what levels of glucose a person will have at different BMI ranges. Features can be further developed into a model which could predict the glucose levels of diabetes patients with their BMI.

2. Dataset

The dataset used in this clustering algorithm is downloaded from the prescribed data repository **DATA.WORLD** under the name **Pima Indians diabetes data.** The dataset contains 769 instances with 8 variables and an output variable (if the person has diabetes or not), the dataset contains the values of the variables as numeric, categorical, and binary data. The dataset is very versatile with all the major causes of diabetes and different glucose and insulin levels. This dataset is highly flexible and can be used with different algorithms. The variables with which we are concerned in this dataset for the clustering algorithm is BMI and glucose levels since they share a higher level of correlation.

3. Explanation and preparation of the dataset

The dataset has the shape of (268, 9). The first 8 variables of the data are the variables are medical records of the patient and the 9th column is the output variable which tells us if the patient is diabetic or not.

Data dictionary

Column name	Data type	Description
Pregnancies	Numeric	Number of times pregnant
Glucose	Numeric	Glucose level
Blood pressure	Numeric	Blood pressure level
Skin thickness	Numeric	Skin fold thickness
insulin	Numeric	Serum insulin level
BMI	Categorical	Body mass index
Diabetes pedigree	Numeric	Diabetes prediction function
Age	Numeric	Age of the patient
Outcome	Binary	Outcome

Data. world is a publicly available site to extract data of different fields and magnitude, this dataset is publicly available for further research and development, this dataset originally belongs to the national institute of diabetes and kidney diseases. It was donated for further research on 2016 and the data was made publicly available. Hence the usage of this data set is highly legal and ethical.

- **Data preparation**

Since the objective of the clustering algorithm is to make groups with the BMI and glucose levels of the patients who has diabetes. The values of patients with diabetes were filtered out with the help of the data filter in Microsoft excel. The data of the values when the outcome is 1 was segregated and a new dataset was formed. by doing this we can extract the data of patients who have diabetes.

To start with the data pre-processing the required libraries to perform clustering algorithms were imported and we check for missing values and standardize the data to a scale are the major aspects of data preparation.

The libraries used in this task are :

Figure – 2.1

```
In [3]: #Importing the libraries
        import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd
        import seaborn as sns
```

And the data was imported using the pd. Read_csv function. and the dimension of the dataset can be known by using the function .shape()

Figure – 2.2

```
In [5]: df = pd.read_csv('diabetesdata.csv')

In [28]: df.shape
Out[28]: (268, 9)
```

Checking for missing values can be done using **.info ()** and **.isna().sum(),** in this case, there are no missing values in the dataset hence no missing value techniques need to use in this task.

Figure – 2.3

```
In [7]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 268 entries, 0 to 267
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               268 non-null    int64
 1   Glucose                   268 non-null    int64
 2   BloodPressure             268 non-null    int64
 3   SkinThickness             268 non-null    int64
 4   Insulin                   268 non-null    int64
 5   BMI                       268 non-null    float64
 6   DiabetesPedigreeFunction  268 non-null    float64
 7   Age                       268 non-null    int64
 8   Outcome                   268 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 19.0 KB

In [30]: df.isna().sum()

Out[30]: Pregnancies               0
         Glucose                   0
         BloodPressure             0
         SkinThickness             0
         Insulin                   0
         BMI                       0
         DiabetesPedigreeFunction  0
         Age                       0
         Outcome                   0
         dtype: int64
```

4. Implementation

- **K-means clustering**

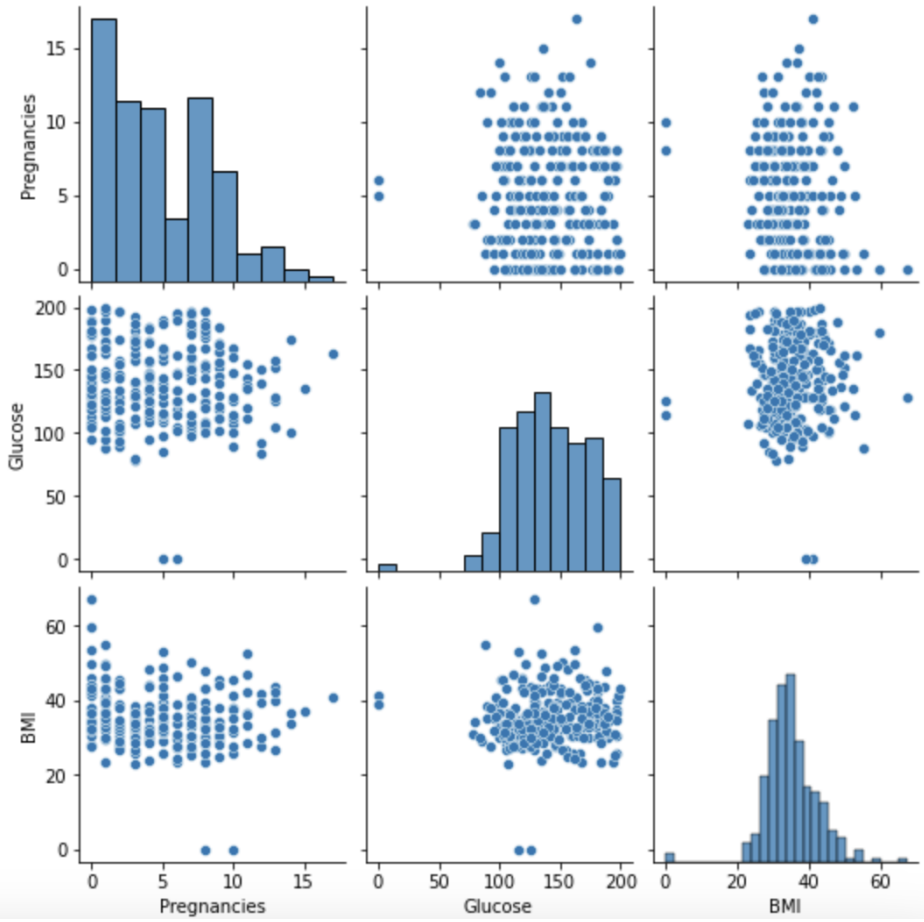
k-means clustering is a very widely used clustering algorithm. K-means clustering involves in grouping unlabeled datasets into various clusters. The k denotes the number how many clusters that should be created in the clustering process. For example, when the value of k = 5 we are supposed to create 5 clusters. This enables us to investigate or discover different similarities and connections in the data set. K- means clustering is based on the centroid in the cluster. The main objective of k-means clustering is to minimize the distance between data points and the cluster they belong to. There are multiple advantages to using the k-means clustering algorithm as it can be used in large sets of data and can adapt to changes very quickly and at ease also it is comparatively easy to implement than other algorithms.

(Machine learning, n.d.) With all the capabilities and easy-to-implement nature of the algorithm also comes with some negatives to, since we manually choose the k value there are high chances of getting it wrong. And when the data set has hidden outliers, the centroid gets dragged away from the actual point or at times the outliers themself can become a cluster

Pair plots are highly effective in visualizing the pairs of variables and also give a histogram representation of each variable with it.

Figure – 2.4

```
In [17]: sns.pairplot(df.iloc[:,[0,1,5]])
Out[17]: <seaborn.axisgrid.PairGrid at 0x7ff5a96afca0>
```



By investigating the scatter plots of glucose against BMI we are able to detect there are almost 3 possible clusters in it.

We will be working on the glucose and BMI data to attain our research objective. Using the iloc () function we have selected the respective columns and put the data through the scaling process so that the data are scaled and we get a more accurate result.

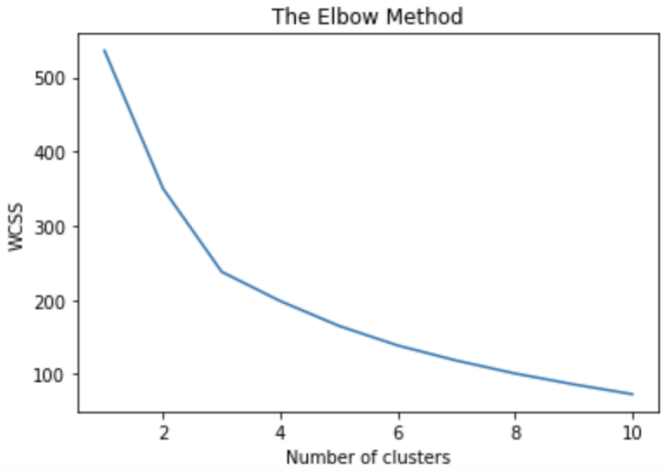
Figure – 2.5

```
In [18]: from sklearn.preprocessing import StandardScaler
         X=df.iloc[:,[1,5]].values
         sc_X=StandardScaler()
         X=sc_X.fit_transform(X)
```

To identify the optimal number of clusters, the k-elbow method can be used to know the number of clusters, the k value will represent it. In this case, the elbow is at k=3. Hence, we will be creating 3 clusters.

Figure – 2.5

```
In [11]: from sklearn.cluster import KMeans
wcss=[]
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,init='k-means++',random_state=42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



Once the optimal number of clusters has been found using the elbow method, the fit_predict () method can be deployed to train the dataset using the kmeans () estimator and get an array output which is equal to the length of the dataset.

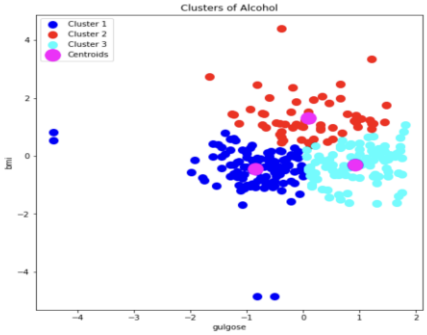
Figure – 2.6

```
In [12]: kmeans=KMeans(n_clusters=3,init='k-means++',random_state=42)
y_kmeans=kmeans.fit_predict(X)
```

Visualizing the clusters can be done as stated below:

Figure – 2.7

```
In [13]: #Visualising the clusters
plt.figure(figsize=(8,8))
plt.scatter(X[y_kmeans==0,0],X[y_kmeans==0,1],s=100,c='blue',label='Cluster 1')
plt.scatter(X[y_kmeans==1,0],X[y_kmeans==1,1],s=100,c='red',label='Cluster 2')
plt.scatter(X[y_kmeans==2,0],X[y_kmeans==2,1],s=100,c='cyan',label='Cluster 3')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s=300,c='magenta',label='Centroids')
plt.title('Clusters of Alcohol')
plt.xlabel('gulgose')
plt.ylabel('bmi')
plt.legend()
plt.show()
```



- **DBSCAN**

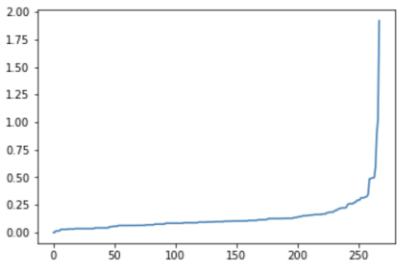
DBSCAN stands for density-based spatial clustering of application with noise, DBSCAN has the capability to trap the arbitrary form of clusters and also make clusters with a dataset with outliers effectively. DBSCAN decides if the data point belongs to a group by checking to which group it shares many closer points. Points are split into three types- core, outliners, and border. One of the major positives of DBSCAN is we don't have to calculate or specify any number of clusters to run the algorithm and it is highly capable when it comes to handling outliers. DBSCAN has a fair share of cons, extracting the results and determining the proximity between the data points need heavy domain knowledge.

The preprocessing method and the scaling methods are the same for both k-means and DBSCAN since we are using the same dataset. Firstly, we plot the distance (nearest neighbours) to every data point and arrange them in ascending order. From this plot, we can see where the distance starts to increase and determine our epsilon. In this case, it's around – 0.50.

Figure – 2.7

```
In [14]: from sklearn.neighbors import NearestNeighbors
neighbours=NearestNeighbors(n_neighbors=2)
distances,indices=neighbours.fit(X).kneighbors(X)
distances=distances[:,1]
distances=np.sort(distances,axis=0)
plt.plot(distances)

Out[14]: [<matplotlib.lines.Line2D at 0x7f93f80b0e80>]
```



By using the fit-predict () method we now generate an array the same as kmeans.

Figure -2.8

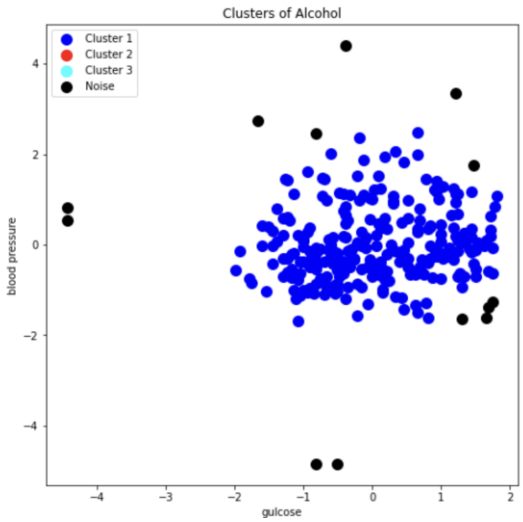
```
In [18]: from sklearn.cluster import DBSCAN
dbscan=DBSCAN(eps=0.50,min_samples=5)
y_dbscan=dbscan.fit_predict(X)

In [19]: y_dbscan

Out[19]: array([ 0, -1,  0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,  0,  0,
                 0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,
                 0,  0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1, -1,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0])
```

Visualizing the clusters can be done as stated below:

```
In [20]: plt.figure(figsize=(8,8))
         plt.scatter(X[y_dbscan==0,0],X[y_dbscan==0,1],s=100,c='blue',label='Cluster 1')
         plt.scatter(X[y_dbscan==1,0],X[y_dbscan==1,1],s=100,c='red',label='Cluster 2')
         plt.scatter(X[y_dbscan==2,0],X[y_dbscan==2,1],s=100,c='cyan',label='Cluster 3')
         plt.scatter(X[y_dbscan==-1,0],X[y_dbscan==-1,1],s=100,c='black',label='Noise')
         plt.title('Clusters of Alcohol')
         plt.xlabel('gulcose')
         plt.ylabel('blood pressure')
         plt.legend()
         plt.show()
```



5. Result analysis

After extracting two different types of clusters from Kmeans and DBSCAN clustering algorithms. we were able to analyze that Kmeans was able to extract 3 different groups and we could analyze that cluster one has multiple outliers in the data points. And it has not been handled well since the centroids get pulled away from the actual point. But in DBSCAN the outliers are handled in a better manner.

By using these different clusters, we can have an idea about the different groups present with different levels of glucose and BMI. There are three different groups of people with a particular level of BMI and connected to a particular level of glucose. from making a clustering model we can make assumptions on which BMI range a patient will fall when we know the patient's glucose levels. With more domain knowledge this model could be future understood and draw more connections between BMI and the Glucose level of a patient

6. Conclusion

The main objective of this assignment is to make a clustering analysis of the medical records of the diabetes patient. And try to make clusters with the highest leading factor for diabetes which is glucose level and BMI. When we did the clustering, we were able to identify three potential groups for which we can make assumptions of what range of BMI a patient could be in or can assume what range of glucose level a patient with a particular range of BMI could have. With future research and domain knowledge, we can make a model based on this cluster group to predict what level range of BMI or range of glucose the patient might fall.