SYNOPSIS

Data Preprocessing :

Data preprocessing in Machine Learning refers to the technique of preparing the raw data to make it suitable for building and training Machine Learning models.

Steps in Data Preprocessing are as follows -

1. Acquire the dataset

   To build and develop Machine Learning models, we must first acquire the relevant dataset. There are several online sources from where we can download datasets or we can also create a dataset by collecting data via different Python APIs. Once the dataset is ready, we must put it in CSV, or HTML, or XLSX file formats.

2. Import all the libraries required

   The libraries that will be needed for this task are

   NumPy - It is the fundamental package for scientific calculation in Python. Hence, it is used for inserting any type of mathematical operation in the code. Using NumPy, we can also add large multidimensional arrays and matrices in our code.

   Pandas - It is an open-source Python library for data manipulation and analysis. It is extensively used for importing and managing the datasets.

   Matplotlib - It is a Python 2D plotting library that is used to plot any type of charts in Python.

   sklearn - It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python.

   Seaborn - It is a library in Python predominantly used for making statistical graphics.

3. Import the dataset

   We can import the dataset using the “read_csv()” function of the Pandas library. This function can read a CSV file and also perform various operations on it. The read_csv() is written as:

   data_set = pd.read_csv(‘filename.csv’)

   Here, data_set is the variable name where we have stored our dataset.

4. Identifying and handling missing/null values

   We can handle null values by the following ways

   - Deleting rows with missing values
     This method is useful only if the number of missing values is very less as compared to the entire dataset as it can lead to loss of a lot of data and the model may work poorly if the percentage of missing values is excessive in comparison to the complete dataset

   - Imputing missing values with mean/median
     This method can prevent the loss of data and it works well with a small dataset and is also easy to implement but it works only with numerical continuous variables.

   - Prediction of missing values
     This method gives a better result than the previous methods. By using the other features which don’t have null values we can predict the missing values.

   Any of these methods can be used for handling null values depending upon the dataset that is used and how many null values does this dataset contain.
   To get the count of null values in each column we can use

   data_set.isnull().sum()

5. Encoding the categorical data

   Categorical data refers to the information that has specific categories within the dataset. In this task, job, marital status, etc are some categorical variables. Machine Learning models are based on mathematical equations, so keeping the categorical data in the equation will cause certain issues. That is why we have to convert this categorical data into numerical data. To do so we can use the LabelEncoder() class from the sci-kit learn (sklearn) library.

Clustering Algorithms :

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

Some of the most used clustering algorithms are -

1. K-Means

   K-means is a centroid-based algorithm, or a distance-based algorithm, where We calculate the distances to assign a point to a cluster. In K-Means, each cluster is associated with a centroid.The main objective of the K-Means algorithm is to minimize the sum of distances between the points and their respective cluster centroid. It is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group.

2. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

   DBSCAN begins with an arbitrary starting data point that has not been visited. The neighborhood of this point is extracted using a distance epsilon ε (All points which are within the ε distance are neighborhood points). If there are a sufficient number of points (according to minPoints) within this neighborhood then the clustering process starts and the current data point becomes the first point in the new cluster. Otherwise, the point is labelled as noise. All the points within the ε distance neighborhoods become part of the same cluster. This continues until all points in the cluster are determined. Once a cluster is made, the procedure is repeated for another unvisited point. This process continues until all the points are visited.

3. MeanShift

   Mean shift clustering is a sliding-window-based algorithm that attempts to find dense areas of data points. It is a centroid-based algorithm meaning that the goal is to locate the center points of each group/class, which works by updating the candidates for center points to be the mean of the points within the sliding-window. These candidate windows are then filtered in a post-processing stage to eliminate near-duplicates, forming the final set of center points and their corresponding groups.In contrast to K-means clustering, there is no need to select the number of clusters as mean-shift automatically discovers this.

For this task I will be using the K-Means clustering algorithm because it is more suitable for customer segmentation. It is one of the most commonly used clustering algorithms and it is also simple and easy to implement. What I have understood about this algorithm is that it partitions n observations into K clusters. The value of K, that is the number of clusters to be formed, has to be specified by us. First we randomly select K data points. These K data points are the centroids of K clusters. Then we one by one calculate the distance between each data point and the centroids and assign each data point to its nearest centroid. Those data points which are near to the particular centroid, create a cluster. The mean of all the data points in a cluster is then calculated for all of these K clusters and these means calculated for each cluster are taken as the new centroids. Then again the distance between the data points and new centroids is calculated and each data point is reassigned to the nearest new centroid thus forming new clusters. The new clusters formed are different from the original ones. This process is repeated until the centroids of newly formed clusters do not change or the data points remain in the same cluster or we can also specify the number of iterations to be performed.

Determination of number of clusters (K) :

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. So it is very important to choose the optimal number of clusters. We can determine the value of K by the following methods

1. The Elbow Method

   In this method, WCSS (within cluster sum of squares) is calculated for all the data points. It is the sum of squares of distances between each data point and its centroid within a cluster. To find the best value of K we have to minimize the value of WCSS and also keep the number of clusters as small as possible.

   To find the optimal value of clusters, the elbow method follows the below steps:

   - It executes the K-means clustering on a given dataset for different K values ranging from 1-10.
   - For each value of K, the WCSS value is calculated.
   - Then a curve between the calculated WCSS values and the number of clusters (K) is plotted.
   - The graph shows a drastic decrease in WCSS as we increase the value of K and after a certain value of K there is no significant decrease in the WCSS value. This value of K is the optimal number of clusters that can be formed.

2. The Silhouette Method

   In this method we calculate the silhouette value for each data point. The Silhouette score can be easily calculated in Python using the metrics module of the sklearn library. The Silhouette value measures how similar a point is to its own cluster compared to other clusters. The range of the Silhouette value is between +1 and -1. A high value is desirable and indicates that the point is placed in the correct cluster. If many points have a negative Silhouette value, it may indicate that we have created too many or too few clusters. To find the optimal number of clusters a graph against Silhouette score and K is plotted. The Silhouette Score reaches its global maximum at its optimal K. So the peak of the Silhouette score vs. K graph is the best value of K.

Sometimes the graph of the elbow method may not give a clear and sharp edge, so in such ambiguous cases we may use the silhouette method. We don’t have to consider these methods as alternatives to each other for finding the optimal K rather we can use the combination of these two methods to make a more confident decision.

Conclusion :

In this synopsis, I have explained the various data preprocessing steps that have to be performed in order to increase the accuracy and efficiency of the model. I have also explained the K-Means, DBSCAN and the MeanShift clustering algorithms and I have chosen the K-Means clustering algorithm for my model because it is a versatile algorithm that can be used for any type of grouping and is most suitable for customer segmentation models. Choosing the best value of K is very important for this algorithm to work efficiently so, to achieve that I have used the combination of the Elbow method and the Silhouette method to get the optimal value of the number of clusters K.