Role of Various Distance Metrics on Clustering Results

1. Clustering

Clustering is a type of unsupervised machine learning that categorizes the elements of data according to similarity in their attributes. A distance metric is important to measure the efficacy of clustering results because it specifies the processes that were employed in assessing the likeness of data. This course addresses the effect of different distance measures on clustering results, especially in relation to the formation of clusters and the performance of the model. (Anon., n.d.)

Getting the nuances of various distance metrics right is key in embedding more advanced clusters into your datasets. In this study, we discuss the influence differential metrics have on K-means and Hierarchical clustering, where we explain three metrics: Manhattan, Cosine, and Euclidean distances. Depending on the goals of the clustering and the features of the data being analyzed, we will demonstrate which one is to be used.

2. Goal

At the end of this tutorial, readers will be able to know, how distance is utilized in clustering algorithms. Some of the distance measures and their definitions. The influence of the distance measure on the outcomes of K-means and hierarchical clustering. Through a practical example in matplotlib and scikit-learn using Python.

3. Some Basics Terminologies:

- **Silhouette Scores**

  A metric called Silhouette Score is used to evaluate how well clustering is done. It takes into account both separation (the degree of distance between clusters) and cohesiveness (the proximity of points inside a cluster). The range of the score is -1 for erroneous clustering and +1 for well-clustered. (Anon., n.d.)

- **Distance Metrics**

  A distance metric can be defined as a mathematical measure that defines how

close or how far apart two points within a dataset are. Relationships among the dataset points are represented through several of these.

- **K-means Clustering**
  The well-known K-means clustering algorithm divides a dataset into a predetermined number of clusters, represented by the letter K. This iterative approach assigns each data point to the closest cluster center (centroid) in order to reduce the within-cluster variation. The algorithm first initializes K centroids at random, and then it alternates between two steps: assignment and update. Each data point in the assignment step is allocated to the cluster with the nearest centroid, which is usually determined by the Euclidean distance or alternative distance metrics such as the Manhattan or cosine distances. The mean of the data points inside each cluster is used to recalculate the centroids in the update step. This process is repeated until centroids stabilize or a maximum number of iterations is achieved, at which point convergence takes place.
- **Hierarchical Clustering**
  A method of clustering that employs either the divisive or agglomerative approach in obtaining parts of a hierarchy of clusters.

4. Distance Metrics Overview

The distance measures discussed in this session may be summarized in the following way (Anon., n.d.):

- **Euclidean Distance**
  The simplest understandable and most widely used distance in multidimensional space which measures the straight-line distance from point A to point B. appropriate for clusters that are round in nature.
- **Manhattan Distance**
  The absolute difference of the coordinates is known as the absolute distance and is often described as the sum. The method is effective for high-dimensional data.
- **Cosine Similarity**
  Calculates the cosine of the angle between two vectors, treating scale as more important than angle focus. Best suited for sparse high dimensional datasets and text data.

5. Agglomerative Clustering

The principles of agglomerative clustering, a hierarchical technique frequently employed in unsupervised learning, can also be used in supervised contexts for feature engineering or data preprocessing. In order to determine how similar or dissimilar two data points are, a **distance metric** is used to iteratively merge the closest pairs of data points or clusters. The choice of distance metric has a big impact on the clustering result. Common metrics include cosine similarity, Manhattan distance, and Euclidean distance. By creating a hierarchy until every point is a part of one cluster or a predetermined number of clusters, agglomerative clustering creates a dendrogram that shows relationships at different levels. Agglomerative clustering does not necessitate pre-specifying the number of clusters, in contrast to K-means, which begins with a predetermined number of centroids and allocates data points to clusters by minimizing within-cluster variance. It is quite interpretable and versatile due to its versatility and capacity to manage different distance measures. Although K-means requires spherical clusters and is subject to random initialization, it is computationally faster and more efficient for large datasets. By using customized distance metrics to capture subtle associations, agglomerative clustering, on the other hand, is more computationally demanding but excels at producing meaningful hierarchical structures, especially for smaller datasets.

6. Dataset Overview

We will use the Wine Quality Dataset which has the chemical information for white and red wines for the analyses. It has a target variable of the quality of wine and includes 11 features such as ethanol content, pH, sugar, and acidity. The multi-dimensional features and pattern diversity in this dataset make it very ideal for clustering. For more details, check out the UCI Datasets - Wine Quality. (Anon., n.d.)

7. Practical Demonstration of Clustering with Different Distance Metrics

In this part, we will discuss the results of several distance metrics subsequent to applying Agglomerative means clustering on the Wine Quality Dataset.

8. Import Required Libraries

First, we'll import all the Python libraries that help us to use the required functions and models for this tutorial.

```
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.spatial.distance import cdist
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.cluster import AgglomerativeClustering
```

9. Load and Explore the Dataset

We will load our dataset file into a data frame for manipulation and exploration.

```
# Load the Wine Quality Dataset
data = pd.read_csv('winequality-red.csv')

# Display the first few rows
print(data.head())

# Inspect the structure and check for missing values
print(data.info())
print(data.isnull().sum())
```

10. Data Preprocessing

To ensure that all dimensions have the same impact weight, perform normalization of the features by use of the **'StandardScaler'** method. And remove the target variable from the data frame.

```
# Drop the target variable
features = data.drop('quality', axis=1)

# Normalize the features
scaler = StandardScaler()
scaled_data = scaler.fit_transform(features)

# Check the scaled data
print(scaled_data[:5])
```

11. Apply Agglomerative-Means Clustering with Different Distance Metrics

To evaluate the difference metrics, we will first apply the Euclidean distance with Agglomerative Means clustering and find out the silhouette score.

```
# Perform Agglomerative Clustering with Euclidean distance
agg_euclidean = AgglomerativeClustering(n_clusters=3, metric='euclidean',  linkage="average")
agg_euclidean_labels = agg_euclidean.fit_predict(scaled_data)

# Evaluate with silhouette score (Euclidean)
euclidean_score = silhouette_score(scaled_data, agg_euclidean_labels)
print(f"Silhouette Score (Euclidean): {euclidean_score}")
```

After that, we will apply a similar algorithm but for Manhattan distance to the scaled data.

```
# Perform Agglomerative Clustering using Manhattan distance
agg_manhattan = AgglomerativeClustering(n_clusters=3, metric='manhattan', linkage="average")
agg_manhattan_labels = agg_manhattan.fit_predict(scaled_data)

# Evaluate with silhouette score (Manhattan)
manhattan_score = silhouette_score(scaled_data, agg_manhattan_labels)
print(f"Silhouette Score (Manhattan): {manhattan_score}")
```

After that, we will apply this algorithm with the cosine similarity metric.

```python
# Compute Cosine similarity for the dataset
cosine_similarities = cosine_similarity(scaled_data)

# Perform Agglomerative Clustering using Cosine similarity
agg_cosine = AgglomerativeClustering(n_clusters=3, metric='precomputed',  linkage="average")
agg_cosine_labels = agg_cosine.fit_predict(1 - cosine_similarities)

# Evaluate with silhouette score (Cosine)
cosine_score = silhouette_score(scaled_data, agg_cosine_labels)
print(f"Silhouette Score (Cosine): {cosine_score}")
```

12. Comparison

To evaluate the model's performance on different distance metrics, we are using Silhouette Scores. Outcomes obtained from different metrics are:

```python
print("\nComparison of Silhouette Scores:")
print(f"Euclidean Distance: {euclidean_score}")
print(f"Manhattan Distance: {manhattan_score}")
print(f"Cosine Distance: {cosine_score}")
```

```
Comparison of Silhouette Scores:
Euclidean Distance: 0.5631668418422066
Manhattan Distance: 0.5597325806675031
Cosine Distance: 0.1552374499995812
```

In clustering, Euclidean distance is the most widely used distance metric. If this metric's silhouette score is the highest, it indicates that the wine dataset can be more easily separated using absolute differences between characteristics (such as alcohol percentage, acidity levels, etc.).

When feature value variances are more linear or grid-like, Manhattan distance performs best. A lower silhouette score for this metric could indicate that the data's pattern is not well suited for Manhattan distance, which might not be as effective as Euclidean distance at maintaining the data's complexity.

Cosine similarity finds the cosine of the angle between two vectors, focusing on direction rather than magnitude. Silhouette score using cosine similarity suggests that the relative patterns—like the proportion of alcohol to acidity or volatile compounds—play a more important role in creating clusters than their absolute values. If you are less concerned

about differences in magnitude but expect the features to be closely connected, cosine similarity might be more suitable. (Anon., n.d.)

13. Summary

- If your Euclidean score is the highest, it means that the dataset naturally clusters based on absolute differences in feature values.
- If the Manhattan score is lower, it could indicate that the dataset is less naturally separable based on grid-like or linear relationships between feature values.
- A higher cosine similarity score would suggest that clustering the data based on relative feature proportions rather than absolute differences leads to more meaningful clusters.

Works Cited

Anon., n.d. [Online]
Available at: https://www.geeksforgeeks.org/clustering-in-machine-learning/
Anon., n.d. [Online]
Available at: https://medium.com/@hazallgultekin/what-is-silhouette-score-f428fb39bf9a
Anon., n.d. [Online]
Available at: https://medium.com/@hazallgultekin/what-is-silhouette-score-f428fb39bf9a
Anon., n.d. [Online]
Available at: https://archive.ics.uci.edu/dataset/186/wine+quality
Anon., n.d. [Online]
Available at: https://www.linkedin.com/advice/0/what-appropriate-distance-metric-clustering-methods-lxn4f

References

Anon., n.d. [Online]
Available at: https://www.geeksforgeeks.org/clustering-in-machine-learning/
Anon., n.d. [Online]
Available at: https://medium.com/@hazallgultekin/what-is-silhouette-score-f428fb39bf9a
Anon., n.d. [Online]
Available at: https://medium.com/@hazallgultekin/what-is-silhouette-score-f428fb39bf9a
Anon., n.d. [Online]
Available at: https://archive.ics.uci.edu/dataset/186/wine+quality

Anon., n.d. [Online]
Available at: https://www.linkedin.com/advice/0/what-appropriate-distance-metric-clustering-methods-lxn4f