HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

MASTER THESIS

Mr
Hasan Lodhi

Reconstruction of Disrupted Online Time Series Data in Measurements for Fluctuating Production of Energy

2021

Faculty of **Applied Computer Sciences and Biosciences**

MASTER THESIS

Reconstruction of Disrupted Online Time Series Data in Measurements for Fluctuating Production of Energy

Author:
Hasan Lodhi

Study Programme:
Applied Mathematics in Networks and Data Science

Seminar Group:
MA18w1-M

First Referee:
Prof. Dr. Thomas Villmann

Second Referee:
Mr.Tilo Schwarz

Mittweida, May 2021

Bibliographic Information

Lodhi, Hasan: Reconstruction of Disrupted Online Time Series Data in Measurements for Fluctuating Production of Energy, 63 pages, 20 figures, Hochschule Mittweida, University of Applied Sciences, Faculty of Applied Computer Sciences and Biosciences

Master Thesis, 2021

Abstract

Over the past few years, wind and solar power plants have increasingly contributed to energy production. However, due to fluctuating energy sources, the energy production data contain disruption. Such disrupted data lead to the wrong prediction performance, and they need to be estimated by other values. In this thesis, we provide a comparative study to estimate the online disrupted data based on the data of similar groups of power plants, We apply three estimation techniques, e.g., mean, interpolation, and k-nearest neighbor to estimate the disruption on training data. We then apply four clustering algorithms, e.g., k-means, neural gas, hierarchical agglomerative, and affinity propagation, with two similarity measures, e.g., euclidean and dynamic time warping to form groups of power plants and compare the results. Experimental results show that when KNN estimation is applied to data, and neural gas and agglomerative with dtw are used to cluster the data, the cluster quality scores and execution time give better results compared to others. Therefore, we conclude and choose KNN estimation to reconstruct the online disrupted data on each group of a similar power plants.

I. Contents

Contents I
List of Figures II
List of Tables III
Nomenclature IV
Preface V
1 Introduction 1
1.1 Related Work 1
1.2 Problem Statement 3
1.3 Thesis Structure 4
2 Distance Measures 7
2.1 Euclidean Distance 9
2.2 Dynamic Time Warping 10
2.3 Comparison of Distance Measures 13
3 Clustering Algorithm 15
3.1 Partition based clustering 16
3.1.1 K-means 16
3.1.2 Neural Gas 18
3.1.3 Affinity Propogation 20
3.2 Hierarchical clustering 23
3.2.1 Agglomerative Nesting: 24
3.2.2 Divisive Algorithm 27
4 Clustering Evaluation Strategies 29
4.1 Silhouette Score 29
4.2 Calinski-Harabasz Index 30
4.3 Davies-Bouldin Index 31
4.4 Comparison of Cluster Validity Indices 32
5 Estimation of Disrupted data 33
5.1 Introduction 33

5.2 Missing Data Mechanism 34
5.3 Estimation Methods for Treating Missing Data 35
5.3.1 Traditional Methods 35
5.3.2 Modern Methods 36
5.3.3 Comparison of Estimation Methods 36
6 Experiment and Results 39
6.1 Data 39
6.1.1 Data Description 39
6.1.2 Data Prepossessing 39
6.1.3 Data Cleaning 40
Cleaning of Missing Data 40
Cleaning of Uncorrelated Data 41
6.1.4 Feature Scaling 43
6.2 Determining the Number of Clusters 44
6.3 Implementation 45
6.3.1 Computing Distance Matrices 45
6.3.2 Implementation of Clustering Algorithm and Quality Index 46
6.4 Results and Analysis 47
6.4.1 Data set I: Wind Plant Data 47
6.4.2 Data set II: Solar Plant Data 52
7 Conclusion and Future Work 57
Bibliography 59

II. List of Figures

2.1 Two time series C and Q that are alligned in time t 8
2.2 ED aligns C and Q by implementing one-to-one matches between the pairs of sequences. 9
2.3 Euclidean distance of high-dimensional time series data with disruptions 10
2.4 DTW aligns C and Q by implementing one-to-many matches between the pairs of sequences and the warp the points to account for the shifts in time. 11
2.5 Cost matrix showing the optimal warping path P of time series C and Q. 11
3.1 Description of update rule 19
3.2 Description of messages 21
3.3 A typical representation of how affinity propogation works. 22
3.4 A typical representation of a dendrogram. 24
3.5 Basic working of agglomerative algorithm. 25
6.1 Missingness in Wind PP 41
6.2 Missingness in Solar PP 41
6.3 Correlation Matrix of Wind PP Data 43
6.4 Correlation Matrix of Solar PP Data 43
6.5 Elbow graph for wind PP data 45
6.6 Elbow graph for solar PP data 45
6.7 Visualization of the six clusters that are obtained when applying the K-means clustering algorithm with the DTW distance and KNN estimation to estimate disruption on wind PP time series data. 50
6.8 Example of one of the wind PP time series data from each cluster to visualize the estimation techniques applied to reconstruct disrupted data. X-Axis represent the time and Y-Axis represent the energy production value of each PP 51
6.9 Visualization of the five clusters that are obtained when applying the Agglomerative clustering with the DTW distance and KNN estimation to estimate disruption on solar PP time series data. This clustering assignment is the best for solar PP in our experiments. 55

6.10 Example of one of the Solar PP time series data from each cluster to visualize the estimation techniques applied to reconstruct disrupted data. X-Axis represent the time and Y-Axis represent the energy production value of each P 56

III. List of Tables

2.1 Comparison of Distance Measures 13
4.1 Comparison of cluster validity indices 32
6.1 Overview of the computation times of the different distance measures for determining the distance matrix 46
6.2 Clustering performance evaluation results for wind PP when using KNN estimation technique to estimate disrupted data 48
6.3 Clustering performance evaluation results for wind PP when using Interpolation technique to estimate disrupted data 48
6.4 Clustering performance evaluation results for wind PP when using mean estimation technique to estimate disrupted data 49
6.5 Clustering performance evaluation results for solar PP when using KNN estimation technique to estimate disrupted data 52
6.6 Clustering performance evaluation results for solar PP when using Interpolation technique to estimate disrupted data 53
6.7 Clustering performance evaluation results for solar PP when using mean estimation technique to estimate disrupted data 53

IV. Nomenclature

$d(C,Q)$ ..... The distance between time series or data vectors C and Q, Seite 7

ED ..... Euclidean Distance, Seite 9

n ..... Number of data points in data vector, Seite 9

$d_{euc}(C,Q)$ ..... Euclidean distance between two data vectors C and Q, Seite 9

DTW ..... Dynamic Time Warping, Seite 10

$CM(i,j)$ ..... Cost Matrix, Seite 10

$d(i,j)$ ..... Distance between data point c_i and q_j where $c_i \in C$ $q_j \in Q$, Seite 10

P ..... Warping path, Seite 10

$d_{DTW}(C,Q)$ ..... Dynamic time warping between two data vectors C and Q, Seite 12

k ..... Number of clusters, Seite 16

M ..... Set of clusters, Seite 16

w ..... Prototype, Seite 16

B ..... Set of feature vectors, Seite 16

$\mathbb{R}$ ..... The real numbers, Seite 16

λ ..... The neighbourhood range, Seite 19

$K_i(c,w_i)$ ..... The rank of prototype, Seite 19

R ..... The responsibility matrix, Seite 20

A ..... The availability matrix, Seite 20

Pr ..... Preference, Seite 20

Λ ..... The damping factor, Seite 22

SSE ..... Sum of squared error, Seite 26

SC ..... Silhouette Coefficient, Seite 29

SS_{bc} ..... Sum of square distance between cluster, Seite 30

SS_{wc} ..... Sum of square distance within cluster, Seite 30

μ_m ..... Center of cluster m, Seite 30

$\rho(c,q)$ ..... Pearson correlation coefficient between two data vectors C and Q, Seite 42

$d_{corr}(c,q)$ ..... Correlation between two data vectors C and Q, Seite 42

V. Preface

This thesis was written as a part of master degree at Hochschule Mittwedia University of Applied Sciences. The research topic was suggested by Mr.Tilo Schwarz together with Prof.Thomas Villmann. They were always willing to help me through out the whole research and analysis. Their guidance and support help me achieve this thesis on time. The research conducted at BayWa renewable energy GmbH is presented in this thesis.

From Hochschule Mittweida, I would like to thank Miss.Marika Kaden and my first supervisor Prof.Thomas Villmann for proofreading the text and give timely and excellent feedback.

Over the past two and a half year I had a great time in BayWa renewable energy GmbH. I would like to thank every one in the department of Methods and Models especially Mr.Sebastian Sambale, Mr.Torsten Kasner and my second supervisor Mr.Tilo Schwarz to have weekly meetings and provide valuable feedback to determine which steps to take during this research.

I would like to thank my dear wife and my son who has been supporting me since the beginning of this journey.

In the end, special thanks to my late father and mother, from the beginning they taught me how valuable I am. They showed me the best choices for me that have directed me to the right path. There is no way I can ever repay them but I am honored. Thank you for everything.

Hasan Lodhi - Mittweida, May 2021

1 Introduction

Power generation and consumption are among some of the newly emerging and concerning the problem of this modernized world of technology. The rapidly increasing population and increased economic growth are one of the major contributors to this global power consumption problem along with many others. To deal with this increased energy demand, a significant increase in smart grid systems using solar and wind energy sources can be seen. The energy production in these smart grid stations is monitored using advanced sensor technologies and is connected to the Supervisory control and data acquisition (SCADA) system to keep track of the data for energy production per minute to per hour depending on the efficiency of the electric systems. Most of the solar and wind energy production facilities are required to predict their hourly production to increase profitability and efficiency. This energy prediction is very important for economic growth, a performance measure of the solar panels, wind turbines, and also the stability of the system. When we talk about forecasting energy production, one of the biggest issues faced is the large ramps, variabilities, and disruption in the data set received by the system. This happens as the wind and solar energy sources are fluctuating in nature as well as there is sensor malfunction with other random errors. All of this result in prediction error which directly influence the reliability and efficiency of the grid station

This disrupted data is classified as online time series data collected at successive time intervals. At present many different approaches and methods are implementing to forecast anomalies in this online time series data which include Statistical Methods, Machine Learning Methods, Data Representation, Synthetic Anomaly Generation (e.g., GANs), and other Note-Worthy Libraries. All these various studies conducted for the modeling of disrupted online time series data including Machine learning models which work with real valued signal for pattern detection and for predicting future data pattern is most common for a more complex data-set. It deals with significantly more complex uncertainties that require an advance and more precise prediction whereas the statistical method as well as other methods are application based and are more efficient if a non-complex data set is being analyzed.

1.1 Related Work

One of the earliest works on time series data analysis can be found in article [1] presented in 1960 in a journal of basic engineering, the study aimed at the state space modeling of time series data. The model results are shown for optimal prediction error by formulating a co-variance matrix as well as noise filtering of the dynamic data. Furthermore, the state space modeling technique for time series data analysis is discussed in more detail

in this study [2]. In general time series analysis, an assumption of identical Gaussian distribution for the squared noise terms with zero mean is considered. However, these assumptions are not valid for independent and different losses other than squared loss hence the approach requires additional modeling that emphasizes the noise distribution e.g. t-distribution and the ARCH model proposed in [3, 4]. A study [5] is conducted in Electric Reliability Council of Texas (ERCOT) for the wind power ramp prediction using a swinging door algorithm that results in a cost effective and reliable solution for power ramping forecasting for a wind turbine power production.

Many studies have been conducted to find whether the statistical methods or the machine learning methods are better for online time series data analyses. A comparative analysis is carried in the article [6] for predicting electric energy utilization in smart buildings. Both techniques including statistical and machine learning are implemented and compared for best results. Another contribution of the study is determining the best sample size of the data for optimal prediction. The best results are obtained with the Machine learning approach on a seven days data log sample obtained from 13 smart buildings located in Spain at UPO university campus in capital city Seville. Similar work based on regression techniques for hourly and daily electrical consumption can be studied in detail in articles [7–9]. Among machine learning approaches, an extensively used and successful approach is presented in [10]. It uses an Artificial Neural Network (ANN) technique on per hour temperature and power load data gathered from the zones of Seattle and Tacoma for 1.5 years, to predict 1 hour and 24-hour electric load prediction values.

Clustering strategies have also been implemented within the domain of solar power data modeling, such as in [11] where clustering technique is used for modeling spectral solar irradiance data set and performance measure of the photovoltaic modules. K-means clustering is one of the foremost utilized techniques commonly utilized in different domains such as data mining, factual learning, and finding patterns in data.

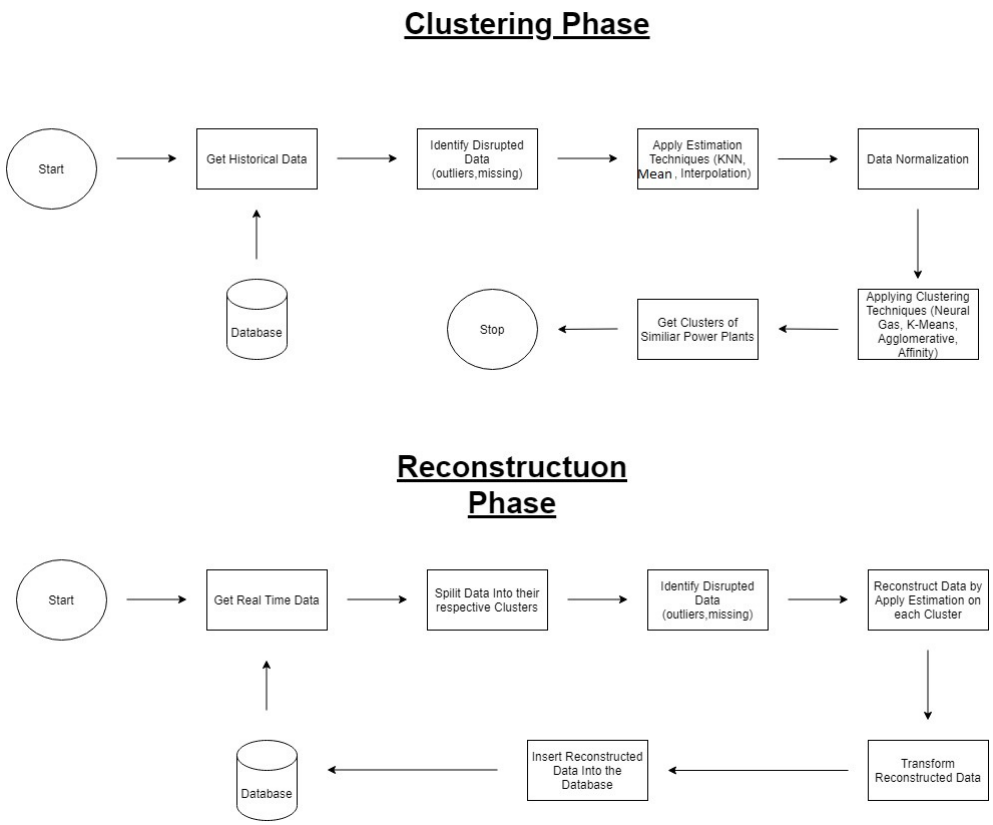
There are many other recent types of research on the application of clustering techniques for online time series data. Among these, the publication [12] presents a very novel approach where the autoregressive moving average (ARMA) model is combined with the K-means clustering algorithm for the fluctuating wind power prediction and is compared with the simple ARMA model. The results show that the hybrid clustering-based model outperforms in performance analysis and optimal prediction error. A similar study focusing on predicting the impact on wind power fluctuation on electric grid stations is presented in [13]. The data consists of wind speed, humidity, pressure, and temperature from four different grid stations. K-means clustering technique is used where the four inputs are analyzed for cluster formation and centroids for each cluster are found. This model resulted in 31 days of wind speed prediction data.

1.2 Problem Statement

The main objective of this thesis is a reconstruction of disrupted online time-series data points of each Power Plant (PP) by estimating it from similar groups of power plants. For example, if plant 1 is having disruption at time t, then there should be a mechanism to reconstruct or estimate that disrupted data from similar PP data. To form groups of similar PP, the clustering is performed on historical data, and once the clustering is successfully implemented then the reconstruction or estimation techniques are applied to real-time online data. Therefore, the study is divided into two phases, Clustering Phase and Reconstruction Phase.

1) Clustering Phase: The working data set is energy time series data collected per minute data log of the year 2019 obtained from automatic monitoring systems (wind and solar power plants). As these are fluctuating energy sources thus the training data set contains disruption like missing data values. The first working step of the study is to prepare this training data and solve the missing data problem so that we have a well-structured training data set for our clustering model. K-nearest neighbors algorithm, Linear Interpolation and Mean estimation are implemented to estimate the missing values. Once the training data set is prepared, four clustering algorithms are applied including K-means, Neural gas, Agglomerative clustering, and Affinity propagation to segregate the data set into clusters. Both dynamic time warping and Euclidean distance measures are used for each clustering algorithm. All the clustering models form groups of similar power plants based on similar trends and serve as a filtration model for online time series data. Performance analysis for clustering algorithms is done in the last step for determining the most efficient clustering model for our data set in terms of cluster quality score, time, and space complexities.

2) Reconstruction Phase: For reconstruction, online data streaming is obtained for every 5 minutes time stamp. The data is then split into their respective cluster. Once the data is apportioned in their matching group, the disruption is detected. Once the disruption is detected, the disrupted values are estimated. We select the estimation technique that gives the best clustering results on training data. The results of this research show a practical approach for the analysis of disrupted online time series data that is supported with experimental outcomes in later sections.



1.3 Thesis Structure

The thesis is organized as follows:

In chapter 2, we define and compare two popular distance measures used in ML for time series data and compare them.

Chapter 3 addresses some well known and important clustering algorithm which comes under the subcategory of partition based clustering and hierarchical clustering.

In chapter 4, we present several performance measures such as the Silhouette score, Calinski-Harabasz index, Davies-Bouldin index to evaluate model performance and also discuss the importance of evaluation criteria for clustering.

In chapter 5 we describe estimation techniques for the reconstruction of disrupted data.

In Chapter 6 first we give knowledge of our data set and discuss important data pre-processing steps. Then, we show the implementation of clustering algorithms and cluster quality indices on our data set. Furthermore, we present the comparison of clustering algorithms and estimation techniques based on the results of an experiment.

Chapter 7 consists of the conclusion and future work.

2 Distance Measures

A numerical value that describes how close or distant objects are from one another is a distance. A distance function or metric, calculate the distance between pairs of observations in the data sets, presented in the form of a distance matrix or (dis)similarity matrix.

Many clustering methods use distance measures as a notion of (dis)similarity between any pair of observations. The less is the distance between observations, the closer they are, and more similar they are considered [14]. The distance measure between two time series $C = (c_1, c_2, ..., c_i, ..., c_n)$ of length $n \in \mathbb{N}$ and $Q = (q_1, q_2, ...q_j, ..., q_m)$ of length $m \in \mathbb{N}$ is a function taking time series as inputs and returning the distance d(C,Q) between these series fulfilling following conditions;

- d(C,Q) $\geq$ 0 (non negativity)
- d(C,Q) = d(Q,C) (symmetry)
- d(C,V) $\leq$ d(C,Q) + d(C,V) (triangular inequality)

They are useful, as the clustering algorithms crucially rely on them to define a meaningful relationship between the data. For instance, neural gas algorithm, k-means, nearest-neighbors classifiers, agglomerative clustering, etc. are based on the optimal choice of distance measures, as algorithm results may vary in terms of time complexities, cluster sizes, and shapes depending on the choice of distance measures. [15].

In this section, we briefly define and compare some commonly used distance measures in time series data. But, first, an overview of four broad categories of distance measures discussed in [16] by Esling and Agon (2012) are presented, namely, shape-based, feature-based, edit-based, and structure-based distance measures.

Shape based distance measures: It compares time series based on shape formed by their actual values and patterns and are independent of the number of times the patterns get repeated. Euclidean distance, Dynamic time warping, and Minkowski distance come under this category.

Feature-based distance measures: First it extracts features from time series data and then measures the difference between them by defining a distance between features. Usually, it is used when dealing with large data sets of noisy data to reduce the noise and dimensions by extracting features such as sine waves.

Edit based distance measures: It compares series by quantifying the number of operations applied on a series to transform it into another, like update and delete operations, etc.

Structure based distance measures: It shows the similarity between the time series using the high level structure of data that is obtained either after modeling or compression of data.

As we are analyzing time series data in our thesis based on similar patterns and trends for anomaly detection, therefore, we specifically present shaped-based distance measures in this thesis. The clustering algorithms that use shape-based distance measures allocate time series of similar patterns to the same clusters. They are further divided into lockstep measures and elastic measures [17].

Euclidean distance, Manhattan, and Pearson all are **lockstep measures**. They are sensitive to time and phase shift as they can only compare the time stamp i of time series C with the time stamp i of times series Q i.e one-to-one matching and do not allow one-to-many matching between two time series data values. Dynamic time warping is an **elastic measure** because it is good at handling local scaling and time invariance and allows one-to-many or one-to-one matching. It will be discussed in detail in later sections 6. Figure 2.1 shows two time series C and Q that are equal in length and aligned in time.

Figure 2.1: Two time series C and Q that are alligned in time t

Background of distance measures: At the same time, if we shed a light on the little background story of distance measures, various studies conducted on this topic concluded that no one measure has proved to be so far the best one for all types of data. As data is obtained from various sources, it is of different types and limited to a specific domain, so the behavior and performance of different distance measures vary on different data. They are chosen on basis of signal distortions and the domain of the work. For example, Boriah et al in [18] conducted a study for outlier detection in categorical data. They could not decide on a single distance measure to be good but reported the situation for good and poor performing distance measures in the context of outlier detection using ML algorithms. The one from Al Khalifa et.al in [19] tries to access the chemical databases for the compound selection and worked on twelve different methods of distance finding techniques. Not any single method proved to work on all methodologies of clustering. The simplest is the Euclidean distance that is discussed below and the method to overcome its shortcomings is mentioned.

2.1 Euclidean Distance

Euclidean distance (ED) is the distance between the two points and is absolute value for the difference between their coordinates. For two data vectors $C = (c_1, c_2, ..., c_n)$ and $Q = (q_1, q_2, ..., q_n)$ of same length $n \in \mathbb{N}$, it could be expressed in d-dimensional space by using the formula [14]:

$$d_{euc}(C, Q) = \sqrt{\sum_{i=1}^{n} (c_i - q_i)^2} \tag{2.1}$$

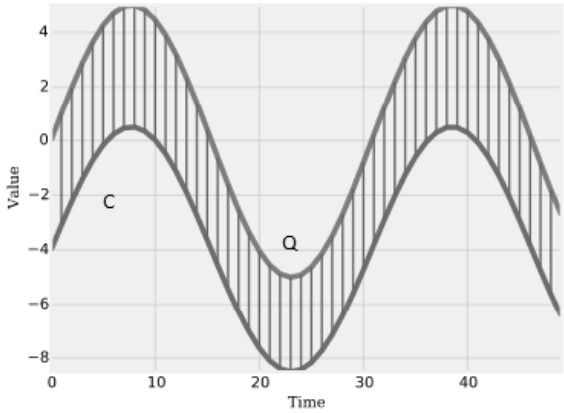


Figure 2.2: ED aligns C and Q by implementing one-to-one matches between the pairs of sequences [20].

Figure 2.2 shows the visualization of ED one-to-one mapping between the two time series C and Q. It is the most popular distance measure for time series or data vectors of the same length based on applicability and effectiveness [15]. It is commonly used in time series or data vectors that are aligned in time and competitive for many problems [21]. For example, for large data sets, many vectors of the same lengths may be present in the data sets so using ED would be sufficient, it calculates the distance between two vectors in one-to-one manner [16]. Nevertheless, in time-series data set, distortion and missing values is a common problem. It leads to miscalculating the distance so we need to either remove the distortion first by normalizing the data then use ED or use other robust distance measures. Even if the problem of noisy data and scaling is handled in a pre-processing step using ED is still not ideal for warping and outliers issues [16] [22].

Below figure 2.3 interpolates the flaws when using ED for determining the distance in high dimension space for disrupted time series data. As it can be seen in the figure that both time series vary in length and the data get missed at q5, one-to-one matching fails and using ED in such case may result in an error.

Since it is not a robust solution to deal with time-series data that is distorted or noisy, dynamic time warping came forward [22].

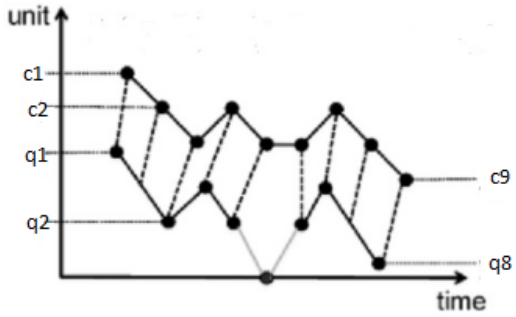


Figure 2.3: Distance of high-dimensional time series data with multiple distortions [14].

2.2 Dynamic Time Warping

When we talk about time series comparison, Dynamic time warping (DTW) is an important and interesting measure. It is a technique for nonlinear alignment of time series [23]. For C and Q time series or data vectors such that $C = (c_1, c_2, ..., c_i, ..., c_n)$ of length $n \in \mathbb{N}$ and $Q = (q_1, q_2, ...q_j, ..., q_m)$ of length $m \in \mathbb{N}$, the elements from C maps to Q or vice versa [15]. The elements $i \in [1 : n]$ and $j \in [1 : m]$ represent the distance between the two points of C and Q in the form of cost matrix $CM \in \mathbb{R}^{n \times m}$ defined by $CM(i,j) := cm(c_i, q_j)$ [24]. A distance is the square of difference between pair of sequence given as,

$$d(i,j) = (c_i - q_j)^2 \tag{2.2}$$

The objective [21] is to form a distance function between the two input data vectors by selecting an optimal warping path such that, $max(n,m) \leq L \leq m+n-1$, where L is the length of warping path. A warping path $P = (p_1, p_2, ..., p_L)$ defines an alignment between data vectors or time series C and Q by associating elements c_i and q_j if $(i,j) \in P$. . An example of DTW is illustrated in Figure 2.4 and Figure 2.5, where for data vectors C and Q the minimum warping path is shown along with the warping matrix that shows an optimal warping path by calculating the distance matrix.

As the possible warping path are exponentially large in number, going through all of them is computationally long [15]. Therefore, for efficiency purposes, it is important to bound the number of possible warping path $P = (p_1, p_2, ..., p_L)$ by introducing constraints that are listed below:

Boundary Condition: The first element from sequence C must map to the first element from the second set of data vector Q or more $p_1 = (1,1)$. The last element from first series must match to last element from second series or more $p_L = (n,m)$. The condition is also called a lower envelope and upper envelope That represents the maximum allowed warping.

Continuity (step-size) condition: The path is restricted to traverse adjacent points in

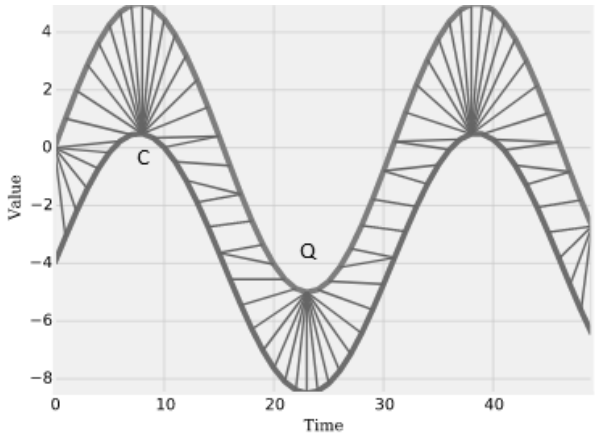


Figure 2.4: DTW aligns C and Q by implementing one-to-many matches between the pairs of sequences [20].

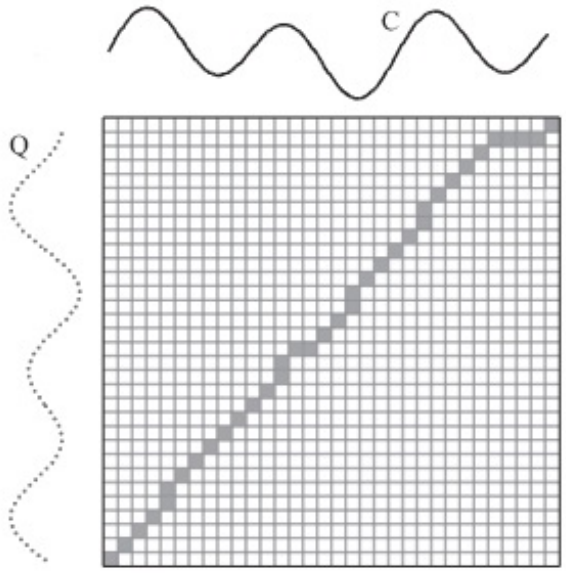


Figure 2.5: Cost matrix showing the optimal warping path P of time series C and Q.

time (not jumping in time). So if, $p_q = (i, j)$ then p_{q+1} is either element $(i+1, j), (i, j+1)$ or $(i+1, j+1)$ for $q = (1, ..., l-1)$ and $i = (1, ..., n-1)$ and $j = (1, ..., m-1)$.

Monotonicity condition: The mapping from first series to other series must be in monotonically increasing order such that $i_{t-1} \leq i_t$ and $j_{t-1} \leq j_t$.

Warping window condition: A fixed warping window size of some width ω (a positive integer) is defined as $|i_t - j_t| \leq \omega$

Slope condition: Slope condition is restricted by avoiding extreme movements in one direction, consequently, the warping path would not fall to extreme minima or maxima.

A warping path is optimal if it satisfies the above-mentioned set of conditions and less computation cost is needed to calculate the distance between the values of each matched pair.

We can now determine the DTW distance by adding and averaging the elements of the warping path with least commulative distance as,

$$d_{DTW}(C,Q) = min\left\{\sqrt{\sum_{l=1}^{L}(p_l)^2}\right. \tag{2.3}$$

where p_l is the distance that corresponds to the l^{th} element of warping path P. Keeping in mind that in the case where the lengths of the vectors n and m are equal, this distance is equal to the ED and only the diagonal of the CM is traversed.

Algorithm 1 Psuedocode of DTW algorithm.

Input: Data vectors $C = (c_1, c_2, \ldots, n)$ and $Q = (q_1, q_2, \ldots, m)$ of length n and m. CM be the Cost Matrix to store similarity measure such that $CM[0,\ldots,m \quad 0,\ldots,m]$ and i and j are loop index. Cost is an integer.

```
CM[0,0]:=0    //Initialize the cost matrix.
for i = 1,2,...,m do
    CM[0,i] := ∞
end for
for i = 1,2,...,n do
    CM[i,0] := ∞
end for

for i = 1,2,...,n do
    for i = 1,2,...,m do
//function to measure the distacne between the two points
        cost := d(C[i],Q[j])
        CM[i,j] := cost + Min(CM[i-1,j],        //increment
                              CM[i,j-1],        //decrement
                              CM[i-1,j-1])      //match
    end for
end for
Return CM[n,m]
```

One of the applications of DTW is in speech recognition for sound pattern matching. For example, a single phrase is spoken by the same person at two different speeds. DTW would compare the phrase and identify it either it matches or not. It is helpful in data mining tasks as it can be used for anomaly and outlier detection [16].

2.3 Comparison of Distance Measures

Among the commonly used distance measures, DTW and ED are the important ones in time series data with a strong intuition that DTW is among the best and ED has some shortcomings to deal with disrupted time series data. ED cannot handle time series of variable length and works on linear alignment of time axis whereas, DTW applies an elastic transformation to a time axis and deals in a better way with non-aligned time series data vectors of variable length. ED holds linear computational complexity of O(n) whereas time complexity of DTW depends on the length of two data vectors to be mapped and is given as O(n x m).

To conclude, DTW can be thought of as an extension of ED, unlike ED it can work with distorted and noisy data. One of the disadvantages of DTW is that it is computationally expensive and time-demanding [15] [16]. The calculation of a minimal path requires a lot of effort. Also, it does not obey the condition of triangular inequality, sometimes giving inadequate results when applied with clustering algorithms [16].

Table 2.1: Comparison of Distance Measures

Distance metric	Time Complexity
Euclidean distance	O(n)
Dynamic Time Warping	O(n x m)

To justify the introduction of these novel distance measures, we have compared them in our experiments presented in section 6. There are some other noteworthy options as well, such as some similarity measures for time series clustering as discussed in [25] are the Cosine measure which is good for different length pattern matching, and extended Jaccard similarity measures, that can be understood as a combination of Euclidean distance and Cosine as it exhibits the properties of both measures. Other popular metrics such as Hamming distance or Levenshtein distance are often used for other forms of data such as text or non-numerical data. Whereas, we have not used them in our experiments due to an increase in time complexities and general elastic measures outperforming the traditional measures. Also, in comparison to ED, no other metric is used as much commonly as ED, according to a review [22] on cluster analysis in health psychology research. Furthermore, the effectiveness of distance measures also depends on the nature of data [15].

3 Clustering Algorithm

Data analysis involves various important steps including the clustering of similar data. The main purpose of clustering is to categorize unlabeled data into subsets to gain insight into the distribution of data. The distribution of these subsets depends upon the exclusive characterization of the data such as the degree of association is strong within the data in the same group than the data in other groups. Data division helps to observe the characteristics of each cluster and to use the insight for further analysis. In the last few decades, clustering of time series sequences has received significant attention not only as a powerful stand-alone tool but also as a pre-processing step or subroutine for other Data mining (DM) tasks [26].

Time series clustering can be achieved using two of the approaches [16]:

Subsequence clustering: Extract a portion of a time series data and perform clustering on it provided there is individual time series data.

Whole series clustering: Perform clustering on sets of time series data and group similar series into a cluster provided there is a set of more than one time series data.

We have used the later one in our experiments as we have sets of time series data.

In this section, we discuss methods for obtaining clustering on the time series data. Many traditional methods are available in the literature and they are classified mainly into five types: hierarchical clustering, partition-based clustering, density-based clustering, model-based clustering, and grid-based clustering [16]. However, traditional clustering methods can be applied on time series data but the selection of distance measures is even more important than the selection of clustering algorithm [16]. According to the literature partition and hierarchical based clustering methods have appeared to be most popularly used on time series data, we discuss these two types of clustering methods with some of the algorithms that are available under these categories, such as K-means, Neural Gas Algorithm, Affinity propagation, and Agglomerative clustering.

Based on the assignments of data points to groups clustering methods can be broadly distinguished [27] as:

Hard clustering methods: In hard clustering, every observation belongs to exactly one cluster. That means all the data points are grouped in such a way that no data point can belong to more than one cluster, resulting in non-overlapping clusters.

Soft clustering methods: In soft clustering, observations can belong to multiple clusters, often accompanied by probabilities of belonging to each cluster, often resulting

in overlapping clusters.

For both types of clustering, there are different algorithms. In this chapter, we discuss some of those algorithms.

3.1 Partition based clustering

Partition based clustering divides the dataset Z of total number of d-dimensional observations $C = (c_1, c_2, ..., c_n)$, $C \subseteq Z$ into k number of clusters $M = \{M_1, ..., M_k\}$ such that $k \leq n$ when k is pre-defined. The goal of such clustering methods is to optimize clustering by minimizing the sum of the distance of each point in the cluster to the center. Given the input vector $C \in \mathbb{R}^{\mathrm{d}}$, a finite set of feature vectors B (or codebook) $w_i \in B$, (i,....,n) is employed to describe a winning feature vector w_j of B. A winning feature vector is the one with least squared error $d(c, w_j) = \|c - w_j\|^2$ and is minimal. A commonly used partitioning algorithm is K-means that form clusters around the mean value of the observations which have variations of Clustering Large Applications (CLARA) and Clustering Large Applications based on Randomized Search (CLARANS) popularly used today for clustering [23]. Another important algorithm for partition-based clustering is Neural Gas which is the generalization of K-means. And the last one which we have discussed is the Affinity propagation that is exemplar based clustering.

3.1.1 K-means

The idea of the general k-means algorithm was first put forward by Stuart Lloyd in 1956 which was later mentioned also by Edward W. Forgy, therefore sometimes referred to as Forgy's Algorithm. However, The term K-means was first introduced in 1967 by J.B. MacQueen [28]. It is still a popularly used clustering algorithm.

K-means comes from the family of center-based clustering algorithms since the data points are represented using several cluster centers or centroids [29]. The first step of K-means is to input an initial number of clusters k to be formed. As soon as the value of k is decided, the next step is to decide centroids called prototypes. For this, we can either use a random approach given by Forgy Method where prototypes from the observations are picked up at random. Or we can use the other method called as Random Partitioning method, where we randomly assign each observation to a cluster and take the mean of each cluster to initialize prototypes [29]. For standard K-means working, the Forgy method is preferable. For a detailed overview on the comparisons of different initialization methods with their time complexities, we can refer to Celebi et al. research [28].

Working of K-Means Algorithm:

1. Given the set of n observations $(c_1, c_2, ..., c_n) \in \mathbb{R}^{(d)}$, number of desired clusters k and set of prototype as $(w_1, w_2, ..., w_k)$.

2. Assign each data point or observation c_i to closest prototype w_j i.e with least squared ED and form k clusters $M = \{M_1, ..., M_k\}$ such that $k \leq n$

$$M_j{}^{(t)} = \{c_i : \|c_i - w_j^2\|^2 \leq \|c_i - w_i^2\|^2 \quad \forall i, 1 \leq i \leq k\}$$

   where each observation is assigned to only one cluster M_j where (j=1,...,k) in each iteration t.
3. Update the centroids or mean for assigned observations to each cluster.

$$w_j{}^{(t+1)} = \frac{1}{|M_j{}^{(t)}|} \sum_{c_i \in M_j^{(t)}} c_i$$

4. Repeat steps 3 and 4 until assignments are not changing anymore.

The algorithm is based on alternating between step 3 and step 4, assignments of observations to closest prototype, and updating the mean of clusters. Geometrically, it results in partitioning the data space into Voronoi cells formed by prototypes. This process is repeated until the centroid value of the clusters is not showing a considerable change for a couple of iterations.

The goal of the algorithm is to optimize the cost function defined as:

$$J = \min_{M} \sum_{j=1}^{k} \sum_{c \in M_j} \|c - w_j\|^2 \tag{3.1}$$

where w_j is the mean of data points in Mj, for j=(1,...,k).

Advantages:

- K-means is believed to process large data efficiently because of its simple structure formation and economical computation [29].
- Always guarantees convergence.
- Smartly adjust to new observations.

Disadvantages:

- It is believed to be an NP-hard problem in nature i.e it is hard computationally that it finds global minimum in polynomial time. Trying out all possible combinations

of clustering solutions, (which is impractical even for small data sets) may converge at local optima [29].

- It does not promise to produce the same results at each run as it is sensitive to initialization. Different initialization produces different results.
- It is sensitive to noisy data such as outliers.
- The choice of distance measure is significant. For example, K-mean with ED is found to be not as efficient in time series data as time and phase shifts may occur within the time series in the same clusters. Whereas, with elastic measures such as DTW the results are proved to be improve presented in chapter 6.

3.1.2 Neural Gas

Neural Gas (NG) is a neural network-based clustering algorithm introduced by Klaus Schulten and Thomas Martinetz in 1991 [30]. NG is a generalised form of K-means algorithm and is used for the optimal representation of the data based on number of typical prototypes called feature vectors. Utilizing a concept of neighbourhood ranking additional semantical insight is gained in NG algorithm. Neighbourhood ranking $(w_{i_0}, w_{i_1}, ..., w_{i_{n-1}})$ of the feature vectors with (w_{i_0}) determined as the closest to c_i and (w_{i_1}) as second closest within the range of (w_{i_k}), $(k = 0, ..., n)$ with $\|c - w_j\| \leq \|c - w_{i_k}\|$.

The algorithm is called "neural gas" because of the dynamics of the feature vectors during the adaptation process, which distribute themselves like a gas within the data space [30].

Working of Neural Gas:

1. Suppose data vectors $(c_1, c_2, ..., c_n) \in R^d$ that are distributed in a data space described as P(c). NG aims to locate and rank the prototypes $w_j \in R^d$, (j= 1,...,k) using the distance function such that the cost function [27] is minimised given as:

$$E_{NG}(w) = \frac{1}{2C(\lambda)} \sum_{j=1}^{k} \int h_\lambda(K_j(c, w)).\|c - w_j\|^2.P(c)dc \tag{3.2}$$

where $\|c - w_j\|^2$ is the squared ED and $K_i(c, w_i) = |\{w_j | d(c, w_j) \leq d(c, w_i)\}|$ is the rank of the prototypes sequenced according to the distances from the minimum to maximum distance value and λ indicate the neighbourhood range such that $\lambda \geq 0$. $C(\lambda)$ is the constant term $\sum_{j=1}^{k} h_\lambda(K_j)$ and $h_\lambda = exp(\frac{t}{\lambda})$ is the Gaussian shaped curve .

2. The adaptation step of the neural gas can be interpreted as gradient descent on a cost function yielding learning rule for prototypes as,

$$\Delta w_j = \epsilon . h_\lambda (K_j(c_i, w_i)).(c_i - w_j) \tag{3.3}$$

The w_j obeys the stochastic gradient descent on the cost function in NG with the increase in rank of the prototypes of feature vectors, the step size is decreasing and adapted not just by the closest feature vector but all them promising a robust convergence than K Means [27].

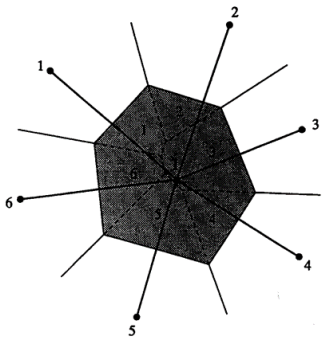


Figure 3.1: Description of update rule for the connections of the network [30]

.

Fig 3.1 illustrates the schematics of neural nets that are connected by an adaptation rule. Neural unit i is indicated as a winner and nominated for the input signal shown in a shaded area of the Voronoi polygon related to the neurons. The number ranges from 1 to 6 represents those prototypes that are second much closer to the input signal [30]. After sufficiently many adaptation steps the feature vectors cover the data space with minimum representation error.[5]

Advantages:

- The NG model does not delete a node and also does not create new nodes.
- NG is not sensitive to initialization.
- Apart from a reduction of the influence of initialization, NG offers a possible solution as a robustly converging algorithm by covering the feature space with less representation error [27].
- By adapting not only the closest feature vector but all of them with a step size decreasing with increasing distance order, compared to k-means clustering a much more robust convergence of the algorithm can be achieved.

Disadvantages:

- The determination of neighbourhood ranking takes high computation power then K-means on large data sets.

3.1.3 Affinity Propogation

A new type of clustering approach known as "Affinity Propagation (AP)" is the partition-based clustering method. It has been evolved over some time to overcome clustering issues like pre-specifying the number of clusters. It was published by Frey and Dueck in 2007 [31] [32] [33], since then it is of great importance in clustering tasks, with fewer errors and speedy performance comparative to traditional methods. The solution of corresponding clusters gradually emerges in AP and the clusters formed contain "exemplar" data points that act as the representative of clusters [31].

Particularly, AP aims to search exemplars by passing the messages between data points and selecting a candidate exemplar. It can be thought of as an election as every data point can vote for other data points and some candidates have more votes than the other to be selected as the exemplar. By continuous message exchanging the deserving exemplar is chosen to represent a cluster. Most importantly, every data point is considered as a potential exemplar, initially [33]. Further, we discuss the mechanism of how this all works together to select exemplar and form clusters.

The input is a set of real-valued pairwise (dis)similarities between data points in a distance matrix d(i,j), where $\forall i \in \{1, ..., N\}$ and j is exemplar. It shows how well suited is the point j to be selected as an exemplar for i points. Commonly, the (dis)similarity between samples is expressed by negatively squared ED measure as:

$$d(i,j) = -\|c_i - q_j\|^2, \quad i \neq j \tag{3.4}$$

In the diagonal of the distance matrix is the global shared preference 'Pr' [34]. 'Pr' is used as a control knob to govern the number of clusters found by AP, often called a set of self-similarity d(i,i) = Pr. Pr should be a common value as the chances for all data points to be a potential exemplar is the same [31]. Pr is a significant factor, taking Pr minimum would result in less number of clusters where Pr as median similarity may result in forming a moderate number of clusters. To conclude the importance of selecting Pr, the higher values of Pr lead to form more clusters, while the effect with low preference is the opposite of it.

Moreover, the output is the exemplar for each cluster. The continuous message passing between the candidate exemplar and the data points indicates the affinity they have for one another. The two messages being exchanged between data points are the responsibility R(i,j) and availability A(i,j) represented in the respective matrix R, A $\in \mathbb{IR}^{(n*n)}$.

Figure 3.2 demonstrate the relationship between the two messages being passed between data points in AP as responsibility is the message sent by data point i to j to express how good j is to represent i, consequently, j sends an availability message to i data point to notify how well-suited j is as its exemplar.

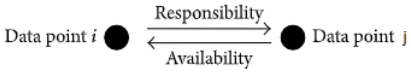


Figure 3.2: The relationship between the two messages of responsibility and availability [30]

.

Working of Affinity Propogation:

1. Initialize R and A with zero.
2. The responsibility messages are updated as follows:

$$R^{(t)}(i,j) = d(i,j) - \max_{h \neq j} \left\{ A^{(t-1)}(i,h) + d(i,h) \right\} \tag{3.5}$$

   where $R^{(t)}(i,j)$ represent the degree that data point i supports j to be an exemplar, $A^{(t-1)}(i,h) + d(i,h)$ measure the ability of keeping data point h to be an exemplar of i. The greater value of $R^{(t)}(i,j)$ show maximum likelihood j as an exemplar of i.

3. The availability messages are updated as follows:

$$A^{(t)}(i,j) = \min \left\{ 0, R^{(t)}(j,j) + \sum_{h \neq i, h \neq j} \max\{0, R^{(t)}(h,j)\} \right\} \tag{3.6}$$

   where $A^{(t)}(i,j)$ represent the likelihood of j as an exemplar based on its positive responsibilities, $\max\{0, R^{(t)}(h,j)\}$ indicate that only positive values of $R^{(t)}(h,j)$ are taken. Equation (3.6) limits the entire sum to zero in order to limit the effects of significant positives $R^{(t)}(j,j)$,

4. It progresses by updating responsibility given the availability, and updating availability given the response at each iteration. For diagonal values A(i,i) matrix is updated as:

$$A^{(t)}(i,i) = \sum_{h \neq i} \max \left\{ 0, R^{(t)}(h,i) \right\} \tag{3.7}$$

5. Recent cluster assignments can be obtained at any time by adding responsibility and availability messages together. Where, the max$\{A(i,k) + R(i,k)\}$ yields the exemplar point k for each point i.

6. The algorithm terminates when the values of A and R remain constant for a certain number of iterations [32].

The understanding of AP execution can be made more clear through figure 3.3, which depicts the standard working of the AP algorithm step by step at each iteration. The

data points are spread in a network-like structure and the blue arrows show the message exchanging procedure between them and the black arrows shows how they identify the exemplar by combining availabilities and responsibilities for each data point.

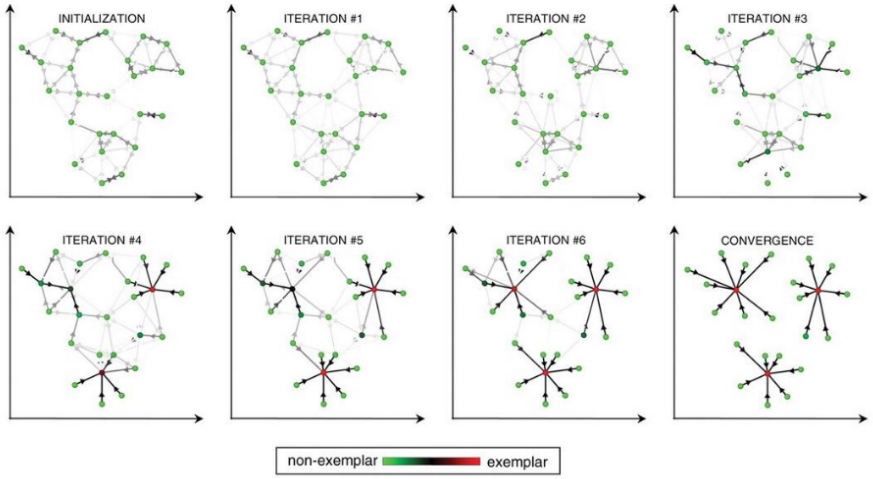


Figure 3.3: A typical representation of how affinity propogation works [35]

While updating messages, AP encounters oscillation. This happens when there are more than one exemplar suitable for a cluster [36]. The oscillation needs to be addressed to avoid confusion in selecting the optimal exemplar. The damping factor Λ, which ranges from 0 to 1, is used for this purpose. Each message's value is multiplied by Λ times its previous iteration's value plus $1-\Lambda$ times its recommended updated value. R and A updates with Λ can be rewritten as:

$$R^{(t)}(i,j) = (1-\Lambda)d(i,j) - \max_{h \neq j}\left\{A^{(t-1)}(i,h) + d(i,h)\right\} + \Lambda R^{(t-1)}(i,j) \tag{3.8}$$

$$A^{(t)}(i,j) = (1-\Lambda)\min\left\{0, R^{(t)}(j,j) + \sum_{h \neq i, h \neq j} \max\{0, R^{(t)}(h,j)\}\right\} + \Lambda A^{(t-1)}(i,j) \tag{3.9}$$

$$A^{(t)}(i,i) = (1-\Lambda)\sum_{h \neq i} \max\left\{0, R^{(t)}(h,i)\right\} + \Lambda A^{(t-1)}(i,i) \tag{3.10}$$

Equations (3,8–3,10) are used by AP to repeatedly update R and A. Data point i is an exemplar, if A(i, j) + R(i, j) is the maximum and i = j; otherwise, j is the exemplar, and i is allocated to the cluster whose exemplar is j. After identifying exemplars AP distributes non-exemplar samples to the cluster with the shortest Euclidean distance between exemplars [36].

Advantages:

- Powerful clustering algorithm compared to traditional clustering methods because of its effective and accurate results.
- General applicability and good performance for sparse data sets.
- Do not need to specify the number of clusters initially, automatic selection of several cluster formations.
- It handles asymmetric data when input data is different types of proximity data.
- Not sensitive to initialization, unlike k-means.

Disadvantages:

- Damping the messages with large value can some time slow the process of clustering [37].
- For large-scale datasets, AP performance becomes inefficient as AP works by estimating three matrices, responsibility, availability, and similarity, its time complexity scale quadratically with the number of data points.
- To reach the desired cluster solution it sometimes becomes hard to choose optimal Pr because it requires to re-run the algorithm every time the Pr-value is changed.
- Despite that AP is effective and accurate in normal data clustering, the clustering algorithm cannot deal with the nested clusters, which have different data densities.

3.2 Hierarchical clustering

Hierarchical clustering [38], as the name suggests, creates a multilevel hierarchy of clusters organized in a cluster tree called a dendrogram. In a cluster tree, initially, all the observations are considered as single element clusters, as the level rises, they merge into bigger clusters, step by step, based on similarity. Unlike k-means, it does not require to specify an optimal number of clusters, but the choice of distance measure is important [39].

The dendrogram represents data graphically which helps the data analyst to closely look into the data distribution, it has a complete record of splits or merges of clusters. In figure 3.4, a dendrogram with 22 observations can be observed. Y-axis shows the formation of the cluster by observations, while the x-axis shows the distance between them. The joining of two horizontal lines represents the fusion of clusters and the distance between them can be interpreted by looking at the vertical small bars on the X-axis. Observations 6 and 13 are outliers as they are merged at much higher distances.

Hierarchical clustering can happen from either downwards to upwards or upwards to downwards in a tree. Both the approaches have names, commonly known as Agglom-

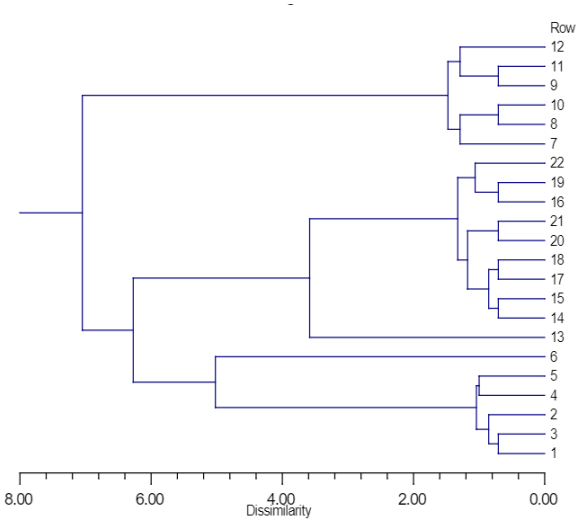


Figure 3.4: A typical representation of a dendrogram [40] .

erative Nesting (AGNES) and Divisive algorithms. Where the latter one is a little less known than the first one.

3.2.1 Agglomerative Nesting:

The commonly used hierarchical algorithm Agglomerative Nesting (AGNES) merges the similar data into clusters at each level until k number of clusters are formed, or it converges after the formation of one last cluster (root) of the tree.

Working of AGNES Algorithm:

1. Initialize n observations as n initial clusters.
2. Calculate the (dis)similarity between each pair of the cluster using distance measure and form a distance matrix.
3. Find the cluster with the least distance in between.
4. Combine the closest cluster.
5. Update the distance matrix.
6. Repeat steps 2 to 6 until n=1 or All clusters agglomerate to form one.

The figure 4.3 above demonstrates the simple working of AGNES. At the initial level, pairwise distance of i and j cluster is the lowest, so they merge to form a new cluster $m_{ij} = i \cup j$. The row and column from the distance matrix of the observation i and j is omitted. A distance matrix is updated at this point by adding a new cluster m_{ij}. Now the new lowest distance clusters are searched by calculating distance again but this time using the formula from Lance and Williams distance update formula,

$$d(i \cup j, k) = \alpha_i d(i,k) + \alpha_j d(j,k) + \beta d(i,j) + \gamma |d(i,k) - d(j,k)| \tag{3.11}$$

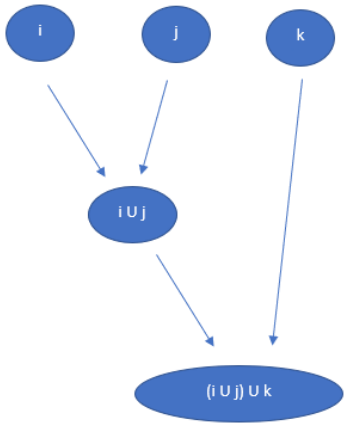


Figure 3.5: Basic working of agglomerative algorithm

Here $d(i \cup j, k)$ is the distance between clusters m_{ij} and m_k and α_i, α_j , β and γ are parameters that together with the distance function, determine the method for agglomerative hierarchical clustering. According to this formula, the lowest distance clusters m_{ij} and m_k are merged to form a new cluster at level 3 and iteratively it converges as soon as a single cluster is formed to represent all observations at the final level. Certain approaches to perform similarity checks between clusters are important.

Single Linkage (SLINK) It is the standard form of agglomerative clustering which finds the minimum distance between a pair of observations and merges them. This approach is believed to be sensitive to outliers [41]. Also referred to as nearest neighbour clustering. It is given [42] as:

$$d(i \cup j, k) = min\{d(i,k), d(j,k)\} \tag{3.12}$$

The coefficients for Lance-William distance update rule is replaced as $\alpha_i = \alpha_j = 0.5, \beta = 0, \gamma = -0.5$

Complete Linkage(CLINK) It tends to find a maximum distance between pairs of observations by traversing and calculating the distance between all the possible pairs of two clusters and merge the highest distance pair to form a cluster. It is less likely to get affected by outliers and noise in the data, often yield well-separated clusters but the drawback of this approach is that sometimes it breaks massive clusters [40] formulated as:

$$d(i \cup j, k) = max\{d(i,k), d(j,k)\} \tag{3.13}$$

The coefficients for Lance-William distance update rule is replaced as $\alpha_i = \alpha_j = 0.5, \beta = 0, \gamma = -0.5$

Average Linkage (ALINK) It is the combination of above two strategies, and is

defined as the average distance between each member of clusters, formulated as:

$$\frac{1}{|M_i|}\frac{1}{|M_j|}\sum_{i\in M_i, j\in M_j} d(i,j) \tag{3.14}$$

The coefficients for Lance-William distance update rule is replaced as $\alpha_i = \frac{n_i}{n_k}, \alpha_i = \frac{n_j}{n_k}, \beta = 0, \gamma = 0$

Ward method Joe H. Ward Jr. In 1963 [43] came forward with this method where the criteria were based on the objective function of a sum of square error. It aims to optimize the objective function by searching for the clusters that increase lower inter-cluster variance after merge operation is performed on them. Distance between two clusters is defined with less increase in sum of squared error as:

$$d(i,j) = SSE_{i\cup j} - SSE_i - SSE_j \tag{3.15}$$

where i and j are two clusters. The coefficients for Lance-William distance update rule can be replaced as: $\alpha_i = \frac{n_i+n_j}{n_k+n_m}, \alpha_j = \frac{n_j+n_m}{n_k+n_m}, \beta = \frac{-n_m}{n_k+n_m}, \gamma = 0$

The benefit of this method is that it decreases inter-cluster variance.

Apart from the linkage criterion, appropriate distance measures as mentioned in chapter 2 are also applicable to decide which cluster to merge with the other. The choice of distance metric plays a vital role in deciding the algorithm time complexity. Besides, it is notable that SLINK, CLINK, AND ALINK apply to random distance metrics while the Ward method is limited to ED. The similarity distance results may vary while using different distance measures to calculate pairwise similarities between clusters, as it influences the shape and size of clusters [38].

As the distance is calculated at each step for each pair of observations, handling large sets of data is a slow process in Agglomerative clustering, so time complexity as given in [38] is $O(n^3)$, where N is the number of observations in the dataset. The complexities are superior for some efficient methods of agglomerative clustering such as SLINK and CLINK are $O(n^2)$ [44]. Thus, hierarchical methods are preferable for smaller datasets [38].

Where AGNES has proved to be used widely in many science and medical fields today, it has a profound impact on ML. For example, it has been commonly used for social networking analysis nowadays and for outlier detection in ML. At the same time, we cannot agree more with the authors in [42] that it has great room for improvements and moderation. This topic is of great interest and requires some attention.

Advantages:

- No need to specify the number of optimal clusters to be formed.
- Desirable number of clusters can be achieved by pruning the tree at any level.
- Well managed and ordered form of data is informative for data analysis.

Disadvantages:

- The merging or splitting cannot be undone once performed in a dendogram. i.e it is less flexible to change.

3.2.2 Divisive Algorithm

Divisive or DIANA (Divise Analysis) typically means a single cluster division into sub-clusters. It is an inverse of AGNES and uses a top-down approach to form a hierarchy of clusters. This is however possible when all data points belong to a single cluster initially and later divides into two, at each level, in an iterative manner, until all the data points are split to represent a cluster. Divisive clustering with an extensive search is $O(2^n)$, which is even worse than AGNES [44].

4 Clustering Evaluation Strategies

The evaluation of the clustering algorithm is considered as one of the important steps for the success of the clustering problem [45]. In this chapter, we answer the question of how to evaluate clusters that are formed using the clustering algorithm discussed in chapter 3 and how to determine the value of k i.e number of the cluster?

To evaluate the cluster quality [46], the goal is to find out the ideal cluster partition. The ideal partition is possible if the compactness is minimum and separation is maximum between clusters. Compactness is the term that is used to define the closeness of data points that belong to the same cluster. Separation defines how separate the clusters are from one another.

There are three different criteria to evaluate the results of clustering algorithms [45]:

External criteria: External criteria measure performance of clustering based on previous information about the data.

Internal criteria: Internal criteria measure performance of clustering based on information inherent to the data themselves.

Relative criteria: Relative criteria measure the performance of clustering by comparing it with different clustering models.

External and Internal criteria take high computation power and depend on statistical measures [45]. Therefore, we will focus on cluster validity indices that come under the category of relative criteria. There are several cluster validity indices for evaluating cluster quality based on relative criteria [47]. A few of them are the Silhouette score, Davies Bouldin score, and Calinski Harabasz score discussed further.

4.1 Silhouette Score

Silhouette Coefficient (SC) is the evaluation score that is calculated for each sample of data and is proposed by Kaufman and Rousseeuw (1990). The evaluation score determines a numerical value indicating how similar the data sample is to its cluster than other clusters. The resulting value is illustrated for its range to facilitate the evaluation interpretation. It can be calculated using the Euclidean distance measure. It works on the principle of compactness and separation and can be calculated for a single

sample c_i from a cluster $M \in \{1, ..., k\}$ as [47],

$$S(c_i) = \frac{a_{m,c_i} - b_{m,c_i}}{\max(a_{m,c_i}, b_{m,c_i})} \tag{4.1}$$

where a_{m,c_i} is the average distance between a sample point c_i with other samples c_j in its own cluster m and can be given as:

$$a_{m,c_i} = \frac{1}{|m| - 1} \sum_{j \in m, c_i \neq c_j} d(c_i, c_j) \tag{4.2}$$

whereas b_{m,c_i} shows the average distance between the sample c_i with all the samples of neighbouring cluster m_k except for the cluster the sample is in and can be given as :

$$b_{m,c_i} = \min_{k \neq i} \frac{1}{|m_k|} \sum_{c_k \in m_k} d(c_i, c_k) \tag{4.3}$$

where $m_k \neq m$. The final silhouette score can be achieved by calculating the mean silhouette score of each sample also known as the Silhouette width criteria (SWC).

$$SWC = \frac{1}{N} \sum_{j=1}^{N} S(c_j) \tag{4.4}$$

The score ranges from [-1,1]. For each sample, a score close to 1 indicates the best matching to its cluster. The negative score indicates, that the point is assigned to a non-similar cluster whereas the value equal to zero shows overlapping clusters.

The optimal value of k according to silhouette index is the one that maximizes the value of SWC. To determine the optimal value of k the elbow graph with a scoring parameter metric set to the silhouette is presented in chapter 6.

4.2 Calinski-Harabasz Index

Calinski-Harabasz (CH) [47] index also referred as variance ratio criterion is proposed by Calinski and Harabasz (1974). CH is the ratio of sum of squared distance between the clusters and within clusters and computed for each value of k as:

$$CH(k) = \left(\frac{n - k}{k - 1} \right) \frac{SS_{bc}}{SS_{wc}} \tag{4.5}$$

where SS_{bc} is the sum of squares distances between cluster centroids or between-cluster dispersion matrix, SS_{wc} is the sum of squares distances within cluster or within-cluster dispersion, n is the total number of sample data points and k is the number of clusters.

$$SS_{bc} = \sum_{m=1}^{k} n_m(\mu_m - \mu)(\mu_m - \mu)^T \tag{4.6}$$

where n_m is the data points in cluster m and μ_m is the center of cluster m whereas μ is the center of data set.

$$SS_{wc} = \sum_{m=1}^{k} \sum_{i=1}^{n_m} n_m(c_i - \mu_m)(c_i - \mu_m)^T \tag{4.7}$$

where c_i are data points within cluster m and μ_m is cluster centre.

Therefore, CH index can be rewritten as:

$$CH = \left[\frac{SS_{bc}}{k-1}\right] \left[\frac{SS_{wc}}{n-k}\right] \tag{4.8}$$

The score value ranges from $[0, \infty]$. For better cluster partitioning the score should be large. To determine the optimal value of k the elbow graph with a scoring parameter metric set to Calinski-Harabasz is presented in chapter 6.

4.3 Davies-Bouldin Index

Davies-Bouldin (DB) [47] Index was first formulated by David L. Davies Donald W. Bouldin (1979) and is today used for the evaluation of clustering algorithms. Given the distance measure Y_{ij}, which is the ratio of similarities within cluster M_i and M_j over the distance between similar clusters, it is formulated as:

$$Y_{ij} = \frac{S_{M_i} + S_{M_j}}{d_{ij}} \tag{4.9}$$

where S_{M_i} and S_{M_j} is the average distance of all data points to its cluster center and d_{ij} is the distance between cluster centers.

DB index is defined as the sum of the average similarity of clusters Mi,...,Mn with its most similar cluster. It can be equated as :

$$DB = \frac{1}{M} \sum_{i=1}^{k} \max_{i \neq j} Y_{ij} \tag{4.10}$$

If M is the total number of clusters, then, the higher the DB index, the bad is the clustering results, and the lower the DB index, the better is the result of clustering. It

normally fluctuates from zero and above, where zero is considered as the lowest possible result for better clustering evaluation.

4.4 Comparison of Cluster Validity Indices

When comparing the above indices we can say that only an SC out of the three is bounded to a range of 1 to -1 whereas, for better partitioning, the CH index and SC should be large whereas the DB index should be small. The SC has high time complexity whereas the CH index is fast to compute and DB is simpler in computation than a silhouette. The drawback of the DB index is that it is limited to use ED as the only distance metric. The SC, DB, and CH are generally higher for convex clusters than density-based clusters.

Table 4.1: Comparison of cluster validity indices

Performance Metric	Time Complexity	Distance Metric	Benchmark
Silhouette Score	$O(mn)^2$	All	Higher value better result
Davies Bouldin Index	$O(nC)^2$	Euclidean	Lower value better result
Calinski Harbasz Index	$O(nm)$	All	Higher value better result

5 Estimation of Disrupted data

5.1 Introduction

Disrupted data such as missing values is a concerning problem as most ML algorithms require complete data for data analysis. It can harm classification, regression problems, and forecasting tasks. The values of missing data need to be filled with reasonable values by some estimation methods in the data preparation stage called pre-processing. Pre-processing is an initial step in DM in which problems related to data quality are identified and handled. Missing values handling is included in the pre-processing stage [48].

For identification and handling the disruption in the data, first, we need to understand what type of data in our dataset is inappropriate or does not lie in the category of normal data. For example, If the data values are too noisy to handle as it has outliers then it is considered an anomaly or disruption. In the next chapter, we will discuss the type of outliers we encountered in our data set and also show practically how we handled them.

Disrupted data handling techniques consist of popular data estimation methods. For more general data sets there are simple methods often called traditional methods and others are a bit more complex than traditional methods known as imputation methods. These techniques do not let the data set compromise it's quality by imputing or estimating the values where the data is being disrupted.

Some traditional methods include mean methods, deletion methods, and mode methods which are used widely due to their simplicity but they are effective when there is a low percentage of missingness. Moreover, they can cause bias to the data and are not considered as the best methods for dealing with missing values [48]. Other methods produced by some algorithms such as K-Nearest-Neighbour (KNN) etc. provide a good solution to decrease the partiality caused by the traditional method to the data. So that at final, the data will be complete and prepared to utilize for the next step of DM [48].

A comprehensive classification and comparison of the various methods divide data imputation techniques into subclasses as deletion-based, statistical, regression-based, classification-based, nearest-neighbor-based, expectation-maximization (EM) based Multi-Layer perceptron (MLP) based and deep learning (DL) based.

Since missing data is an immense field of study in DM, in recent decades the solution to handle it has gained much attention in the field of AI, ML, and statistical analysis. [37][38][[39]. Missing data handling techniques have wide application areas of business, healthcare, education, etc, There are so many techniques that have been proposed in

the literature from the simplest one to the complex one. Each with its advantages and disadvantages.

5.2 Missing Data Mechanism

Data is measured under certain circumstances and sometimes it is important to know the way data gets missed, MDM specifies the standard ways the data is being missed. Rubin (1976) et al. [49] outlined a classification system defining three categories of missing data which are discussed below.

Missing completely at random (MCAR):

In this category, the data being missed entirely at random occasions, which means the probability for a specific variable being missed is independent of the value of any other variable and also the rest values of a variable itself that is being missed. Let suppose ci is the missing variable. In our case, as we have univariate data, where there is no other variable involved other than the implicit variable time, the above two conditions can be replaced by just one condition as the probability for a specific variable data ci being missed is independent of a moment this variable is observed in the series.

For example, sensor data is recorded of power plants and sent to the backend, the transmission is missed due to some unknown reasons and sensor recording is failed at random time intervals. Since it has no dependency on the data that is being missed it is referred to as MCAR.

Missing at random (MAR):

MAR mechanism specifies that the probability of the samples being missed is random. That means the probability of the samples, suppose ci being missed is independent of the values of the sample ci held by itself but dependent on the values of other variables qi. For the case of univariate time series data, since there are no other variables than the time itself, so it can be assumed that the values being missed depend on the specific moment this sample is observed in time series data.

For example, nuclear power plants have a lifespan of 20 to 40 years so it is likely that they will shut down after a certain age limit, due to such uncertainty at a particular time the sensor data may get disrupted. Another example considering data obtained through solar plants, it is likely to get more missing values at night as compared to day time because of the absence of sun at night.

Not missing at random (NMAR):

If the above-mentioned two mechanisms are not the case then the way data is missed

can be the NMAR mechanism, which is quite different. Here the probability of the data being missed for a variable ci is related to the hypothetical values of the observation ci itself. Sometimes it may or may not depend on other variable values.

For example, the sensor for monitoring temperature may stop showing values if the temperature decreases below -50 degrees C or rises above 100 degrees C.

For this case, as there is a reason that the respondent did not fill up the field with reasonable data, data is missed due to a reason so it becomes important to search what is the reason, we need to inspect, before straight away jumping to the step of imputing the values.

5.3 Estimation Methods for Treating Missing Data

In the next section, we discuss some available methods to estimate missing values used in this field of study. We divide simple methods as traditional methods and a bit complex ones as modern methods. The discussion aims to help us gain basic knowledge of methods commonly used now for data imputation along with some of the pros and cons of using each one of them.

5.3.1 Traditional Methods

From the list of statistical methods, some simple missing data handling techniques are: mean, mode, median imputation, deletion, and ignoring methods. They rely on the particular column to impute missing values and are called univariate or single imputation techniques. The input values by looking in a single feature that has a missing value to be imputed.

One way to deal with missingness is to completely leave it as it is by ignoring them completely, however, it is a bad idea if the percentage of missingness is large. It will affect further data analysis in a poor way. Another simple method is to delete the entire set of values that do not behave as normal data. It is risky in a way because in case of a high percentage of missingness, deleting too many data values can cause information loss.

Another traditional method is the **Linear interpolation method**. It is a deterministic method that searches for a line between two variables and interpolates in between the interval of these point variables with the help of the slope function. For a complete data set, linear interpolation is achieved by the aggregation of a small linear interpolant between each pair of data. As a result, a continuous curve is formed. Hence, the missing point (suppose x) between two linearly spaced points (Suppose Xa and Xb) is interpolated using this method.

The next method is the **Mean imputation method** as discussed in [48]. This method works by simply estimating the missing values by taking the average of other (available) values of the same column in the data. This comes forward as a solution to deletion and ignoring methods as the number of data values remains the same after the imputation in this method.

These methods may solve the missing value problem but give rise to other types of data disruption issues. A large batch of missing values will be replaced by mean values causing too many similar values or in some cases outlier values. Moreover, the drawback is that this method is known to cause bias in the data by the amount of too many values which are similar to each other.

Other similar types of methods are the mode method and median methods that deal with missing values by replacing them with median and mode values of observed data. These still do not provide the best solution for a large amount of missingness and may give rise to other problems such as contextual outliers in the case of many values missing at the same interval. Therefore we seek better methods that can be used in time series data for missing data imputation.

5.3.2 Modern Methods

From the list of modern imputation methods, we discuss K-Nearest-Neighbour(KNN).

KNN estimation

The KNN estimation uses the K-nearest-neighbour model to predict missing values. It belongs to the class of neighbor-based imputation methods as it looks into the k nearest neighbors to replace the missing value in time series data. The nearest neighbors are those which are closest to missing values and are calculated using ED (by default). For example, depending on the input value of K, the values of neighbors are taken mean or weighted by distance to impute missing data value.

It can be applied to any type of data be it discrete, continuous, or categorical, which makes its applicability easy, also that it does not imply a specific missingness mechanism and works fine under any missingness mechanism. It surpasses the common methods of imputing such as averaging or imputing zeros in the missing value places.

5.3.3 Comparison of Estimation Methods

To conclude, the traditional methods of imputation are acceptable approaches to handle missing data under certain assumptions about missingness. For example, 5% missingness and MCAR mechanism, but such cases are ideal and rarely occur. Traditional methods

are simpler, easier to implement than modern methods but rely on strong missing assumptions and routinely result in inefficient estimates, biased inferences, and standard error. Whereas, modern methods have many advantages over other imputation methods. Modern methods such as KNN account for uncertainty in the imputation and are the potential of creating better estimates by trying not to produce biased estimates.

Identifying disruption in time series data [48] generally requires either consider a single time series or multiple time series in the time series database. Single time series analysis considers subsequence or some part of it as anomalous by analyzing the trends in data and analyzing multiple time series in time series databases identify a few sequences as anomalous. When imputation the data, It is important to know the percentage of disruption in the data to identify what type of imputation methods can be applied to normalize the data containing missing values. The amount of disruption in the data will directly affect the imputation method. In general, some researchers have researched data that contains around 1-80 % of missing values out of complete data while others analyzed different imputation methods when 5-50 % data is disrupted. Different methods which aim to reduce missing value problems vary with the percentage of missingness but it is still not clear from various studies which methods are well suited to treat missingness depending on solely missingness percentage factor [37].

6 Experiment and Results

In this chapter, we will apply the estimation techniques with the clustering algorithm discussed in chapter 5 and 3 on actual time series data of wind and solar power plants. We experiment with all these methods on a Lenovo ThinkPad with a 2.20 GHz Intel Core i7 processor and 16 GB of RAM. We use Python 3 programming language to train the machine learning models and test them on online time series data.

To set up the experiment we divide this chapter into four sections. In section 6.1, we describe the data set used in our experiment and describe data prepossessing steps. In section 6.2, we describe the process to determine the number of clusters i.e. the value of k. In section 6.3, we describe the python implementation of clustering algorithms with similarity measure and quality indices. In section 6.4, we present our results and compare the performance of each estimation techniques with each clustering algorithm and visualize the results of the algorithm that performs best on our data.

6.1 Data

6.1.1 Data Description

The data set we work in our experiment is energy time series data of 300 power plants. 220 power plants are of wind and 80 power plants are of solar energy. The data is collected per minute data log of the year 2019 obtained from automatic monitoring systems (wind and solar power plants). We resample the data to 5 min interval to make the calculation memory efficient. Each power plant data contain 525600 rows and a column featuring the energy production value in Kwh. We divide the data set into training and testing data. We take data from 1-Jan-2019 to 15-Sep-2019 as training data and from 16-Sep-2019 to 31-Dec-2019 as testing data. For testing purposes, we assume the testing data as real time production data. Due to company data policy, we cannot mention the name of the power plants. However, we rename each power plant to show the visualization.

6.1.2 Data Prepossessing

This section explains the important preprocessing steps that are necessary before implementing the algorithm. Pre-processing plays an important role in improving the performance of the model. Some of the preprocessing steps that are necessary for our time series data are described below.

6.1.3 Data Cleaning

Data cleaning plays a vital role in the field of machine learning. It is the process of identifying incomplete and inaccurate data and estimating it or removing it according to the requirement. We decided to remove those PP data that are not correlated and contain more missing values compare to true values.

Cleaning of Missing Data

As our data is obtained from fluctuating energy sources; thus the training data set contains disruption. We categorize this disruption into three categories. These categories are missing data points, long series of consecutive duplicates values also called contextual anomalies and big negative integers.

We detect the contextual anomalies and big negative integers by writing a coding logic in a python programming language. Once these disruptions are detected we replace them with null values and visualize and calculate the percentage of missing values in each time series data of PP. Since the percentage of missingness will directly affect the results of estimation techniques so we remove all the power plants that have more than 50 percent of missing values, and estimate all the remaining missing values by using the KNN estimation technique, interpolation method and mean estimation technique defined in chapter 5.

We use the python sklearn package to implement the KNN estimator to estimate missing values. However, to interpolate and mean estimate we use python pandas data frame functionality to estimate the missing values. We select the one with the best clustering results.

To show the mechanism of missing data in all power plants we take the sample data from 15 power plants of wind and solar and present them in Figure 6.1 and 6.2 respectively. The white lines in each column represent the missing time series data. The more the white line in each column the more it will affect the performance of the clustering model.

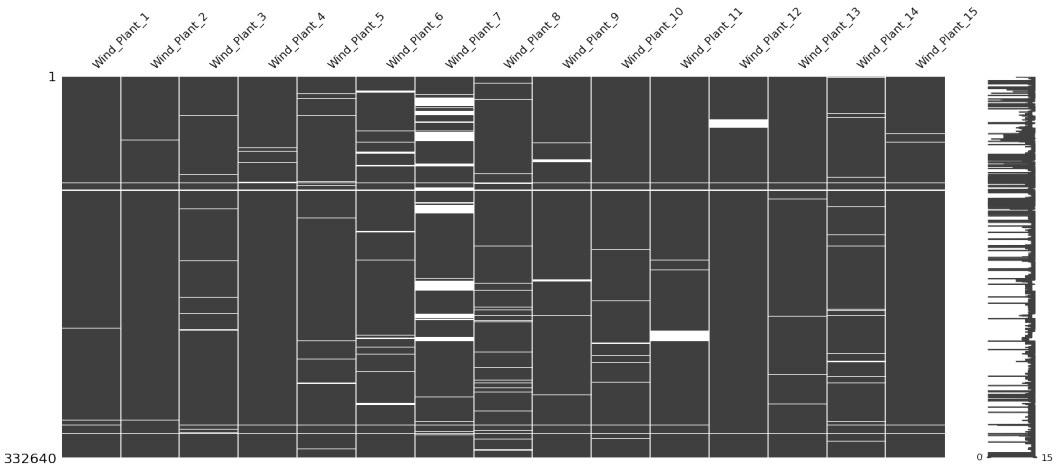


Figure 6.1: Missingness in Wind PP

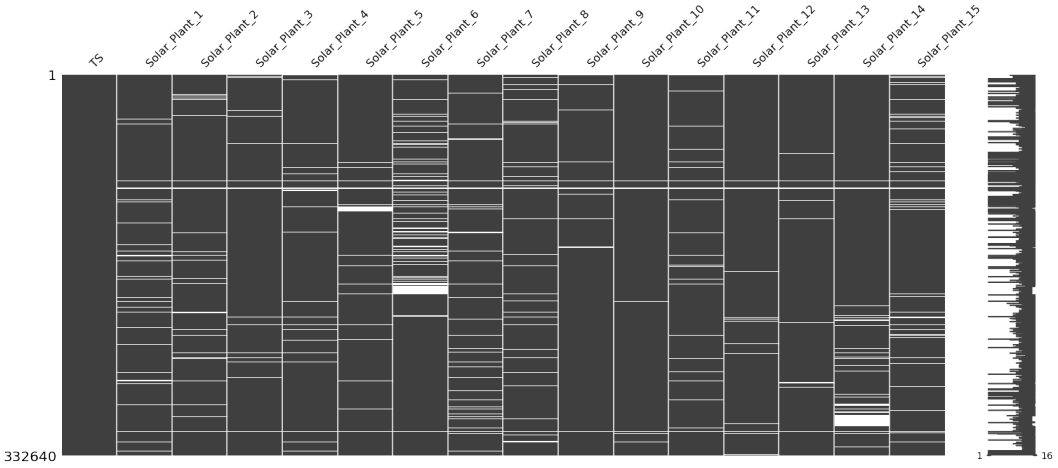


Figure 6.2: Missingness in Solar PP

Cleaning of Uncorrelated Data

As is already mentioned in the problem statement that the purpose of this thesis is to estimate disrupted data based on similar groups of PP. Therefore, to find out the correlation between the values of power plant play essential role in exploring similar patterns in the data. If the time series data of any PP is uncorrelated with other PP then it affect the accuracy of the model. In our case, we remove all PP that are uncorrelated (i.e close to 0) to increase the accuracy of the model.

We use Pearson correlation to find out correlation between power plants. Pearson correlation coefficient (PCC) is also known as bivariate correlation or Pearson product-moment correlation coefficient (PPMCC). As the name suggests, it is used to define a degree of linear relationship between two data vectors $C = (c_1, c_2, ..., c_i, ..., c_n)$ and $Q = (q_1, q_2, ...q_j, ..., q_n)$ of equal length $n \in \mathbb{N}$. It the ratio of covariance over the product of standard deviations of two data points c and q that are invariant to scaling.

Mathematically represented as,

$$\rho(c,q) = \frac{Cov(c,q)}{\sigma_c \sigma_q} \tag{6.1}$$

where, Cov(c,q) is covariance of data vector c and q and σ_c is the standard deviation of c and σ_q is the standard deviation of q, then,

$$\rho(c,q) = \frac{\sum_{i=1}^{n}(c_i - \mu_c)(q_i - \mu_q)}{\sqrt{\sum_{i=1}^{n}(c_i - \mu_c)^2}\sqrt{\sum_{i=1}^{n}(q_i - \mu_q)^2}} \tag{6.2}$$

where μ_c is the mean of data points of C data vector and μ_q is the mean of data points of Q data vector.

Alternatively, it can be expressed using mean and expectation as,

$$\rho(c,q) = \frac{E(c - \mu_c)(q - \mu_q)}{\sigma_c \sigma_q} \tag{6.3}$$

PCC value ranges from [-1,+1]. The value equal to 0 shows no relationship at all, whereas, the value 1 denotes a perfect positive and -1 denotes a perfect negative relationship between C and Q. For distance measures that are based on correlation, for better correlation, the two vectors should generate low distance values between them to show a positive correlation. Then it can be redefined as,

$$d_{corr}(c,q) = 1 - \rho(c,q) \tag{6.4}$$

To visualize the correlation we create a matrix of 15 wind PP presented in figure 6.3 and 15 solar PP presented in figure 6.4. Here the values in dark show a bad correlation between PP and values in light color show a good correlation.

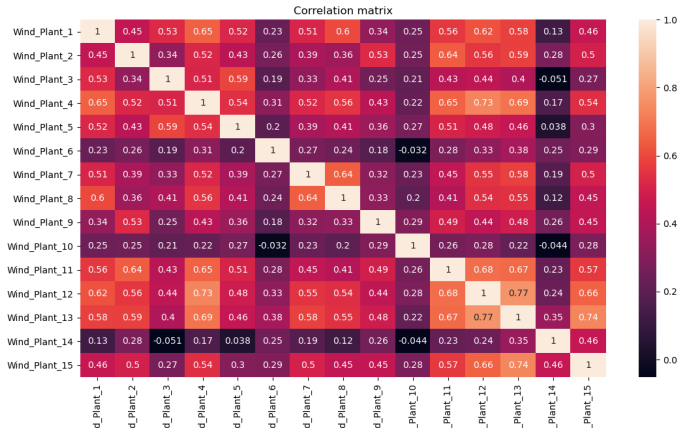


Figure 6.3: Example of correlation Matrix to determine correlation between Wind PP

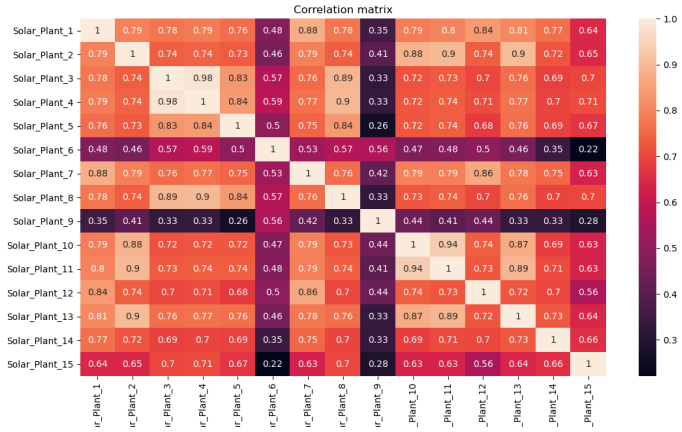


Figure 6.4: Example of correlation Matrix to determine correlation between Solar PP

6.1.4 Feature Scaling

Once the data is clean the next important step in our case is feature scaling. It is a technique to change the values of a feature that has a different scale. Not all the algorithms in machine learning require feature scaling. In our case, we are using an algorithm that is using a distance measure to find out the similarity between data points so there is a possibility that the higher weightage is given to features with higher values. Therefore, we scale our data before calculating the distance between each time series. Two common methods of scaling the feature are standardization and normalization. The normalization is performed by using the formula:

$$C_{norm} = \frac{C - C_{min}}{C_{max} - C_{min}}$$

where C is the data vector, C_{min} is the minimum value, and C_{max} is the maximum value in the vector. In normalization, the data is scaled into a range [0,1]. The standardization

is performed by using the formula:

$$C_{std} = \frac{C - \mu}{\sigma}$$

where μ is the mean and σ is the standard deviation of all the values in the data vector C. In our case, we chose normalization to scale each time series to the range [0,1]. We chose this because we want to cluster the data based on similar shapes in the time series rather than similar variances.

6.2 Determining the Number of Clusters

Determining the value of a number of clusters k in a data set is an important step in partition-based clustering, The partition-based clustering algorithm such as k-means require the value k as an input.

The optimal value of k is the one that gives the ideal partition. The definition of ideal partition is already stated in chapter 4, Ideal partition is possible under certain circumstances such as if there is some prior domain knowledge to identify an optimal number of clusters. For example, in determining the optimal number of clusters for the Iris data set, if we have prior knowledge of the type of species, we would know that how many clusters we want to form. But, if there is no knowledge of the domain then a data-driven approach such as the elbow method and statistical approach such as gap statistic helps.

In our case, we use the elbow method along with the performance metric set to silhouette score and CH score. We try the elbow method for each algorithm described in chapter 3 except affinity propagation as it does not require the value of k as an input parameter. The Elbow method forms a graph where we define several clusters at the x-axis and the performance metric at the y-axis. If the graph shows an arm like structure, then the point where the curve starts to flatten is called an elbow and indicates the optimal number of clusters to be formed denoted as k. As k increases, initially, the intra-cluster variance decreases, but at some point, the change in variance slows.

Figure 6.5 shows the elbow graph for wind PP data set by using a performance metric of silhouette and CH score with the k means model. In our case, we get the optimal value of k at k $= 6$ for the wind plant data set.

The figure 6.6 show the elbow graph for solar PP by using a performance metric of silhouette and CH score with k means model. In our case, we get the optimal value of k at k $= 5$ for solar plant data set.

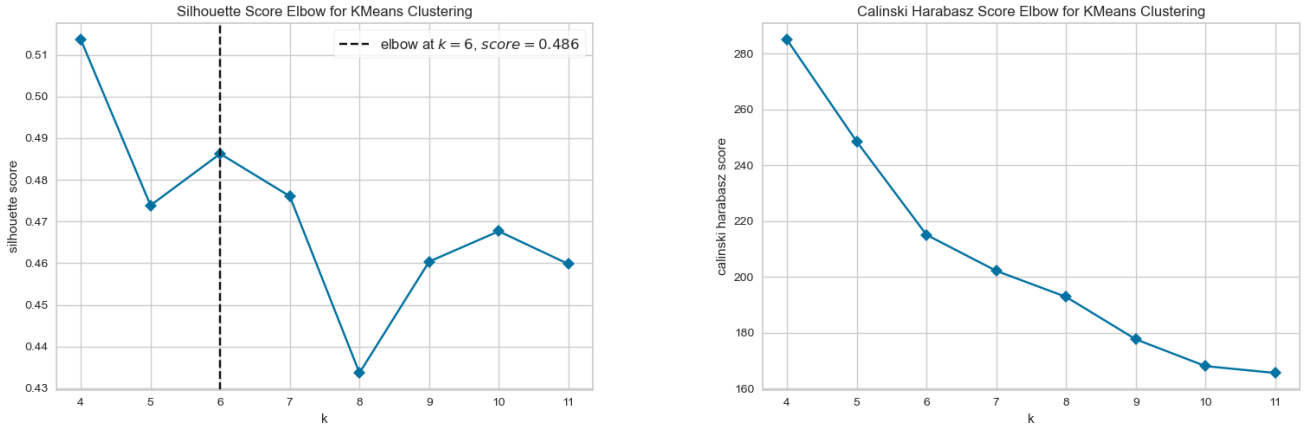


Figure 6.5: Elbow graph to determine the optimal value of no of cluster k for wind PP data

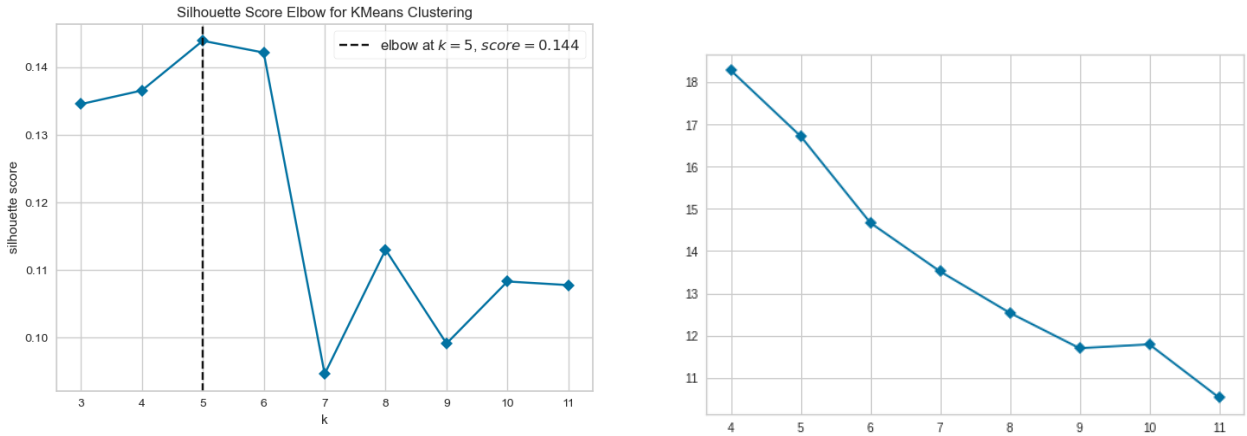


Figure 6.6: Elbow graph to determine the optimal value of no of cluster k for solar PP data

6.3 Implementation

In this section, we will discuss the python implementation of the distance matrices, clustering algorithm, and cluster quality measure discussed in this thesis.

6.3.1 Computing Distance Matrices

We compute the distance matrices for solar and wind plants data set. For both the data set we consider training data to compute distance matrices. Below we describe the python implementation of the distance measure and show the computation time taken to calculate the distance matrix in table 6.1.

Dynamic Time Warping

It is implemented using the funtion *dtw.distance.matrix .fast* from the package *dtaidistance*. It can take high computation power and CPU time to compute the distance matrix. However, in *dtaidistance* package various options are available to reduce the time complexity. We use the function that runs the algorithm in the C programming language to make computation fast. Parallelization can also achieve using the parallel argument.

Euclidean Distance

It is implemented using the python metric module from the scikit-learn library. It takes low computation power and CPU time.

In Table 6.1 the overview of computation times for both the distance measure can be found. The time is measured in seconds. In our case, it took 120 seconds to compute the distance matrix for the wind plant data set and 75 seconds for the solar plant data set when using euclidean distance. And 14400 seconds to compute the distance matrix for wind plant data set and 2700 seconds for solar plant data set when using dynamic time warping as a distance measure.

Distance measure	Wind Plant Data	Solar Plant Data
Euclidean	120	75
DTW	14400	2700

Table 6.1: Overview of the computation times of the different distance measures for determining the distance matrix.

6.3.2 Implementation of Clustering Algorithm and Quality Index

In this section, we describe the python implementation of the clustering algorithm described in chapter 3.

K-Means: K-Means with euclidean as a similarity measure is implemented using python sklearn library with cluster module. K-Means with DTW is implemented using python tslearn library with clustering module.

Agglomerative Agglomerative with euclidean as a similarity measure is implemented using python sklearn library with cluster module. Agglomerative with DTW as a similarity measure is implemented using python dtaidistance library with dtw module.

Affinity Propogation Affinity propogation is implemented using python sklearn library with cluster module.

Neural Gas No implementation of the neural gas algorithm was found. We programmed this method ourselves in python. It takes high computation power and memory on our data set. It leads to memory problems when we apply it with dtw. Therefore, we programmed neural gas with only euclidean distance metric and keep neural gas with dtw as future work. In our case, we choose the learning rate of 0.5, epoch value equal to 100 value of lambda equal to 0.33.

Silhouette Score: Implemented using the python scikit-learn library with metrics module.

CH Score: Implemented using the python scikit-learn library with metrics module.

DB Score: Implemented using the python scikit-learn library with metrics module.

6.4 Results and Analysis

In this section, we evaluate the performance of the clustering algorithm using the cluster quality index described in chapter 4 and computation time. Based on the performance of the clustering algorithm we select the estimation technique to reconstruct online time series data. Below we first present the results for wind plant data and show the visualization of the algorithm that performs best on wind data. We then present the results for solar plant data and show the visualization of the algorithm that performs best on the solar data.

6.4.1 Data set I: Wind Plant Data

In table 6.2, 6.3, and 6.4 an overview of performance for each combination of distance matrix and clustering algorithm can be found when using the **KNN estimator**, **interpolation** and **mean estimator** to estimate disrupted data. The best score for each cluster quality index column is represented in bold font.

We compare the results of table 6.2, 6.3, and 6.4. We observe the best results for the neural gas and k-means with DTW with a good silhouette and CH score compare to other algorithms. We observe that the cluster quality scores of every algorithm increased when we apply dtw as a similarity measure. Agglomerative clustering is the quickest at producing clustering and also gives a good DB score, but we must keep in mind that these CPU times do not include the time it takes to compute the individual distance measure as we precompute the distance matrix. However, in the case of k-means and neural gas, the distances are not precomputed. In the end, we compare three estimation technique and observe that KNN estimation in table 6.2 give the best cluster quality result compare to interpolation and mean estimation. We concluded that the neural gas algorithm outperforms others as it is only implemented with euclidean if it is implemented

with DTW the score could be much better. However, due to high computation time, we do not consider neural gas for the final implementation of the model and consider the second best algorithm that is k-means with DTW. Furthermore, based on clustering results we consider KNN estimation as the final estimation technique to reconstruct the online time series data of wind PP.

KNN Estimation						
Clustering Algorithm	Similarity Measure	Linkage	Silhouette Score	CH Score	DB Score	CPU Time
K-Means	Euclidean	-	0.11	23	2	12.22
	DTW	-	0.51	202	1.85	1500
Agglomerative	Euclidean	Complete	0.10	20	1.78	1.74
	DTW	Complete	0.21	64	0.91	0.006
	Euclidean	Single	0.14	7	0.80	1.70
	DTW	Single	0.23	19	0.62	0.005
	Euclidean	Average	0.19	7	0.98	1.73
	DTW	Average	0.36	31	**0.62**	0.005
	Euclidean	Ward	0.19	8	0.98	1.72
Affinity Propogation	Euclidean	-	0.11	10	1.65	0.26
	DTW	-	0.40	113	1.09	0.02
Neural Gas	Euclidean	-	**0.58**	**249**	1.25	2100

Table 6.2: Clustering performance evaluation results for wind PP when using KNN estimation technique to estimate disrupted data

Interpolation						
Clustering Algorithm	Similarity Measure	Linkage	Silhouette Score	CH Score	DB Score	CPU Time
K-Means	Euclidean	-	0.10	21	2.3	12.12
	DTW	-	**0.47**	208	1.29	1525
Agglomerative	Euclidean	Complete	0.11	22	1.9	1.71
	DTW	Complete	0.16	88	1.26	0.01
	Euclidean	Single	0.19	5	0.58	1.70
	DTW	Single	0.21	14	**0.46**	0.005
	Euclidean	Average	0.22	7	0.96	1.7
	DTW	Average	0.34	29	0.61	0.005
	Euclidean	Ward	0.09	22	2.38	1.71
Affinity Propogation	Euclidean	-	0.10	10	1.57	0.25
	DTW	-	0.39	110	1.09	0.03
Neural Gas	Euclidean	-	**0.47**	**240**	1.17	2100

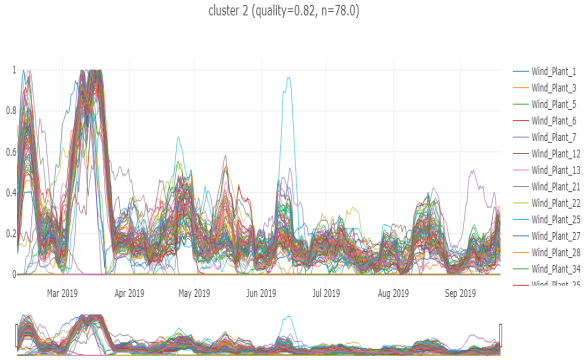
Table 6.3: Clustering performance evaluation results for wind PP when using Interpolation technique to estimate disrupted data

Mean Estimation						
Clustering Algorithm	Similarity Measure	Linkage	Silhouette Score	CH Score	DB Score	CPU Time
K-Means	Euclidean	-	0.10	21	2	12.04
	DTW	-	0.44	**176**	1.50	1500
Agglomerative	Euclidean	Complete	0.09	20	2.03	1.72
	DTW	Complete	0.23	91	0.90	0.01
	Euclidean	Single	0.14	6	0.81	1.70
	DTW	Single	0.34	20	**0.65**	0.004
	Euclidean	Average	0.18	9	1.24	1.70
	DTW	Average	0.40	42	0.99	0.005
	Euclidean	Ward	0.07	21	2.38	1.73
Affinity Propogation	Euclidean	-	0.10	9.6	1.70	0.26
	DTW	-	0.30	102	1.09	0.03
Neural Gas	Euclidean	-	**0.51**	242	1.18	2100

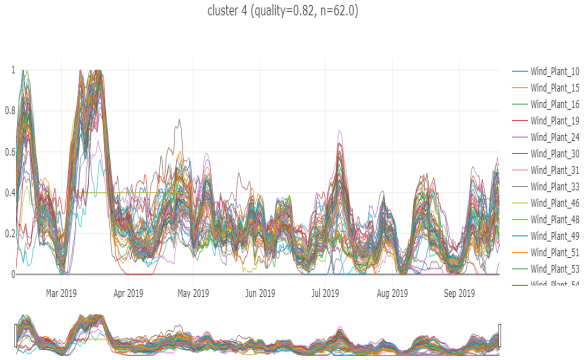
Table 6.4: Clustering performance evaluation results for wind PP when using mean estimation technique to estimate disrupted data

The visualization of six clusters obtained by using the K-Means algorithm with DTW is seen in figure 6.7. In terms of efficiency, this is the second-best algorithm. We choose this algorithm as the final implementation because it needs less CPU time than others, and our data is large and growing by the minute, so to avoid memory problems in the future. We also calculate the correlation between power plant data of each cluster to observe the similarity between values. Size represents the number of power plants in each cluster and quality represents the correlation between values in each cluster. The X-axis represents the date and Y-axis represents the normalized values range from 0 to 1.

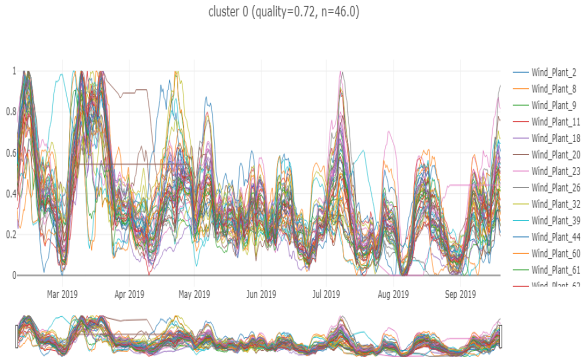
We take one PP data from each cluster shown in figure 6.8 and present the visualization of three estimation techniques applied to it. The X-axis represents date and time and Y-axis represents the value of the PP. For reconstructing or estimating disrupted values on real time online data we choose the estimation technique that gives the best results for clustering algorithm when applied to training data to estimate disruption.



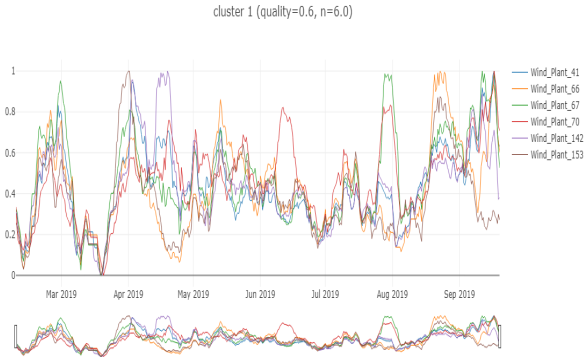
(a) Cluster: 2 , Quality: 0.82, Size: 78



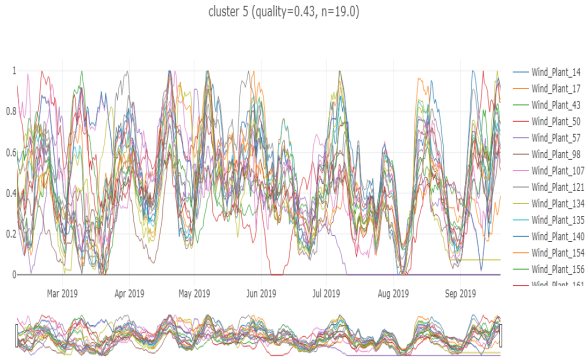
(b) Cluster: 4 , Quality: 0.82, Size: 62



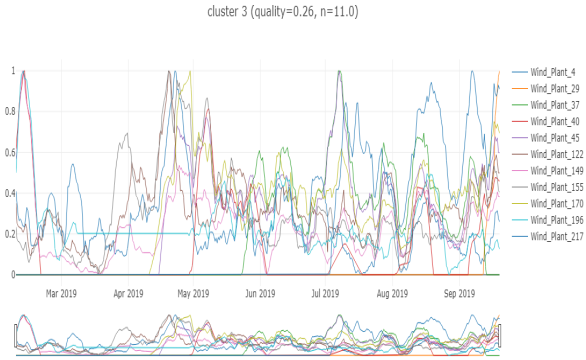
(c) Cluster: 0 , Quality: 0.72, Size: 46



(d) Cluster: 1 , Quality: 0.6, Size: 6

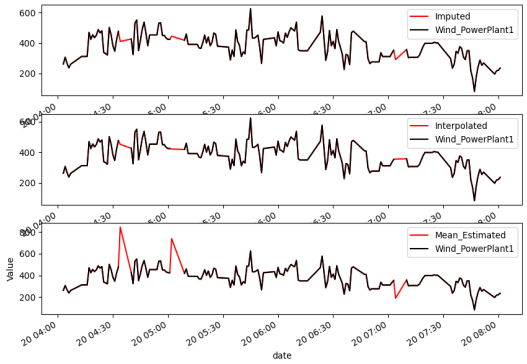


(e) Cluster: 5 , Quality: 0.43, Size: 19

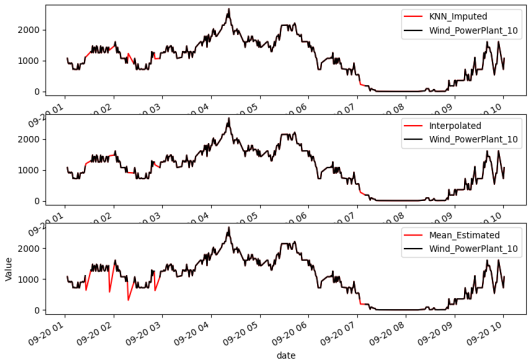


(f) Cluster: 3 , Quality: 0.26, Size: 11

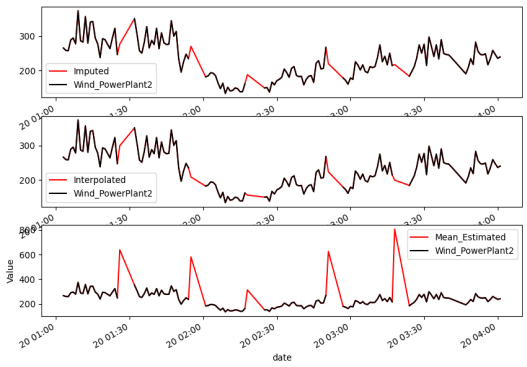
Figure 6.7: Visualization of the six clusters that are obtained when applying the K-means clustering algorithm with the DTW distance and KNN estimation to estimate disruption on wind PP time series data.



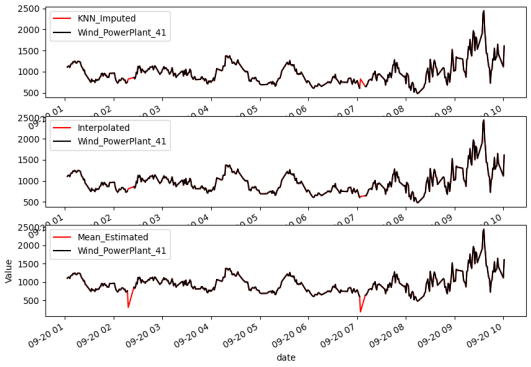
(a) Cluster: 3 , Wind Plant 1



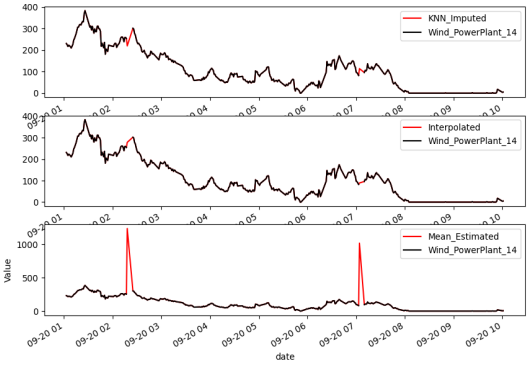
(b) Cluster: 4 , Wind Plant 10



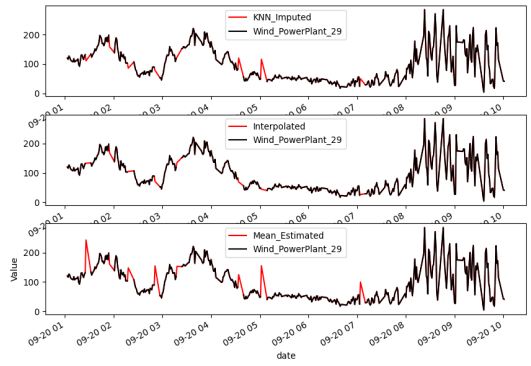
(c) Cluster: 0 , Wind Plant 2



(d) Cluster: 1 , Wind Plant 41



(e) Cluster: 5 , Wind Plant 14



(f) Cluster: 3 , Wind Plant 29

Figure 6.8: Example of one of the wind PP time series data from each cluster to visualize the estimation techniques applied to reconstruct disrupted data. X-Axis represent the time and Y-Axis represent the energy production value of each PP

6.4.2 Data set II: Solar Plant Data

In table 6.5, 6.6, and 6.7 an overview of performance for each combination of distance matrix and clustering algorithm can be found when using the **KNN estimator**, **interpolation** and **mean estimator** to estimate disrupted data. The best score for each cluster quality index column is represented in bold font.

We compare the results of Tables 6.5, 6.5, and 6.7. We observe the best results for the agglomerative with DTW with a good DB and CH score compare to other algorithms. However, the result of neural gas and k-means give the best silhouette score compare to others, We observe that the cluster quality scores of every algorithm increased when we apply DTW as a similarity measure. Agglomerative clustering is also the quickest at producing clustering, but we must keep in mind that these CPU times do not include the time it takes to compute the individual distance measure as we precompute the distance matrix. However, in the case of k-means and neural gas, the distances are not precomputed. In the end, we compare three estimation technique and observe that KNN estimation in table 6.5 give the best cluster quality result compare to interpolation and mean estimation. We concluded that the agglomerative clustering algorithm with DTW outperforms others and we consider it for the final implementation of the model. Furthermore, based on clustering results we consider KNN estimation as the final estimation technique to reconstruct the online time-series data of solar PP.

KNN Estimation						
Clustering Algorithm	Similarity Measure	Linkage	Silhouette Score	CH Score	DB Score	CPU Time
K-Means	Euclidean	-	0.9	10	2.20	3.19
	DTW	-	0.41	21	1.66	900
Agglomerative	Euclidean	Complete	0.11	11	1.78	0.18
	DTW	Complete	0.17	**151**	1.29	0.001
	Euclidean	Single	0.18	8	0.87	0.18
	DTW	Single	0.02	42	**0.79**	0.002
	Euclidean	Average	0.17	8	0.86	0.17
	DTW	Average	0.19	75	1.34	0.004
	Euclidean	Ward	0.13	12	2.06	0.17
Affinity Propogation	Euclidean	-	0.10	8	1.62	0.07
	DTW	-	0.61	68	0.67	0.04
Neural Gas	Euclidean	-	**0.68**	32	0.56	1200

Table 6.5: Clustering performance evaluation results for solar PP when using KNN estimation technique to estimate disrupted data

Interpolation						
Clustering Algorithm	Similarity Measure	Linkage	Silhouette Score	CH Score	DB Score	CPU Time
K-Means	Euclidean	-	0.12	11	2.09	3.55
	DTW	-	**0.66**	83	0.58	878
Agglomerative	Euclidean	Complete	0.10	11	1.92	0.20
	DTW	Complete	0.22	**88**	1.19	0.01
	Euclidean	Single	0.05	2	**0.75**	0.20
	DTW	Single	0.3	0.9	1.42	0.003
	Euclidean	Average	0.23	9	0.83	0.20
	DTW	Average	0.21	76	1	0.002
	Euclidean	Ward	0.11	12	2.11	0.19
Affinity Propogation	Euclidean	-	0.10	8	1.62	0.06
	DTW	-	0.59	77	0.67	0.03
Neural Gas	Euclidean	-	0.59	19	1.09	1200

Table 6.6: Clustering performance evaluation results for solar PP when using Interpolation technique to estimate disrupted data

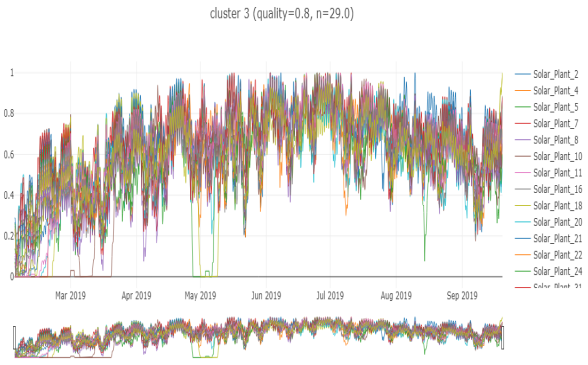
Mean Estimation						
Clustering Algorithm	Similarity Measure	Linkage	Silhouette Score	CH Score	DB Score	CPU Time
K-Means	Euclidean	-	0.10	10	2.16	3.49
	DTW	-	**0.64**	119	0.52	880
Agglomerative	Euclidean	Complete	0.11	11	1.84	0.19
	DTW	Complete	0.26	**123**	**0.19**	0.004
	Euclidean	Single	0.19	10	0.85	0.20
	DTW	Single	-0.2	0.98	1.25	0.002
	Euclidean	Average	0.19	10	1.01	0.20
	DTW	Average	0.29	78	0.82	0.003
	Euclidean	Ward	0.10	12	2.29	0.19
Affinity Propogation	Euclidean	-	0.09	7	1.7	0.06
	DTW	-	0.59	81	0.68	0.01
Neural Gas	Euclidean	-	0.38	17	1.23	1200

Table 6.7: Clustering performance evaluation results for solar PP when using mean estimation technique to estimate disrupted data

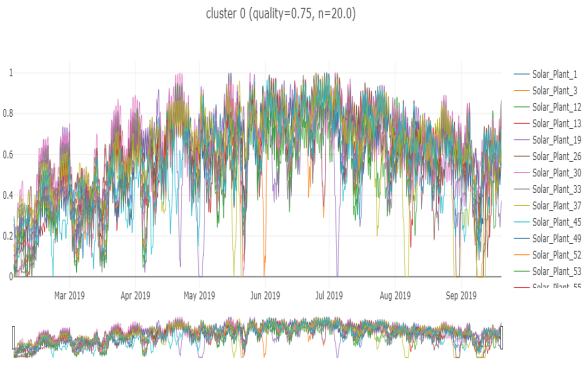
The visualization of six clusters obtained by using the agglomerative algorithm with DTW is seen in figure 6.9. In terms of efficiency, this is the best algorithm. We choose this algorithm as the final implementation. We also calculate the correlation between power plant data of each cluster to observe the similarity between values. Size represents the number of power plants in each cluster and quality represents the

correlation between values in each cluster. The X-axis represents the date and Y-axis represents the normalized values range from 0 to 1.

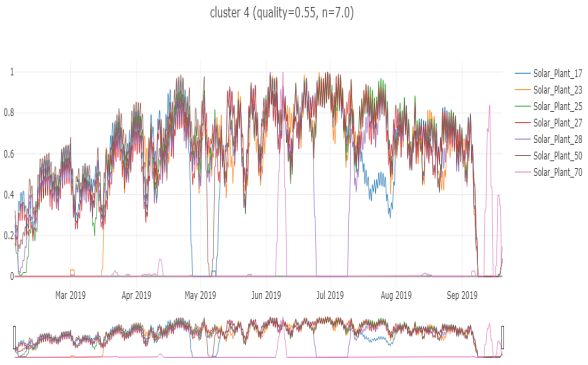
We take one PP data from each cluster shown in figure 6.10 and present the visualization of three estimation techniques applied to it. The X-axis represents date and time and Y-axis represents the value of the PP. For reconstructing or estimating disrupted values on real time online data we choose the estimation technique that gives the best results for clustering algorithm when applied to training data to estimate disruption.



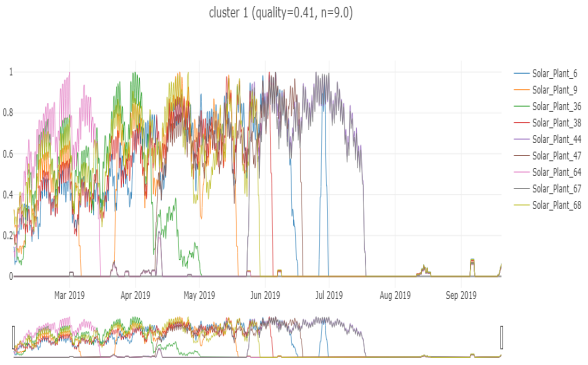
(a) Cluster: 3 , Quality: 0.8, Size: 29



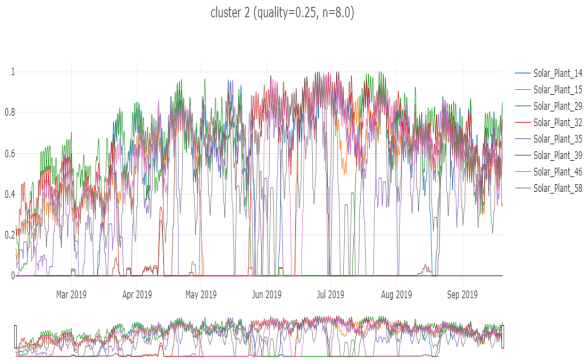
(b) Cluster: 0 , Quality: 0.75, Size: 20



(c) Cluster: 4 , Quality: 0.55, Size: 7

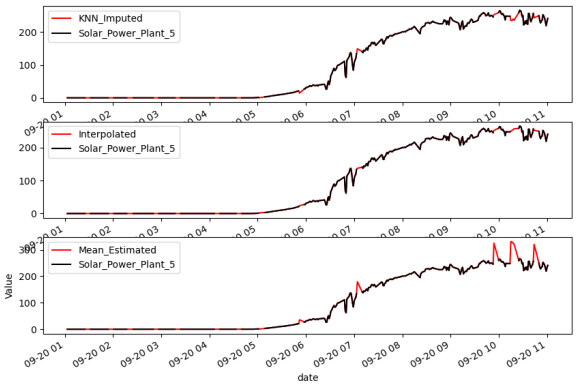


(d) Cluster: 1 , Quality: 0.41, Size: 9

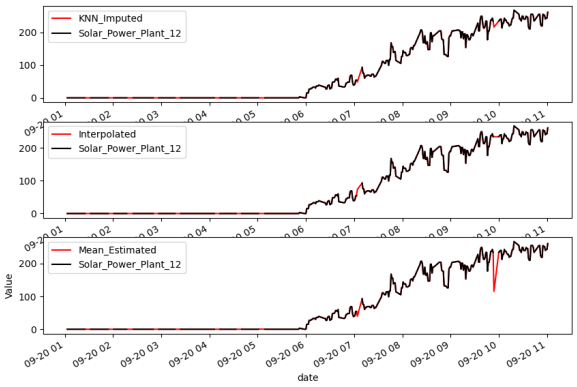


(e) Cluster: 2 , Quality: 0.25, Size: 8

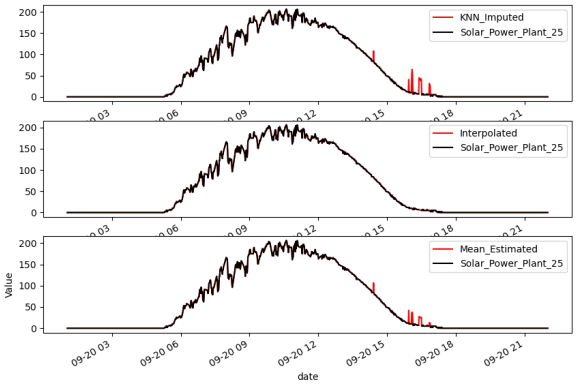
Figure 6.9: Visualization of the five clusters that are obtained when applying the Agglomerative clustering with the DTW distance and KNN estimation to estimate disruption on solar PP time series data. This clustering assignment is the best for solar PP in our experiments.



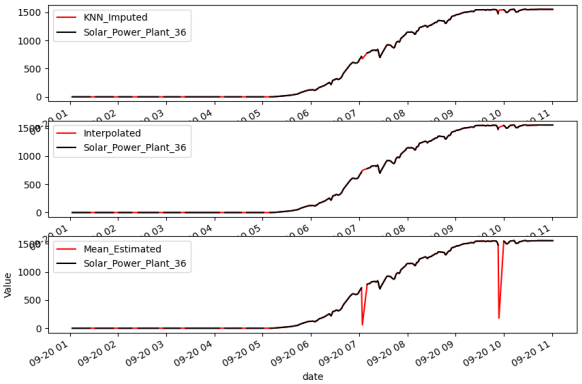
(a) Cluster: 3 , Solar Plant 5



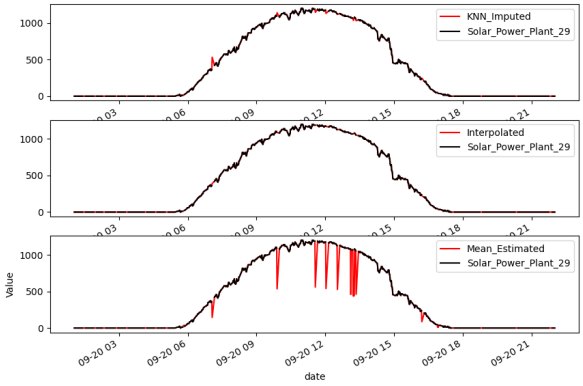
(b) Cluster: 0 , Solar Plant 12



(c) Cluster: 4 , Solar Plant 25



(d) Cluster: 1 , Quality: 0.41, Size: 9



(e) Cluster: 2 , Quality: 0.25, Size: 8

Figure 6.10: Example of one of the Solar PP time series data from each cluster to visualize the estimation techniques applied to reconstruct disrupted data. X-Axis represent the time and Y-Axis represent the energy production value of each P

7 Conclusion and Future Work

In this thesis, we focus on the problem of disrupted online time series data and estimate it by applying cluster analysis.

Discussion on different types of clustering algorithms and estimation techniques have been presented. We worked on real experiments and showed that the clustering algorithms when applied with dtw as a similarity measure on time series data give meaningful results. We showed that modern estimation techniques like KNN outperformed traditional estimation techniques and improves the quality of clusters. We also showed that neural gas algorithm gives remarkable results compare to k-means and other algorithm using euclidean distance as its does not stick in the local minima.

We can extend this thesis by applying a neural gas algorithm with dtw because it gives memory leakage problem on our data set therefore solution to solve this problem is left for the future. Many modern estimation techniques like Multivariate Imputation by Chained Equation (MICE), random forest, decision tree, etc are left for the future due to lack of time because the experiments with real data take a lot of time and analysis. The performances of all the estimation techniques described in chapter 5 on online time series data have not been presented and left for the future. The reason was that it requires considerable execution time to evaluate each power plant data set.

Bibliography

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, pp. 35–45, 1960.

[2] X. Liu and A. Goldsmith, "Kalman filtering with partial observation losses," in *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, vol. 4. IEEE, 2004, pp. 4180–4186.

[3] E. Damsleth and A. El-Shaarawi, "Arma models with double-exponentially distributed noise," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 51, no. 1, pp. 61–69, 1989.

[4] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation," *Econometrica: Journal of the Econometric Society*, pp. 987–1007, 1982.

[5] J. Zhang, M. Cui, B.-M. Hodge, A. Florita, and J. Freedman, "Ramp forecasting performance from improved short-term wind power forecasting over multiple spatial and temporal scales," *Energy*, vol. 122, pp. 528–541, 2017.

[6] F. Divina, M. García Torres, F. A. Goméz Vela, and J. L. Vázquez Noguera, "A comparative study of time series forecasting methods for short term electric energy consumption prediction in smart buildings," *Energies*, vol. 12, no. 10, p. 1934, 2019.

[7] D. W. Schrock and D. E. Claridge, "Predicting energy usage in a supermarket," *6th Symposium on Improving Building Systems in Hot and Humid Climates*, 1989.

[8] J. Nowotarski, B. Liu, R. Weron, and T. Hong, "Improving short term load forecast accuracy via combining sister forecasts," *Energy*, vol. 98, pp. 40–49, 2016.

[9] H. Rahman, I. Selvarasan, and J. Begum A, "Short-term forecasting of total energy consumption for india-a black box based approach," *Energies*, vol. 11, no. 12, p. 3442, 2018.

[10] D. C. Park, M. El-Sharkawi, R. Marks, L. Atlas, and M. Damborg, "Electric load forecasting using an artificial neural network," *IEEE transactions on Power Systems*, vol. 6, no. 2, pp. 442–449, 1991.

[11] R. M. Sáez, M. Sidrach-De-Cardona, and L. Mora-López, "Data mining and statis-

tical techniques for characterizing the performance of thin-film photovoltaic modules," *Expert systems with applications*, vol. 40, no. 17, pp. 7141–7150, 2013.

[12] B. Uzunoglu, "Analysis of financial impact of wind speeds in wind farms using anova, ar and arma and six sigma processes," in *18th International Conference on Computing in Economics and Finance. Society for Computational Economics*, 2012.

[13] Z. Guo, D. Chi, J. Wu, and W. Zhang, "A new wind speed forecasting strategy based on the chaotic time series modelling technique and the apriori algorithm," *Energy Conversion and Management*, vol. 84, pp. 140–151, 2014.

[14] S. Spiegel, "Time series distance measures : Segmentation, classification, and clustering of temporal data," 2015.

[15] E. Xing, M. Jordan, S. J. Russell, and A. Ng, "Distance metric learning with application to clustering with side-information," *Proceeding of the 15th International Conference on Neural Information Processing System*, vol. 15, pp. 521–528, 2002.

[16] P. Esling and C. Agon, "Time-series data mining," *ACM journal*, vol. 45, p. 34, 2012.

[17] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Mining and Knowledge Discovery*, vol. 26, no. 2, pp. 275–309, Feb. 2012.

[18] K. V. Boriah S, Chandola V, "Similarity measures for categorical data: A comparative evaluation," *Proceedings of the 2008 SIAM International Conference on Data Mining*, p. 243–254, 2008.

[19] H. J. Khalifa A Al, Haranczyk M, "Comparison of non binary similarity coefficients for similarity searching, clustering and compound selection," *Journal of Chemical Information and Modeling*, vol. 49(5), p. 1193–201, 2009.

[20] P. Schäfer, "Scalable time series similarity search for data analytics," Ph.D. dissertation, 10 2015.

[21] F. elix Iglesias and W. Kastner, "Analysis of similarity measures in times series clustering for the discovery of building energy patterns," *Energies*, vol. 6(2), pp. 579–597, 2013.

[22] A. D. S. G.Batista, J Keogh, "An efficient complexity-invariant distance for time series," *Data Mining and Knowledge Discovery*, 2013.

[23] T. Liao, "Clustering of time series data—a survey," vol. 38, pp. 1857–1874, 2005.

[24] "Dynamic time warping," in *Information Retrieval for Music and Motion*. Springer Berlin Heidelberg, 2007, pp. 69–84.

[25] M. R. Strehl A, Ghosh J, "Impact of similarity measures on web-page clustering," *Workshop on Artificial Intelligence for Web Search*, p. 58–64, 2000.

[26] J. Paparrizos and L. Gravano, "k-shape: Efficient and accurate clustering of time series," *ACM SIGMOD Record*, vol. 45, no. 1, pp. 69–76, Jun. 2016.

[27] M. Cottrell, B. Hammer, A. Hasenfuss, and T. Villmann, "Batch and median neural gas," *Neural Networks*, vol. 19, no. 6-7, pp. 762–771, Jul. 2006.

[28] H. A. K. Celebi, M. E. and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Syst. Appl*, vol. 4(1), p. 200–210, 2013.

[29] G. Hamerly and C. Elkan, "Alternatives to the k-means algorithm that find better clusterings," p. 600–607, 2002.

[30] T. Martinetz and K. Schulten, "A 'neural-gas' network learns topologies," *Artificial Neural Networks*, vol. 1, pp. 397–402, 1991.

[31] Y. Jiang, Y. Liao, and G. Yu, "Affinity propagation clustering using path based similarity," *Algorithms*, vol. 9, no. 3, p. 46, 2016.

[32] A. M. R. Refianti and A. Syamsudduha, "Performance evaluation of affinity propagation approaches on data clustering," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 3, p. 46, 2016.

[33] A. M. Serdah and W. M. Ashour, "Clustering large-scale data based on modified affinity propagation algorithm," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 6, no. 1, pp. 23–33, 2016.

[34] A. F. Moiane and A. M. L. Machado, "Evaluation of the clustering performance of affinity propogation algorithm considering the influence of preference parameter and damping factor," *Boletim de Ciencias Geodesicas*, vol. 24, no. 4, pp. 426–441, 2018.

[35] F. Torrent-Fontbona, "Decision support methods for global optimization," 2012.

[36] Y. Jiang, Y. Liao, and G. Yu, "Affinity propagation clustering using path based similarity," *Algorithms*, vol. 9, no. 3, p. 46, Jul. 2016.

[37] Vol. 7, 2016. [Online]. Available: https://doi.org/10.14569/ijacsa.2016.070357

[38] P. K.Sasirekha, "Agglomerative hierarchical clustering algorithm-a review," *International Journal of Scientific and Research Publications*, vol. 3, pp. 2250–3153, 2013.

[39] X. Wang, K. Smith-Miles, and R. Hyndman, "Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series," *Neurocomputing*, vol. 72, no. 10-12, pp. 2581–2594, Jun. 2009. [Online]. Available: https://doi.org/10.1016/j.neucom.2008.10.017

[40] "Chap 445 hierarchical clustering and dendrograms," *NCSS (Number Cruncher Statistical System)*, pp. 15–50, 2015.

[41] P. Rodrigues, J. Gama, and J. Pedroso, "Hierarchical clustering of time-series data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 5, pp. 615–627, 2008.

[42] S. Miyamoto, "Refinement properties in agglomerative hierarchical clustering," pp. 259–267, 2009.

[43] J. H. W. Jr., "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[44] K. S. P. Baby, "Agglomerative hierarchical clustering algorithm- a review," 2013.

[45] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods," *ACM SIGMOD Record*, vol. 31, no. 2, pp. 40–45, Jun. 2002.

[46] M. J. A. Berry and G. Linoff, "Data mining techniques for marketing, sales and customer support," pp. 40–45, 1996.

[47] E. H. Vendramin L, B Campello, "Relative clustering validity criteria: A comparative overview," *Statistical Analysis and Data Mining*, vol. 3(4), pp. 209–235, 2010.

[48] I. Pratama, A. E. Permanasari, I. Ardiyanto, and R. Indrayani, "A review of missing values handling methods on time-series data," 2016.

[49] R. K. Elissavet, "Missing data in time seriesand imputation methods," 2017.

Erklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Mittweida, im May 2021