


**A WEB-BASED IMPLEMENTATION OF  
K-MEANS ALGORITHMS**

**LEE QUAN**

**UNIVERSITI TUNKU ABDUL RAHMAN**

**DECLARATION**

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature :   
\_\_\_\_\_

Name : LEE QUAN  
\_\_\_\_\_

ID No. : 18UEB01846  
\_\_\_\_\_

Date : 10 May 2022  
\_\_\_\_\_

**APPROVAL FOR SUBMISSION**

I certify that this project report entitled **“A WEB-BASED IMPLEMENTATION OF K-MEANS ALGORITHMS”** was prepared by **LEE QUAN** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Hons) Software Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : *kckhor*

Supervisor : Khor Kok Chin

Date : 11/5/2022

Signature : \_\_\_\_\_

Co-Supervisor : \_\_\_\_\_

Date : \_\_\_\_\_

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2022, Lee Quan. All right reserved.

## ACKNOWLEDGEMENTS

First and foremost, I would like to extend my sincerest gratitude to Dr. Khor Kok Chin, my supervisor, for his continued guidance and patience throughout the entire duration of this project. This project would not have been possible without his invaluable advice and input at every stage.

Furthermore, I am eternally grateful to my loving and caring parents for their encouragement and cheers from the side-lines. They were with me every step of the way, helping me push on till I managed to cross the finishing line.

Lastly, let's not forget my academic peers, who provided much needed advice and suggestions during the design and development phases of this project.

## ABSTRACT

The K-means algorithm has been around for over a century. While a rather simplistic and dated algorithm, it remains widely used and taught till this day. The K-means algorithm requires two inputs for it to be applied onto a data set, the value K, and a proximity measure. Picking the right inputs is of utmost importance if one wishes to achieve good results with the algorithm, especially the proximity measure. There are plenty of different proximity measures available in the world, all of them best suited for different types of applications and data sets. Yet knowing this, most modern data mining tools only offer a handful of proximity measures to the user, with the most common ones being Euclidean distance and Manhattan distance. This stinginess of proximity measures in data mining tools is stifling the performance of the algorithm. This is where *k-luster* comes in.

*k-luster*, the web application developed as a result of this project, implements the K-means and K-means++ algorithm along with ten proximity measures, seven of which are distance measures and whereas the remaining three are similarity measures. The project was planned using the Kanban development methodology, and was built using HTML, CSS, JavaScript, Django, NumPy and pandas. The completed web application is then hosted on Heroku. *k-luster* allows users to upload their own data set, or choose from one of three samples if they just want to try out the application. Playing around with different settings and comparing the results obtained, it is clear how large of an impact choosing the right proximity measure can make.

In conclusion, this project has accomplished what it first set out to achieve. However, there is still much room for improvement. Firstly, *k-luster* could incorporate additional clustering algorithms, or even classification algorithms in the future. Furthermore, the web application could save the users' past work, so that they may resume their work at a later time without skipping a beat.

## TABLE OF CONTENTS

<b>DECLARATION</b>	<b>ii</b>
<b>APPROVAL FOR SUBMISSION</b>	<b>iii</b>
<b>ACKNOWLEDGEMENTS</b>	<b>v</b>
<b>ABSTRACT</b>	<b>vi</b>
<b>TABLE OF CONTENTS</b>	<b>vii</b>
<b>LIST OF TABLES</b>	<b>xi</b>
<b>LIST OF FIGURES</b>	<b>xii</b>
<b>LIST OF SYMBOLS / ABBREVIATIONS</b>	<b>xvi</b>
<b>LIST OF APPENDICES</b>	<b>xvii</b>

## CHAPTER

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 General Introduction	1
	1.2 Problem Background	1
	1.3 Problem Statement	3
	1.4 Project Objectives	3
	1.5 Project Solution	3
	1.6 Project Approach	4
	1.7 Project Scope	5
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>6</b>
	2.1 The K-means Algorithm	6
	2.1.1 Brief Introduction to Data Mining	6
	2.1.2 The K-means Algorithm	11
	2.1.3 The K-means++ Algorithm	14
	2.1.4 Proximity Measures	15
	2.2 Review of Common Data Mining Tools	25

2.2.1	amap (R package)	25
2.2.2	scikit-learn (Python package)	26
2.2.3	WEKA	26
2.2.4	Oracle Data Mining	27
2.2.5	Summary	27
2.3	Web Application	28
2.3.1	Introduction to Web Applications	28
2.3.2	Static and Dynamic Web Pages	28
2.3.3	General Architecture of a Web Application	29
2.3.4	Summary	29
2.4	Development Methodology	30
2.4.1	Agile	30
2.4.2	Kanban	30
<b>3</b>	<b>METHODOLOGY AND WORK PLAN</b>	<b>33</b>
3.1	Introduction	33
3.2	Development Methodology	33
3.3	Work Plan	35
3.3.1	Work Breakdown Structure	35
3.3.2	Project Schedule	37
3.3.3	Gantt Chart	39
3.4	Tools and Frameworks	41
3.4.1	HTML and CSS	41
3.4.2	JavaScript	41
3.4.3	Python	41
3.4.4	Django	42
3.4.5	NumPy	42
3.4.6	pandas	42
3.4.7	Matplotlib	42
3.4.8	Desmos API	43
3.4.9	Visual Studio Code	43
3.4.10	Git	43
3.4.11	GitHub	43
3.4.12	Trello	44



3.4.13	Heroku	44
<b>4</b>	<b>PROJECT SPECIFICATION</b>	<b>45</b>
4.1	Introduction	45
4.2	Functional Requirements	45
4.2.1	Upload data sets module	45
4.2.2	Data pre-processing module	45
4.2.3	Clustering module	45
4.2.4	Results module	46
4.3	Non-functional Requirements	46
4.3.1	Operational requirements	46
4.3.2	Reliability requirements	46
4.3.3	Performance requirements	46
4.4	Use-cases	47
4.4.1	Use-case diagram	47
4.4.2	Use-Case Descriptions	48
4.5	Prototype	60
<b>5</b>	<b>SYSTEM DESIGN</b>	<b>63</b>
5.1	System Architecture	63
5.2	Data Flow Diagrams	64
5.3	Activity Diagrams	66
5.4	Page Designs	69
<b>6</b>	<b>SYSTEM IMPLEMENTATION</b>	<b>71</b>
6.1	Introduction	71
6.2	Migrating Prototype Code	71
6.3	Initialise a Git Repository and Push Code to GitHub	71
6.4	Implementing K-means++	72
6.5	Implementing Additional Proximity Measures (I)	73
6.6	Deploying Project to Heroku	74
6.7	Attribute Exclusion	76
6.8	Attribute Normalisation	77

6.9	Seed Specification	77
6.10	Improving Result Visualisation Using Desmos API	77
6.11	Detection of Attribute Data Types	79
6.12	Implementing Sessions	80
6.13	Handling Missing Values in Uploaded Data sets	81
6.14	Sample Data sets	82
	6.14.1 Iris Data set	82
	6.14.2 Diabetes Data set	83
	6.14.3 Breast Cancer Data set	83
6.15	Download Results	84
6.16	Classes to Clusters Evaluation	84
6.17	Implementing Additional Proximity Measures (II)	85
<b>7</b>	<b>SYSTEM TESTING</b>	<b>87</b>
7.1	SUS Testing	87
	7.1.1 Test Procedure	87
	7.1.2 Calculating SUS Scores	88
	7.1.3 Interpreting SUS Scores	88
	7.1.4 Conducting the SUS Test	90
<b>8</b>	<b>CONCLUSIONS AND RECOMMENDATIONS</b>	<b>95</b>
8.1	Conclusion	95
8.2	Recommendations for future work	95
	<b>REFERENCES</b>	<b>96</b>
	<b>APPENDICES</b>	<b>99</b>

## LIST OF TABLES

Table 2.1: K-means Algorithm Pseudocode (Han, Kamber and Pei, 2011)	12
Table 2.2: K-means++ Algorithm Pseudocode (Arthur and Vassilvitskii, 2007)	14
Table 3.1: Project Schedule	37
Table 4.1: Upload Data Set Use-Case Description	48
Table 4.2: Fill Missing Values Use-Case Description	49
Table 4.3: Choose Sample Data Set Use-Case Description	50
Table 4.4: Cluster Data Use-Case Description	51
Table 4.5: Choose K-means Variant Use-Case Description	52
Table 4.6: Choose Proximity Measure Use-Case Description	53
Table 4.7: Enter Number of Clusters Use-Case Description	54
Table 4.8: Exclude Attributes Use-Case Description	55
Table 4.9: Enter Seed Use-Case Description	56
Table 4.10: Normalise Attributes Use-Case Description	57
Table 4.11: Show Classes to Clusters Evaluation Use-Case Description	58
Table 4.12: Download Cluster Results Use-Case Description	59
Table 7.1 Adjective Rating Scale (Bangor, Kortum and Miller, 2009)	89
Table 7.2 Average SUS Scores for each Adjective Rating (Bangor, Kortum and Miller, 2009)	90
Table 7.3 Summary of Responses Received	90
Table 7.4 Converted SUS Scores	91

## LIST OF FIGURES

Figure 1.1: Web Application Overview	3
Figure 1.2: Kanban Board Sample (Kirovska and Koceski, 2015, p.29)	4
Figure 2.1: General Steps in Data Mining (Han, Kamber and Pei, 2011)	6
Figure 2.2: K-means Algorithm in Action (Han, Kamber and Pei, 2011)	13
Figure 2.3: Built-in Proximity Measures in WEKA	26
Figure 2.4: Error Shown When Picking an Unsupported Distance Metric	26
Figure 2.5: General Architecture of a Web Application	29
Figure 2.6: Kanban Board Sample (Kirovska and Koceski, 2015, p.29)	31
Figure 3.1: Kanban Board Sample (Kirovska and Koceski, 2015, p.29)	33
Figure 3.2: Gantt Chart	39
Figure 3.3: Gantt Chart (continued)	40
Figure 4.1: Use-Case Diagram	47
Figure 4.2: Home page	60
Figure 4.3: Clustering page	61
Figure 4.4: Clustering page with results	62
Figure 5.1: System Architecture	63
Figure 5.2: Context Diagram	64
Figure 5.3: Level 0 Data Flow Diagram	64
Figure 5.4: Level 1 Data Flow Diagram: Cluster Data set	65
Figure 5.5: Activity Diagram: Upload Data set	66

Figure 5.6: Activity Diagram: Choose Sample Data set	67
Figure 5.7: Activity Diagram: Cluster Data set	68
Figure 5.8: k-luster Home Page	69
Figure 5.9: k-luster Missing Values Prompt	69
Figure 5.10: k-luster Cluster Page	70
Figure 5.11: k-luster Cluster Page with Cluster Results	70
Figure 6.1: k-luster GitHub Repository	72
Figure 6.2: Algorithm Variant Selection Field	72
Figure 6.3: K-means and K-means++ Code	73
Figure 6.4: Chebyshev Distance Code	73
Figure 6.5: Average Distance Code	74
Figure 6.6: Canberra Distance Code	74
Figure 6.7: <i>k-luster</i> App on Heroku	75
Figure 6.8: <i>Requirements.txt</i> Contents	75
Figure 6.9: Procfile Commands	76
Figure 6.10 Attribute Exclusion Field	76
Figure 6.11: Drop Excluded Attributes	76
Figure 6.12: Normalise Attributes Field	77
Figure 6.13: Attribute Normalisation Code	77
Figure 6.14: Seed Specification Field	77
Figure 6.15: Seed Specification Code	77
Figure 6.16: Importing the Desmos API	78
Figure 6.17: Cluster Result View	78
Figure 6.18: Update Result Visualisation	78
Figure 6.19: x-axis and y-axis Dropdowns	79

Figure 6.20: x-axis and y-axis Dropdown Event Listeners	79
Figure 6.21: Check to Distinguish Numeric and Textual Attributes	79
Figure 6.22: Textual Attribute Disabled in Attribute Exclusion Field	80
Figure 6.23: Textual Attribute Disabled in Axis Dropdown	80
Figure 6.24: Saving an Uploaded Data set	80
Figure 6.25: Loading an Uploaded Data set	81
Figure 6.26: Check for Missing Values	81
Figure 6.27: Filling in Missing Values	82
Figure 6.28: Check for Missing Values in the Iris Data set	82
Figure 6.29: Replacing Values Under Outcome in the Diabetes Data set	83
Figure 6.30: Replacing Values Under Diagnosis in the Breast Cancer Data set	83
Figure 6.31: Downloaded Results	84
Figure 6.32: Classes to Clusters Evaluation	84
Figure 6.33: Chord Distance Code	85
Figure 6.34: Mean Character Difference Code	85
Figure 6.35: Cosine Measure Code	86
Figure 6.36: Czekanowski Coefficient Code	86
Figure 6.37: Index of Association Code	86
Figure 7.1: Plot of Percentile Against SUS Score (Sauro, 2018)	89
Figure 7.2: Overlapping of Clusters	92
Figure 7.3: No Overlapping between Clusters	92
Figure 7.4: From Left to Right, Download Button, Classes to Clusters Button, Results Text	93

Figure 7.5: Popup Message Pointing at Classes to Clusters  
Button

**LIST OF SYMBOLS / ABBREVIATIONS**

JIT	Just-In-Time
GIGO	Garbage in, garbage out
GUI	Graphical user interface
URL	Uniform Resource Locator
IP	Internet Protocol
DNS	Domain Name System
HTTP	Hypertext Transfer Protocol
IDE	Integrated development environment
UI	User interface
JSON	JavaScript Object Notation
SUS	System usability scale
CSV	Comma separated values



## LIST OF APPENDICES

APPENDIX A: SUS Responses

99

## CHAPTER 1

### INTRODUCTION

#### 1.1 General Introduction

This project aims to resolve the constant lack of supported proximity measure options for the K-means algorithm in data mining tools and packages by developing a web application that implements the K-means and K-means++ algorithm along with a plethora of proximity measures, including both similarity and dissimilarity measures.

#### 1.2 Problem Background

The K-means algorithm is a partitioning clustering algorithm that works by iteratively recalculating the centroid of each cluster and reassigning cluster memberships for each data point if necessary (Han, Kamber and Pei, 2011). A proximity measure is used to determine the cluster of a data point. There are many proximity measures available, including Euclidean distance, Weighted Euclidean distance, Manhattan distance, Minkowski distance, Average distance, Chord distance, Mahalanobis distance, Pearson coefficient, and many others (Shirkhorshidi, Aghabozorgi and Wah, 2015). The algorithm is halted when there are no more changes to any of the clusters.

There has always been a shortage of choices for proximity measures available for use with the algorithm in many data mining tools. The following are a few examples of popular data mining tools and the available distance metrics for the K-means algorithm:

1. amap (R package) (RDocumentation, n.d.)
  - Euclidean distance
  - Maximum distance
  - Manhattan distance
  - Canberra distance
  - Binary distance
  - Pearson correlation coefficient
  - Absolute Pearson correlation coefficient
  - Correlation coefficient
  - Absolute correlation coefficient
  - Spearman rank correlation coefficient
  - Kendall rank correlation coefficient
2. scikit-learn (Python package):
  - Euclidean distance
3. Weka:
  - Euclidean distance
  - Manhattan distance
4. Oracle Data Mining (Oracle, n.d.)
  - Euclidean distance
  - Cosine distance

As seen above, only the amap package for R supports more than 2 proximity measures. This is the main inspiration behind this project.

### 1.3 Problem Statement

Many popular data mining tools such as scikit-learn the Python package, Weka, and Oracle Data Mining do not implement a large variety of proximity measures for K-means clustering. Most of them only include common similarity measures such as Euclidean distance.

### 1.4 Project Objectives

This project aims to solve the problem above by achieving the following objectives:

- To implement K-means and K-means++ algorithm using web scripting languages.
- To integrate two categories of proximity measures, namely, similarity and dissimilarity measures into the K-means and K-means++ algorithms.

### 1.5 Project Solution

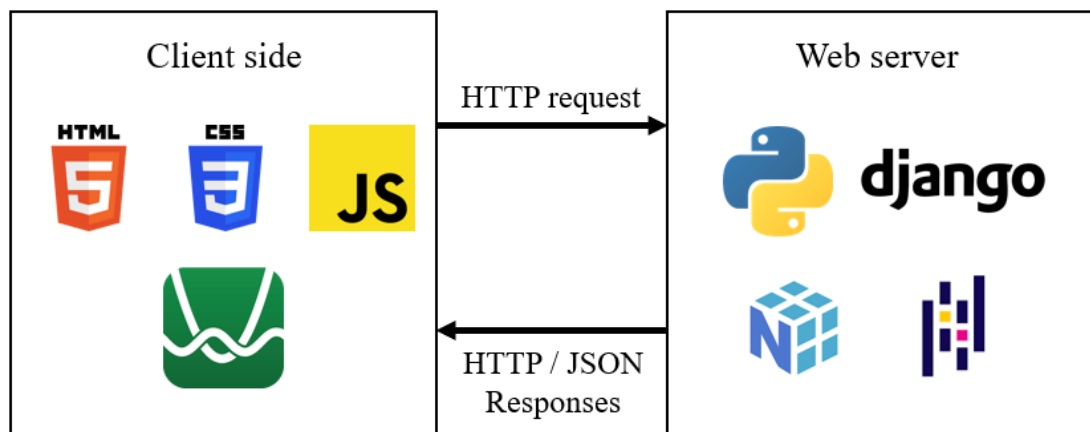


Figure 1.1: Web Application Overview

A web application that implements the K-means and K-means++ algorithm was developed. Figure 1.1 shows the languages and libraries that were used to develop the system. The client side or front-end of the application is handled by HTML, CSS JavaScript and the Desmos API, whereas the back-end or the web server is coded using Python, Django, NumPy and pandas. Django is responsible for providing web services. NumPy and pandas are used for their blazingly fast speed when manipulating data sets. The entire web application is hosted on Heroku.

Both the K-means and K-means++ clustering algorithms are written in Python. The web application allows users to upload their data sets and input their desired settings for the K-means algorithm. These include the proximity metric used, the number of clusters, excluded attributes, seed and attribute normalisation. The web application also allows the user to choose from three sample data sets, in case the user does not want to bother with uploading their own data set. The clustering task is done on the web server with Python, pandas and NumPy. Once complete, the web server returns the cluster results to the client where JavaScript will update the page dynamically.

## 1.6 Project Approach

The aforementioned application was built with Kanban, an agile development methodology. Kanban, meaning “signboard” in Japanese (Corona and Pani, 2013, p.3), is a concept used by Toyota’s Just-In-Time (JIT) production system during the 1950s (Kirovska and Koceski, 2015, p.25). When applied to software development, a Kanban board is used to keep track and visualise the project’s progress.

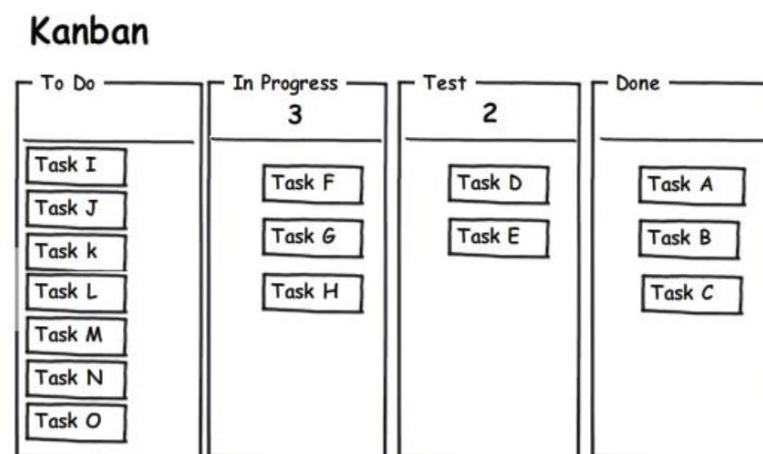


Figure 1.2: Kanban Board Sample (Kirovska and Koceski, 2015, p.29)

## **1.7 Project Scope**

The final deliverable is a web application designed to work on desktops or laptops with modern browsers such as Mozilla Firefox, Google Chrome, and Opera. The web application features the K-means and K-means++ clustering algorithms, and allow users to upload their own data sets and configure the algorithm settings according to their wishes. The application was built using HTML, CSS, JavaScript, Python Django, NumPy and pandas. The application is hosted on Heroku, so that users who are interested may easily access and use it.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 The K-means Algorithm

This section introduces the K-means algorithm, explains its inner workings and ties it with the need for this project.

##### 2.1.1 Brief Introduction to Data Mining

In the simplest way possible, data mining is the process of exacting valuable insights and actionable knowledge from heaps of data (Han, Kamber and Pei, 2011). As technology slowly advanced over the past decades, humans have steadily amassed tons of data. However, without proper analysis, these collected data have no practical use. This is where data mining comes in.

Data mining is a practice that consists of steps to process and analyse the collected data. The output at the end of the process is valuable knowledge.

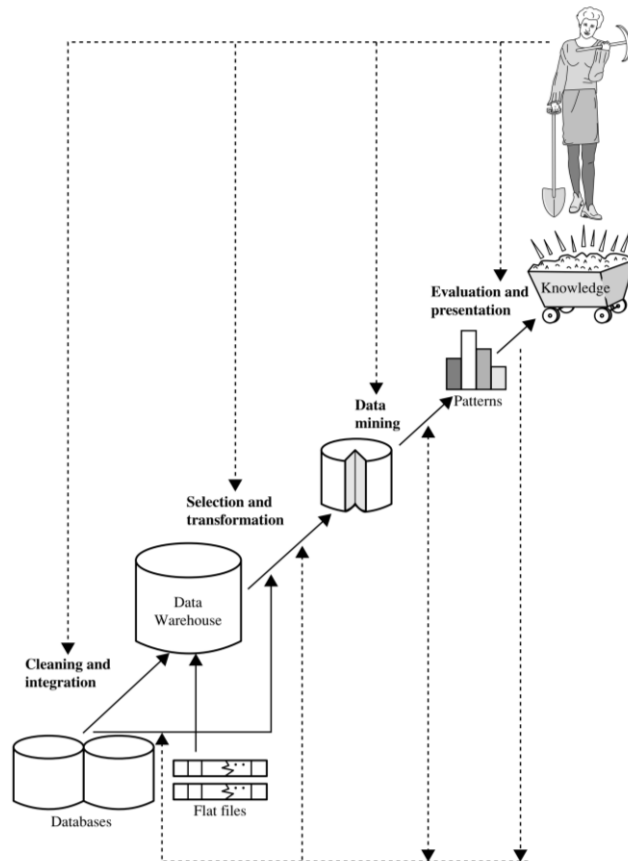


Figure 2.1: General Steps in Data Mining (Han, Kamber and Pei, 2011)

In general, data mining boils down to the steps shown in figure 2.1. Data collected has to be pre-processed first before learning algorithms can be applied onto it.

### **2.1.1.1 The Need for Data Pre-processing**

There is a famous saying in computer science that goes “garbage in, garbage out”, or GIGO for short. Learning algorithms must not be blindly applied on any data set without first pre-processing it. The data sets fed into learning algorithms are directly being used to train the model. Pre-processing is done to ensure the data is accurate and reliable. Without trustable data, the resulting model will not be able to output accurate results, hence the phrase GIGO. If only garbage data is provided to a model, the results obtained will be garbage as well.

Data pre-processing can be broken down into a few steps, which are: data cleaning, data integration, data reduction, and data transformation. Pre-processing is usually the longest phase of data mining. The techniques and tools used for pre-processing depends on the data. There is no one-size-fits-all approach to data pre-processing.

#### **2.1.1.1.1 Data Cleaning**

Data gathered from the real world is often noisy, incomplete and inconsistent (Han, Kamber and Pei, 2011). Noisy is used to describe data filled with outliers; incomplete means certain fields or columns have been left out either intentionally or by accident; inconsistent means data from different fields contradict each other. Data cleaning refers to the transforming and processing of such data in order to make it usable for training models. There are different methods for tackling each issue mentioned above.

For missing values in the data set, the easiest solution is to remove the affected tuples entirely. Otherwise, a value can be inferred using data from other tuples. Noisy data or outliers can be smoothed out to better fit with the rest of the data using binning and regression algorithms. Other than that, clustering algorithms such as DBSCAN can be applied to detect and remove outliers from the data set (Liu, et. al., 2019).



#### **2.1.1.1.2 Data Integration**

Data collected for model training will likely come from different sources, be it different databases, data warehouses, data cubes, or raw files. As such, the data collected will be in different formats and schemas. Thus, techniques are needed to merge data from different data stores into a single coherent data set for analysis. This merging process is known as data integration (Han, Kamber and Pei, 2011).

The main concerns when conducting data integration are the entity identification problem, data duplication and data conflicts (Han, Kamber and Pei, 2011). The entity identification problem describes the challenge of identifying and matching up data belonging to the same entity. For example, when building a customer profile using data collected from multiple stores and shops, how can all data pertaining to the same user be identified and joined together? Merely merging them based on similar or equivalent names is insufficient, as users may provide their real name to some stores and nicknames to others. Furthermore, the customer names might not even be provided out of respect for the customers' privacy. The only identifying attribute given would be their customer IDs from different databases. In cases like these, how can different customer IDs with no correlation between each other be matched up?

Data duplication refers to the repetition of tuples in the combined data set. Duplicated tuples have to be removed from the data set prior to analysis as it will throw off the model's accuracy and degrade its performance both when learning and predicting. An example of a data conflict is inconsistent data values for the same entity received from different sources. This could be due to a variety of reasons, including different units or scales used, data entry errors, or the entity simply giving different values to both sources by accident.

#### **2.1.1.1.3 Data Reduction**

In most cases, the amount of data made available to us for analysis is too large to be fed into an algorithm entirely. It would take an exceptionally long time for the algorithm to learn from the entire data set. Thus, steps need to be taken to alleviate this issue. Data reduction is the process of reducing the cardinality and dimensionality of the data, yet still closely preserving the overall integrity and feature distributions of the original data set (Han, Kamber and Pei, 2011).

The most common technique to reduce cardinality is random sampling. This means to randomly select a certain percentage of tuples from the original data set to form the new reduced data set. Sampling can be done either with or without replacement. Other than that, data can be aggregated to dramatically reduce the number of tuples. For example, data on profits gained daily can be totalled up to be monthly, quarterly or even yearly. For reducing dimensionality, unrelated attributes are simply dropped before the data set is passed to the algorithm.

#### **2.1.1.1.4 Data Transformation**

Lastly, data transformation is the process of transforming the data to optimise the mining process and to achieve a better and more accurate result (Han, Kamber and Pei, 2011). Usual steps taken in the data transformation phase include normalisation, discretisation, and attribute construction (Han, Kamber and Pei, 2011). Normalisation is used to scale a numerical attribute so that all data values within it fall in a certain range. Normalisation is usually done to enforce a consistent range across multiple numerical features. Discretisation means to convert numerical data into discrete data. A common example is the conversion of age numbers into a categorical label such as young, teens, middle-aged or elderly. Attribute construction refers to the derivation of new attributes from existing features.

### 2.1.1.2 Data Mining Algorithms

Once data has been pre-processed, data mining algorithms can be applied onto them to obtain insights and knowledge. Data mining algorithms can be categorised into several types, the most prominent ones being classification, regression, and clustering.

Classification problems are where given a certain set of known data values, predict a discrete class which the set belongs to (Han, Kamber and Pei, 2011). For example, given a person's age, daily lifestyle habits, medical history, blood pressure and BMI, predict if the patient is at risk of developing a heart attack. Popular classifier algorithms include decision tree classifiers such as ID3, C4.5 and CART (Han, Kamber and Pei, 2011). Regression tasks are similar to classification tasks, but instead of predicting a discrete class label, a continuous numerical value is predicted.

However, clustering is conceptually different from classification and regression. In classification and regression, a class label and a numeric value is to be predicted respectively. For clustering tasks, the goal is to group data points that "belong together" into the same cluster. The method used to determine if points belong in the same cluster vary depending on the clustering algorithm used. Examples of clustering algorithms include affinity propagation, mean-shift, agglomerative clustering, DBSCAN and K-Means (scikit-learn, n.d.), which is the focus of this project.

## **2.1.2 The K-means Algorithm**

This section explains the origin of the K-means algorithm, how the algorithm works, its strengths and drawbacks.

### **2.1.2.1 Brief History of the K-means Algorithm**

The K-means algorithm is by no means a modern idea. The three most commonly used variations of the K-means algorithm are the Lloyd algorithm, the Forgy algorithm, the MacQueen algorithm, and the Hartigan-Wong algorithm (Morissette and Chartier, 2013). The Lloyd algorithm and the Forgy algorithm were developed in 1957 and 1965 respectively, but the Lloyd algorithm was not published until 1982. The clustering steps in these two algorithms are exactly the same, with the only difference being the consideration of data distribution (Morissette and Chartier, 2013). The Lloyd algorithm was defined for discrete data, whereas the Forgy algorithm considered continuous data. The MacQueen algorithm, which was proposed in 1967, is a slight variation on the Lloyd and Forgy algorithms. The Hartigan-Wong algorithm was first described by Hartigan in 1975, but was improved upon in 1979 by Hartigan and Wong (Hartigan and Wong, 1979). The first variation of the K-means algorithm appeared over 60 years ago, and yet this clustering technique still remains used till this day.

### 2.1.2.2 Explanation of the K-means Algorithm

This section describes the Lloyd/Forgy algorithm, which is the variant used in this project.

The first step in using the algorithm is to pick a value for  $K$ , which is the number of clusters to segregate the data points into. Then,  $K$  number of data points are randomly chosen to be the initial cluster centroids. Next, each point in the data set is assigned to the cluster it is most similar to. To determine the point's similarity with each cluster, a certain proximity measure is used. Proximity measures will be further explained in section 2.1.3. After each value has been assigned a cluster membership, all cluster centroids are recalculated to be the mean of all points assigned to them. Then, each point's similarity with all cluster centroids are re-evaluated, and its cluster membership is reassigned if necessary. After that, each cluster centroid is recalculated again. This repeats until there are no further changes in cluster memberships and cluster centroids.

Thus, the pseudocode for the K-means algorithm is as follows:

Table 2.1: K-means Algorithm Pseudocode (Han, Kamber and Pei, 2011)

(1)	Choose a value for $K$
(2)	Randomly choose $K$ points from the data set to act as initial cluster centroids
(3)	DO: For each point, assign/reassign cluster memberships to the most similar cluster centroid Evaluate new cluster centroids by calculating the mean of all points belonging to the cluster WHILE there are still changes to cluster memberships and centroids

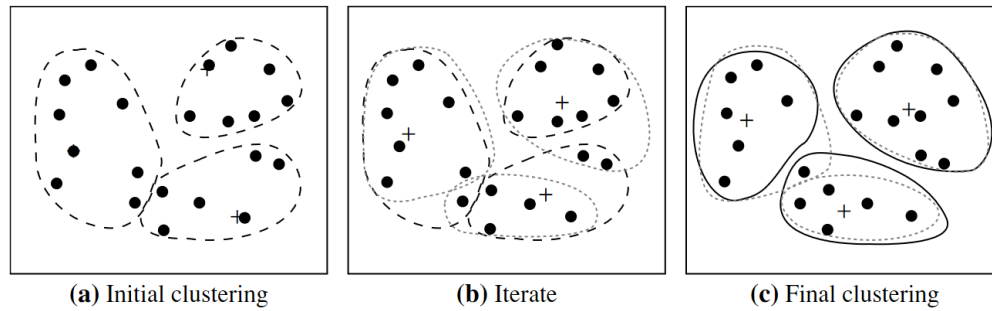


Figure 2.2: K-means Algorithm in Action (Han, Kamber and Pei, 2011)

Figure 2.2 shows an example of the K-means algorithm being applied on an arbitrary data set. In this example, the value of K is 3. The plot labelled (a) shows the initial clusters based on the random points chosen to be the cluster centroids. Once cluster memberships have been assigned to each point, the cluster centroids are calculated as the mean of all points within the cluster. The centroids are labelled with a “+” in the second plot. Then, each data point is reassigned a new cluster if needed, and the cluster centroids are recalculated. The final plot on the right shows the final clusters, the output of the algorithm.

### 2.1.2.3 Strengths and Drawbacks of the K-means algorithm

The most obvious advantage of the K-means algorithm is its simplicity and ease of implementation. The algorithm’s low computational cost and memory usage also contributed to it remaining popular throughout the decades (Morissette and Chartier, 2013) Furthermore, the algorithm also scales relatively well since it has a time complexity of  $O(nKt)$ , where  $n$  is the number of data points,  $k$  is the number of clusters,  $t$  is the number of iterations (Han, Kamber and Pei, 2011).

On the other hand, the K-means algorithm is sensitive to outliers and noise data (Han, Kamber and Pei, 2011), as it uses a mean measure to determine the cluster centroids. Other than that, some data scientists view the need for specifying the number of clusters beforehand to be a weakness (Han, Kamber and Pei, 2011). Other clustering methods such as hierarchical clustering may be able to operate without first specifying the number of clusters, but they are not as efficient as the K-means algorithm.

### 2.1.3 The K-means++ Algorithm

K-means++ is a variant of the K-means algorithm that modifies the process of picking the initial cluster centroids to favour more spread out arrangements. The algorithm was proposed by Arthur and Vassilvitskii in 2007. The pseudocode for the K-means++ algorithm is as follows:

Table 2.2: K-means++ Algorithm Pseudocode (Arthur and Vassilvitskii, 2007)

(1)	Choose a value for K
(2)	Randomly choose a single point from the data set, with uniform chance for all points to be chosen
(3)	WHILE number_of_clusters < K FOR all points in the data set Calculate probability of choosing that point with $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$ Choose a random point with the probabilities calculated
(4)	DO: For each point, assign/reassign cluster memberships to the most similar cluster centroid Evaluate new cluster centroids by calculating the mean of all points belonging to the cluster WHILE there are still changes to cluster memberships and centroids

where

$D(x)$  = distance from that particular data point to the closest cluster centroid

Notice how the only difference between this algorithm and the regular K-means algorithm is the way the initial centroids are picked. By assigning a higher probability of being picked to data points that are further away, the initial chosen centroids tend to be more spread out. Also note that step 4 in table 2.2 is the exact same as step 3 in table 2.1.

#### **2.1.4 Proximity Measures**

In section 2.1.2.2, it is mentioned that a certain proximity measure is needed to determine a point's similarity with each cluster centroid, which is then used to assign the point to a certain cluster. In this section, the purpose of proximity measures in data mining will be explained in more depth. This section also presents a list of proximity measures that can be used with the K-means algorithm.

In data mining, there are many algorithms that require a proximity measure to function. Examples include partitioning clustering algorithms such as K-means and K-medoids (Shirkhorshidi, Aghabozorgi and Wah, 2015). In most cases, Euclidean distance is selected as the distance metric. The proximity measure chosen to be used with these algorithms directly affects the algorithm's performance and results. Due to the significance of a proximity measure's effect, much research has been done looking into new measures and comparing them to existing metrics in terms of performance, advantages and disadvantages when applied to various different types of data such as categorical data and binary data (Shirkhorshidi, Aghabozorgi and Wah, 2015). It should also be noted that proximity measures are not universally applicable to all types of data. Furthermore, there is no single "best" proximity measure for all applications (Shirkhorshidi, Aghabozorgi and Wah, 2015). The optimal metric for each situation needs to be discovered through experimentation.

Proximity measures fall into one of two categories, similarity and dissimilarity measures (Shirkhorshidi, Aghabozorgi and Wah, 2015). For this project, emphasis will be given on proximity measures meant for continuous data. The following proximity measures are all featured in the web application.



### 2.1.4.1 Similarity Measures

Metrics listed under this subsection measure the similarity between two data points. The higher the value, the more alike the two points are. When used with the K-means algorithm, points with a higher value will be placed into the same cluster.

#### 2.1.4.1.1 Czekanowski Coefficient

This proximity measure is known to give reliable results when used with the K-means algorithm for medium-dimensionality data (Shirkhorshidi, Aghabozorgi and Wah, 2015).

$$d = 1 - \frac{2 \sum_{i=1}^n \min(x_i, y_i)}{\sum_{i=1}^n (x_i + y_i)} \quad (2.1)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.1.2 Coefficient of Divergence

Evidence has shown that this proximity metric is able to produce accurate results when used with the K-means algorithm (Shirkhorshidi, Aghabozorgi and Wah, 2015).

$$d = \sqrt{\frac{1}{n} \sum_{i=1}^n \left( \frac{x_i - y_i}{x_i + y_i} \right)^2} \quad (2.2)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

### 2.1.4.1.3 Mean Character Difference

In contrast to Czekanowski coefficient, research has shown that mean character difference proximity measure tends to produce inaccurate results when used with the K-means algorithm for data sets with a high number of dimensions (Shirkhorshidi, Aghabozorgi and Wah, 2015).

$$d = \frac{1}{n} \sum_{i=1}^n |x_i - y_i| \quad (2.3)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

### 2.1.4.1.4 Index of Association

This measure is known to produce inaccurate results when used with K-means.

$$d = \frac{1}{n} \sum_{i=1}^n \left| \frac{x_i}{\sum_{i=1}^n x_i} - \frac{y_i}{\sum_{i=1}^n y_i} \right| \quad (2.4)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.1.5 Pearson Coefficient

This measure is commonly used for clustering gene expression data (Shirkhorshidi, Aghabozorgi and Wah, 2015). One of its biggest disadvantage is that it is easily affected by outliers.

$$Pearson(x, y) = \frac{\sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^n (x_i - \mu_x)^2} \sqrt{\sum_{i=1}^n (y_i - \mu_y)^2}} \quad (2.5)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

$\mu_x$  = mean of all attribute values in  $x$

$\mu_y$  = mean of all attribute values in  $y$

#### 2.1.4.1.6 Cosine Measure

A similarity measure that is usually used for determining document similarity (Shirkhorshidi, Aghabozorgi and Wah, 2015).

$$Cosine(x, y) = \frac{\sum_{i=1}^n x_i y_i}{\|x\|_2 \|y\|_2} \quad (2.6)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

$$\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$$

$$\|y\|_2 = \sqrt{\sum_{i=1}^n y_i^2}$$

### 2.1.4.2 Dissimilarity Measures

Measures listed under this subsection is the exact opposite to those listed in section 2.1.3.1. The measures here quantify the distance between two points. When used with the K-means algorithm, points with a lower dissimilarity will be placed in the same cluster.

#### 2.1.4.2.1 Euclidean Distance

The most common proximity measure used with the K-means algorithm, and also the default option in most cases, if not the only option. This metric works well in most cases, but is however sensitive to outliers and easily affected by largely-scaled attributes (Shirkhorshidi, Aghabozorgi and Wah, 2015). To solve the issue of largely-scaled features, normalisation is usually performed beforehand. The equation for computing the Euclidean distance between two points is as follows:

$$d = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (2.7)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.2.2 Weighted Euclidean Distance

A variation of Euclidean distance that allows weights to be assigned for each attributes. This allows the algorithm to put more emphasis on certain attributes when during the clustering phase (Shirkhorshidi, Aghabozorgi and Wah, 2015). The equation for this metric is as follows:

$$d = \sqrt{\sum_{i=1}^n w_i (x_i - y_i)^2} \quad (2.8)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

$w_i$  = weight assigned to attribute  $i$

#### 2.1.4.2.3 Average Distance

Another variant of Euclidean distance, designed to dampen the effects of outliers within the data set (Shirkhorshidi, Aghabozorgi and Wah, 2015) by averaging out the sum of distances for all attributes. The formula for this is as below:

$$d = \sqrt{\left(\frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2\right)} \quad (2.9)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.2.4 Manhattan Distance

A common alternative to Euclidean distance, has the same strengths and drawbacks as Euclidean distance (Shirkhorshidi, Aghabozorgi and Wah, 2015). Manhattan distance is also known as city block distance (Cha, 2007).

$$d = \sum_{i=1}^n (|x_i - y_i|) \quad (2.10)$$

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.2.5 Minkowski Distance

A general case for Euclidean distance and Manhattan Distance. In fact, both Euclidean distance and Manhattan distance are measures that belong under the Minkowski family (Shirkhorshidi, Aghabozorgi and Wah, 2015). When  $m = 1$ , the formula gives the same result as Manhattan distance. When  $m = 2$ , the equation gives the same result as Euclidean distance. The equation for Minkowski distance is as follows:

$$d = (\sum_{i=1}^n |x_i - y_i|^m)^{\frac{1}{m}}, m \geq 1 \quad (2.11)$$

where

$n$  = number of attributes

$m$  = a positive real number

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.2.6 Chebyshev Distance

An extension of Minkowski distance. As the value of  $m$  approaches infinity, the distance obtained is known as the Chebyshev distance (Cha, 2007). The distance can be approximated by finding the maximum absolute distance across all attributes. The following is the equation for that approximation:

$$d = \max_i |x_i - y_i| \quad (2.12)$$

where

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

#### 2.1.4.2.7 Chord Distance

A modified version of Euclidean distance developed to overcome the weaknesses of Euclidean distance, including issue of it being heavily affected by largely scaled attributes (Shirkhorshidi, Aghabozorgi and Wah, 2015). It is defined as “the length of the chord joining two normalized points within a hypersphere of radius one” (Shirkhorshidi, Aghabozorgi and Wah, 2015). The following is the equation for Chord distance:

$$d = \sqrt{2 - 2 \frac{\sum_{i=1}^n x_i y_i}{\|x\|_2 \|y\|_2}} \quad (2.13)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

$$\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$$

$$\|y\|_2 = \sqrt{\sum_{i=1}^n y_i^2}$$

#### 2.1.4.2.8 Canberra Distance

This dissimilarity measure was first introduced by Lance back in the 1960's and further improved on the next year by Williams (Faisal, Zamzami and Sutarman, 2020). Research conducted shows that this metric is able to provide accurate clustering results, provided that the data has been properly preprocessed. (Faisal, Zamzami and Sutarman, 2020; Thakare and Bagal, 2015). The performance of this metric is comparable to the performances of Euclidean distance and Manhattan distance. However, the Canberra distance is easily affected by largely-scaled attributes. Normalising the data before clustering will rectify this issue.

$$d = \sum_{i=1}^n \frac{|x_i - y_i|}{|x_i| + |y_i|} \quad (2.14)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point



### 2.1.4.2.9 Dissimilarity Approach

The above measures are all known as similarity measures. They measure the similarity between two data values. However, proximity measures can be broken up into similarity and dissimilarity measures. Thus, some dissimilarity measures need to be implemented in the project as well.

Wang (2013) describes a dissimilarity based variation of the K-means algorithm in their conference paper. Dissimilarity measures work in the opposite way as compared to similarity measures. The smaller the value, the closer the two points are together. There are two equations involved, one to calculate attribute dissimilarity, and the other to calculate object dissimilarity. Attribute dissimilarity measures the dissimilarity between two values of the same attribute in two data points.

$$att\_dissimilarity(x_i, y_i) = \frac{||x_i - \mu_i| - |y_i - \mu_i||^2}{\max(i) - \min(i)} \quad (2.15)$$

where

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

$\mu_i$  = mean of all values for attribute  $i$

$\max(i)$  = mean of all attribute values in  $y$

$\min(i)$  = mean of all attribute values in  $y$

Object dissimilarity is simply the average of the attribute dissimilarities for all attributes in two points.

$$obj\_dissimilarity(x, y) = \frac{\sum_{i=1}^n att\_dissimilarity(x_i, y_i)}{n} \quad (2.16)$$

where

$n$  = number of attributes

$x_i$  = value of attribute  $i$  for first data point

$y_i$  = value of attribute  $i$  for second data point

## 2.2 Review of Common Data Mining Tools

The main inspiration behind this project is the constant lack of choices for proximity measures in data mining tools that support the K-means algorithm. In this section, several tools are reviewed and a list of proximity measures supported by the tool will be presented. Tools reviewed include libraries or frameworks for programming languages as well as standalone tools.

### 2.2.1 amap (R package)

R is a programming language designed for statistical computing (R, n.d.). The amap package for R provides an implementation of the K-means algorithm. According to RDocumentation (n.d.), the supported proximity measures are as follows:

- Euclidean distance
- Maximum distance
- Manhattan distance
- Canberra distance
- Binary distance
- Pearson coefficient
- Absolute Pearson coefficient
- Correlation coefficient
- Absolute correlation coefficient
- Spearman rank correlation coefficient
- Kendall rank correlation coefficient

### 2.2.2 scikit-learn (Python package)

Python is a programming language that was first released 3 decades ago. It has gained immense popularity in recent years due to its simple and easy to understand syntax. scikit-learn is a very popular Python package among beginners and veterans alike in data science. It provides implementations for most of the commonly used data mining algorithms, including the K-means algorithm. According to the documentation, the only available proximity measure to be used with the K-means algorithm is **Euclidean distance** (scikit-learn, n.d.).

### 2.2.3 WEKA

WEKA is an open-source data mining tool developed at the University of Waikato in New Zealand (Frank, Hall and Witten, 2016). WEKA is an acronym for Waikato Environment for Knowledge Analysis. The tool provides common data mining algorithms in a user-friendly graphical user interface (GUI).

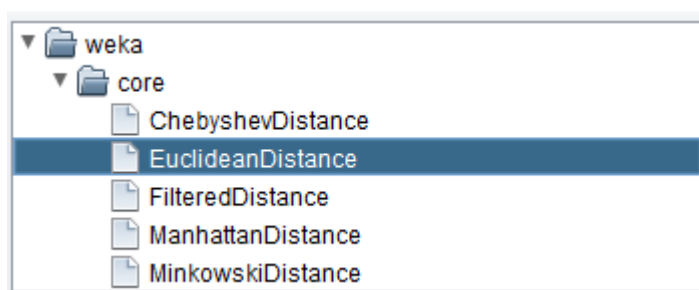


Figure 2.3: Built-in Proximity Measures in WEKA

Figure 2.3 shows all available proximity measures in WEKA. However, the K-means algorithm implementation only supports **Euclidean** and **Manhattan distance**. Picking any other metric yields the following error:

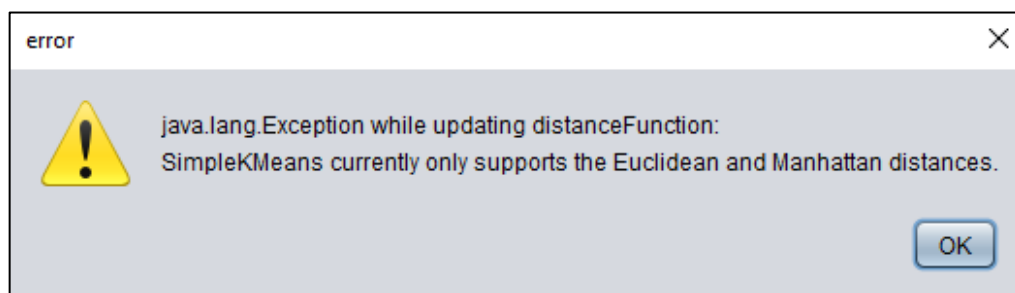


Figure 2.4: Error Shown When Picking an Unsupported Distance Metric

#### 2.2.4 Oracle Data Mining

Oracle Data Mining is a data mining tool built into Oracle Database that offers a variety of data mining algorithms (Oracle, n.d.). It was designed to work on extremely large data sets. Based on the documentation, the available proximity measures for the K-means algorithm are **Euclidean distance** and **Cosine distance** (Oracle, n.d.).

#### 2.2.5 Summary

In summary, out of all data mining tools reviewed in this section, only the amap package for R supports more than 2 proximity measures. The reason for the limited choices of proximity measures in most popular data mining tools is unknown. However, it is clear that the proximity measure chosen has a huge impact on the clustering results obtained from the K-means algorithm. Considering how crucial it is to select the right proximity measure to be used with the K-means algorithm, it is counter-intuitive how few proximity measures are supported by most popular data mining tools. This forms the crux of the issue that this project sets out to solve.

## **2.3 Web Application**

This project aims to solve the constant lack of supported proximity measures for the K-means algorithm in popular data mining tools. This objective of this project is to develop a web application that implements the K-means algorithm and provides numerous choices for proximity measures to use with the algorithm. This section will briefly explain web applications and their general architecture.

### **2.3.1 Introduction to Web Applications**

Basically put, web applications are applications that are hosted entirely online and accessed through a web browser (Jazayeri, 2007). A Uniform Resource Locator (URL) is used to tell the browser where a web page is located. A URL consists of a domain name and a path to the resource. The domain name is converted into an IP address with a Domain Name System (DNS), which points the browser to the computer or server where the application is located. The path points to the object to load.

Hypertext Transfer Protocol (HTTP) is a protocol for communication between clients and web servers (Jazayeri, 2007). There are eight operations defined in the protocol. The most commonly used ones are GET and POST.

### **2.3.2 Static and Dynamic Web Pages**

Web pages fall within one of two groups, static pages and dynamic pages. Static pages are web pages which run entirely on the client side. In other words, static pages are websites developed using only HTML, CSS and JavaScript.

Dynamic pages require a server-side scripting language. The major difference between static and dynamic pages is that dynamic pages are first processed by the server before being displayed to the client (Jazayeri, 2007). For example, the web server can receive a data set uploaded by the client and format the next page accordingly before sending it to the client.

For the web application this project aims to build, a dynamic web page is needed as the contents displayed to the user will change based on the uploaded data set.

### 2.3.3 General Architecture of a Web Application

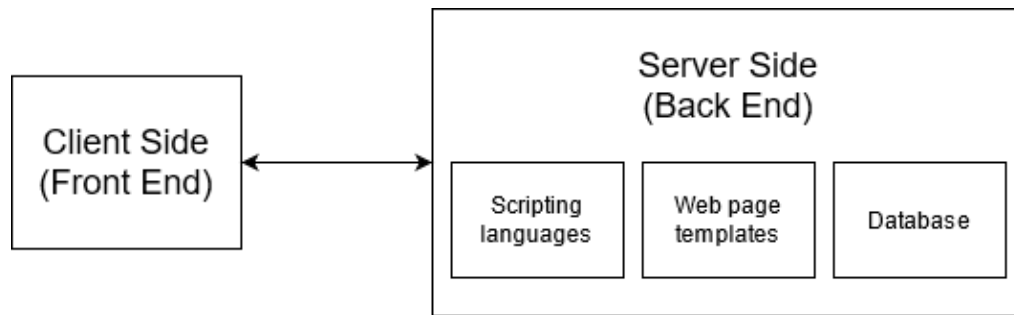


Figure 2.5: General Architecture of a Web Application

The server will process the template programmatically using scripting languages and many more before sending it to the client. The templates are typically written using HTML, CSS and JavaScript. Web scripting languages include PHP, PERL, Python, and many more.

### 2.3.4 Summary

In conclusion, the web application proposed by this project needs to be a dynamic web page, since the cluster calculations are done on the server side. Besides that, the appearance of the web page needs to change based on the data set that the user uploaded or chose.

## **2.4 Development Methodology**

This section introduces and explains an Agile methodology for software development.

### **2.4.1 Agile**

Agile methodologies deviate from traditional structured development methodologies such as Waterfall, Spiral and so on. Agile methodologies emphasise on flexibility and adapting to change instead of following a concrete plan. There is a common misconception that development teams can choose to follow the Agile methodology. However, Agile methodology is not a single concrete methodology to be followed, but rather a general term referring to methodologies that adhere to the principles stated in the Agile manifesto (Shore and Warden, 2007). Examples of such methodologies are Extreme Programming (XP), Kanban, Scrum, Feature Driven Development and so on.

### **2.4.2 Kanban**

This subsection explains the origins of Kanban and how it has been adapted to software development.

#### **2.4.2.1 Brief Introduction to Kanban**

Kanban got its start in a Toyota production system back in the 1950s (Kirovska and Koceski, 2015). The production system follows the Just-in-time (JIT) concept. This means that the production line only produces parts that are necessary, in just the right quantity and no more. In Japanese, Kanban roughly means signal card. In the context of Toyota's production system, Kanban cards were used to signal what part were needed and the exact quantity.

The first time Kanban was introduced to the software development field was when Microsoft invited David J. Anderson to come up with a method to visualise the work flow within a development team.

### 2.4.2.2 Kanban in Software Development

When applied to software development, a Kanban board is used to visualise the workflow within a team. An example of such a board is shown in figure 2.6 below:

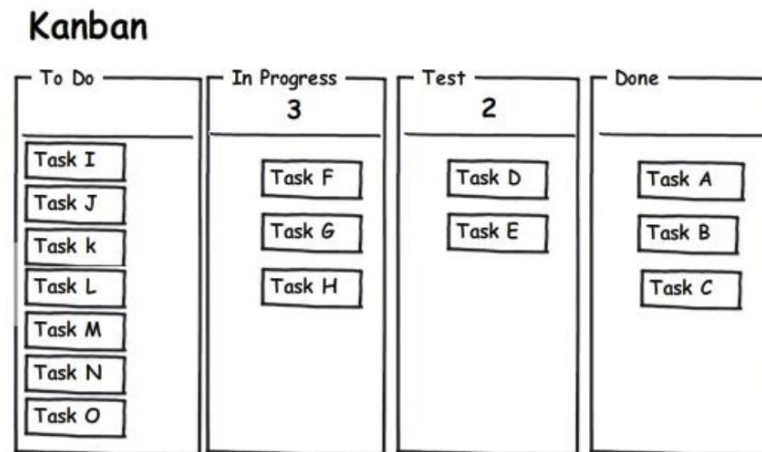


Figure 2.6: Kanban Board Sample (Kirovska and Koceski, 2015, p.29)

There are a few categories or groupings on the board, namely To Do, In Progress, Test, and Done. Under each of these categories are cards labelled with tasks. As work progresses, cards will be moved around from one category to another. For example, when a team member decides to take up a task, the associated card is moved from To Do to In Progress. Similarly, when a task is completed, the associated card is moved to Done. That is the entirety of the Kanban methodology.

This methodology is founded on five core principles (Kirovska and Koceski, 2015):

1. Visualise the workflow
2. Limit items being worked on concurrently
3. Manage flow
4. Make management policies explicit
5. Use models and the scientific method to improve



Principle one is achieved by the Kanban board itself. By having a card for each task, the workflow can be visualised as the project progresses. Principle two is put into place to make Kanban scalable based on the size of the developer team. If there are only a few developers on the team, the manager may consider setting a low limit on the total number of items that can be worked on concurrently. Conversely, if the team was larger, managers could experiment with raising the limit. Principle three is related to principle one.

#### **2.4.2.3 Benefits of Kanban**

The biggest benefit of Kanban is its scalability. As there is only a single developer on the project, selecting a methodology that scales well to the team size is paramount to the success of this project. This is done by setting a limit on the concurrent tasks being worked on during the entire project.

Besides that, Kanban is a relatively simple methodology compared to other Agile methodologies. All developers have to do is to pick a task from the board and work on it until it is completed. This is repeated until the project is done. There is no need for daily stand up meetings and pairwise code reviews. Furthermore, it is quite flexible. The due date for each task can be adjusted based on its complexity instead of following a fixed sprint cycle like Scrum.

## CHAPTER 3

### METHODOLOGY AND WORK PLAN

#### 3.1 Introduction

In this chapter, the methodology chosen for this project will be explained, as well as how it is applied to the project. This chapter also describes the overall plan for completing the project. The tools and frameworks used to build the project are listed in this chapter too.

#### 3.2 Development Methodology

The Kanban development methodology was chosen for this project due to its simplicity, flexibility and scalability. In this methodology, a Kanban board is used to keep track of the project's progress. The board is split into several lists, with each list representing a step in the project workflow. Tasks are moved across lists as the project progresses. Figure 3.1 below an example of a Kanban board.

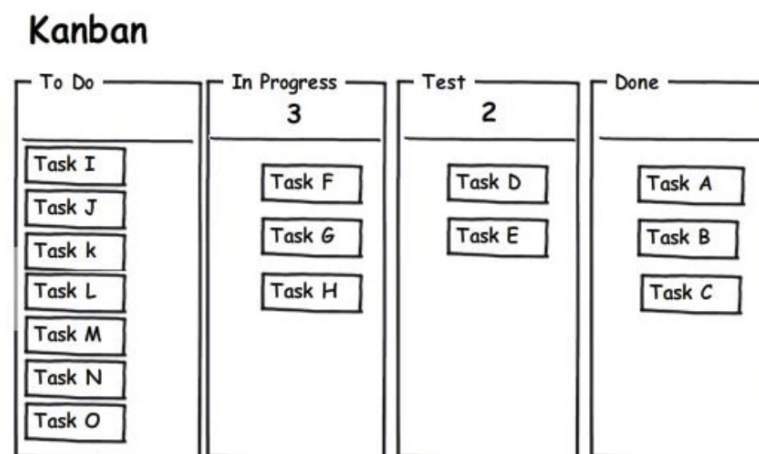


Figure 3.1: Kanban Board Sample (Kirovska and Koceski, 2015, p.29)

For this project, the Kanban board was split into the following lists:

a) Backlog

This list contains all of the project's main tasks, taken from the work breakdown structure. These tasks were arranged from most important to least in the list. Tasks were moved to the prioritise list prior to being worked on.

b) Prioritise

Similar to backlog, but for tasks with a higher priority at that time. These include important tasks picked from the backlog and emergency tasks. An example of an emergency task would be to fix a severe bug that was recently discovered which is holding up the progress of other tasks. This list also acts as a buffer for tasks picked from the backlog. As such, it provides a clearer picture on what needs to be worked on soon and gives a sense that progress is being made.

c) On hold

This list is for tasks that have been started but have hit a roadblock and are unable to be continued on for the time being. There are several possible reasons for this. For example, the task may require another task to have started or completed first;

d) In Progress

Tasks that are currently being worked on will be under this list. Tasks will remain on this list until they have been completed or come to a halt due to some unexpected circumstance. If the task is completed, it is moved to done. If it is unable to progress any further due to an unforeseen reason, it is moved to on hold for the time being.

e) Done

Tasks that have been completed will be placed under this list. Once tasks have been placed into this list, they can no longer be moved out. If a bug is caused by one of the completed tasks, a new task to fix it will be created under backlog or prioritise depending on the severity of the bug.

### **3.3 Work Plan**

In this section, a work plan encompassing the whole project will be laid out.

#### **3.3.1 Work Breakdown Structure**

##### 0.0 A Web-based Implementation of the K-means Algorithm

##### 1.0 Project Initiation

- 1.1 Register title
- 1.2 Kick-start project
- 1.3 Formulate problem statement
- 1.4 Formulate project objectives
- 1.5 Define project scope

##### 2.0 Literature Review

- 2.1 Understand problem domain
  - 2.1.1 Study basic data mining concepts
  - 2.1.2 The K-means algorithm
  - 2.1.3 The K-means++ algorithm
  - 2.1.4 Proximity measures
    - 2.1.4.1 Similarity measures
    - 2.1.4.2 Dissimilarity measures

##### 2.2 Review data mining tools

##### 2.3 Review web frameworks

##### 2.4 Review methodologies

##### 3.0 Project Planning

- 3.1 Select suitable methodology
- 3.2 Develop work breakdown structure
- 3.3 Develop project schedule

### 3.4 Select suitable tools, frameworks and packages

## 4.0 Project Specifications

### 4.1 List functional requirements

### 4.2 List non-functional requirements

### 4.3 Describe use-cases

### 4.4 Develop simple functional prototype

#### 4.4.1 Upload data set

#### 4.4.2 File handling

#### 4.4.3 The K-means algorithm

#### 4.4.4 Euclidean distance

#### 4.4.5 Manhattan distance

#### 4.4.6 Results visualisation

#### 4.4.7 Dummy UI elements

## 5.0 Development

### 5.1 Migrate prototype code

### 5.2 Initialise Git repository and set up GitHub remote repository

### 5.3 K-means++ algorithm

### 5.4 Additional proximity measures

### 5.5 Attribute exclusion

### 5.6 Attribute normalisation

### 5.7 Seed specification

### 5.8 Improve results visualisation

### 5.9 Sample data sets

#### 5.9.1 Iris data set

#### 5.9.2 Diabetes data set

#### 5.9.3 Breast Cancer data set

### 5.10 Sessions implementation

### 5.11 Classes to clusters evaluation

### 5.12 Download results

### 5.13 Deploy to Heroku

## 6.0 Testing

### 6.1 SUS Test

#### 6.1.1 Prepare SUS Test Template

#### 6.1.2 Invite Participants

## 6.1.3 Calculate Final Score

## 7.0 Project Closure

7.1 Deploy web application

7.2 Finalise report

## 3.3.2 Project Schedule

Table 3.1: Project Schedule

Task	Start Date	End Date	Duration (Days)
<b><u>Project Initiation</u></b>			
Register title	5-6-2021	5-6-2021	1
Kick-start project	17-6-2021	17-6-2021	1
Formulate problem statement	19-6-2021	24-6-2021	6
Formulate project objectives	19-6-2021	24-6-2021	6
Define project scope	19-6-2021	24-6-2021	6
<b><u>Literature Review</u></b>			
Study basic data mining concepts	27-6-2021	7-7-2021	11
The K-means algorithm	2-7-2021	11-7-2021	10
The K-means++ algorithm	2-7-2021	11-7-2021	10
Similarity measures	8-7-2021	26-7-2021	19
Dissimilarity measures	8-7-2021	26-7-2021	19
Review data mining tools	27-7-2021	2-8-2021	7
Review web frameworks	30-7-2021	6-8-2021	8
Review methodologies	4-8-2021	13-8-2021	10
<b><u>Project Planning</u></b>			
Select suitable methodology	16-8-2021	17-8-2021	2
Develop work breakdown structure	17-8-2021	24-8-2021	8
Develop project schedule	17-8-2021	24-8-2021	8
Select suitable tools, frameworks and packages	22-8-2021	22-8-2021	1
<b><u>Project Specifications</u></b>			
List functional requirements	20-8-2021	24-8-2021	5
List non-functional requirements	20-8-2021	24-8-2021	5
Describe use-cases	20-8-2021	24-8-2021	5

Upload data set	14-8-2021	15-8-2021	2
File handling	14-8-2021	15-8-2021	2
The K-means algorithm	15-8-2021	17-8-2021	3
Euclidean distance	15-8-2021	17-8-2021	3
Manhattan distance	15-8-2021	17-8-2021	3
Results visualisation	16-8-2021	18-8-2021	3
Dummy UI elements	14-8-2021	18-8-2021	5
<b><u>Development</u></b>			
Migrate prototype code	26-1-2022	2-2-2022	8
Initialise Git repository and set up GitHub remote repository	1-2-2022	6-2-2022	6
K-means++ algorithm	5-2-2022	8-2-2022	4
Attribute exclusion	7-2-2022	9-2-2022	3
Attribute normalisation	8-2-2022	10-2-2022	3
Seed specification	10-2-2022	11-2-2022	1
Improve results visualisation	9-2-2022	13-2-2022	5
Iris data set	21-2-2022	23-2-2022	3
Diabetes data set	21-2-2022	23-2-2022	3
Breast Cancer data set	21-2-2022	23-2-2022	3
Sessions implementation	3-2-2022	25-2-2022	23
Classes to clusters evaluation	15-2-2022	17-2-2022	3
Download results	21-2-2022	24-2-2022	4
Deploy to Heroku	3-3-2022	4-3-2022	2
<b><u>Testing</u></b>			
Prepare SUS Test Template	6-3-2022	8-3-2022	3
Invite Participants	9-3-2022	20-3-2022	12
Calculate Final Score	21-3-2022	21-3-2022	1
<b><u>Project Closure</u></b>			
Finalise report	4-4-2022	10-4-2022	7

### 3.3.3 Gantt Chart

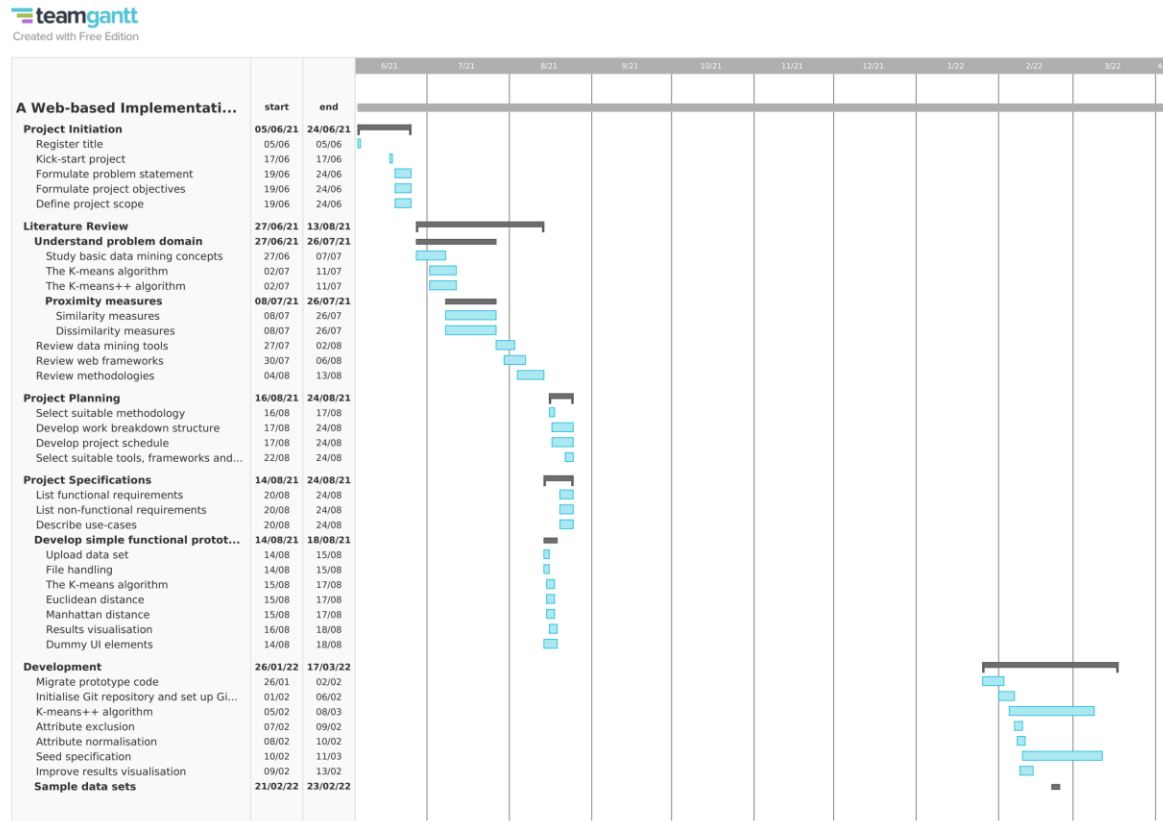


Figure 3.2: Gantt Chart



teamgantt  
Created with Free Edition

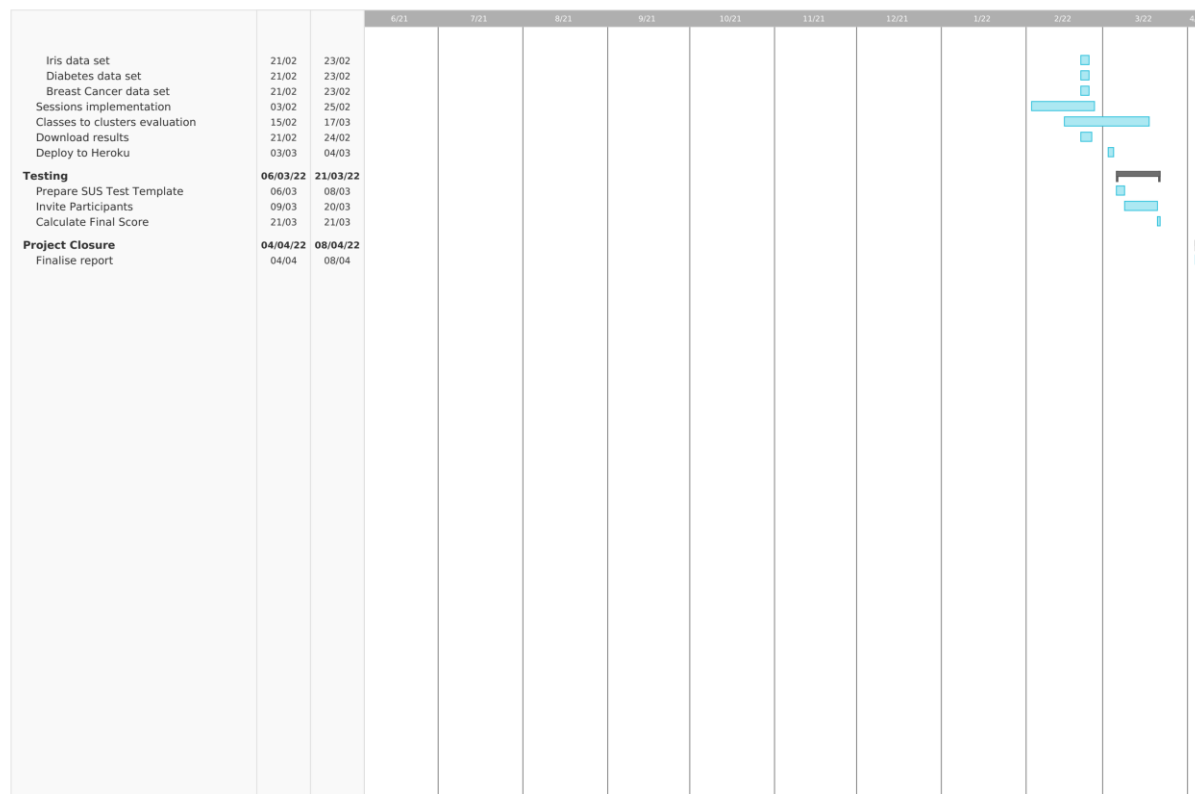


Figure 3.3: Gantt Chart (continued)

### **3.4 Tools and Frameworks**

This section lists the tools and frameworks used to develop the project, and provides a brief explanation of their purpose.

#### **3.4.1 HTML and CSS**

HTML stands for HyperText Markup Language, and CSS stands for Cascading Style Sheets. Together, these two languages form the basic building blocks of websites and web applications. The entire interface of the web application was written using these two languages.

#### **3.4.2 JavaScript**

JavaScript is a popular scripting language that runs on the client side. It is often referred to as the programming language of the web (MDN Web Docs, n.d.). JavaScript scripts can be included with the web pages sent by the web server. These scripts can be made to trigger on certain events, such as buttons being pressed or checkboxes being marked. For this project, JavaScript was used to dynamically update the page.

#### **3.4.3 Python**

Python is a high-level programming language that has grown in popularity in the past few years. Python was chosen for this project due to its simplicity and easy to read code. The K-means algorithm and various proximity measures stated in section 2.1.3 was coded in Python.

Besides that, Python was also chosen because of the wide range of available packages for the language, some of which were utilised in this project. The packages used for this project will be explained in the following subsections.

### **3.4.4 Django**

Django is a popular web framework for Python. Django is used in this project to provide web services so that the web application can be run. Django is responsible for receiving the uploaded data set and other inputs from the client side. The data set and inputs are then passed off to a Python script where the actual clustering takes place. The results are then returned to the client by Django. Django was also chosen over other web frameworks due to its support for dynamic web pages, which is a must have for this project.

### **3.4.5 NumPy**

NumPy is a Python library that adds support for arrays and matrices. While Python supports lists out of the box, the performance is extremely slow compared to NumPy arrays. This is because at its core, NumPy implemented in C. NumPy arrays are used throughout the project wherever possible to speed up performance as much as possible.

### **3.4.6 pandas**

pandas is a powerful Python package designed for data analysis and manipulation. pandas provides a data structure known as DataFrames, and complementary methods to manipulate data stored within them. Most importantly, DataFrames provide a huge performance boost over the built-in data structures in Python such as Lists and Dictionaries. The uploaded data sets are loaded into Python and pre-processed using the pandas package before the K-means or K-means++ algorithm is applied onto it.

### **3.4.7 Matplotlib**

Matplotlib is a Python library for creating data visualisations. It is extremely versatile, and plots can be tailored to meet specific needs. However, it does not provide any intractability. As such, Matplotlib was only used in the prototype to plot the cluster results.

### **3.4.8 Desmos API**

Desmos is a graphing calculator made to aid the learning and teaching of Mathematics. Students and teachers from all over the globe use Desmos to visualise Mathematical functions. It is available as a web application and a mobile application. However, Desmos also provides developers with an API which they can import into their own projects. This allows developers to add an interactive Desmos graphing calculator to their own pages. The Desmos API is used in this project to provide interactive data and results visualisation.

### **3.4.9 Visual Studio Code**

Visual Studio Code, usually abbreviated as VS Code, is a free, lightweight code editor developed by Microsoft. It is not as feature-packed as other IDEs, but still contains most of the commonly used features such as debugging, task running and version control (Visual Studio Code, n.d.). VS Code also has a variety of plugins to assist with development no matter the language used. VS Code was the main editor of choice when developing this project. All Python scripts, HTML, CSS and JavaScript files were written using VS Code.

### **3.4.10 Git**

Git is a version control system developed by Linus Torvalds, the man behind Linux. It was developed to facilitate version tracking and easy collaboration in software development projects. Git was used to keep track of the changes made to the project, and to enable swift rollbacks and rollforwards whenever necessary.

### **3.4.11 GitHub**

GitHub is a remote repository hosting website owned by Microsoft. The website is usually used hand in hand with Git. Aside from providing hosting services, GitHub also provides GitHub Actions which allows for the creation of continuous integration workflows and scripts to help automate repetitive tasks. However, for this project, GitHub was just used as an archive for the project folder.

### **3.4.12 Trello**

Trello is a virtual Kanban board that has integrations with multiple other productivity tools such as JIRA, Bitbucket and Confluence to streamline a project's workflow make managing projects as seamless as possible. Trello was a perfect addition to this project's toolbox as the Kanban methodology was chosen. Trello was used to manage and visualise the project's workflow and progress.

### **3.4.13 Heroku**

Heroku is a platform-as-a-service that allows developers to host their apps for free. Developers only need to pay once they decide to upscale their apps or once they have used up their monthly quota and require more. The web application is hosted on Heroku to provide easy access so that anyone can play around with it on their own devices.

## CHAPTER 4

### PROJECT SPECIFICATION

#### 4.1 Introduction

In this chapter, the functional and non-functional requirements of the web application will be listed. Different use-cases for the web application will also be presented along with their accompanying use-case descriptions. Finally, a functional prototype developed using Django will be presented.

#### 4.2 Functional Requirements

This section lists the functional requirements of the system.

##### 4.2.1 Upload data sets module

- The system shall allow users to upload their data sets in a .csv format.
- The system shall allow users to select from at least 3 pre-prepared data sets.

##### 4.2.2 Data pre-processing module

- The system shall be able to display summary statistics of a data set.
- The system shall allow users to fill in missing values from their uploaded data sets.
- The system shall allow users to exclude certain attributes from being used in the clustering process.
- The system shall allow users to normalise attributes prior to clustering.

##### 4.2.3 Clustering module

- The system shall be able to apply the K-means and K-means++ algorithms on data sets uploaded or selected by the user.
- The system shall allow users to pick a desired proximity measure to be used with the K-means or K-means++ algorithm.
- The system shall allow users to specify the value of K, the number of clusters to form.

- The system shall feature at least 10 proximity measures, consisting of a mix of similarity and dissimilarity measures.
- The system shall allow users to specify a seed to be used when choosing the initial cluster centroids.

#### 4.2.4 Results module

- The system shall be able to display the clustering results in the form of an interactive plot to the user.
- The system shall allow the user to download their cluster results.
- The system shall be able to evaluate the clustering model's performance by comparing the formed clusters with the class label.

### 4.3 Non-functional Requirements

This section lists the non-functional requirements of the system.

#### 4.3.1 Operational requirements

- The system shall work on modern web browsers, including but not limited to, Mozilla Firefox, Google Chrome, and Opera.

#### 4.3.2 Reliability requirements

- The system shall have an availability rate of at least 95%.

#### 4.3.3 Performance requirements

- The system shall be able to return the cluster results within a certain amount of time of starting the cluster task, relative to the number of tuples in the data set. This amount of time includes time for pre-processing and clustering.

Number of tuples	Response time
< 50	< 10 seconds
51 - 200	< 30 seconds
201 – 500	< 2 minutes
501 – 1000	< 5 minutes
> 1000	No time limit

## 4.4 Use-cases

In this section, the use-cases of the web application will be presented.

### 4.4.1 Use-case diagram



Figure 4.1: Use-Case Diagram



#### 4.4.2 Use-Case Descriptions

Accompanying use-case descriptions for the use-cases shown in figure 4.1.

##### 4.4.2.1 Upload Data Set

Table 4.1: Upload Data Set Use-Case Description

Use case name: Upload data set	ID: 1	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Upload their data set for clustering		
Brief Description: Allows users to upload their data set in the form of a .csv file for clustering		
Trigger: Every time the user launches the web application		
Relationship: Association: User Include: - Extend: - Generalization: -		
Normal flows of event: <ol style="list-style-type: none"> <li>1. The user loads the web application.</li> <li>2. The user is brought to the home page.</li> <li>3. If the user decides to upload their own data set, subflow S-1 is performed.  If the user decides to pick one of the sample data sets, the “Choose sample data set” use case (ID: 3) is performed.</li> </ol>		
Sub-flows: S-1: <ol style="list-style-type: none"> <li>1.) The user browses for a data set to upload from their local machine.</li> <li>2.) The user clicks the upload button.</li> <li>3.) The user is redirected to the clustering page.</li> </ol>		
Alternate/ Exceptional Flows: 2.a.) If the uploaded data set contains any missing values, the “Fill missing values” use case (ID: 2) is performed.		

#### 4.4.2.2 Fill Missing Values

Table 4.2: Fill Missing Values Use-Case Description

Use case name: Fill missing values	ID: 2	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Fill in missing values found in their uploaded data set		
Brief Description: Allows users to pick a method to fill in the missing values found in the uploaded data set		
Trigger: When the user uploads a data set with missing values		
Relationship: Association: User Include: - Extend: Upload data set (ID: 1) Generalization: -		
Normal flows of event: <ol style="list-style-type: none"> <li>1. The user picks one of the methods to fill in the missing values.</li> <li>2. The user is redirected to the clustering page.</li> </ol>		
Sub-flows: N/A S-1: 1.) The user clicks on Show Data Set Details button 2.) The web application shows the data sets summary statistics by attribute		
Alternate/ Exceptional Flows: N/A		

### 4.4.2.3 Choose Sample Data Set

Table 4.3: Choose Sample Data Set Use-Case Description

Use case name: Choose sample data set	ID: 3	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Choose a sample data set.		
Brief Description: Allows users to choose one of the sample data sets prepared so that they do not have to upload their own.		
Trigger: When the user decides to choose one of the sample data sets instead of uploading their own		
Relationship: Association: User Include: - Extend: - Generalization: -		
Normal flows of event: <ol style="list-style-type: none"> <li>1. The user clicks one of the buttons corresponding to a sample data set.</li> <li>2. The user is redirected to the clustering page.</li> </ol>		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.4 Cluster Data

Table 4.4: Cluster Data Use-Case Description

Use case name: Cluster data	ID: 4	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Apply the K-means algorithm on the uploaded or chosen sample data set		
Brief Description: Allows users to cluster the data using the chosen settings		
Trigger: When the user clicks on the Cluster! button		
Relationship: Association: User Include: Extend: - Generalization: -		
Normal flows of event: <ol style="list-style-type: none"> <li>1. The user clicks on the Cluster! button.</li> <li>2. The system clusters the data using the algorithm settings provided by the user.</li> <li>3. The system displays the cluster results to the user.</li> </ol>		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.5 Choose K-means Variant

Table 4.5: Choose K-means Variant Use-Case Description

Use case name: Choose K-means Variant	ID: 5	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Choose their desired K-means Variant		
Brief Description: Allows users to choose between the K-means and K-means++ algorithms		
Trigger: When the user wishes to change the K-means variant chosen		
Relationship: Association: User Include: - Extend: Cluster data (ID: 4) Generalization: -		
Normal flows of event: 1. The user chooses either the K-means option or the K-means++ option.		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.6 Choose Proximity Measure

Table 4.6: Choose Proximity Measure Use-Case Description

Use case name: Choose proximity measure	ID: 6	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Choose their desired proximity measure		
Brief Description: Allows users to choose their desired proximity measure to be used with the K-means algorithm		
Trigger: When the user wishes to change the proximity measure used		
Relationship: Association: User Include: - Extend: Cluster data (ID: 4) Generalization: -		
Normal flows of event: 1. The user chooses a proximity measure from the dropdown.		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.7 Enter Number of Clusters

Table 4.7: Enter Number of Clusters Use-Case Description

Use case name: Enter number of clusters	ID: 7	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Enter the number of clusters		
Brief Description: Allows users to enter the number of clusters (the value of k) to form from the data set		
Trigger: When the user wishes to change the number of clusters to form		
Relationship: Aassociation: User Include: - Extend: Cluster data (ID: 4) Generalization: -		
Normal flows of event: 1. The user enters the number of clusters to form.		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.8 Exclude Attributes

Table 4.8: Exclude Attributes Use-Case Description

Use case name: Exclude attributes	ID: 8	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Exclude attributes from the clustering task		
Brief Description: Allows users to exclude some attributes from the clustering task.		
Trigger: When the user wishes to change the attributes included in the clustering task		
Relationship: Association: User Include: - Extend: Cluster data (ID: 4) Generalization: -		
Normal flows of event: 1. The user unchecks the checkboxes next to the attributes they want excluded.		
Sub-flows: N/A		
Alternate/ Exceptional Flows: 2.a.) If the user attempts to cluster when less than two attributes are excluded, an error message will be displayed and the cluster task will not start.		



#### 4.4.2.9 Enter Seed

Table 4.9: Enter Seed Use-Case Description

Use case name: Enter seed	ID: 9	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Specify seed to use		
Brief Description: Allows users to specify the seed to use during the initial cluster centroid selection		
Trigger: When the user wishes specify a seed to use		
Relationship: Association: User Include: - Extend: Cluster data (ID: 4) Generalization: -		
Normal flows of event: 1. The user enters a seed to use.		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.10 Normalise Attributes

Table 4.10: Normalise Attributes Use-Case Description

Use case name: Normalise attributes	ID: 10	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Normalise attributes before clustering		
Brief Description: Allows users to normalise all attributes before the K-means algorithm is applied on the data set		
Trigger: When the user wishes to normalise all attributes		
Relationship: Association: User Include: - Extend: Cluster data (ID: 4) Generalization: -		
Normal flows of event: 1. The user checks or unchecks the checkbox indicating if all attributes should be normalised prior to clustering or not.		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.11 Show Classes to Clusters Evaluation

Table 4.11: Show Classes to Clusters Evaluation Use-Case Description

Use case name: Show classes to clusters evaluation	ID: 11	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – View classes to clusters evaluation results		
Brief Description: Allows users to view the results of classes to clusters evaluation		
Trigger: When the user wishes to view the classes to clusters evaluation results		
Relationship: Association: User Include: - Extend: Generalization: -		
Normal flows of event: <ol style="list-style-type: none"> <li>1. The user clicks on the Classes to Clusters Evaluation button.</li> <li>2. The Classes to Clusters popup is displayed.</li> </ol>		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

#### 4.4.2.12 Download Cluster Results

Table 4.12: Download Cluster Results Use-Case Description

Use case name: Download cluster results	ID: 12	Importance level: High
Primary actor: User	Use case type: Detailed, Essential	
Stakeholders and Interests: User – Download their cluster results		
Brief Description: Allows users to download a csv file containing the original data and an additional column containing the clustering results		
Trigger: When the user wishes to download their clustering results		
Relationship: Association: User Include: - Extend: Generalization: -		
Normal flows of event: <ol style="list-style-type: none"> <li>1. The user clicks on the Download button.</li> <li>2. The system starts the download.</li> <li>3. The downloaded file is a csv file containing the original data and an additional column containing the clustering results</li> </ol>		
Sub-flows: N/A		
Alternate/ Exceptional Flows: N/A		

## 4.5 Prototype

A functional prototype was developed using the Django framework for Python. It supports the uploading of data sets from users, but has been hardcoded to only work with the Iris data set for now. The K-means algorithm has also been programmed into the prototype, along with two proximity measures, namely, Euclidean distance and Manhattan distance. The prototype also uses Matplotlib to plot the cluster results and saves it as an image. The image is then shown to the client-side using Django.

Other UI elements which do not directly serve the above functionalities have been added to the prototype as dummies. This was done to provide a better representation of the final application design. Furthermore, the prototype has been hardcoded to always find 3 clusters using the K-means algorithm.



Figure 4.2: Home page

Figure 4.2 shows the home page of the prototype. The page is split into two sections. The top section is for users to upload their own data sets in the form of a .csv file. The bottom section is for users who wish to pick from the pre-prepared data sets. For this prototype, three dummy buttons have been created, each representing a different data set, namely, the Iris data set, the Diabetes data set, and the Breast Cancer Wisconsin data set respectively.

Once the user has uploaded their data set or picked one from the pre-prepared data sets, they will be brought to the following page:

Figure 4.3: Clustering page

The left page displays all the settings for the K-means algorithm. This includes the proximity measure to use, the value for  $k$  (the number of clusters), the attributes to include in the clustering task, and whether to normalise the attributes before clustering. Finally, there is a button at the bottom to start the K-means algorithm.

The right pane shows the results of the clustering task. Before the user clicks on the Cluster! button, this pane will simply display a message prompting the user to click on the button. Once the user clicks on the button, the page will update to display the results.

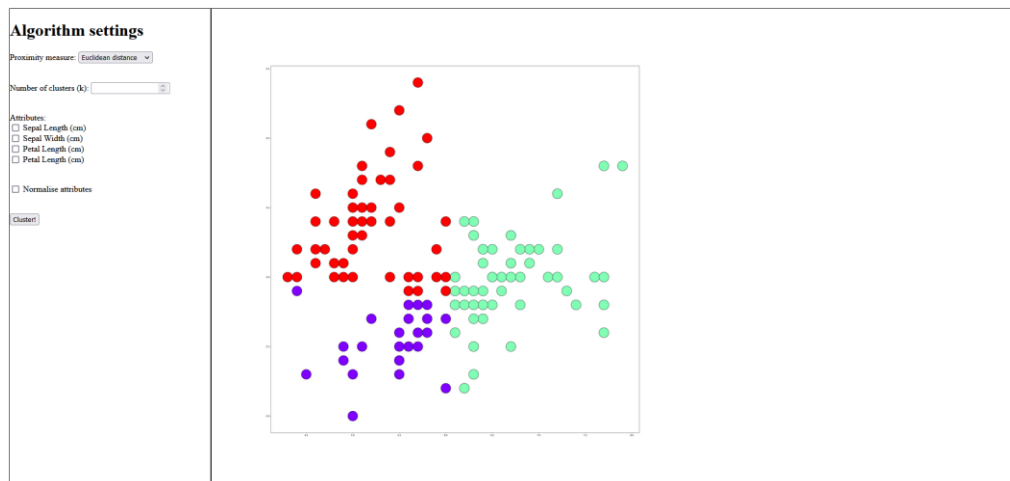


Figure 4.4: Clustering page with results

Figure 4.4 shows the clustering page with results displayed. Subsequent clicks on the Cluster! button will refresh this page with the new results.

## CHAPTER 5

### SYSTEM DESIGN

#### 5.1 System Architecture

There are two components to the web application, the client side (front-end) and the web server (back-end). The client side is responsible for receiving data and clustering results from the web server, and presenting them to the user. The web server will store the data sets and cluster the data based on the settings sent from the client. Additionally, the web server will also render page templates using a template engine before they are sent to the client for display.

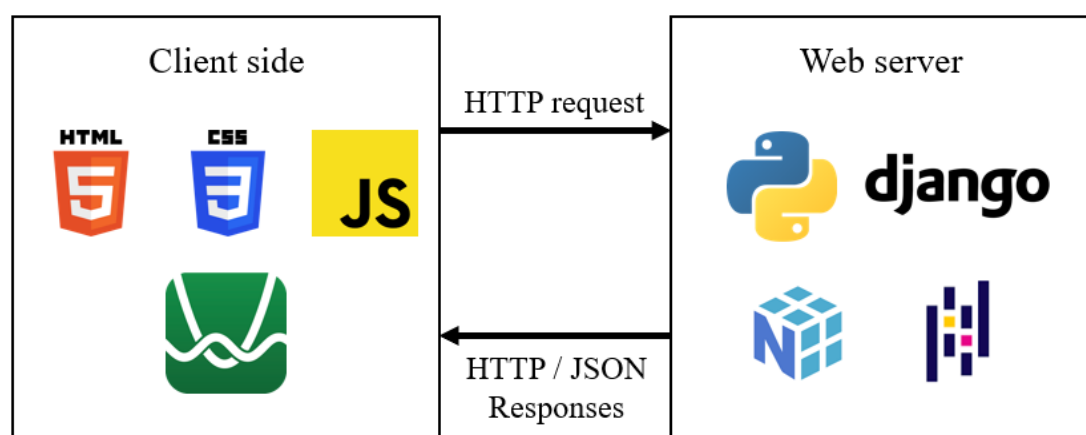


Figure 5.1: System Architecture

The client side is using HTML, CSS, JavaScript and the Desmos API. HTML and CSS are used to form the layout of the page. JavaScript is used to update the page without having to reload the entire web page from scratch. The Desmos API is used to visualise the data set.

As for the web server, Python is the language of choice, with the Django package being used to provide web services. The NumPy and pandas packages are used to manipulate the data set and cluster the data.

Finally, the entire system is hosted on Heroku to provide easy access to users.



## 5.2 Data Flow Diagrams

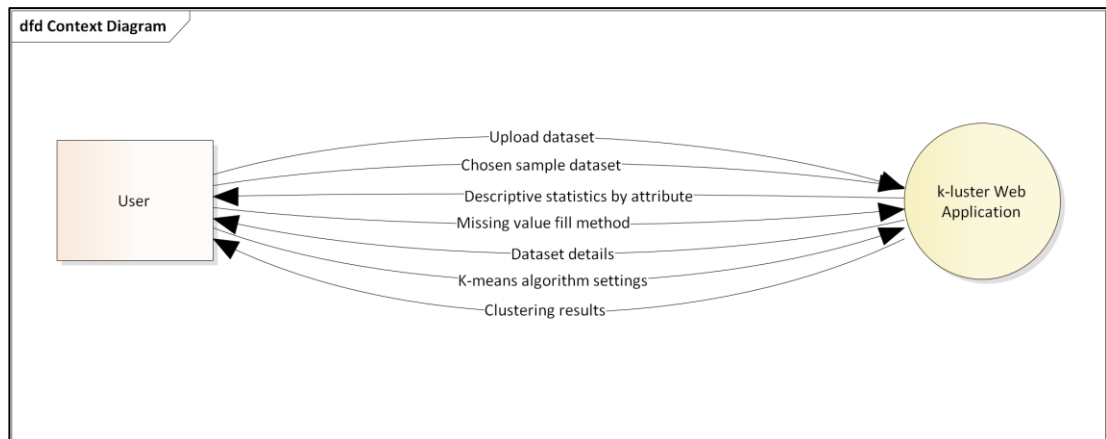


Figure 5.2: Context Diagram

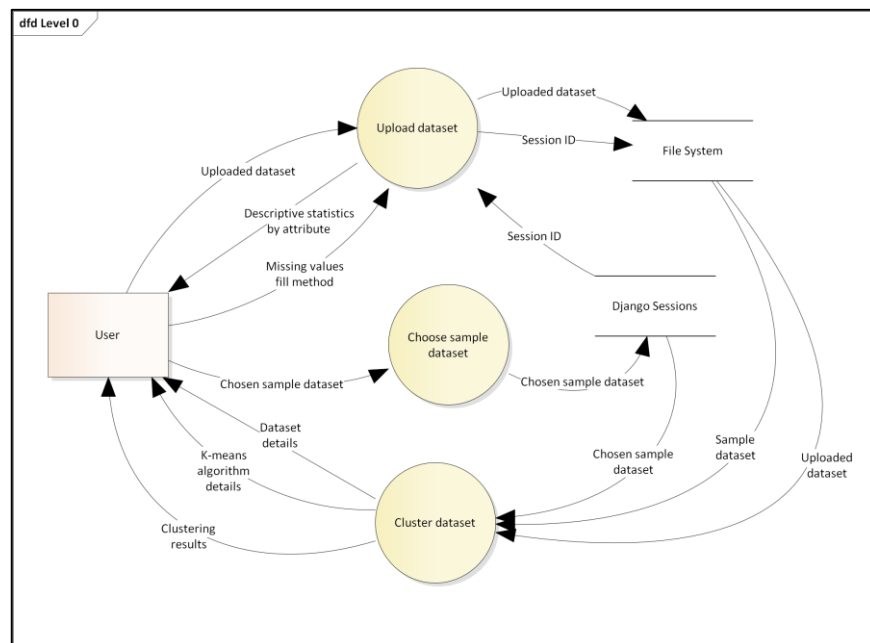


Figure 5.3: Level 0 Data Flow Diagram

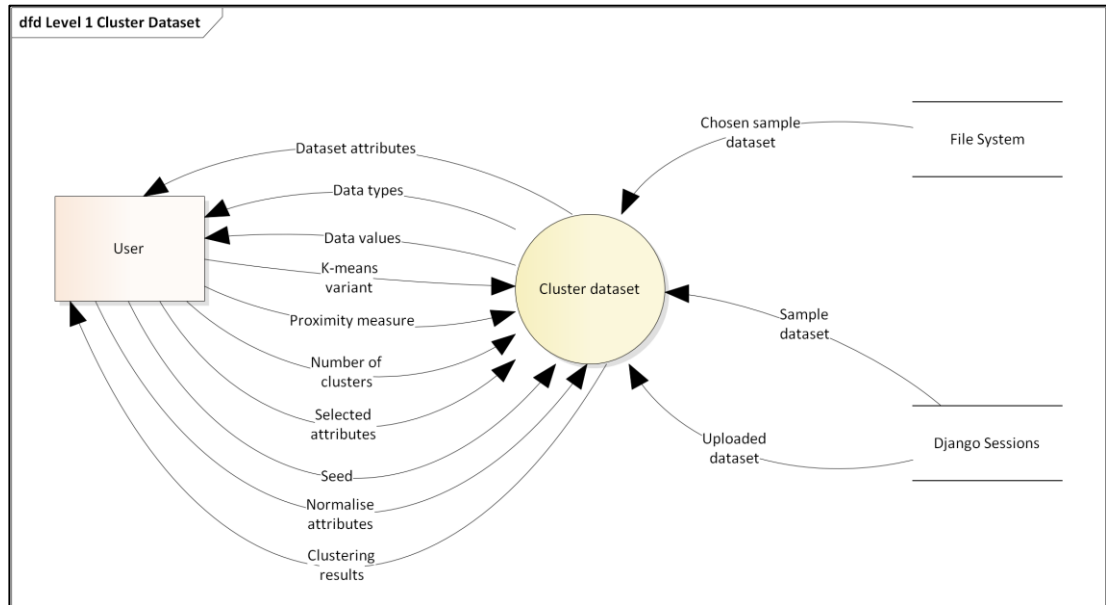


Figure 5.4: Level 1 Data Flow Diagram: Cluster Data set

### 5.3 Activity Diagrams

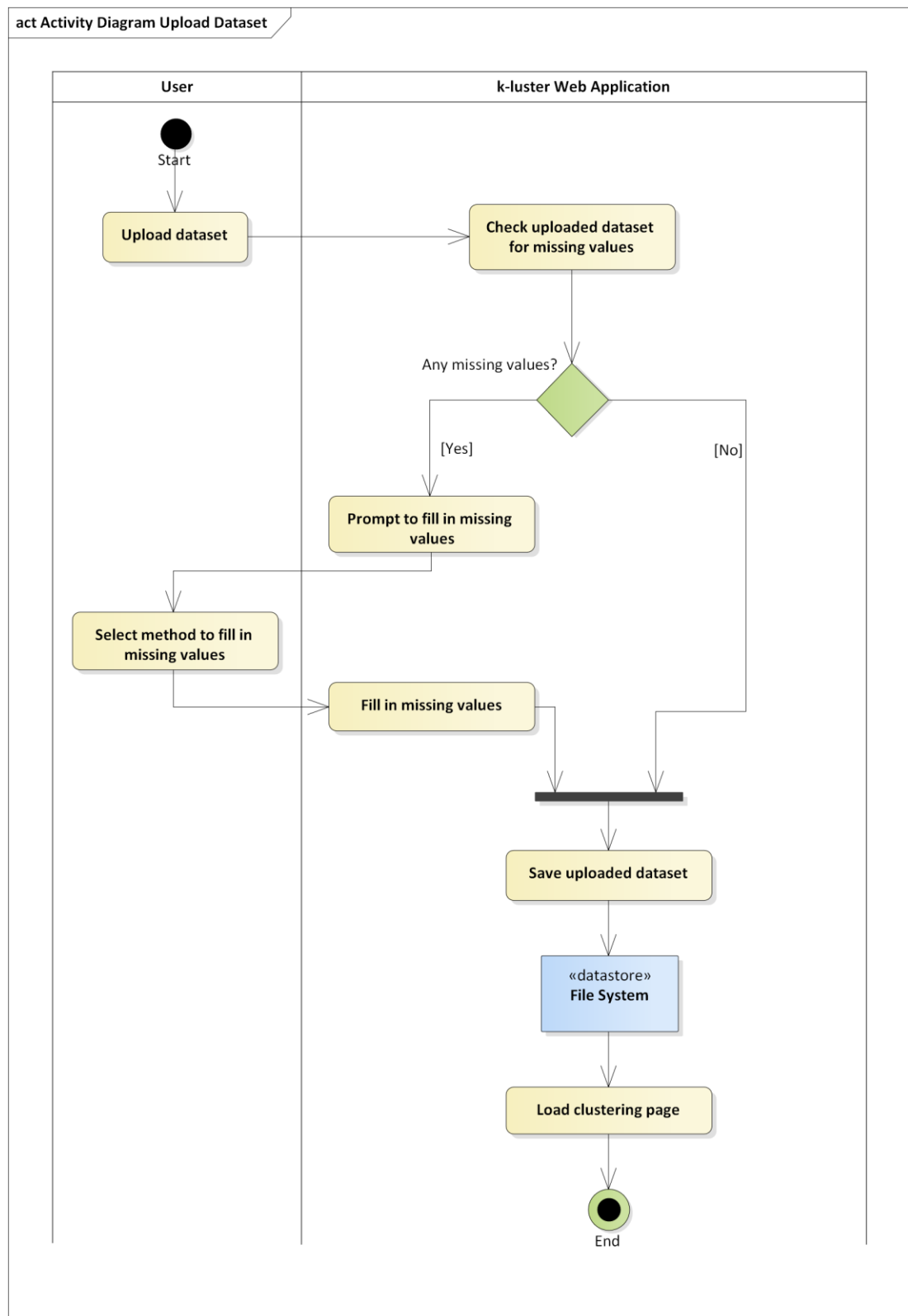


Figure 5.5: Activity Diagram: Upload Data set

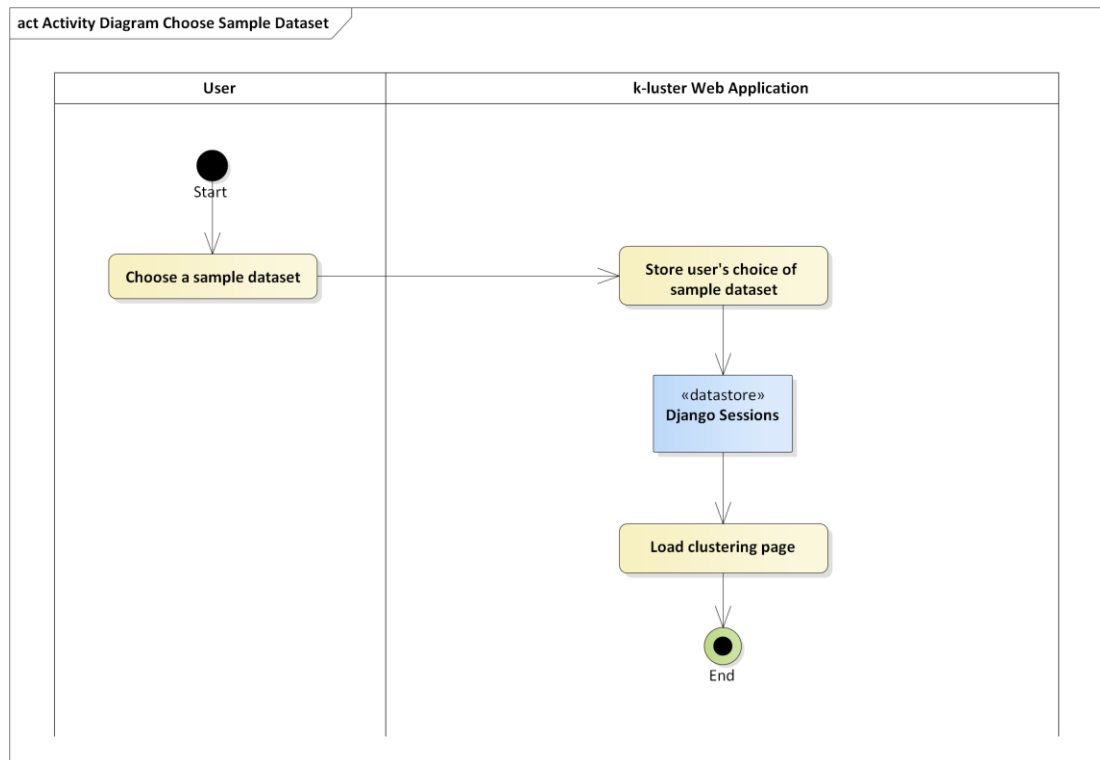


Figure 5.6: Activity Diagram: Choose Sample Data set

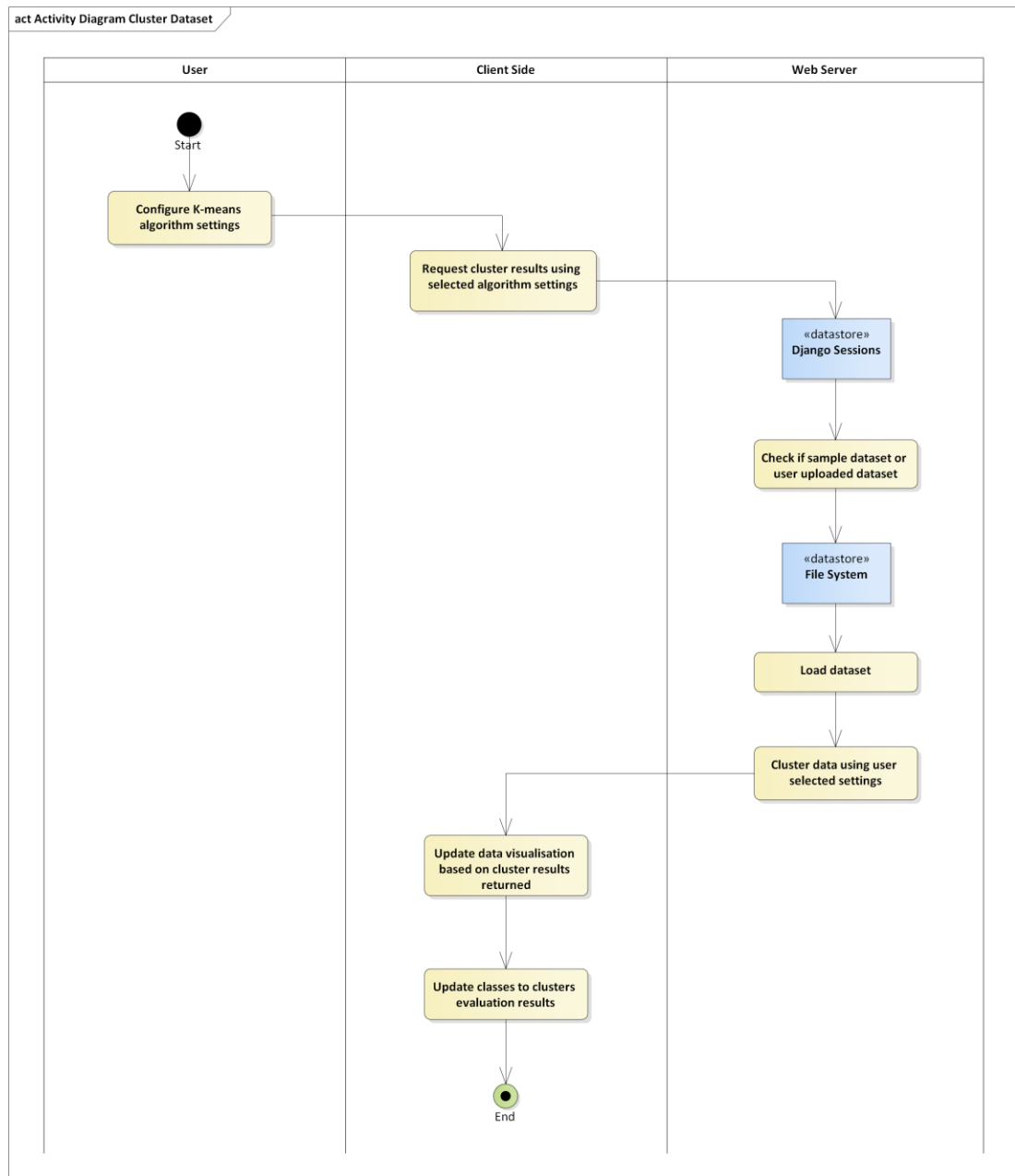


Figure 5.7: Activity Diagram: Cluster Data set

## 5.4 Page Designs

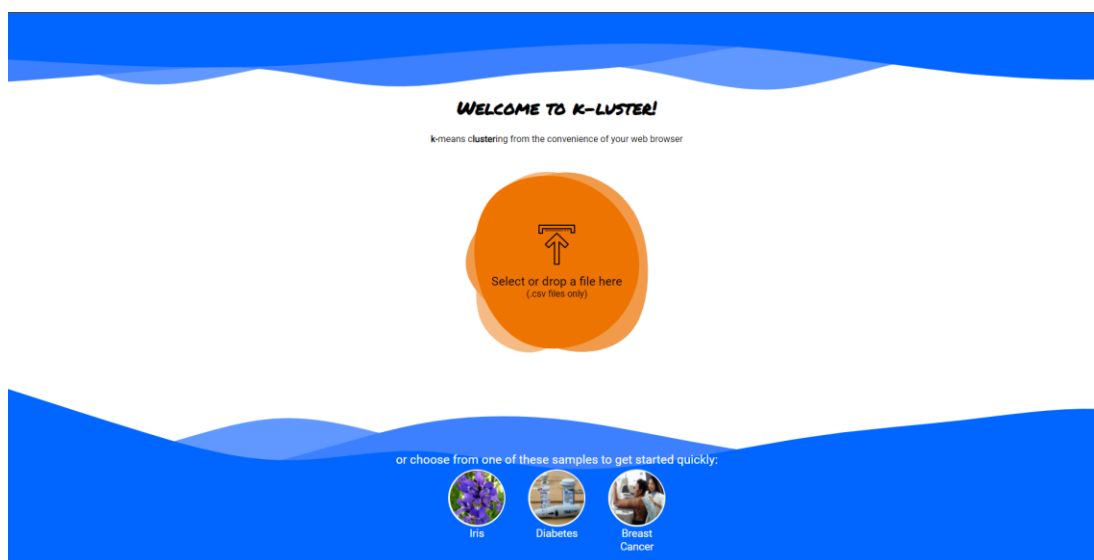


Figure 5.8: k-luster Home Page

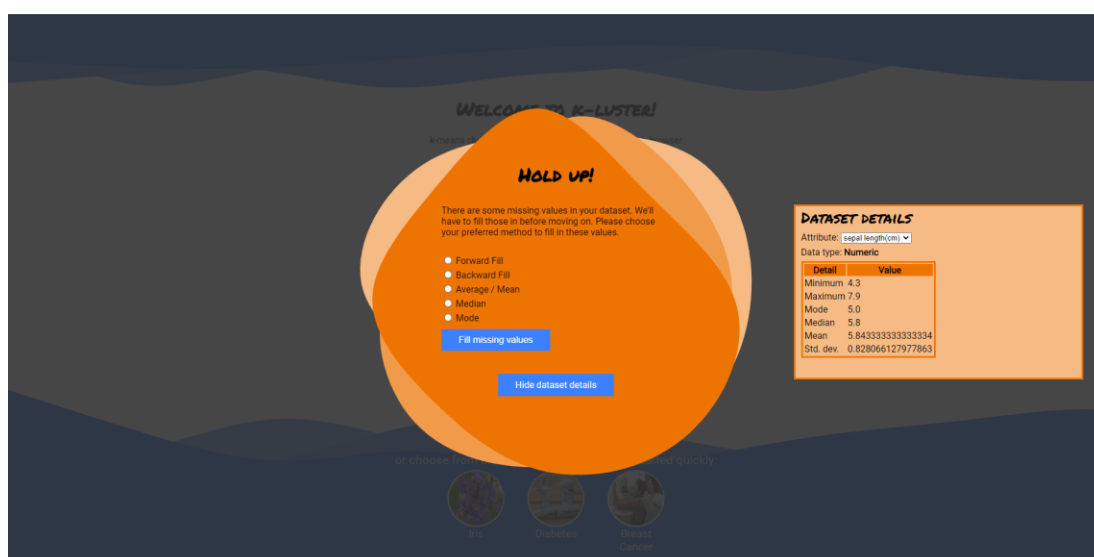


Figure 5.9: k-luster Missing Values Prompt

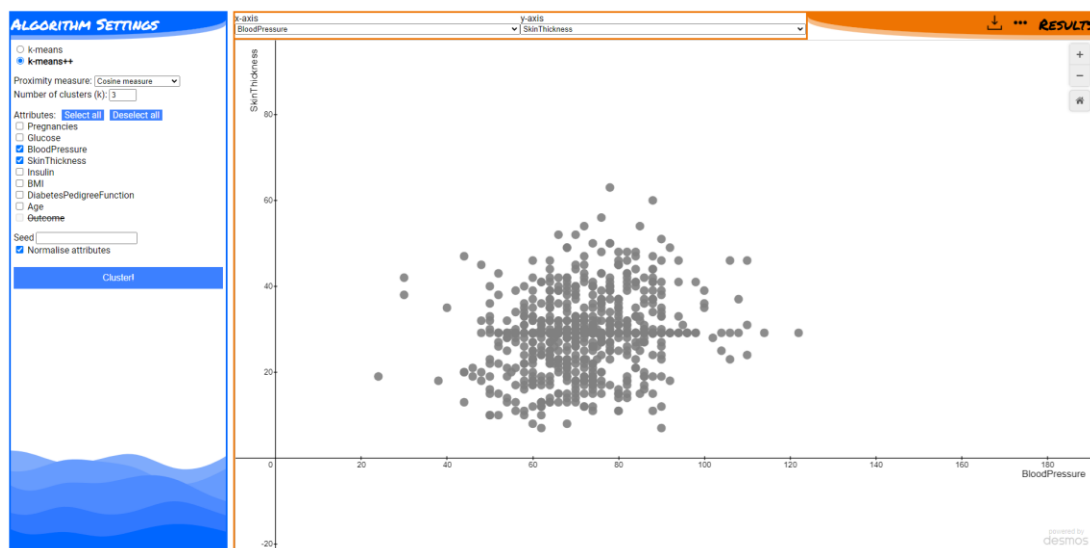


Figure 5.10: k-luster Cluster Page

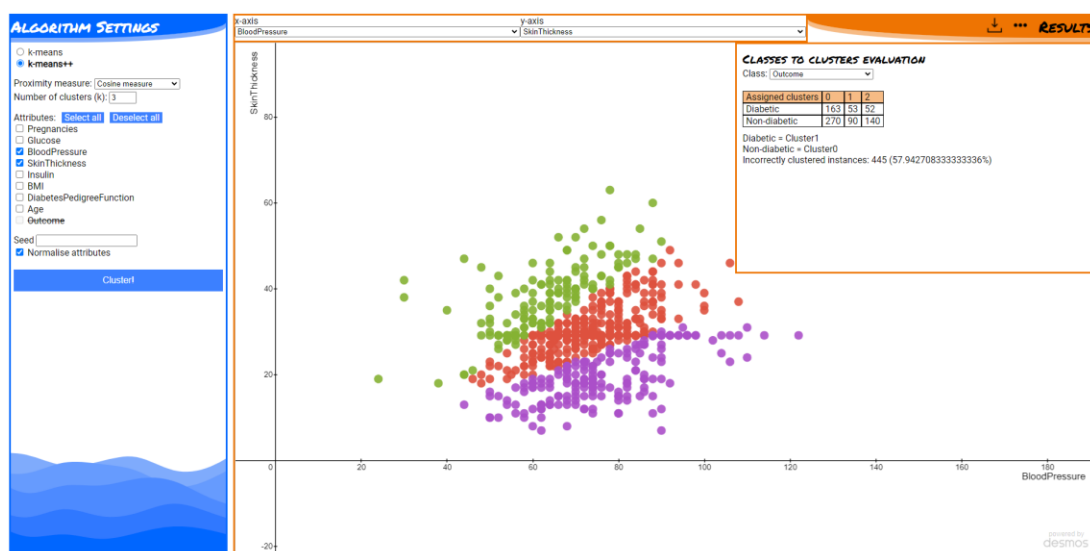


Figure 5.11: k-luster Cluster Page with Cluster Results

## CHAPTER 6

### SYSTEM IMPLEMENTATION

#### 6.1 Introduction

The Kanban development methodology was adopted for this project. It is an Agile development methodology that maps out all tasks that need to be carried out on a Kanban board. Every sub-section following this introduction describes an individual task carried out during the development phase. Thus, this chapter can be thought of as a progression of events leading up to the completion of the system.

#### 6.2 Migrating Prototype Code

Referring to the Gantt chart in chapter three, there exists a gap between the time the prototype was developed and the time development for the full system started. During that gap, a new major release of Django, the package providing web services, was made available. Therefore, the code from the prototype needed to be migrated over to a new project that utilises the latest version of Django at that time.

Referring to the Django documentation, the basics are still exactly the same. First, a new Django project was created with the command line. After that, a one-for-one copy of the essential files, such as the view templates, route file and data clustering scripts were made and moved to the new project folder.

#### 6.3 Initialise a Git Repository and Push Code to GitHub

A new Git repository was initialised with the new project folder. A Git repository provides system versioning and enables easy rollbacks in case a major bug occurs. In addition, the project code was also pushed to a remote GitHub repository as a means of archiving the project.



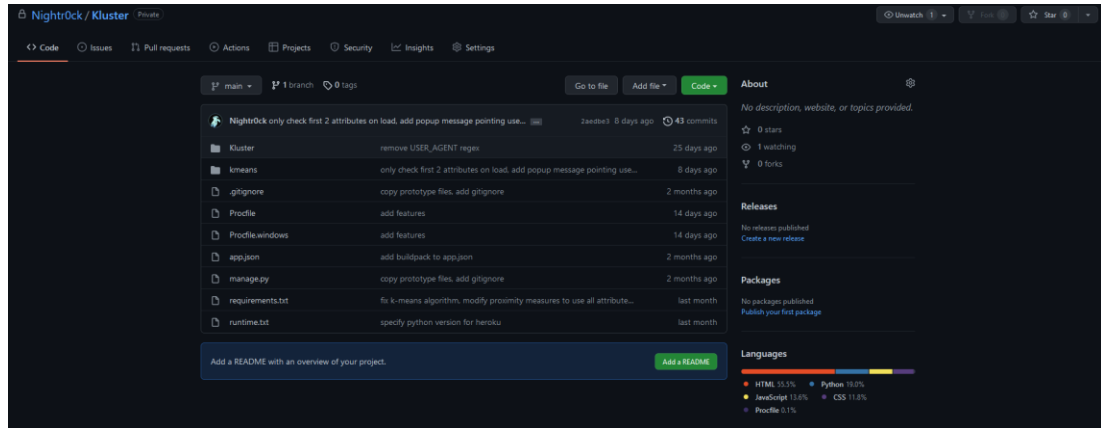


Figure 6.1: k-luster GitHub Repository

## 6.4 Implementing K-means++

In the prototype, only the K-means algorithm has been implemented. The K-means++ algorithm had to be added into the web application. First, a field was created on the client side to allow users to pick between the two algorithm variants.

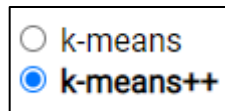


Figure 6.2: Algorithm Variant Selection Field

Then, the data clustering script was modified to include the K-means++ algorithm. The final code is as follows:

```

centroids = np.empty(0, dtype=np.intc) # numpy array containing coordinates for current centroids

if plusplus: # if k-means++ is used
    centroids = np.append(centroids, random.choice(range(0, dataset.shape[0]))) # choose first centroid randomly, temporary store its index in the dataset instead of the coordinates
    for i in np.arange(0, k-1): # for the rest of the centroids needed to be chosen
        distances_squared = np.empty(0) # distance between each point and the closest centroid
        total_distances_squared = 0.0 # sum of distances in the array directly above

        for index, data in dataset.iterrows():
            shortest_distance = np.empty(0) # calculate the distances between this point and all current chosen centroids and find the closest one

            for centroid in centroids:
                shortest_distance = np.append(shortest_distance, proximity[metric](data.to_numpy(), dataset.iloc[centroid].to_numpy()))

            shortest_distance = np.min(shortest_distance)**2
            distances_squared = np.append(distances_squared, shortest_distance)
            total_distances_squared += shortest_distance

        distances_squared = (distances_squared / total_distances_squared).tolist()
        centroids = np.append(centroids, random.choices(range(0, dataset.shape[0]), weights=distances_squared)[0])

    centroids = dataset.iloc[centroids].to_numpy() # get coordinates of all chosen centroids from dataset

else: # if k-means is used
    centroids = dataset.iloc[random.choices(range(0, dataset.shape[0]), k=k)].to_numpy() # get selected points to be initial cluster from dataset and assign to centroids as a numpy array

cluster_memberships = np.zeros(dataset.shape[0], dtype=np.intc) # numpy array containing cluster memberships of each point in dataset
change = True # indicates if there is a change in centroids
while change:
    change = False

    for index, data in dataset.iterrows(): # iterate through all rows
        distances = np.empty(0) # distances between this point and all centroids
        for centroid in centroids: # iterate through all centroids
            distances = np.append(distances, proximity[metric](data.to_numpy(), centroid)) # calculate the distances of current row to all centroids

        if cluster_memberships[index] != np.where(distances == np.min(distances))[0][0]:
            cluster_memberships[index] = np.where(distances == np.min(distances))[0][0]
            change = True

    if change:
        for i in np.arange(0, k):
            centroids[i] = dataset.iloc[np.where(cluster_memberships == i)[0].mean(axis=0)].to_numpy()

return cluster_memberships.tolist()

```

Figure 6.3: K-means and K-means++ Code

## 6.5 Implementing Additional Proximity Measures (I)

Up until now, only the Euclidean distance and Manhattan distance proximity measures had been implemented. Three additional distance measures were implemented into the web application, Chebyshev distance, Average distance and Canberra distance.

```

def chebyshev(x, y):
    distances = np.empty(0)

    for i in np.arange(0, x.size):
        distances = np.append(distances, abs(x[i] - y[i]))

    return np.max(distances)

```

Figure 6.4: Chebyshev Distance Code

```

def average(x, y):
    total_squared = 0.0
    n = 0

    for i in np.arange(0, x.size):
        total_squared += (x[i] - y[i])**2
        n += 1

    return math.sqrt(total_squared / n)

```

Figure 6.5: Average Distance Code

```

def canberra(x, y):
    total = 0.0

    for i in np.arange(0, x.size):
        if abs(x[i]) + abs(y[i]) != 0:
            total += (abs(x[i] - y[i])) / (abs(x[i]) + abs(y[i]))

    return total

```

Figure 6.6: Canberra Distance Code

## 6.6 Deploying Project to Heroku

At this point, concerns arose that the web application would not work as intended when deployed to Heroku. The main concern was with loading the data sets from the file system. The web application was developed and tested locally on a Windows machine, yet Heroku hosts applications on a UNIX-based machine. Furthermore, when hosted on Heroku, the web server would be started by Gunicorn with a separate config file instead of through Python directly. Thus, the decision was made to ensure that the web application would work when deployed to an entirely different environment before proceeding with the development the rest of the features.

Firstly, a new Heroku account was set up and a new app named *k-luster* was created under the account.

Personal > k-luster

Overview Resources **Deploy** Metrics Activity Access Settings

**Add this app to a pipeline**  
Create a new pipeline or choose an existing one and add this app to a stage in it.

**Add this app to a stage in a pipeline to enable additional features**

Pipelines let you connect multiple apps together and **promote code** between them. [Learn more.](#)

Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. [Learn more.](#)

Choose a pipeline

**Deployment method**

- Heroku Git: Use Heroku CLI
- GitHub: Connect to GitHub
- Container Registry: Use Heroku CLI

**Deploy using Heroku Git**  
Use git in the command line or a GUI tool to deploy this app.

**Install the Heroku CLI**  
Download and install the [Heroku CLI](#).  
If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

**Clone the repository**  
Use Git to clone k-luster's source code to your local machine.

```
$ heroku git:clone -a k-luster
$ cd k-luster
```

Deploy your changes

Figure 6.7: *k-luster* App on Heroku

After that, the Git URL to the newly created app is added as a new remote repository with Git. Then, a new text file called *requirements.txt* is also created. This text file lists all the dependencies of the application. Heroku will read the file and install all packages listed before attempting to build and run the application.

```
django
gunicorn
django-heroku
pandas
```

Figure 6.8: *Requirements.txt* Contents

Gunicorn is used to run the web server, whereas django-heroku is used to automatically load in the secret key stored in the config vars of the Heroku app.

Besides that, a new file named *Procfile* is created in the project's root directory. This file contains a list of commands that will be executed every time changes to the app are pushed, or the app is restarted.

```
release: python manage.py migrate
web: gunicorn Kluster.wsgi
```

Figure 6.9: Procfile Commands

The first command will create a new database file in the project containing all of Django's default tables. The second command starts the web server and listens for incoming HTTP requests.

The two new files were committed and the entire project was pushed onto Heroku. Within a few minutes, the project was successfully built and the web server was online. The Heroku app is accessible at <https://k-luster.herokuapp.com/>.

## 6.7 Attribute Exclusion

There are times where a user would want to exclude certain attributes from the clustering process in order to achieve a better result. In order to incorporate this feature, an additional field is created that allows users to uncheck attributes they wish to be excluded.

Attributes: Select all Deselect all

- Pregnancies
- Glucose
- BloodPressure
- SkinThickness
- Insulin
- BMI
- DiabetesPedigreeFunction
- Age
- Outcome

Figure 6.10 Attribute Exclusion Field

When sending a request for cluster results to the web server, the included attributes are sent in the request body. Then, on the web server, the excluded attributes are dropped from the data set prior to the clustering process.

```
excludedAttributes = dataset.columns.drop(attributes)
dataset = dataset.drop(columns=excludedAttributes)
```

Figure 6.11: Drop Excluded Attributes

## 6.8 Attribute Normalisation

Normalisation is a very important data pre-processing step. Thus, it is crucial to provide the user with the option to normalise all attributes before clustering.

Normalise attributes

Figure 6.12: Normalise Attributes Field

```
if normalise:
    for column in dataset.columns:
        if pd.api.types.is_numeric_dtype(dataset[column].dtype) and not pd.api.types.is_complex(dataset[column].dtype):
            dataset[column] = (dataset[column] - dataset[column].min()) / (dataset[column].max() - dataset[column].min())
```

Figure 6.13: Attribute Normalisation Code

## 6.9 Seed Specification

Any pseudo-random number generator requires a seed to work. By default, Python will use the current system time as the seed. However, this behaviour can be overwritten to produce consistent results.

Seed

Figure 6.14: Seed Specification Field

```
if seed:
    random.seed(a=seed)
else:
    random.seed(a=None)
```

Figure 6.15: Seed Specification Code

## 6.10 Improving Result Visualisation Using Desmos API

In the prototype, result visualisation was built by displaying an image of a plot returned by the web server. This image was rendered using Matplotlib. This approach brings about several disadvantages. First of all, the plot is not interactive. The user is not able to drag it around, zoom in or zoom out. Secondly, the larger the data set, the longer it will take for the web server to render the image, thus delaying the HTTP response from the web server. This also takes up additional resources. Lastly, the user is not able to freely switch between different attributes

Hence, result visualisation will be handled by the Desmos API instead of Matplotlib in the full system. This also shifts the load from the web server to the client side.

The Desmos API can be imported into the project with a single script tag on the client side.

```
<script src=
"https://www.desmos.com/api/v1.6/calculator.js?apiKey=dcb31709b452b1cf9dc26972add0fda6"
></script>
```

Figure 6.16: Importing the Desmos API

On the back-end, a JSON object containing the cluster results is returned instead of an image.

```
def cluster_result(request):
    if request.method == "POST":
        body = json.loads(request.body.decode(encoding="utf-8"))
        return JsonResponse(kmeans(request.session.session_key, body["plusplus"], body["metric"],
int(body["k"]), body["attributes"], body["seed"].strip(), body["normalise"]), safe=False)
    else:
        return HttpResponseNotAllowed(["POST"]) # code 405, method not allowed
```

Figure 6.17: Cluster Result View

Once the client receives the JSON object containing the results, the visualisation is updated dynamically without the need to reload the entire page.

```
const updateResultsVisualisation = (clustersChanged) => { // update the plot
    calculator.removeExpressions(expressions);
    calculator.updateSettings({
        xAxisLabel: document.getElementById("xAxisSelector").selectedOptions[0].text,
        yAxisLabel: document.getElementById("yAxisSelector").selectedOptions[0].text,
    });

    if (clustersChanged) {
        let generatedColors = paletteGenerator.generate(document.getElementById("k").value, (color) => {
            let hcl = color.hcl();
            return hcl[0]>=90 && hcl[0]<=360
                && hcl[1]>=60 && hcl[1]<=100
                && hcl[2]>=35 && hcl[2]<=80;
        }, false, 50, false, 'Default');
        generatedColors = paletteGenerator.diffSort(generatedColors, 'Default');

        expressions = Array.from((length: document.getElementById("k").value, (element, index) => {
            return {
                "id": `clusters${index}`,
                "color": `rgba(${generatedColors[index]["_rgb*"].map((value) => { return Math.round(value); }).join(",")})`,
            };
        }));

        let latex;
        if (dataPoints[0]["cluster"] == null) { // user hasn't clustered once yet
            latex = [dataPoints.reduce((previous, current) => {
                previous.push(`${current["values"][document.getElementById("xAxisSelector").selectedIndex]},${current["values"][document.getElementById("yAxisSelector").selectedIndex]}`);
                return previous;
            }, []).join(",")];
        } else { // user has already clustered
            latex = dataPoints.reduce((previous, current) => {
                previous[current["cluster"]].push(`${current["values"][document.getElementById("xAxisSelector").selectedIndex]},${current["values"][document.getElementById("yAxisSelector").selectedIndex]}`);
                return previous;
            }, Array.from((length: document.getElementById("k").value, () => { return []; })).map((points) => { return points.join(","); }));

        for (let i = 0; i < latex.length; i++) {
            calculator.setExpression({
                "id": expressions[i]["id"],
                "latex": latex[i],
                "color": expressions[i]["color"],
                "pointSize": 15
            });
        }
    };
};
```

Figure 6.18: Update Result Visualisation

Two dropdowns labelled x-axis and y-axis were also added to the top of the page. Changing the selections in the dropdowns will update the visualisation accordingly.

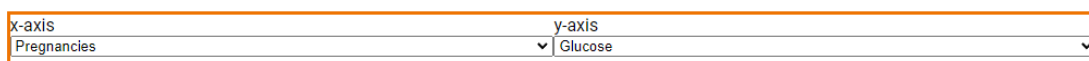


Figure 6.19: x-axis and y-axis Dropdowns

```
document.getElementById("xAxisSelector").addEventListener("change", () => { updateResultsVisualisation(false); });
document.getElementById("yAxisSelector").addEventListener("change", () => { updateResultsVisualisation(false); });
```

Figure 6.20: x-axis and y-axis Dropdown Event Listeners

### 6.11 Detection of Attribute Data Types

Both the K-means and K-means++ algorithms only work on numerical data, since they work by calculating the distance between different points. Similarly, the result visualisation implemented in the previous subsection is only able to plot numerical data. Therefore, there needs to be a check to distinguish between numerical and textual attributes.

```
for column, series in dataset.iteritems():
    if pd.api.types.is_numeric_dtype(series.dtype) and not pd.api.types.is_complex(series.dtype):
        pass
    else:
        pass
```

Figure 6.21: Check to Distinguish Numeric and Textual Attributes

On the client side, textual attributes are disallowed from being included in the clustering process. These attributes are not allowed to be selected for any of the axes as well.



Attributes: Select all Deselect all

- Pregnancies
- Glucose
- BloodPressure
- SkinThickness
- Insulin
- BMI
- DiabetesPedigreeFunction
- Age
- Outcome

Figure 6.22: Textual Attribute Disabled in Attribute Exclusion Field

x-axis

Pregnancies
▼

- Pregnancies
- Glucose
- BloodPressure
- SkinThickness
- Insulin
- BMI
- DiabetesPedigreeFunction
- Age
- Outcome

Figure 6.23: Textual Attribute Disabled in Axis Dropdown

## 6.12 Implementing Sessions

Session implementation is needed so that multiple users may use the web application concurrently. Without sessions, a user uploading a new data set will overwrite the data set uploaded by a separate user. Sessions were implemented using Django's sessions framework. The framework assigns each user a unique session key, and creates a new row in the `django_sessions` table with the session key as the primary key. The session data can be retrieved from the database as a session object, which is essentially just a dictionary of stored values.

When a user uploads a new data set, it is saved using the user's unique session key as the file name.

```
with open(f"kmeans/uploaded/{session_key}.csv", "wb") as dataset_file:
    dataset_file.write(uploaded)
```

Figure 6.24: Saving an Uploaded Data set

Then, when the web server receives a cluster request, the data set with a matching name as the user's session key is loaded.

```
session = SessionStore(session_key = session_key)
dataset = pd.read_csv(f"kmeans/uploaded/{session_key}.csv")
```

Figure 6.25: Loading an Uploaded Data set

### 6.13 Handling Missing Values in Uploaded Data sets

Sometimes, the data sets uploaded by users are incomplete and have missing values. This presents an issue as the implementation of the K-means and K-means++ algorithm assumes that there will be no missing values. The same goes for the result visualisation with the Desmos API. Therefore, missing values found in an uploaded data set needs to be handled before proceeding to the clustering page. Currently, there are a few methods available for users to pick from to fill in the missing values, namely, forward fill, backward fill, average or mean, median, and mode. The prompt for users to choose a fill in method also displays descriptive statistics of the data set by attribute.

```
dataset = pd.read_csv(f"kmeans/uploaded/{session_key}.csv")
if dataset.isna().any().any(): # null values detected
    metadata = {}
    for column, series in dataset.iteritems():
        if pd.api.types.is_numeric_dtype(series.dtype) and not pd.api.types.is_complex(series.dtype):
            metadata[column] = {}
            metadata[column]["isNumeric"] = True
            metadata[column]["details"] = {
                "Minimum": series.min(),
                "Maximum": series.max(),
                "Mode": ", ".join(series.mode().to_string(header=False, index=False).split("\n")),
                "Median": series.median(),
                "Mean": series.mean(),
                "Std. dev.": series.std()
            }
        else:
            metadata[column] = {}
            metadata[column]["isNumeric"] = False
            metadata[column]["details"] = series.value_counts().to_dict()
    return {"result": 1, "metadata": metadata}
else:
    return {"result": 0}
```

Figure 6.26: Check for Missing Values

```

def fill_nulls(method, session_key):
    dataset = pd.read_csv(f"kmeans/uploaded/{session_key}.csv")
    match method:
        case "forwardFill":
            dataset = dataset.fillna(method="ffill")
        case "backwardFill":
            dataset = dataset.fillna(method="bfill")
        case "mean":
            dataset = dataset.fillna(dataset.mean(numeric_only=True))
        case "median":
            dataset = dataset.fillna(dataset.median(numeric_only=True))
        case "mode":
            dataset = dataset.fillna(dataset.mode(numeric_only=True).iloc[0])

    dataset.to_csv(f"kmeans/uploaded/{session_key}.csv", index=False)
    return 0

```

Figure 6.27: Filling in Missing Values

## 6.14 Sample Data sets

A few popular data sets will be chosen and pre-processed. These data sets will serve as samples for users who do not wish to look for and upload their own data set, and just want a convenient way to try out the web application. Three data sets have been chosen for this purpose, the iris data set, the diabetes data set, and the breast cancer data set.

### 6.14.1 Iris Data set

This is a very common data set used to teach beginners the basics of data mining. This data set was obtained from Kaggle (MathNerd, 2018) and has five attributes, one of them being the class label. The data set was introduced by Ronald Fisher in 1936, and is now in the public domain (MathNerd, 2018). This means that the data set is no longer protected under copyright law, and can be used freely.

No pre-processing steps were taken for this data set as there were no missing values within the dataset.

```

import pandas as pd
dataset = pd.read_csv("kmeans/static/kmeans/sample-datasets/iris.csv")
[1] ✓ 0.4s

dataset.isna().any().any()
[2] ✓ 0.4s

... False

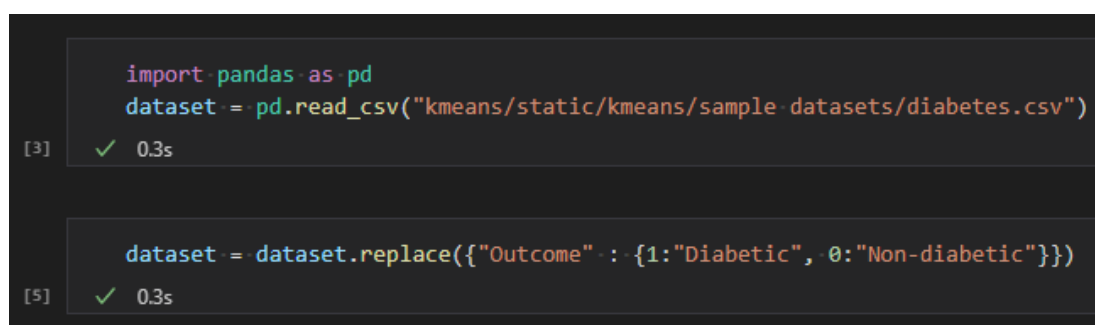
```

Figure 6.28: Check for Missing Values in the Iris Data set

### 6.14.2 Diabetes Data set

The diabetes data set is also a very popular data set, usually introduced as a toy data set to students. The data set was obtained from Kaggle (UCI Machine Learning, 2016b). The data was collected and released by the National Institute of Diabetes and Digestive and Kidney Diseases (UCI Machine Learning, 2016b). Similar to the iris data set, this data set is also under the public domain.

The only pre-processing step taken with this data set is replacing the values under the Outcome attribute. 1's were replaced with the word "Diabetic", whereas 0's were replaced with the word "Non-diabetic".



```
[3] ✓ 0.3s
import pandas as pd
dataset = pd.read_csv("kmeans/static/kmeans/sample_datasets/diabetes.csv")

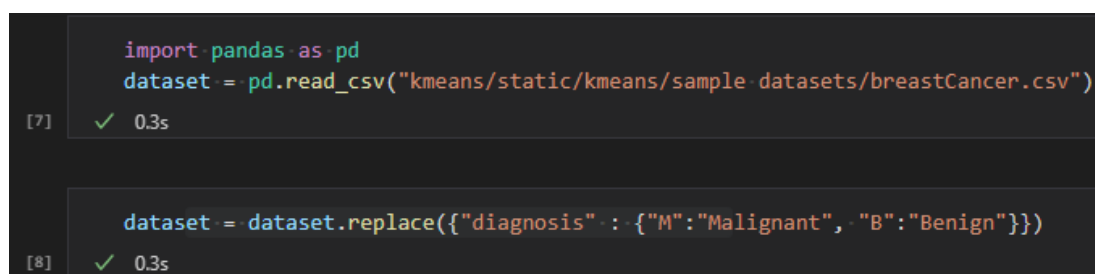
[5] ✓ 0.3s
dataset = dataset.replace({"Outcome": {1:"Diabetic", 0:"Non-diabetic"}})
```

Figure 6.29: Replacing Values Under Outcome in the Diabetes Data set

### 6.14.3 Breast Cancer Data set

This is the largest data set out of the three. This dataset is obtained from Kaggle (UCI Machine Learning, 2016a). The data set is under the Creative Commons NonCommercial-ShareAlike license. As the name suggests, the usage of this data set for non-commercial purposes is permitted. Redistribution and modification of the original data set is also allowed.

Similar to the diabetes data set, the values under the diagnosis attribute have been replaced. M has been replaced with "Malignant", whereas B has been replaced with "Benign".



```
[7] ✓ 0.3s
import pandas as pd
dataset = pd.read_csv("kmeans/static/kmeans/sample_datasets/breastCancer.csv")

[8] ✓ 0.3s
dataset = dataset.replace({"diagnosis": {"M":"Malignant", "B":"Benign"}})
```

Figure 6.30: Replacing Values Under Diagnosis in the Breast Cancer Data set

### 6.15 Download Results

Once the user has clustered the data at least once, they will be able to download the cluster results obtained. The downloaded file is the exact same csv file of the data set they uploaded or chose, but with an additional column called k-luster results containing the cluster results.

A	B	C	D	E	F	G
sepal leng	sepal wid	petal leng	petal wid	class	k-luster results	
5.1	3.5	1.4	0.2	Iris-setosa	1	
4.9	3	1.4	0.2	Iris-setosa	1	
4.7	3.2	1.3	0.2	Iris-setosa	1	
4.6	3.1	1.5	0.2	Iris-setosa	1	
5	3.6	1.4	0.2	Iris-setosa	1	
5.4	3.9	1.7	0.4	Iris-setosa	1	
4.6	3.4	1.4	0.3	Iris-setosa	1	
5	3.4	1.5	0.2	Iris-setosa	1	
4.4	2.9	1.4	0.2	Iris-setosa	1	
4.9	3.1	1.5	0.1	Iris-setosa	1	
5.4	3.7	1.5	0.2	Iris-setosa	1	
4.8	3.4	1.6	0.2	Iris-setosa	1	
4.8	3	1.4	0.1	Iris-setosa	1	
4.3	3	1.1	0.1	Iris-setosa	1	
5.8	4	1.2	0.2	Iris-setosa	1	

Figure 6.31: Downloaded Results

### 6.16 Classes to Clusters Evaluation

This feature evaluates the accuracy of the formed clusters using the class label.

CLASSES TO CLUSTERS EVALUATION			
Class:	class <input type="button" value="v"/>		
Assigned clusters	0	1	2
Iris-setosa	1	49	0
Iris-versicolor	34	0	16
Iris-virginica	13	0	37
Iris-setosa = Cluster1			
Iris-versicolor = Cluster0			
Iris-virginica = Cluster2			
Incorrectly clustered instances: 30 (20%)			

Figure 6.32: Classes to Clusters Evaluation

Each class label is assigned one of the formed clusters based on how many instances of that class belong to each of the formed clusters. The formed cluster with the highest instances of that particular class will be assigned to that class.

### 6.17 Implementing Additional Proximity Measures (II)

Additional proximity measures were implemented to achieve the project's objectives. The added proximity measures were chord distance, mean character difference, cosine measure, Czekanowski coefficient, and index of association.

```
def chord(x, y):
    sum_xy_diff = 0.0

    sum_x_squared = 0.0
    sum_y_squared = 0.0
    for i in np.arange(0, x.size):
        sum_x_squared += x[i] ** 2
        sum_y_squared += y[i] ** 2
    sum_x_squared = math.sqrt(sum_x_squared)
    sum_y_squared = math.sqrt(sum_y_squared)

    for i in np.arange(0, x.size):
        sum_xy_diff += ((x[i] / sum_x_squared) - (y[i] / sum_y_squared)) ** 2

    return math.sqrt(sum_xy_diff)
```

Figure 6.33: Chord Distance Code

```
def mean_character_difference(x, y):
    sum_difference = 0.0

    for i in np.arange(0, x.size):
        sum_difference += abs(x[i] - y[i])

    return sum_difference / x.size
```

Figure 6.34: Mean Character Difference Code

```

def cosine(x, y):
    product_xy = 0.0
    sum_x_squared = 0.0
    sum_y_squared = 0.0

    for i in np.arange(0, x.size):
        product_xy += x[i] * y[i]
        sum_x_squared += x[i] ** 2
        sum_y_squared += y[i] ** 2

    return 1 - (product_xy / (math.sqrt(sum_x_squared) * math.sqrt(sum_y_squared)))

```

Figure 6.35: Cosine Measure Code

```

def czekanowski(x, y):
    sum_min_xy = 0.0
    sum_xy = 0.0

    for i in np.arange(0, x.size):
        sum_min_xy += min(x[i], y[i])
        sum_xy += x[i] + y[i]

    return 1 - (2 * sum_min_xy / sum_xy)

```

Figure 6.36: Czekanowski Coefficient Code

```

def index_of_association(x, y):
    sum_xy_diff = 0.0

    sum_x = x.sum()
    sum_y = y.sum()
    for i in np.arange(0, x.size):
        sum_xy_diff += abs((x[i] / sum_x) - (y[i] / sum_y))

    return sum_xy_diff / x.size

```

Figure 6.37: Index of Association Code

## CHAPTER 7

### SYSTEM TESTING

#### 7.1 SUS Testing

System usability scale (SUS) testing is a method of assessing the usability of a system. According to Lewis (2018), it was developed by John Brooke in 1984 for a usability engineering program. In 1986, John Brooke then made it freely available to anyone. Then, in 1996, John Brooke published a chapter in the book titled Usability Evaluation In Industry describing the method. (Lewis, 2018) Since then, the SUS has been used to evaluate several different hardware and software systems such as handphones, computer programmes and websites (Sauro, 2011). While it is often referred to as a “quick and dirty” method, it is capable of producing reliable results with only a small sample size (Sauro, 2011).

##### 7.1.1 Test Procedure

First, the test participant is introduced to the test subject. In the context of the software development, this would usually be a type of application, either on the web or for mobile devices, or some other form of computer system. The test participant would then be allowed to explore the application given. Guidance may be provided to the participant to introduce them to all the major functionalities of the system. Once the participant has sufficiently explored the system, they are given a questionnaire consisting of ten questions (Grier, et al., 2013). For each question, the participant is required to provide a response on a five point Likert scale. The participant is not allowed to leave any questions blank. The ten questions are as follows:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.



7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

The SUS test concludes once the participant has responded to all ten questions.

### **7.1.2 Calculating SUS Scores**

This process converts the participants' responses into a score ranging from 0 to 100 (Lewis, 2018). First, subtract 1 from all the scores given for odd-numbered questions. Next, for even-numbered questions, subtract the given score from 5. Then, sum up all the converted scores and multiply the result by 2.5 (Lewis, 2018). Repeat the same steps for all participants and calculate the average of all the converted scores. The average score is the final SUS test score.

### **7.1.3 Interpreting SUS Scores**

The final SUS score obtained is just a number and does not mean much on its own. A method of interpreting the score is needed to determine if the tested system is well-designed or not.

#### **7.1.3.1 Percentiles**

This is the most common way of evaluating SUS test scores.

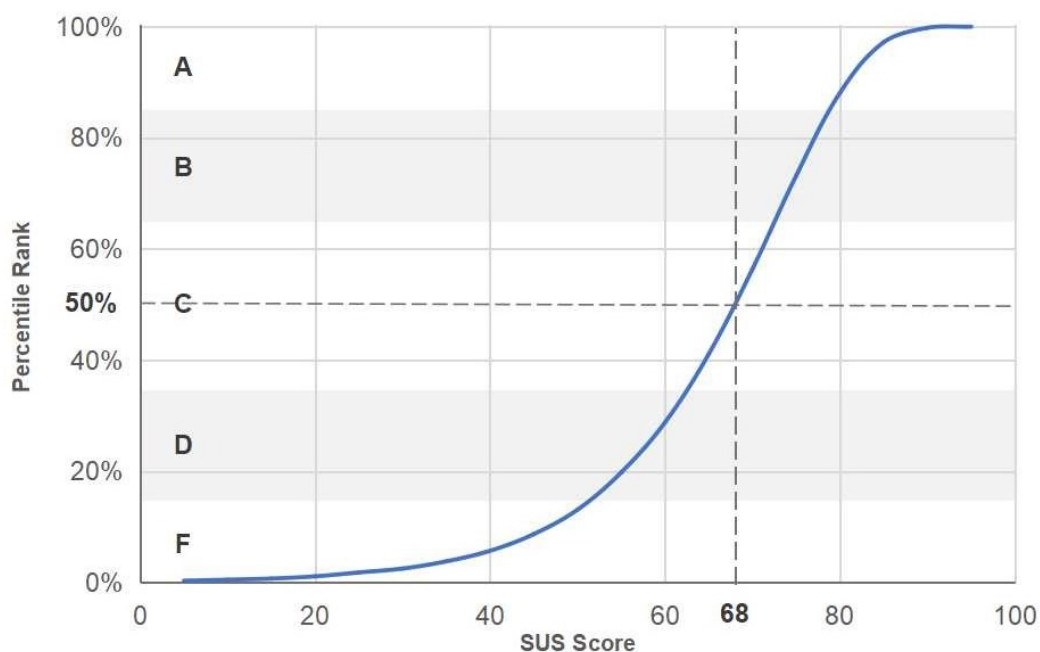


Figure 7.1: Plot of Percentile Against SUS Score (Sauro, 2018)

Figure 7.1 shows that 50 percent of SUS scores obtained for software systems are above 68. Thus, as long as the final obtained score is higher than 68, the tested system is considered to be above average in terms of usability. 68 is also often referred to as the passing mark for SUS tests (Sauro, 2018).

### 7.1.3.2 Adjective Ratings

Bangor, Kortum and Miller (2009) conducted a study in an attempt to add adjective rating scale to the SUS test. The scale shown in table 7.1 was proposed and added to the bottom of the SUS questionnaire.

Table 7.1 Adjective Rating Scale (Bangor, Kortum and Miller, 2009)

Worst Imaginable	Awful	Poor	OK	Good	Excellent	Best Imaginable
---------------------	-------	------	----	------	-----------	--------------------

In addition to the ten usual questions asked in an SUS test, the participants were also asked to select a rating that best describes the usability of the system being tested. Bangor, Kortum and Miller (2009) then calculated the average of the SUS scores associated with each rating.

Table 7.2 Average SUS Scores for each Adjective Rating (Bangor, Kortum and Miller, 2009)

Adjective Rating	Average SUS Score
Worst Imaginable	12.5
Awful	20.3
Poor	35.7
OK	50.9
Good	71.4
Excellent	85.5
Best Imaginable	90.9

#### 7.1.4 Conducting the SUS Test

Five lecturers and one postgraduate student from UTAR were invited to participate in the SUS test for the web application. They were first sent a link leading to the web application and asked to play around with it. Guidance was provided for participants whom were not familiar with data mining. Once the participants were ready, they started filling out the questionnaire.

##### 7.1.4.1 Test Responses

Table 7.3 Summary of Responses Received

Questions \ Participants	Questions									
	1	2	3	4	5	6	7	8	9	10
Participant 1	5	2	4	1	4	2	5	1	5	2
Participant 2	5	2	4	1	4	2	4	1	5	2
Participant 3	3	2	3	4	3	2	4	3	3	3
Participant 4	3	2	4	2	4	3	4	2	3	3
Participant 5	3	2	4	2	3	2	2	3	4	2
Participant 6	4	2	4	3	4	2	4	2	4	2

Table 7.3 summarises all the responses received from the test participants.

Table 7.4 Converted SUS Scores

<b>Participant</b>	<b>SUS Score</b>
Participant 1	87.5
Participant 2	85
Participant 3	55
Participant 4	65
Participant 5	62.5
Participant 6	72.5

Calculating the average of all SUS scores shown in table 7.4, a final SUS score of 71.25 is achieved. Referring to the percentile rank of historical SUS scores, an SUS score of 71.25 is slightly above average. Thus, a passing score has been achieved.

According to table 7.2, a final SUS score of 71.25 is the closest to the “Good” rating.

#### **7.1.4.2 Other Comments**

During the test, the participants were also asked for additional comments and suggestions on improvements to the system. This subsection addresses the feedback obtained the participants.

##### **7.1.4.2.1 Default Algorithm Settings Causes Overlapping of Clusters**

By default, on the clustering page, all selectable attributes are pre-selected. If the user decides to cluster without changing the selection, there will be a lot of overlaps between the clusters.

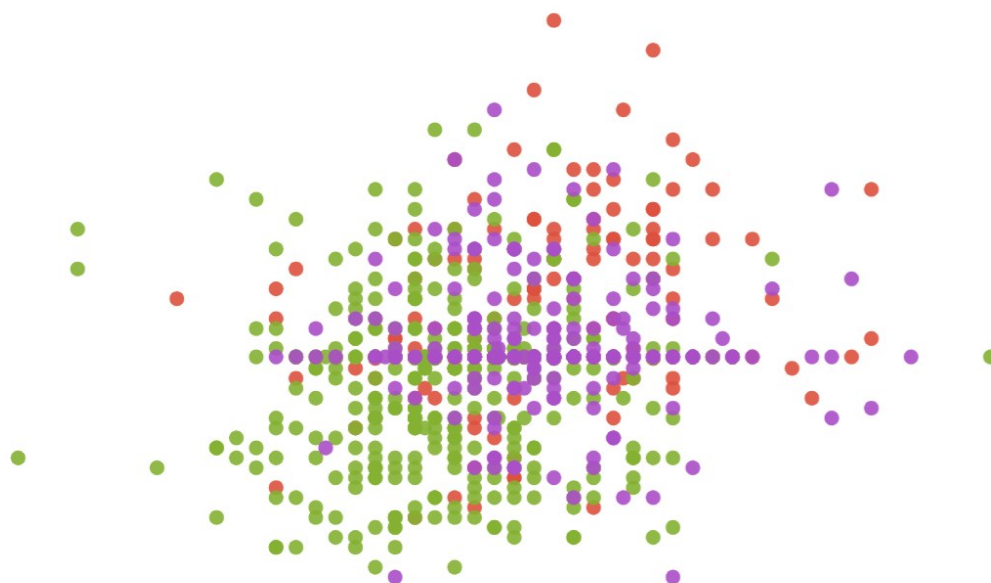


Figure 7.2: Overlapping of Clusters

This is because the visualisation is only able to plot the data in two-dimensions (x-axis and y-axis), yet the algorithm is taking all numeric attributes in the data set to be used in the clustering process. If only the attributes being plotted by are selected for clustering, there would be any overlaps between the clusters, as shown in figure 7.2

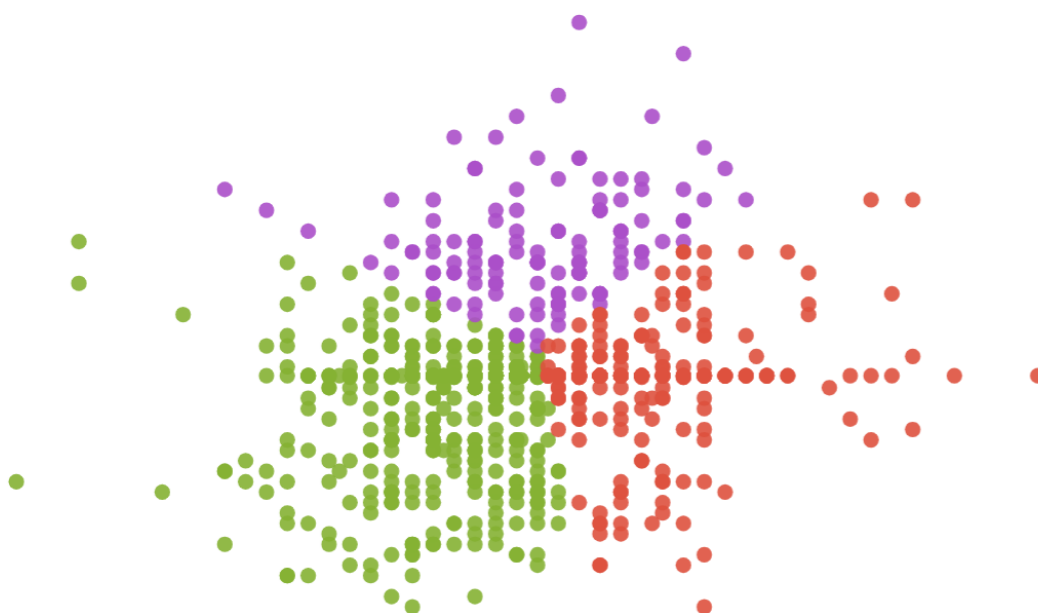


Figure 7.3: No Overlapping between Clusters

This behaviour may be unintuitive to users who are not very familiar with clustering or the k-means algorithm. Thus, the web application has been changed to only pre-select the first two numeric attributes in the data set, which is the same way the attributes for both the x and y axes are selected. This will ensure that the selected attributes match the plotted attributes when the cluster page is first loaded.

#### 7.1.4.2.2 Hard to Notice Classes to Clusters Button

The button to toggle the display of the classes to clusters panel is at the top right corner, in between the download button and the results text. It is denoted with an ellipsis icon.



Figure 7.4: From Left to Right, Download Button, Classes to Clusters Button, Results Text

Since most of the screen is taken up by the data visualiser, some of the participants had difficulty noticing the classes to clusters button. Hence, an additional popup message pointing to the button has been added. This popup only displays the first time the cluster button is clicked for the current session. Reloading or leaving the page will reset the session.

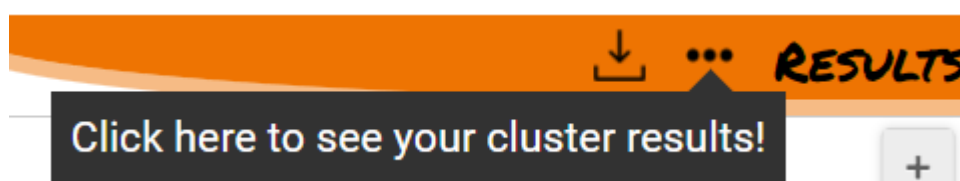


Figure 7.5: Popup Message Pointing at Classes to Clusters Button

#### 7.1.4.2.3 Back Button on Clustering Page

A few participants pointed out the lack of a back button on the clustering page to return to the home page where users may choose or upload a different data set. The intended way for users to navigate back to the home page is to use the browser's built-in back button. The page transition between the two pages is also designed to

accommodate for the back/forward cache (bfcache) of browsers so that users would not be greeted with a blank page when using them. However, seeing how a few test participants have pointed out this shortcoming, it would be best to add a back button in the future.

#### **7.1.4.2.4 Missing Values Handling**

Several participants pointed out that the currently implemented methods for filling in missing values are insufficient. Two of the most requested for methods that are not implemented are to outright ignore the rows with missing values, and to allow the user to specify a certain value to be used. These two methods should be implemented in the near future.

#### **7.1.4.2.5 Map Cluster Colours to Cluster Numbers**

Looking at the output in the classes to clusters evaluation popup, it is not possible to know which colour belongs to which cluster number. Only a single participant pointed out this issue. While it is only a minor detail, it is still an oversight. Thus, to rectify this problem, colour markers matching those shown on the result visualiser can be added to the classes to clusters evaluation popup so that the user would know which colour belongs to which cluster.

## CHAPTER 8

### CONCLUSIONS AND RECOMMENDATIONS

#### 8.1 Conclusion

The project has met all the objectives it set out to achieve. The result is *k-luster*, a web application that allows users to upload or choose a sample dataset and cluster the data using the K-means or K-means++ algorithm. Furthermore, it implements ten proximity measures which the user may pick from to use with the clustering algorithms. Comparing the available proximity measures in *k-luster* against those in other prominent data mining tools, over half the proximity measures implemented in *k-luster* are not seen in any other tools that were researched on. The web application has been hosted on Heroku, so that anyone may access the site through their browser at any time on their devices. An SUS test was also conducted with five lecturers and one postgraduate student from UTAR to evaluate the usability of the system. The final score achieved was 71.25. Admittedly, this is not a great score considering that the passing score is 68, just a mere few points away. However, the testing process provided invaluable insight as to how the web application may be further improved in the future.

#### 8.2 Recommendations for future work

Although all the goals have been met, there are still plenty of ways the web application could be improved. First of all, all feedback pointed out by the SUS test participants should be addressed. The issues brought up are all only require minor changes, yet are capable of greatly improving user satisfaction.

Besides that, the functionality of the web application can be further increased. For example, other clustering algorithms such as k-medoids or DBSCAN can be implemented into the web application. Other than that, support for classification tasks could also be added in the future.

Next, the web application can save a few of the users' recent data sets along with their most recently used algorithm settings for each data set. This will allow users to resume their work right away when coming back to the web application the next day or after restarting their computer.



## REFERENCES

- Arthur, D. and Vassilvitskii, S., 2013. k-means++: The Advantages of Careful Seeding. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035. [Accessed 21 April 2022]
- Bangor, A., Kortum, P. and Miller, J., 2009. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *Journal of Usability Studies*, 4(3), pp. 114-123. [Accessed 19 April 2022]
- Cha, S., 2007. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4), pp.300-307. [Accessed 9 August 2021]
- Corona, E. and Pani, F., 2013. *A Review of Lean-Kanban Approaches in the Software Development*. University of Cagliari. Available at: <<http://www.wseas.us/journal/pdf/information/2013/5709-110.pdf>> [Accessed 9 July 2021]
- Faisal, M., Zamzami, E. M. and Sutarman, 2020. Comparative Analysis of Inter-Centroid K-Means Performance using Euclidean Distance, Canberra Distance and Manhattan Distance. *Journal of Physics Conference Series*.
- Frank, E., Hall, M. A. and Witten, I. H., 2016. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Available at: <[https://www.cs.waikato.ac.nz/ml/weka/Witten\\_et\\_al\\_2016\\_appendix.pdf](https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf)> [Accessed 12 August 2021]
- Grier, R. A. et al., 2013. The System Usability Scale: Beyond Standard Usability Testing. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 57(1), pp. 187–191. [Accessed 18 April 2022]
- Han, J., Kamber, M. and Pei, J., 2011. *Data Mining Concepts and Techniques*. 3rd ed. Waltham, Massachusetts: Morgan Kaufmann. [Accessed 27 June 2021]
- Hartigan, J. A. and Wong, M. A., 1979. Algorithm AS 136: A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1), pp.100-108 [Accessed 10 August 2021]
- Jazayeri, M., 2007. Some Trends in Web Application Development. *Future of Software Engineering (FOSE '07)*, pp.199-213. [Accessed 13 August 2021]

Kirovska, N. and Koceski, S., 2015. Usage of Kanban methodology at software development teams. *Journal of Applied Economics and Business*, [online] Available at: <<http://www.aebjournal.org/articles/0303/030302.pdf>> [Accessed 9 July 2021]

Lewis, J. R., 2018. The System Usability Scale: Past, Present, and Future. *International Journal of Human-Computer Interaction*, pp. 1-14. [Accessed 18 April 2022]

Liu, H., Li, J., Wu, Y., and Fu, Y., 2019. Clustering with Outlier Removal. *IEEE Transactions on Knowledge and Data Engineering*, 33(6), pp.2369-2379. [Accessed 10 August 2021]

MathNerd, 2018. *Iris Flower Dataset*. [online] Available at: <<https://www.kaggle.com/datasets/arshid/iris-flower-dataset>> [Accessed 20 April 2022]

MDN Web Docs, n.d. *JavaScript*. [online] Available at: <<https://developer.mozilla.org/en-US/docs/Web/JavaScript>>

Morissette, L. and Chartier, S., 2013. The k-means clustering technique: General considerations and implementation in Mathematica. *Tutorials in Quantitative Methods for Psychology*, 9(1), pp.15-24. [Accessed 9 August 2021]

Oracle, n.d. *2 Introduction to Oracle Data Mining*. [online] Available at: <<https://docs.oracle.com/database/121/DMCON/GUID-0B1D8B18-218B-46C6-92A1-2A499F961D49.htm#DMCON001>> [Accessed 12 August 2021]

Oracle, n.d. *Database PL/SQL Packages and Types References*. [online] Available at: <[https://docs.oracle.com/database/121/ARPLS/d\\_datmin.htm#ARPLS618](https://docs.oracle.com/database/121/ARPLS/d_datmin.htm#ARPLS618)> [Accessed 24 June 2021]

R, n.d. *What is R?* [online] Available at: <<https://www.r-project.org/about.html>> [Accessed 12 August 2021]

RDocumentation, n.d. *amap (version 0.8-18) Kmeans: K-Means Clustering*. [online] Available at: <<https://www.rdocumentation.org/packages/amap/versions/0.8-18/topics/Kmeans>> [Accessed 20 July 2021]

Sauro, J., 2011. *Measuring Usability with the System Usability Scale (SUS)*. [online] Available at: <<https://measuringu.com/sus/>> [Accessed 18 April 2022]

Sauro, J., 2018. *5 Ways to Interpret a SUS Score*. [online] Available at <<https://measuringu.com/interpret-sus-score/>> [Accessed 19 April 2022]

scikit-learn, n.d. *2.3. Clustering*. [online] Available at: <<https://scikit-learn.org/stable/modules/clustering.html>> [Accessed 12 August 2021]

Shirkhorshidi, A. S., Aghabozorgi, S. and Wah, T. Y., 2015. A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data. *PLoS ONE*, 10(12). [Accessed 20 July 2021]

Thakare, Y. S. and Bagal S. B., 2015. Performance Evaluation of K-means Clustering Algorithm with Various Distance Metrics. *International Journal of Computer Applications*, 110(11), pp.12-16. [Accessed 20 July 2021]

UCI Machine Learning, 2016a. *Breast Cancer Wisconsin (Diagnostic) Data Set*. [online] Available at: <<https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data>> [Accessed 20 April 2022]

UCI Machine Learning, 2016b. *Pima Indians Diabetes Database*. [online] Available at: <<https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database>> [Accessed 20 April 2022]


Visual Studio Code, n.d. *Visual Studio Code FAQ*. [online] Available at: <<https://code.visualstudio.com/docs/supporting/faq>> [Accessed 20 July 2021]

## APPENDICES

### APPENDIX A: SUS Responses

#### k-luster System Usability Testing

**Test participant details:**

<b>Name</b>	Hoo Meei Hao
<b>E-Mail</b>	hoomh@utar.edu.my
<b>Signature</b>	

**Test started on:**

<b>Date</b>	11/4/2022
<b>Time</b>	9.35am

Read through each of the following statements and place a checkmark (✓) next to the score that most accurately depicts how you feel

1.)

<b>I think that I would like to use this system frequently.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
				/

2.)

<b>I found the system unnecessarily complex.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	/			

3.)

<b>I thought the system was easy to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			/	

4.)

<b>I think that I would need the support of a technical person to be able to use this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
/				

5.)

<b>I found the various functions in this system were well integrated.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			/	

6.)

<b>I thought there was too much inconsistency in this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	/			

7.)

<b>I would imagine that most people would learn to use this system very quickly.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
				/

8.)

<b>I found the system very cumbersome to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
/				

9.)

<b>I felt very confident using the system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
				/

10.)

<b>I needed to learn a lot of things before I could get going with this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	/			

**Additional Comments (if any):**

Missing button to go back to the first page where to load the data.

Some feedbacks in selecting attributes if there are changes of attributes in the graphs to ensure the results show is correct. Because may forget to press Cluster button if changes of attributes selection is made.


**Test ended on:**

<b>Date</b>	<b>11/4/2022</b>
<b>Time</b>	<b>9.50am</b>

**Thank you so much for volunteering to participate in this test**

## k-luster System Usability Testing

**Test participant details:**

<b>Name</b>	Too Chian Wen
<b>E-Mail</b>	toocw@utar.edu.my
<b>Signature</b>	

**Test started on:**

<b>Date</b>	13 April 2022
<b>Time</b>	12pm



Read through each of the following statements and place a checkmark (✓) next to the score that most accurately depicts how you feel

1.)

<b>I think that I would like to use this system frequently.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
				✓

2.)

<b>I found the system unnecessarily complex.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

3.)

<b>I thought the system was easy to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

4.)

<b>I think that I would need the support of a technical person to be able to use this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
✓				

5.)

<b>I found the various functions in this system were well integrated.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

6.)

<b>I thought there was too much inconsistency in this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

7.)

<b>I would imagine that most people would learn to use this system very quickly.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

8.)

<b>I found the system very cumbersome to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
✓				

9.)

<b>I felt very confident using the system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
				✓

10.)

<b>I needed to learn a lot of things before I could get going with this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

**Additional Comments (if any):**

-NIL-
-------

**Test ended on:**

<b>Date</b>	<b>13 April 2022</b>
<b>Time</b>	<b>12.35pm</b>

**Thank you so much for volunteering to participate in this test**

## k-luster System Usability Testing

**Test participant details:**

<b>Name</b>	Lai Yen Lung
<b>E-Mail</b>	laiyl@utar.edu.my
<b>Signature</b>	

**Test started on:**

<b>Date</b>	15/4/2022
<b>Time</b>	2:30pm

Read through each of the following statements and place a checkmark (✓) next to the score that most accurately depicts how you feel

1.)

<b>I think that I would like to use this system frequently.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

2.)

<b>I found the system unnecessarily complex.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

3.)

<b>I thought the system was easy to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

4.)

<b>I think that I would need the support of a technical person to be able to use this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

5.)

<b>I found the various functions in this system were well integrated.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

6.)

<b>I thought there was too much inconsistency in this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

7.)

<b>I would imagine that most people would learn to use this system very quickly.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

8.)

<b>I found the system very cumbersome to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

9.)

<b>I felt very confident using the system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

10.)

<b>I needed to learn a lot of things before I could get going with this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

**Additional Comments (if any):**

The developed system included the stimulation for K-means clustering (and K-means++) algorithm which could benefit to the user either in their daily clustering task and learning clustering mechanism. Overall, I found such system is designed to be user friendly and easy to be use on hand for any non-expert user.

**Test ended on:**

<b>Date</b>	15/4/2022
<b>Time</b>	3.00pm

**Thank you so much for volunteering to participate in this test**

## k-luster System Usability Testing

**Test participant details:**

<b>Name</b>	Mohammad Babrdel Bonab
<b>E-Mail</b>	babrdel@utar.edu.my
<b>Signature</b>	<i>Mohammad Babrdel Bonab</i>

**Test started on:**

<b>Date</b>	15 April 2022
<b>Time</b>	8:15pm



Read through each of the following statements and place a checkmark (✓) next to the score that most accurately depicts how you feel

1.)

<b>I think that I would like to use this system frequently.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

2.)

<b>I found the system unnecessarily complex.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

3.)

<b>I thought the system was easy to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

4.)

<b>I think that I would need the support of a technical person to be able to use this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

5.)

<b>I found the various functions in this system were well integrated.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

6.)

<b>I thought there was too much inconsistency in this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

7.)

<b>I would imagine that most people would learn to use this system very quickly.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

8.)

<b>I found the system very cumbersome to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

9.)

<b>I felt very confident using the system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

10.)

<b>I needed to learn a lot of things before I could get going with this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

**Additional Comments (if any):**

--

**Test ended on:**

<b>Date</b>	<b>15 April 2022</b>
<b>Time</b>	<b>8:30pm</b>

**Thank you so much for volunteering to participate in this test**

## k-luster System Usability Testing

**Test participant details:**

<b>Name</b>	Wong Chim Chwee
<b>E-Mail</b>	wongcc@utar.edu.my
<b>Signature</b>	<i>wong</i>

**Test started on:**

<b>Date</b>	14/4/2022
<b>Time</b>	6:00pm

Read through each of the following statements and place a checkmark (✓) next to the score that most accurately depicts how you feel

1.)

<b>I think that I would like to use this system frequently.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

2.)

<b>I found the system unnecessarily complex.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

3.)

<b>I thought the system was easy to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

4.)

<b>I think that I would need the support of a technical person to be able to use this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

5.)

<b>I found the various functions in this system were well integrated.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

6.)

<b>I thought there was too much inconsistency in this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

7.)

<b>I would imagine that most people would learn to use this system very quickly.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

8.)

<b>I found the system very cumbersome to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

9.)

<b>I felt very confident using the system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

10.)

<b>I needed to learn a lot of things before I could get going with this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

**Additional Comments (if any):**

--


**Test ended on:**

<b>Date</b>	<b>14/4/2022</b>
<b>Time</b>	<b>6:45pm</b>

**Thank you so much for volunteering to participate in this test**

## k-luster System Usability Testing

**Test participant details:**

<b>Name</b>	Ho Yan Bing
<b>E-Mail</b>	hyb951104@gmail.com
<b>Signature</b>	

**Test started on:**

<b>Date</b>	15/04/2022
<b>Time</b>	8.28 AM



Read through each of the following statements and place a checkmark (✓) next to the score that most accurately depicts how you feel

1.)

<b>I think that I would like to use this system frequently.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

2.)

<b>I found the system unnecessarily complex.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

3.)

<b>I thought the system was easy to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

4.)

<b>I think that I would need the support of a technical person to be able to use this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
		✓		

5.)

<b>I found the various functions in this system were well integrated.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

6.)

<b>I thought there was too much inconsistency in this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

7.)

<b>I would imagine that most people would learn to use this system very quickly.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

8.)

<b>I found the system very cumbersome to use.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

9.)

<b>I felt very confident using the system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
			✓	

10.)

<b>I needed to learn a lot of things before I could get going with this system.</b>				
1 (Strongly Disagree)	2	3	4	5 (Strongly Agree)
	✓			

**Additional Comments (if any):**

--

**Test ended on:**

<b>Date</b>	15/04/2022
<b>Time</b>	8.54 AM

**Thank you so much for volunteering to participate in this test**