S	
E1	
E2	
For Office Use Only	

Masters Project Final Report

(MCS)

2019

Project Title	Automated News Clustering Using an Unsupervised Learning Model
Student Name	W. P. I. Fonseka
Registration No. & Index No.	2017/MCS/030 17440305
Supervisor's Name	Dr. Ruvan Weerasinghe

For Office Use ONLY

Automated News Clustering Using an Unsupervised Learning Model

A dissertation submitted for the Degree of Master of Computer Science

W. P. I. Fonseka
University of Colombo School of Computing
2019

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.
To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: W. P. I. Fonseka
Registration Number: 2017/MCS/030
Index Number: 17440305

_____

Signature: Date: 20/06/2020

This is to certify that this thesis is based on the work of
Mr. W. P. I. Fonseka
under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:
Supervisor Name: Dr. Ruvan Weerasinghe

_____

Signature: Date:

Abstract

Text clustering is an emerging research topic which involves two main subject areas; Natural Language Processing and Machine Learning. Clustering comes under unsupervised machine learning which is more complex in terms of both implementation and evaluation than its supervised variants. Especially when it comes to a dynamic domain like online news content, it is really difficult because of the unpredictability of cluster labels or number of clusters. With the absence of this prior knowledge, the required knowledge should be acquired from the data set itself. So, the focus of this research is to come up with a proper approach which addresses all these challenges and cluster online news items with higher degree of accuracy.

The main challenge was plenty of noise due to lengthier content of news items. So, focus was given to minimize this noise and extract key features to improve the accuracy and performance of the clustering module. Text preprocessing and feature extraction techniques were experimented empirically under different parameter combinations, to find the optimal method and set of parameters which suits a noisy and lengthier text corpus like ours. Singular Valued Decomposition was used to reduce the dimensionality of the feature matrix, which improved the results generated by the clustering module drastically. K-means Elbow method together with Silhouette Score curves were used for finding the optimal number of clusters. The optimized model was evaluated using data with ground truth information and the quality of the generated clusters was verified using extrinsic methods like Jaccard Score and Adjusted Random Index. Data set for the experiment was generated by extracting sports news from CNN, BBC and Aljazeera news sites programmatically.

When the model was tested using the entire data set without any ground truth information, generated clusters, for the identified optimal number of clusters (k), had a Silhouette Score over 0.4 and Sum of Squared Errors less than 60. Cluster quality measures were not up to expected values because of some overlapping clusters, though most of the clusters exhibited a meaning according to the majority of items assigned to them. The model generated quality results when tested with a sample data set with non-overlapping clusters. Silhouette score was around 0.9 and SSE was reduced to value less than 10 for this sample. Generated clusters for this sample were compared with the ground truth information using extrinsic methods and the calculated value for the Adjusted Random Index was 0.9694 which is an indication of an acceptable end result.

Acknowledgement

First and foremost, my utmost gratitude should go to my supervisor Dr. Ruvan Weerasinghe, senior lecturer of University of Colombo School of Computing, for all the support and guidance throughout this research. He was the key resource person who directed me in the correct path with usual follow ups and support whenever I came across any difficulty during this study.

I also like to express my gratitude to Dr. Randil Pushpananda, senior lecturer of University of Colombo School of Computing, for systematically coordinating this module, which allowed the students to complete this module without any overhead. All my MSc lecturers at UCSC are remembered with sincere gratitude for all the knowledge I gained which was a solid foundation for conducting a research like this.

I can’t forget my family members, my mother, wife and son for all the encouragements and looking after everything, allowing me to focus on my studies.

Table of Contents

List of Figures

List of Tables

Chapter 1 - Introduction

1.1 Introduction

With the rapid growth rates of information technology, there is an explosion of text data via the world wide web. Sometimes with this overwhelming amount of information, it is hard to filter out our exact needs. So, there is a great need of organizing and indexing these data into a manner where we can traverse and search them easily. Doing this manually is impossible. So, we have to find methods to automate this process.

When we consider text in online news feeds, one of the major concerns is the dynamic behaviour of data. Since many new and different news items will be reported day by day, we cannot have a predefined set of categories. There will be many new categories introduced over time. Also, it is not feasible to find a labelled training data set to uniformly represent all the future dynamic news content. So, it is impossible to come up with a supervised data classification method. Because of that, we have to come up with an unsupervised text mining method like text clustering.

When clustering text content like dynamic news feeds, there are two major concerns to be addressed

1. We don’t know cluster labels beforehand
2. Number of clusters are unknown

Another concern is, when we have to deal with fine grained clusters, sometimes we have to consider finer levels of granularity in our clustering algorithms. It is always a challenge to identify up to what extent of granularity we should consider: document level, paragraph level, sentence level or word level, to identify a set of meaningful clusters based on some criteria.

Ambiguous nature of natural languages makes text clustering a complex task. Data scientists conducting experiments on the area of natural language processing find it really difficult, dealing with syntactic and semantic ambiguities of almost all the natural languages in use.

There can be very similar content in two text items, but describe two different incidents, hence should be assigned into two different groups and contextually different news items, but describe the same incident hence should be assigned to the same group. So, when we go for an unsupervised learning model there can be a lot of noise which directly affects the accuracy of our algorithm. This makes identifying accurate cluster boundaries with a set of clusters which make sense, really hard. Identified clusters should have minimum intra cluster distances and maximum inter cluster distances. In other words, identified objects within a cluster should be compact as much as possible while different clusters should be apart from one another as much as possible.

So, the challenge is to develop an algorithm which addresses all these concerns and provide results with a higher degree of accuracy.

1.2 Text Clustering

Clustering is a data mining technique which comes under machine learning. It is an unsupervised machine learning technique which tries to find groups of similar objects, given a set of data objects, based on a similarity function [6]. The similarity between objects is measured using distance functions which determines how similar the objects are based on some predefined properties of those objects. Euclidean distance, Manhattan distance, Minkowski distance and Cosine similarity are some of the commonly used distance functions to measure similarity.

Clustering can be very useful in the text domain. Text clustering involves a combination of machine learning and Natural Language Processing (NLP) to understand and categorize textual data.

Text documents should be pre-processed before feeding into clustering algorithms to improve the performance and accuracy of those algorithms. Lot of pre-processing tasks like stop word removal, tokenization and feature extraction will be performed on input text documents. Then statistical parameters of documents like word frequencies and tf-idf will be calculated, and used to convert text documents into their vector representations. These vectors will be given as input to the clustering algorithms. These techniques will be discussed in detail in future chapters. Different text clustering algorithms like agglomerative clustering algorithms,

partitioning algorithms, and standard parametric modelling-based methods such as the EM-algorithm are available to be used based on the application [6].

Some of the common applications of text clustering are:

- **Document Organization and Browsing:** Clustering can be used to assign a collection of documents into unified clusters and organize them into a hierarchical model [6]. For an example if we are concerned about clustering news items, the top level will be news. The next level can consist of clusters such as sports news, political news, business news etc. Within the sports news cluster there can be sub clusters such as cricket, football, rugger. Within the cricket category there can be sub clusters test, one day, 20-20. Likewise, we can have a hierarchy of clusters which will enable systematic browsing of documents.

- **Corpus Summarization:** Methods such as sentence clustering can be used to generate insights for the overall content of the underlying corpus, hence can be used to generate summaries for a corpus which is related to dimensionality reduction and topic modelling [6].

- **Document Classification:** Clustering can be used to improve the quality of text classification - the supervised variant of text clustering. Word-clusters and co-training methods can be used to improve the accuracy of text classification [6].

1.3 Motivation

Printed media like newspapers and newsmagazines, and broadcast methods like radio and television, were mainly used methods to deliver news to the general public in earlier days. But with the introduction of the internet and web, many online news sources were introduced and people started to move away from older methods of news retrieval, and gather around online news sources. High availability, easy accessibility and instant reporting of news within a few minutes after the incident are the main reasons for the popularity of online news sources. With this new trend, most of the news reporters are more focussed on their online versions, hence there are many online news websites available and many new ones are being introduced day by day. So, with the availability of many news websites, sometimes it can be hard to filter out

our exact needs and this may degrade the usability of this highly valuable online information repository. If we can develop a system which extracts news from available popular news websites, group them and index them in a proper manner and make them available in one place, that will be a need of this century.

When we traverse through literature, many of the news indexing methods have followed supervised learning approaches and have considered categorizing news into a broader predefined set of categories which is not very suitable for the dynamic and unpredictable nature of news contents. Very little focus was given to unsupervised news clustering which identifies a set of dynamic clusters which changes over time as our proposed approach. Also, the number of clusters to be generated should be provided as an input for most of the available clustering algorithms. For this we should know the number of clusters to be generated beforehand which is not really practical with dynamic news clustering because there can be many new clusters that can be introduced with time.

So, there is an emerging need for developing an efficient and accurate unsupervised text clustering algorithm which addresses the dynamic behaviour of news items, which will greatly benefit online news seekers. Also, such a system will not only be limited to the news domain, and can be generalized into any kind of online text domains.

1.4 Objectives

The ultimate objective of this study is to come up with an unsupervised learning model which identifies clusters of news from different news channels and provides results with higher degree of accuracy. Our model should address the concerns below

- Cluster labels are not predefined
- Number of clusters are not predefined

Also, the proposed model should perform well with the presence of noise due to synonymy and ambiguity when clustering text.

1.5 Scope

Our model should be able to identify a set of clear and meaningful news clusters, from different news channels. For this, Twitter news feeds from three major news channels CNN, BBC and Al Jazeera, for the sports category will be used. These feeds are extracted over a specific period of time. The time period will be defined in a way that the size of the extracted data set is sufficient to evaluate the functionality of our algorithm.

Since we are extracting news for the same category sport, from three different channels over the same time period, this data set will include news related to the same sports or events but reported in different ways from different news channels. Our system should be able to identify those news and group them together. This way each identified news cluster should contain news items which exhibit similarities according to a clearly identifiable criterion.

The data set used above is not annotated according to a predefined criterion. So, our algorithm should behave in a totally unsupervised manner to identify the optimal number of clusters (k) and cluster the data set into k meaningful clusters.

Chapter 2 - Literature Review

2.1 Overview

Natural Language Processing is a branch of artificial intelligence which deals with processing, analyzing, understanding and generating languages used by humans, in written and spoken context, especially when they are in digital form and used to interact with computers and machines [2]. In particular NLP deals with programming computers to process and analyze natural language data [1].

Early Natural Language Processing models were based on complex set of hand written rules [1]. The revolution started with the introduction of machine learning algorithms to language processing in the late 1980s [1]. Instead of the algorithms based on decision trees, which produced systems with hard if-then rules, research was focused more into statistical models [1]. These statistical models assigned a real value weights by analyzing the features of the input data and made probabilistic decisions based on those weights [1]. These models generated results with higher accuracy, especially when given unfamiliar and random inputs [1].

Statistical models were supervised learning models which were trained using substantial amount of annotated data [1]. But in many cases finding such an annotated set of data is not feasible. Recent research related to NLP is more focused on semi-supervised and unsupervised learning models which are based on representation learning and artificial neural networks [1]. With these new advances self-learning models were developed which adapts with the input data fed into the system [1].

Text mining is one of the emerging research topics. Researchers have been actively involved in finding better solutions for automatic text categorization [3]. It is now being applied in many contexts, ranging from document indexing based on a controlled vocabulary, to document filtering, automated metadata generation, word sense disambiguation, population of hierarchical catalogues of web resources and more complex tasks like human sentiment analysis [3].

Two main approaches are used; classification which comes under supervised learning and clustering which comes under unsupervised learning. Hybrid approaches of these two methods can be used, which is known as semi-supervised learning. Classification models should be trained using a substantial amount of labelled training data set which is used to train the model. When absence of such data, which is our case, we have to go with unsupervised models like clustering.

The basic idea of a supervised learning model is to define the required categories first, and use a training data set with assigned class labels to train a model by changing the parameters in a way that inputs are assigned to the correct categories according to the pre assigned class labels. Weights are assigned to inputs according to their features such as term frequency (tf) term frequency inverse document frequency (tf-idf) to measure how similar they are to predefined categories [4]. The model is trained by adjusting these parameters until training inputs are assigned to most relevant categories with an expected level of accuracy.

With the absence of properly annotated training dataset, we have to go with unsupervised methods like clustering. Clustering is the task of finding groups of similar data items in a collection of data items [5]. A suitable category for a given item is decided by using a similarity function. Some of the commonly used text clustering algorithms are Hierarchical Clustering, k-means Clustering, Single-pass Clustering, Probabilistic Clustering and Topic Model and Spectral Clustering [5]. There are off the shelf implementations of common clustering algorithms like Lemur, BOW which can be used for the purpose of document clustering [5].

2.2 Text Pre-processing

Text documents cannot be fed into clustering algorithms as it is in the raw format. They have to be pre-processed and converted into vector representations before feeding them into clustering algorithms. Operations like tokenization, stop word removal, stemming will be

performed in the pre-processing stage [8] [9]. Characteristics like tf-idf are used as numerical weights when converting text documents into corresponding vector representations [7] [8].

2.2.1 Text Feature Selection

Feature selection is one of the important tasks in text clustering. Important features should be extracted while filtering out uninformative terms [7] [8]. The aim is to remove too frequent words, which are also referred to as stop words, such as “a”, “and”, “the”, “or”, “of” which don't have any contribution for discriminating documents in text clustering [6]. Available lists of stop words can be found to be used in the process of stop word removal [6]. Not only the frequent words, but also words which occur extremely infrequently will not contribute to most of the commonly used similarity measurements among text documents [6]. In some cases, these infrequent words are due to misspellings or typographical errors in documents [6].

Feature selection is commonly applied in text classification but seldom applied in text clustering due to unavailability of class labels [6]. However unsupervised feature selection methods like term frequency inverse document frequency (tf-idf), term strength, entropy-based ranking and term contribution methods are available for the use of text clustering [6].

Only the selected feature words extracted from documents will be fed into clustering algorithms, to enhance the accuracy and performance of those algorithms.

Recalling our objective, the challenge is to come up with an unsupervised model to cluster dynamic news feeds from different news channels. The two major concerns to be addressed in this approach are, we don’t know the cluster labels and we don’t know the number of clusters prior to clustering.

Automatic news clustering has been a hot research topic over the recent years. Many different approaches can be found when going through literature, trying to find optimized solutions for this problem. Some of these approaches are described below.

2.3 WordNet-based Text Document Clustering

In [10], the authors have proposed an enhancement for k-means clustering for a corpus of news wire articles. In this approach they mainly address two challenges of text clustering;

ambiguity: where there can be two documents with very similar content but should belong to different clusters and synonymity: where there can be contextually different content but should be assigned to the same cluster. Since our goal is to find fine grained clusters in a given news category, we surely have to face these challenges, and address them properly. The basic idea is to enhance the bag of words generated by text pre-processing, by using an external database called WordNet. These enhanced feature sets can then be fed into clustering algorithms which will generate results with higher accuracy levels. Here they have used k-means clustering with cosine similarity measures.

In [11], the authors have done a comparative study on different WordNet databases; Syns, Hyper 5 and Hyper All. By experimental results they have concluded that adding background knowledge from Hyper 5, outperforms the accuracy levels of other methods.

2.4 Clustering News Articles Using Efficient Similarity Measure and N-grams

In another approach where news articles were represented using N-grams and the vector space model of documents were used to dimensionality reduction of the feature vector [12]. Here they have used word-based N-grams like bi-grams and trigrams. Weighted values like Term Frequency (TF) and Inverse Document Frequency (IDF) for the identified N-grams have been used to normalize frequency of those N-grams. These identified feature vectors are input to the clustering algorithms. An improved version of cosine similarity called sqrt-cosine similarity measurement was used to measure similarity between two news items [12].

In most of the above approaches k -means has been used as the clustering algorithm, where we have to define the number of clusters prior to clustering. Since this number is dynamic in our approach, we have to find alternatives to overcome this limitation. Spectral clustering can be used as a good alternative which does not need the number of clusters as the input and can find this value during processing [13]. It shows high accuracy when data's vector model could be presented as bipartite graph [13].

Dimensionality reduction methods which were originally developed for computer vision-based applications, can also be used for document clustering. One major disadvantage of using this method is that they should be initialized randomly which will lead to different results over multiple runs of the algorithm on the same data. But these methods have some

advantages like high performance and some of them can be used to estimate the optimal number of clusters.

Some approaches which tried to address the problem of not knowing cluster labels and number of clusters beforehand are described below.

2.5 News Clustering Based on Similarity Analysis

In [13], authors have proposed an ontology-based method to calculate similarity between news items to improve the accuracy of generated clusters. With this approach items assigned to a cluster will represent one theme or a point of view of that theme. They have used WordNet of English to measure word and sentence similarities by analyzing the semantic similarities and lexical relations by using synonyms.

They have considered few characteristics of mass media news data from social networks when coming up with a suitable clustering algorithm. They are a single news item will have short content (few sentences), number of news items will be very large, number of clusters is unknown and can vary with time and pre-processing stage will output a similarity matrix which represents a bipartite graph [13]. Considering these characteristics, the authors have proposed spectral divide & merge clustering algorithm which shows high accuracy with the presence of these characteristics [13]. This will be a good consideration in our hybrid approach since we are also to address most of these characteristics.

2.6 Automatic Thread Detection in Dynamic Text Message Streams

Dou Shen et al has proposed, using single-pass clustering to assign short text messages to their relevant message threads [14]. Single-pass clustering is a better option for clustering dynamic data like news feeds where cluster labels and number of clusters are not known beforehand, which is our case. Here they have proposed three variants of single-pass clustering algorithms to improve the performance of basic single-pass algorithm. They have taken into consideration, exploiting the temporal behaviour of the content in text messages. One other limitation is text messages contain very short content, usually one to two short

sentences. Addressing this limitation is essential for our approach since we are planning to consider similarity up to sentence level granularity.

The three variants of the single-pass algorithm they proposed are SP with Weighted Centroids: where more recent messages are given more weight when calculating the weighted centroids, SP using Nearest Neighbour: where it uses maximal cosine similarity value between the message and messages already assigned to the thread and SP using weighted nearest neighbour: where both cosine similarity value and time distance is taken into account when calculating the similarity between two messages. They have also proposed an improved version by incorporating linguistic features between adjacent messages to above mentioned single-pass variants. Their experimental results show that SP using Nearest neighbour incorporated with linguistic feature weights have outperformed the other proposed variants [14].

2.7 Agglomerative Cluster Analysis

Agglomerative clustering is a hierarchical clustering method which uses a bottom up approach. In this approach each document starts in its own cluster. Clusters will get merged successively according to a distance function and a linkage criterion [22]. When the linkage matrix is generated, we can decide the granularity of clusters by defining a max distance and consider clusters with linkage similarity values below that max distance. For example, if we plot a dendrogram for our linkage matrix as depicted in figure 2.1 and consider clusters linked under similarity distances below 1, there will be 3 clusters (2, 5, 7), (0, 1, 6), (3,4). If we take the max distance as 3, there will be only 2 clusters (2, 5, 7) and (0, 1, 3, 4, 6).

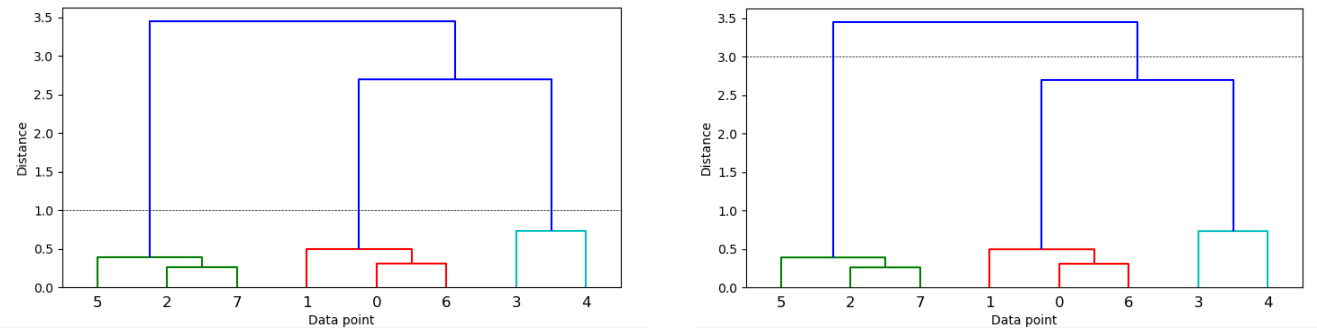

Figure 2.1: Agglomerative clustering with different linkage upper distances

This way we can use this method to generate clusters with different max distances and evaluate the validity of those clusters using intrinsic measures like Silhouette coefficient when obtaining the optimal number of clusters with required level of granularity.

From our literature review it is clear that very few researches have been conducted on clustering dynamic text content with absence of ground truth information like number of clusters or cluster labels. We also came across different methods for finding optimal number of clusters and methods to optimize text feature extraction process in different approaches. So, the aim is to study on these approaches further and come up with an optimized model which suits our text domain to provide accurate results.

Chapter 3 - Methodology

3.1 Problem Analysis

The major challenge addressed by this research is to develop an unsupervised text clustering algorithm to cluster news feeds from different online news sites. There are some important characteristics of online news content which makes this task more challenging.

- Huge number of news items available
- Dynamic nature of news content, hence the number of clusters and cluster labels are not predefined
- A news item will have lengthy content
- High dimensionality of text data
- Ambiguity of natural language
- Noise due to synonymy and ambiguity of news items

With the existence of such challenges the aim is to identify a set of clusters with news items, which exhibits a proper clustering criterion that makes sense. Each cluster should have higher internal coherence, and different clusters should be decoupled from each other substantially.

This approach involves two broad concepts of artificial intelligence- Natural Language Processing (NLP) and Machine Learning. Effective natural language processing techniques should be used to pre-process raw text corpora and extract only the relevant lexical features of the content before performing further analysis. Accuracy and efficiency of the feature extraction process will directly impact the performance and accuracy of our clustering algorithm, hence more focus should be given to the feature extraction methods to be used in our approach. These extracted features should be formed into their vector forms, and feature vectors should be generated for each document in our input data set to suit the clustering algorithms used in our approach. Suitable statistical measures of the features of documents should be taken into account when generating these feature vectors, which will output accurate and meaningful clusters when provided as inputs to our proposed clustering

approach. For example, one can consider the existence or nonexistence of feature words to construct a binary feature vector of a document and another can consider statistical measures like term frequency inverse document frequency when constructing document feature vectors. Each of these forms have pros and cons when applied to different clustering algorithms, hence should select the most suitable one for our approach.

Machine learning plays a vital role in the second stage of our approach, which performs clustering of news items. As described in the previous chapters we have to adapt unsupervised methods to address the dynamic behaviour of news items, since we cannot have a predefined set of categories. The biggest challenge is to identify the optimal number of clusters which satisfies the expected level of granularity and accuracy. Because of the ambiguous and unpredictable behaviour of news items there will be a lot of noise which should be handled properly and mitigated.

3.2. Methodology

A research methodology can be considered as a framework within which a research study is conducted [17]. In other words, it is a systematic way accepted by the research community which has led to many successful researches in the past. It is essential to follow such methods in doing research in any discipline. The choice of research method will mostly be determined by the purpose of the research [17].

3.2.1. Scientific Method

Science is considered as a subject which represents reliable knowledge. It is considered as a strong method of reasoning used by many disciplines [16]. Scientific reasoning is a highly practical based discipline which generates conclusions by using empirical results. If some theory is scientifically proven, people have a high tendency to accept that as a reliable theory. Induction is the method used in science to derive conclusions about a proposed model.

3.2.1.1 Induction

Scientific models are constructed with the aid of a large amount of empirical evidence [16]. This principle where arriving at a conclusion based on a large number of observations is called induction [16]. Deriving conclusion using induction is also referred to as inductive reasoning [16]. In inductive approach the focus will be given to exploring a new phenomenon or looking at a previously researched phenomena from a different perspective [18]. This approach usually requires extensive and repeated sifting through the data and analysing and re-analysing multiple times before arriving on a conclusion [18].

3.2.1.2 Steps in Scientific Method

As described above, scientific method is considered as the most accepted way of discovering reliable knowledge. It mainly has 9 steps.

- Observation
- Preliminary study
- Problem definition
- Theoretical framework
- Hypothesis development
- Experimental design
- Data gathering
- Data analysis
- Conclusion

3.2.2 Approach

A methodology similar to scientific method is followed in this study. There are plenty of online news sources available. It would be a great benefit for the society, if we can develop a model to extract news from different online sources, cluster them into finer grained groups and index them properly, so that people will have one large news repository where they can search and travers news easily and faster.

Motivated by this fact a systematic literature review was conducted and identified that unsupervised text clustering is a suitable approach for this because of the dynamic behaviour of news data. Also, it was identified that very few researches have been conducted on dynamic text clustering. The literature revealed that existing models do not address the need of unsupervised text clustering into a finer level of granularity with an expected level of accuracy.

So, the hypothesis was developed that, by combining some existing text feature extraction methods and unsupervised text clustering methods we could achieve our goal of clustering dynamic text content with a higher degree of accuracy.

In addition to basic text pre-processing tasks like stop word removal, tokenizing and stemming; text feature extraction can be improved by incorporating wordnets and ontology-based methods. Effective text feature extraction can benefit text clustering algorithms by improving their performance and accuracy by a great extent. K-Means clustering associated with Elbow Method and Silhouette Score Analysis can be used when determining the optimal number of clusters when number of clusters and cluster labels are not known, and have to determine them dynamically. When it comes to the news domain the feature matrix is very sparse, which degrades the accuracy and performance of the proposed model. Dimensionality reduction methods like Singular Value Decomposition works well with sparse feature matrices like ours.

Data will be extracted from several twitter sports channels for preparing the input data set. Experiments will be done on this data set by carrying out text pre-processing and clustering, by applying above mentioned methods and combination of those methods. Experiments will be continued by changing the parameters and threshold values used in the proposed model until the required level of accuracy is achieved. Accuracy of clusters will be measured using cluster evaluation metrics such as Purity and Inverse Purity, Clusters and Class Entropy, VI measure, Jaccard Coefficient and Mutual Information; which will provide numerical outputs [19]. So basically, the experiment will be carried out using Inductive Reasoning based on quantitative data analysis. Evaluation can be done using a known and labelled data sample and checking the validity of generated clusters by comparing them with the ground truth values.

Based on the experimental results obtained by performing above described methods and evaluating them thoroughly we can conclude the algorithms and parameters which optimises the performance and the accuracy of our proposed model and suggest the hybrid model which clusters dynamic text contents with a higher degree of accuracy.

3.3. High Level Design

Described below is the high-level design of the proposed approach. Figure 3.1 illustrates the main modules and their inter communication to perform required functionality.

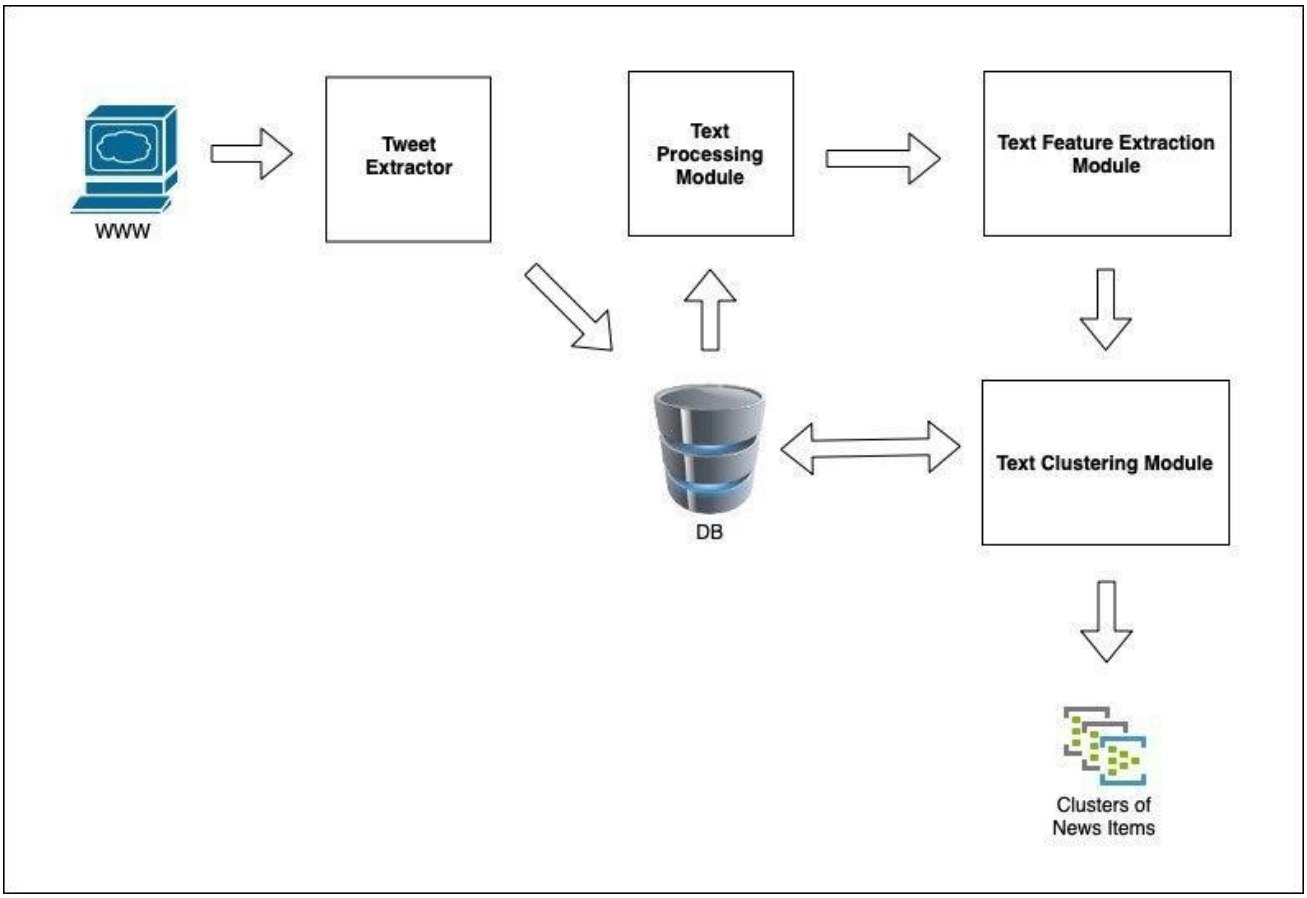

Figure 3.1: High level design of the proposed model

3.3.1 Tweet Extractor

This module extracts tweets from different sports channels of our interest, by using their twitter handlers. This module will continuously extract the content for a defined period of time. The relevant attributes of tweets will be extracted and saved into a relational database to

be used by the other modules of the system. Also, this module extracts actual news content by using the content URLs of tweets. Actual news titles and news bodies from actual news sites (i.e. CNN, BBC and Aljazeera) will be extracted and saved to the database.

3.3.2 Text Pre-processing Module

This module will include methods to perform necessary text pre-processing steps such as stop word removal, tokenization, stemming, lemmatization etc. The module will access the database to retrieve actual news bodies saved from the Tweet Extractor, and generates a set of pre-processed documents.

3.3.3 Text Feature Extraction Module

Given the set of pre-processed documents generated by the Text Pre-processing module, this module will extract the relevant features and generate feature vectors to be fed into clustering algorithms.

3.3.4 Text Clustering Module

Feature matrix of the text corpus, generated by analysing statistical weights of extracted features of documents, will be provided as input to this module. It will process these data using clustering algorithms and produce the required clusters as outputs. This module also contains the classes to perform necessary cluster analysis and rerun the clustering by adjusting the parameters and threshold values, until the generated clusters have accepted level of accuracy.

3.4 Implementation

Any proposed model has no use if we cannot implement it into a workable system. A prototype of the model was implemented for the purpose of the experiment and evaluation of results.

3.4.1 Tools and Technologies

Python is the programming language used for developing the prototype of the proposed model. Python has many off the shelf libraries with many built in algorithms used for the purpose of data mining. From data gathering to pre-processing to clustering, there are many stable and freely available libraries which perform the functionalities required in our study. Set of libraries used for text pre-processing, clustering and formatting in this research are listed below.

- **Natural Language Toolkit (NLTK)** - This helps the entire Natural Language Processing methodology from word tokenizing, stemming, recognizing POS tags, removing stop words and many more functionalities. This was the main toolkit used for text pre-processing in our approach.

- **Numerical Python (NumPy) Library** - NumPy is the core library for scientific computing in Python. One of the main functions it provides is a high-performance multidimensional array object that is a powerful data structure for efficient computation of arrays and matrices. It stores more mathematical and statistical information than a normal python array or a list, that provides faster access and many additional mathematical operations. We used numpy to store feature vectors of our text corpus and perform mathematical calculations on them.

- **Scikit-learn Library** - Scikit-learn is an open source machine learning library that supports most of the supervised and unsupervised learning algorithms. This is really helpful to scientists who conduct experiments on machine learning algorithms. Also, this supports python's core numerical and scientific libraries NumPy and SciPy. This library was used to test different clustering algorithms and generate results in this experiment. Scikit-learn also provides implementation for many cluster validation algorithms and some relevant algorithms were used to evaluate clusters generated by our model.

- **Pandas Library** - Pandas is a high-level data manipulation tool built on Python's NumPy package. Its main data structure is the DataFrames which allows us to store

and manipulate tabular data in rows of observations and columns of variables. In this experiment Pandas was used to visualize feature matrices, cluster linkage matrices and cluster labels.

- **Matplotlib Library** - Matplotlib is a library for creating static, animated, and interactive visualizations in Python. Matplotlib's Pyplot module provides facilities to generate plots using python data structures including NumPy arrays. These facilities are used to generate plots for different experimental results, to visualize and analyze them.

- **BeautifulSoup Library** - A Python library for extracting data from HTML and XML files. We used this library for extracting news titles and body from the original news site, using the news URLs extracted from Twitter. This is a really helpful library for parsing HTML in Python programs.

3.4.2 Preparing the Data Set

Preparing the data set is the most important part of this experiment. Since the aim is to develop a news clustering algorithm, having a proper corpus of news data was the main requirement for continuing this study. This section describes how the data set for the experiment was prepared.

3.4.2.1 Extracting News from Twitter

Twitter was used as an index to extract tweets from twitter channels for sports news from CNN, BBC and Al-jazeera (@cnnsport, @BBCSport, @AJE_Sport are the respective twitter handles used).

There are many freely available tools and libraries to extract content from twitter channels. Among them Tweepy is a solid library available for python developers for accessing the Twitter API. With the use of this library, it requires only a few lines of code to extract required content from any Twitter channel.

After importing the required libraries, we have to get authentication by providing our access keys and tokens before accessing Twitter API using Tweepy. Python code snippet in Figure 3.2 does this required authentication.

```
from tweepy import Stream
from tweepy import OAuthHandler
from tweepy.streaming import StreamListener
import tweepy
import json

ckey="########################"
csecret="###################################"
atoken="###################################"
asecret="###################################"

auth = OAuthHandler(ckey, csecret)
auth.set_access_token(atoken, asecret)

api = tweepy.API(auth)
```

Figure 3.2: Code snippet to authorize the program to access Twitter API

After authenticating, extracting tweets from any public channel is just a single line of code by providing the user id of the channel from which we need to extract tweets and the required tweet count. Since the API supports extracting a maximum of 200 tweets for a single call, we have to run this call periodically, in our case twice a day, to accumulate recent tweets. ID of the last tweet extracted from the particular channel is given as the since_id parameter to avoid repetitions. The code snippet is given in figure 3.3.

```
timeline = api.user_timeline(user_id="265902729", since_id=lastRowId, count=200)
```

Figure 3.3: Code snippet to extract timeline from a particular channel

Tweepy will output extracted tweets in json format. It contains a lot of irrelevant stuff and had to extract only the required properties for our study from the json. Part of a sample of

response json for a tweet is shown in figure 3.4. Code snippets to extract relevant json properties are shown in figure 3.5. Extracted tweets were saved in a relational database.


```
{ ⊖
   "created_at":"Fri Jan 24 23:59:00 +0000 2020",
   "id":1220858569987383296,
   "id_str":"1220858569987383296",
   "text":"Tottenham midfielder Christian Eriksen, 27, will reportedly head to Milan for a medical early next week before comp\u2026 https://t.co/wNJlr4uLPC",
   "truncated":true,
   "entities":{ ⊖
      "hashtags":[ ⊖

      ],
      "symbols":[ ⊖

      ],
      "user_mentions":[ ⊖

      ],
      "urls":[ ⊖
         { ⊖
            "url":"https://t.co/wNJlr4uLPC",
            "expanded_url":"https://twitter.com/i/web/status/1220858569987383296",
            "display_url":"twitter.com/i/web/status/1\u2026",
            "indices":[ ⊖
               117,
               140
            ]
         }
      ]
   },
   "source":"<a href=\"https://about.twitter.com/products/tweetdeck\" rel=\"nofollow\">TweetDeck</a>",
   "in_reply_to_status_id":null,
   "in_reply_to_status_id_str":null,
   "in_reply_to_user_id":null,
   "in_reply_to_user_id_str":null,
   "in_reply_to_screen_name":null,
   "user":{ ⊖
      "id":265902729,
      "id_str":"265902729",
      "name":"BBC Sport",
      "screen_name":"BBCSport",
```


Figure 3.4: Part of a response json of a tweet from Tweepy

```
data_list = []
for status in timeline:
  # print(status)
  row = {}
  #convert to string
  json_str = json.dumps(status._json)
  print(json_str)
  #deserialise string into python object
  parsed = json.loads(json_str)
  row['id'] = parsed['id_str']
  row['created_at'] = parsed['created_at']
  row['text'] = parsed['text']
  row['user_name'] = parsed['user']['name']
  row['screen_name'] = parsed['user']['screen_name']
  row['description'] = parsed['user']['description']

  if parsed['entities']['urls']:
    row['url'] = parsed['entities']['urls'][0]['url']
    row['expanded_url'] = parsed['entities']['urls'][0]['expanded_url']
  else:
      row['url'] = ''
      row['expanded_url'] = ''
  data_list.append(row)
```

Figure 3.5: Code snippet to extract relevant properties from a response from Tweepy

3.4.2.2 Extracting News from News Sites

Twitter is just used as an index for extracting news items. Though the character limit of a tweet is limited to 280 characters, most of the extracted tweets had a short caption and a link to the content page in the original news site. Using these short tweet texts will not be sufficient for having effective analysis of our proposed model. So, the URL for the original news page, obtained from twitter, was used to extract the actual news title and body from the original news page. Doing this manually was infeasible for a large corpus with over 3000 news items. So, the process had to be automated.

After extracting the HTML content from the original news page, BeautifulSoup library was used to parse HTML content to extract news title and body from the page. We have to have an idea about the html structure of the news page for extracting content using BeautifulSoup. By analyzing the sports news pages of the 3 sites of our interest (CNN, BBC and Aljazeera) I observed that each site maintains a static structure for their sports news page templates. All the three sites have structured the news body by using a set of <p> or tags with a common ‘class’ attribute for those tags. If you know the tag type and the ‘class’ or ‘id’ attribute of those tags which comprise the news body, it is just a few lines of code to extract the news title and body by using BeautifulSoup library. If we take a sports news page of CNN as an example, news body was comprised using a parent tag with class attribute ‘pg-

rail-tall__body' and a set of child tags with class attributes 'el__leafmedia--sourced-paragraph' or 'zn-body__paragraph'. Known this information we can extract the title and the body from CNN sports news page using the Python code shown in figure 3.6.

```
def extract_cnn_news_item(tweet):
    # Make a GET request to fetch the raw HTML content
    html_content = requests.get(tweet["expanded_url"]).text

    # Parse the html content
    soup = BeautifulSoup(html_content, "lxml")

    # Extracting news title
    title = soup.title.text

    # Extracting news body
    body = ''
    my_div = soup.findAll("div", {"class": "pg-rail-tall__body"})

    for tag in my_div:
        first_div = tag.find_all("div", {"class": "'el__leafmedia el__leafmed
        for tag_1 in first_div:
            body = body + "<p>" + tag_1.text + "</p> "

        para_divs = tag.find_all("div", {"class": "zn-body__paragraph"})
        for tag_2 in para_divs:
            body = body + "<p>" + tag_2.text + "</p> "
    # Saving news title and body in database
    update_title_and_body_of_tweet(tweet['id'], title, body)
```

Figure 3.6: Code snippet to extract news title and body from CNN sports page

BBC and Aljazeera also maintain a static structure for their news pages. So, like for CNN, content from these pages can be extracted using BeautifulSoup. All the extracted original news titles and bodies were persisted to the database.

3.4.2.3 Handling Missing Data

Not all the tweets had the URL for the original post in the original news site. Also, some URLs were not referred to an actual news page in the original site. Those tweets were ignored and only the tweets which provided a URL to an actual sports news page in the respective news site, were considered as valid items. Text corpus for our study was the set of news bodies extracted from original sites.

3.4.2.4 Removing Duplicates

Some twitter channels tweet the same news again and again for making it trending. So, there were duplicate values in our data set. These duplicate values degrade the performance and accuracy of our model, so we had to remove them from our data set. Duplicate items which had exact same content for both the news title and news body were removed using Pandas DataFrames using the code snippet in figure 3.7

```
data_frame = pd.read_sql(mysql_query, con=connection)
data_frame.drop_duplicates(subset=["news_title", "news_body"], keep='first', inplace=True)
```

Figure 3.7: Code snippet to remove duplicates considering news_title and news_body

After handling missing data and removing duplicates, our final data set consisted of **1999** news items which were used for our experiment.

3.4.3 Text Pre-processing

Text pre-processing is the most important step of this experiment. Raw text items cannot be directly fed into clustering algorithms. They should be pre-processed and converted into a numeric vector representation before feeding them to clustering algorithms. Effectiveness of our text pre-processing methods will directly impact the performance and accuracy of the clustering algorithm.

For our approach, following pre-processing steps were selected to optimize the feature set identified for the corpus.

- **Removing tags and special characters** - Only the actual word frequencies matters when selecting features for our clustering algorithms. So, all the unrelated tags and special characters were removed from our corpus as the first step. BeautifulSoup was used to remove HTML tags and python's regular expression module was used to remove special characters in the text.

- **Lowercasing the corpus** - When taking statistical measures like term frequencies and document frequencies they should be case insensitive. So, all the text in the corpus was converted to lowercase.

- **Tokenizing the corpus** - All the statistical measures in our approach depend on frequencies of words(tokens). So, the corpus is tokenized using the WordPunctTokenizer of NLTK. This method tokenizes text using word punctuations.

- **Removing stop words** - Stop words does not have any significant contribution for the feature extraction process. These words usually have the highest term frequencies in the corpus but not useful for distinguishing documents based on feature words. Word categories like determiners, conjunctions and pronouns are considered as stop words in English. There are many off the shelf stop word lists available for use and we used the English language stop word list from NLTK.

- **Stemming and Lemmatization** - When analyzing the similarity of text contents based on their feature words, considering different forms of the same word as different features, will degrade the efficiency of feature extraction. So, all the words are converted into their root stems before further analysis. This was achieved by using Snowball English stemmer provided by NLTK.

Code snippet which performs all the above pre-processing tasks is shown in figure 3.8

```
def normalize_document(doc):

    # Removing html tags from text
    soup = BeautifulSoup(doc, 'lxml')
    doc = soup.get_text()

    # lower case and remove special characters\whitespaces
    doc = re.sub(r'[^a-zA-Z\s]', '', doc, re.I | re.A)
    doc = doc.lower()
    doc = doc.strip()

    # tokenize document

    tokens = wpt.tokenize(doc)

    # filter stopwords out of document
    filtered_tokens = [token for token in tokens if token not in stop_words]

    stemmer = EnglishStemmer()
    stemmed_tokens = [stemmer.stem(word) for word in filtered_tokens]
    # re-create document from filtered tokens
    doc = ' '.join(stemmed_tokens)

    return doc
```

Figure 3.8: Code snippet which performs text pre-processing

3.4.4 Generating the Feature Matrix for the Corpus

After the above pre-processing tasks, documents in our corpus should be converted into their numerical vector forms, before feeding them into clustering algorithms. Each document is converted into its numeric vector representation, and feature vectors for all the documents in the corpus will form the feature matrix. So, the feature matrix has a number of rows equal to number of documents in the input data set and number of columns equal to number of features taken into account. For example, if we consider unigrams when constructing a feature matrix, the number of columns will be equal to the number of all the identified feature words in our corpus.

Both the count vectorizer and tf-idf vectorizer were used for our experiment to compare the generated results. When using count vectorizer, a text document is represented as a numeric vector where each dimension is a feature from the corpus vocabulary and the value is

document frequency, occurrence (0 or 1) or a weighted value of that feature. The feature can be a unigram, bigram, trigram etc.

TF-IDF model calculates the statistical importance of a word in a document by incorporating a normalizing factor without using the term frequency alone. TF-IDF value for a term w in a document D can be calculated using the following formula.

$$tfidf(w,D) = tf(w,D) \times idf(w,D) = tf(w,D) \times log\left(\frac{C}{df(w)}\right)$$

The term *tf(w, D)* is the term frequency of the word *w* in document *D*. The term *idf(w, D)* is the inverse document frequency for the term *w*, which can be computed as the log transform of the total number of documents in the corpus *C* divided by the document frequency of the word *w*, which is basically the frequency of documents in the corpus where the word *w* occurs.

NLTK implements both these vectorizers with ability to input important parameters to be used when generating the feature vectors. In this experiment the two vectorizers were tested by changing below important parameters to find optimal values for these parameters to be used in our model.

ngram-range - A tuple (min_n, max_n), defining the ngram range to be used when generating the feature matrix. For example, if we give (1, 1) only unigrams will be used. If (2, 2) only bigrams will be used. If (1, 2) both unigrams and bigrams will be used.

min_df - Ignore terms that have a document frequency lower than this value when building the vocabulary. This will be used to ignore words with spelling errors or rare words uniquely used in very few documents which have a very low frequency and have negligible impact for our statistical calculations.

max_df - Ignore terms that have a document frequency higher than this value when building the vocabulary. These values are also called as corpus specific stop words, which are frequently used in majority of the documents in corpus and have no significant importance when distinguishing documents based on text similarities

The code for generating feature vectors based on these parameters is shown in figure 3.9

```
ngram_range = (ngram, ngram)

if vectorizer == Constants.COUNT_VECTORIZER:
    cv = CountVectorizer(min_df=0., max_df=1., ngram_range=ngram_range)
    feature_matrix = cv.fit_transform(norm_corpus)
elif vectorizer == Constants.TFIDF_VECTORIZER:
    tv = TfidfVectorizer(min_df=3, use_idf=True, ngram_range=ngram_range)
    feature_matrix = tv.fit_transform(norm_corpus)
```

Figure 3.9: Code snippet for generating feature vectors

3.4.5 Singular Value Decomposition

Since we used news bodies from the original news sites as our data set the feature matrix generated was very sparse. This degraded the performance and accuracy of our proposed model. One solution is to use dimensionality reduction techniques to select only the important features. Singular Value Decomposition (SVD) is a suitable dimensionality reduction technique for a sparse feature matrix like ours [23]. The computation is done using the eigenvectors of the feature matrix, and eigenvalues of those eigenvectors. We used sklearn's SVD implementation where we can define the required dimensionality of our feature matrix after applying SVD.

3.4.6 Pairwise Document Similarity

When we have the feature matrix for the corpus it can be used to compute pairwise similarity between documents in our corpus. Each pair of documents is compared for their similarity based on a similarity function. Some commonly used similarity functions are Cosine distance, Euclidian distance and Manhattan distance. If the text corpus has **N** documents then we will have a **N x N** size similarity matrix, and each element of the matrix will represent similarity index between the documents representing that row and column, respectively.

We used cosine similarity, which measures the cosine of the angle between two document vectors projected in a multi-dimensional space. Lower the angle between the documents, the closer and more similar they are. Figure 3.10 depicts how the similarity of two document vectors u and v will be based on their cosine similarity.

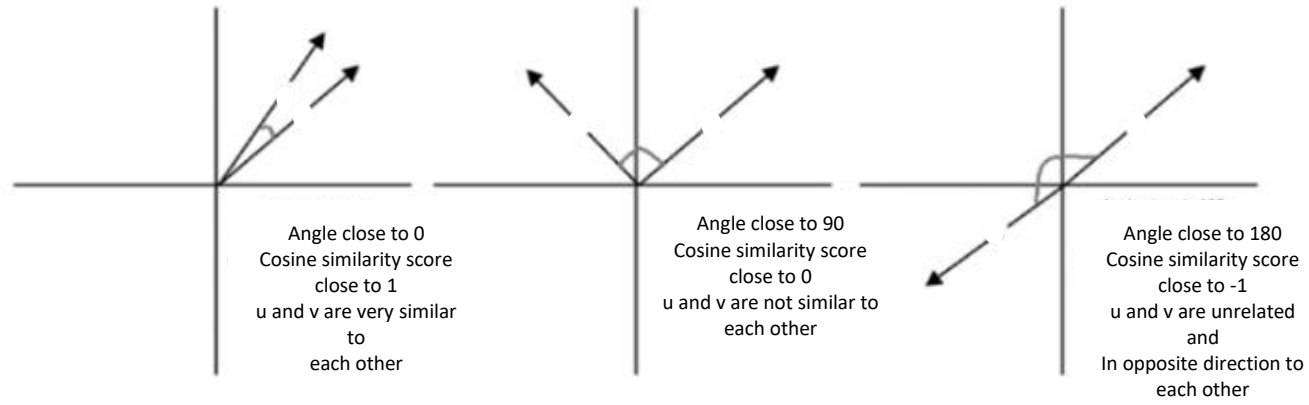

Figure 3.10: Cosine similarity depiction of document feature vectors u and v

Cosine distance can be calculated as

$$Cosine\ Distance = 1 - Cosine\ Similarity$$

We can compute both cosine similarity and distance matrices using sklearn by using the code shown in figure 3.11

```
similarity_matrix = sklearn.metrics.pairwise.cosine_similarity(feature_matrix)

distance_matrix = sklearn.metrics.pairwise.cosine_distances(feature_matrix)
```

Figure 3.11: Code snippet for generating similarity and distance matrices

3.4.7 Elbow Method

Since the objective is to find the optimal number of clusters for the given corpus, Elbow Method with K-Means clustering was experimented with different vectorizers (count and tf-idf) and input parameters (min_df, max_df and ngram_range). Here K is iterated over a range and computed clusters for each K value. The elbow curve is obtained by plotting the average sum of squared errors for clusters at each K against K. If the plotted graph has a shape of an arm then the point of inflection on the graph (the elbow) is a good indication that the underlying model fits best at that point. We take value at that point as the optimal K. For an example we can take the best K value as 3 for the model represented by the elbow curve shown in figure 3.12.

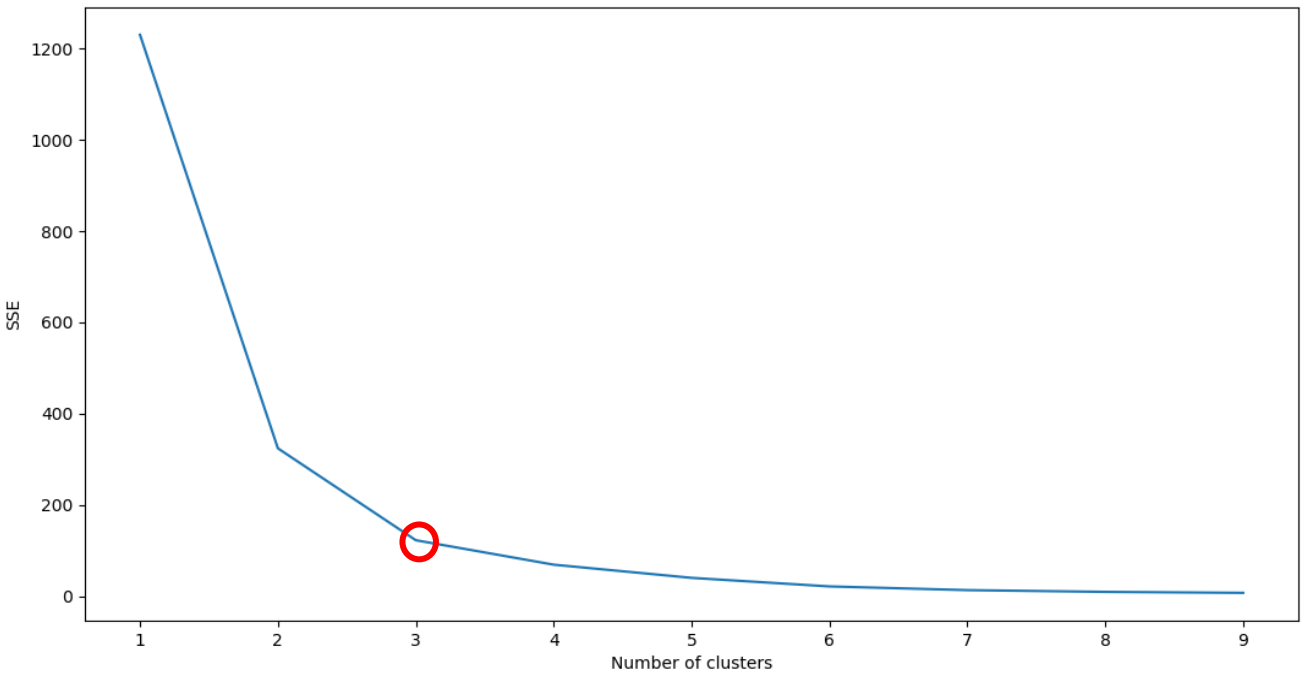

Figure 3.12: Elbow curve with clear elbow

When we evaluate clusters for large data sets the elbow of the curve will not always be clear as in the above figure. Sometimes there can be few bends in the curve and make it confusing to select the optimal K just by examining the graph. We can use other intrinsic cluster evaluation measures like Silhouette Coefficient in such ambiguities.

3.4.8 Silhouette Coefficient

Silhouette coefficient measures how well an object fits to its own cluster compared to other neighbouring clusters. The value ranges from -1 to +1. Higher values closer to 1 indicates better clustering configurations. Silhouette value ***S*** can be calculated using the following formula where ***a*** is mean intra cluster distance and ***b*** is the distance between a sample and the nearest cluster that the sample is not a part of.

$$S = (b - a) / max (a, b)$$

We used both the Silhouette Score curve and Elbow curve to obtain the optimal number of clusters for our model. As an example, take the elbow curve shown in figure 3.13, drawn for a sample data set. We don't see a clear elbow in this curve. There are few elbows for K values from 6 to 9. So, it is difficult to come up with an optimal K by using only this curve.

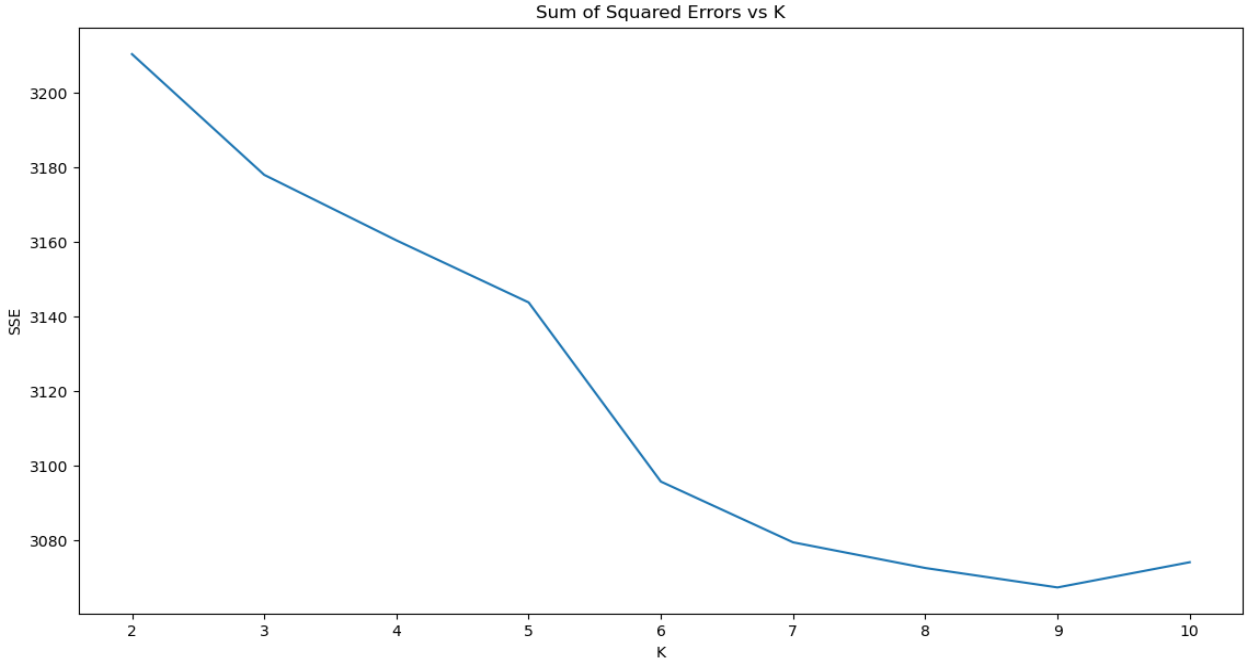

Figure 3.13: Elbow curve with several bends

If we take the Silhouette curve for the same range of K values, we get the curve shown in figure 3.14. We can see a clear peak at K = 8 and from the elbow curve we know that optimal K should be somewhere between 6 and 9. So we can get that K = 8 is a good candidate for the optimal number of clusters and use that value for further analysis.

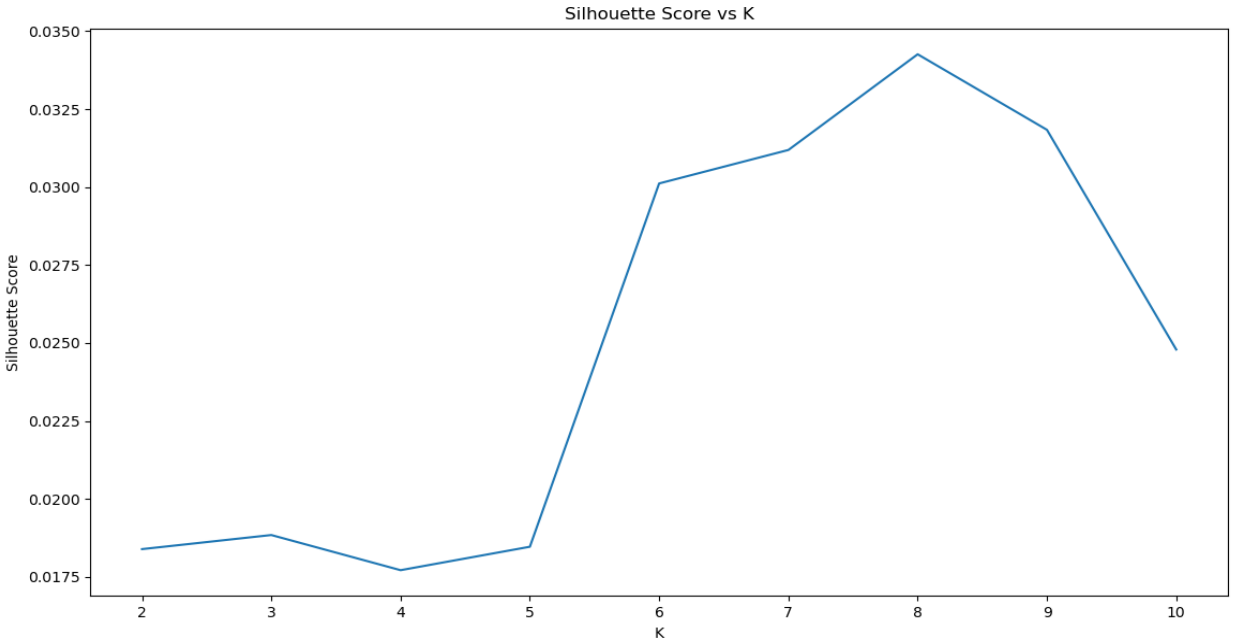

Figure 3.14: Silhouette curve for same range of k

Analyzing our data set to come up with an optimal number of clusters using the combination of elbow method and Silhouette coefficient was performed using the code shown in figure 3.15. Cosine matrix constructed above was used to calculate Silhouette scores for clusters in each K. Matplotlib library's plot function was used for plotting the calculated results.

```
distortions = []
    silhouette_score_values = list()
    K = range(start_k, end_k + 1)
    for k in K:
        print('Generating clusters for k = {}'.format(k))
        kmeanModel = KMeans(n_clusters=k, init='k-means++', max_iter=100, n_init=1)
        kmeanModel.fit(feature_matrix)
        distortions.append(kmeanModel.inertia_)
        cluster_labels = kmeanModel.labels_
        score = sklearn.metrics.silhouette_score(distance_matrix, cluster_labels.ravel(), metric='precomputed')
        silhouette_score_values.append(score)

    plot1 = plt.figure(1, figsize=(16, 8))
    plt.title('Sum of Squared Errors vs K')
    plt.xlabel('K')
    plt.ylabel('SSE')
    plt.plot(K, np.array(distortions).ravel())

    plot2 = plt.figure(2, figsize=(16, 8))
    plt.title('Silhouette Score vs K')
    plt.xlabel('K')
    plt.ylabel('Silhouette Score')
    plt.plot(K, np.array(silhouette_score_values).ravel())
```

Figure 3.15: Code for K-means cluster analysis using elbow method and Silhouette score

Chapter 4 - Evaluation Plan

4.1 Cluster Validation

Cluster validation involves evaluating results obtained by different clustering algorithms and measuring the accuracy and validity of generated clusters [20]. Validating a clustering algorithm is not straight forward as validating a supervised learning algorithm since we do not have class labels [21]. There are three main factors which can be used to evaluate a clustering algorithm: clustering tendency, number of clusters and clustering quality [21].

4.1.1 Clustering Tendency

The data set used for evaluating our clustering algorithm should not be uniformly distributed. It should have a clustering tendency or in other words it should have a set of clearly identifiable clusters. If the data set is uniformly distributed, clusters generated by any clustering approach will be irrelevant [21].

4.1.2 Number of Clusters

Getting the optimal number of clusters is very significant when it comes to cluster analysis. If the number of clusters is too high, each data point will broadly start representing a cluster. If the number of clusters is too low then data points will be incorrectly assigned to the same cluster. Finding the optimal number of clusters is important in cluster analysis [21]. Having a sound knowledge about the domain of used dataset is important when deciding the optimal number of clusters. But with absence of such knowledge, one can use mathematical methods when calculating optimal number of clusters [21].

4.1.3 Clustering Quality

The quality of the generated clusters can be measured by intra-cluster and inter-cluster distances. The quality of clusters is high when intra-cluster distance is minimal and inter-cluster distance is maximal [21].

There are two major methods to assess the quality of generated clusters

(i) Extrinsic Measures like Jaccard scores, Adjusted Rand index, Fowlkes-Mallows scores, Homogeneity, Mutual information-based scores, Completeness and V-measure can be used to evaluate clusters when the ground truth labels are available for the data set. [21].

(ii) Intrinsic Measures such as Silhouette Coefficient, Calinski-Harabasz Index, Davies-Bouldin Index etc can be used when ground truth labels are not available[21].

4.2 Evaluation of the Proposed Algorithm

The entire evaluation process should be focussed on validating the unsupervised behaviour of our algorithm to cluster text documents with the presence of following two problems.

1. Cluster labels are not predefined.
2. Number of clusters is not predefined.

The entire data set with 1999 news documents were evaluated with the proposed approach, and clusters generated for the optimal K value, predicted by our approach, were interpreted to evaluate the validity of the clustering criteria. Since the sports domain is a familiar domain, we can use ground truth knowledge for this analysis. Also, both extrinsic and intrinsic indexes, involving mathematical calculations, were used to evaluate the quality of the generated clusters. The data set was analyzed under different input parameters to come up with a set of parameter combinations which best suit our dataset.

Once the model is finalized with the set of parameters to be used when generating clusters, a set of 315 manually and randomly selected news items from our data set, with assigned ground truth labels were used to evaluate the validity of our model. Data points were selected in a way that they belong to a one of 5 predefined categories as listed below.

1. Football - 100 news items
2. Cricket - 35 news items
3. Races - 80 news items

4. Fights - 35 news items
5. Olympics - 65 news items

This sample data set will be evaluated using our model to test whether the optimal number of clusters (K) will be predicted correctly and the generated clusters for that K value represent one of the predefined categories above. The validity of the generated clusters will be compared with ground truth values using intrinsic methods. Our model did not have any knowledge about the number of clusters or cluster labels when generating these optimal sets of clusters. This verifies our model's ability to generate accurate clusters with the absence of ground truth knowledge.

4.3 Results

The entire data set of 1999 news items were analyzed with our proposed approach to evaluate the validity of the optimal K value and clusters generated . The experiment was performed with different parameter values for feature extraction and clustering algorithms. Generated results are discussed and analyzed below.

4.3.1 Selecting Optimal Values for min_df and max_df parameters

First attempt was to find an optimal set of parameters which optimize the feature extraction process of our model. We tried using different values for min_df and max_df for feature extraction and obtained results are depicted below.

Different values for max_df were tested keeping min_df fixed at 0, by running the K-means algorithm for a range of k from 2 to 10. TF-IDF vectorizer was used and we limited the maximum number of iterations of k-means for each k (max_iter) to 100 and number of times k-means will run with different cetriod seeds (n_init) to 10 considering the processing power of the machine to cater a sparse feature matrix like ours. Observed results are depicted in table 4.1. From the results we can observe that keeping the max_df value equal to the number of items in our data set is sufficient to filter out the corpus specific stop words, and using less values degrades the clustering quality.

min_df	max_df	max_iter	n_init	Silhouette Score Range	SSE range	Elapsed time	Number of features
0	1999	100	10	0.0114 - 0.0321	1780 - 1870	15:14	21352
0	1000	100	10	0.0099 - 0.0292	1796 - 1889	22:55	21337
0	500	100	10	0.0133 - 0.0302	1830 - 1920	23:29	21250
0	100	100	10	0.0063 - 0.0246	1910 - 1970	05:07	20488

Table 4.1: Test Results for feature extraction using different max_df values

Having decided a suitable value for max_df, our next attempt was to find an optimal value for min_df. We tested different values for min_df, keeping max_df fixed at 1999 (optimal value decided by the above test results). We used the same values, we used for finding max_df, for the other parameters as well. Obtained results are depicted in table 4.2. From the results we can decide that changing min_df to higher values does not have much effect in enhancing feature extraction for our text corpus. So, 3 was used for min_df to remove infrequent words including spelling mistakes and typos.

min_df	max_df	max_iter	n_init	Silhouette Score Range	SSE range	Elapsed time	Number of features
3	1999	100	10	0.0162 - 0.0314	1760 - 1860	12:42	9476
6	1999	100	10	0.0177 - 0.0374	1740 - 1850	11:29	6072
10	1999	100	10	0.0185 - 0.0407	1720 - 1840	10:22	4398

Table 4.2: Test Results for feature extraction using different min_df values

Graphs of SSE and Silhouette Score plotted against k, by running k-means algorithm with above parameters and setting min_df to 3 and max_df to 1999, are depicted in figure 4.1 and figure 4.2.

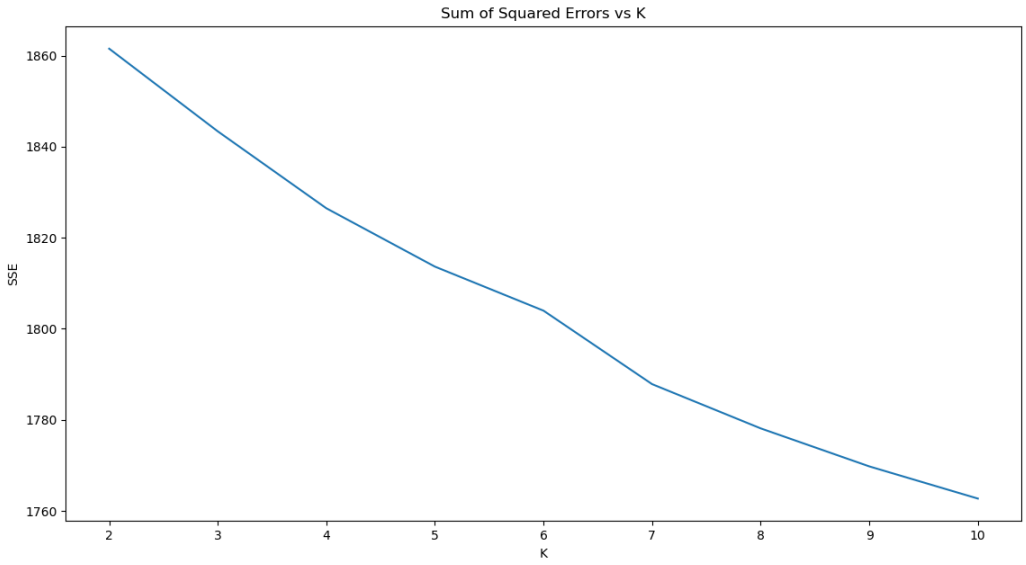

Figure 4.1: SSE plot against k for optimal values of min_df and max_df

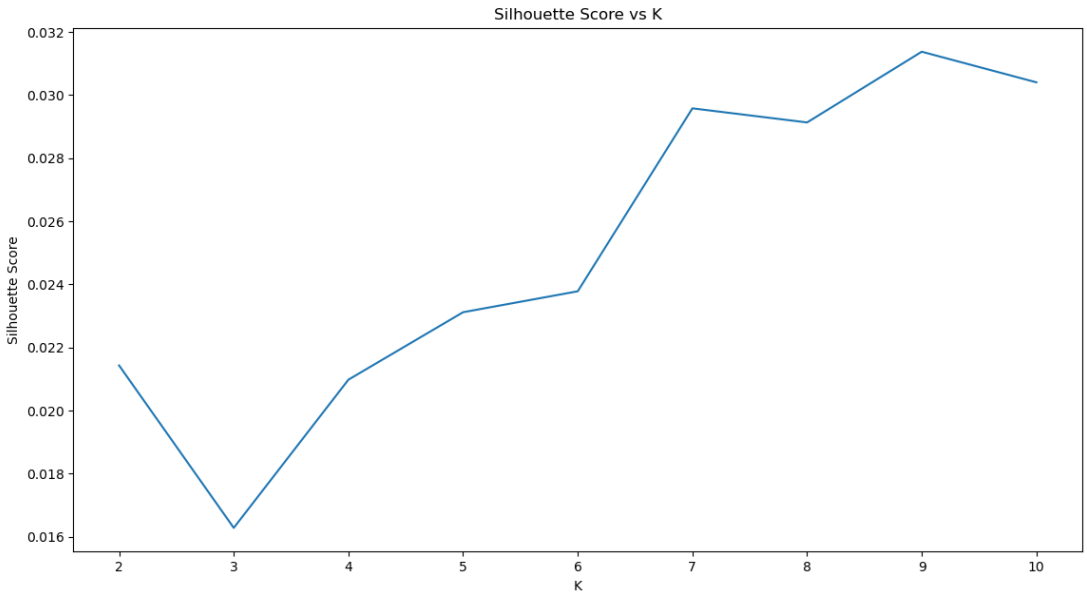

Figure 4.2: Silhouette score plot against k for optimal values of min_df and max_df

From these results we can see that the Sum of Squared Errors are very high and Silhouette scores are very low. So, we continued the experiment on finding better parameters for our feature extraction process to improve the quality and accuracy of generated clusters.

4.3.2 Testing with Count vectorizer

Our next attempt was to use count vectorizer instead of tf-idf vectorizer and check whether our results are improved. Figure 4.3 and figure 4.4 depicts plots of SSE and Silhouette Score against k, with keeping all the input parameters same as for the above test.

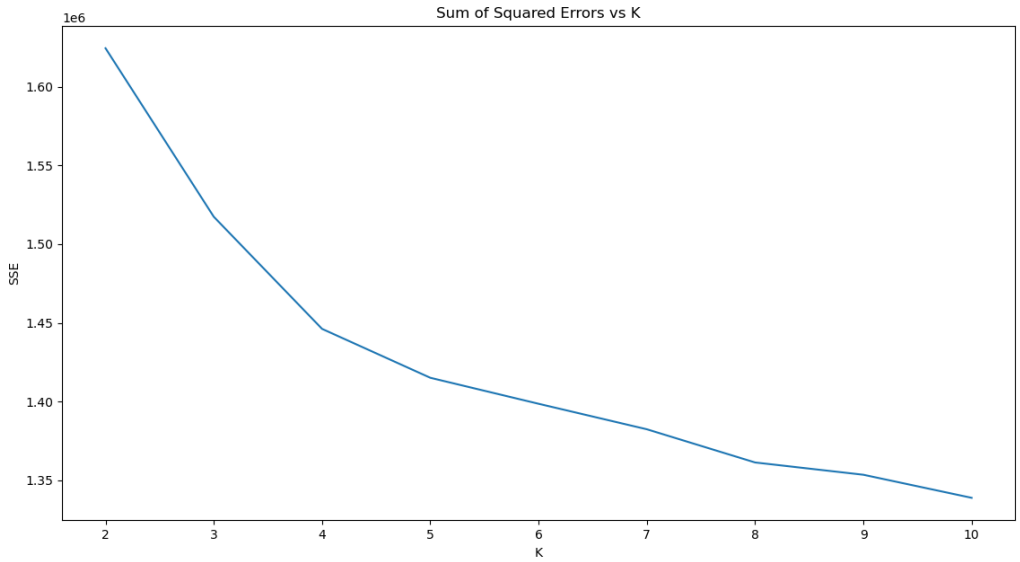

Figure 4.3: SSE plot against k when using count vectorizer

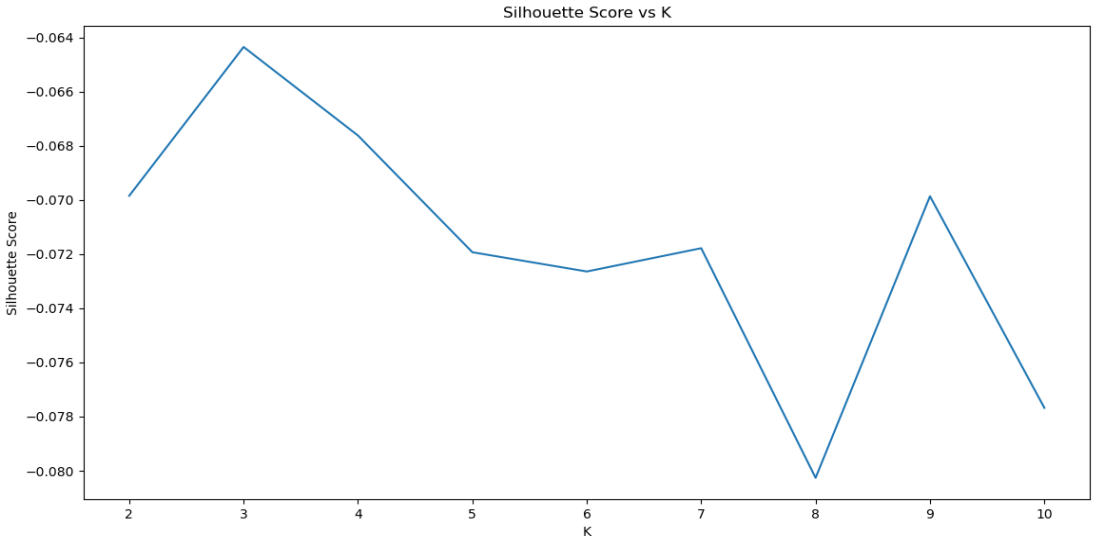

Figure 4.4: Silhouette score plot against k when using count vectorizer

When using count vectorizer SSE values are very large (in range of 10^6) and silhouette score values are negative. So, it is clear that the tf-idf vectorizer produces much better results with the same parameter settings. So, we move forward with the tf-idf vectorizer.

4.3.3 Testing with Bigrams

The experiment was continued to check whether using bigrams instead of unigrams. All the above results were generated by using unigrams as features. Tests were done to visualize the effect of using bigrams instead of unigrams while keeping all the other parameters unchanged and using the optimal parameters we have identified up to now. Plots of SSE and Silhouette score against k were generated and depicted in figure 4.5 and 4.6. From the results we can see that using unigrams produced better results for our text corpus.

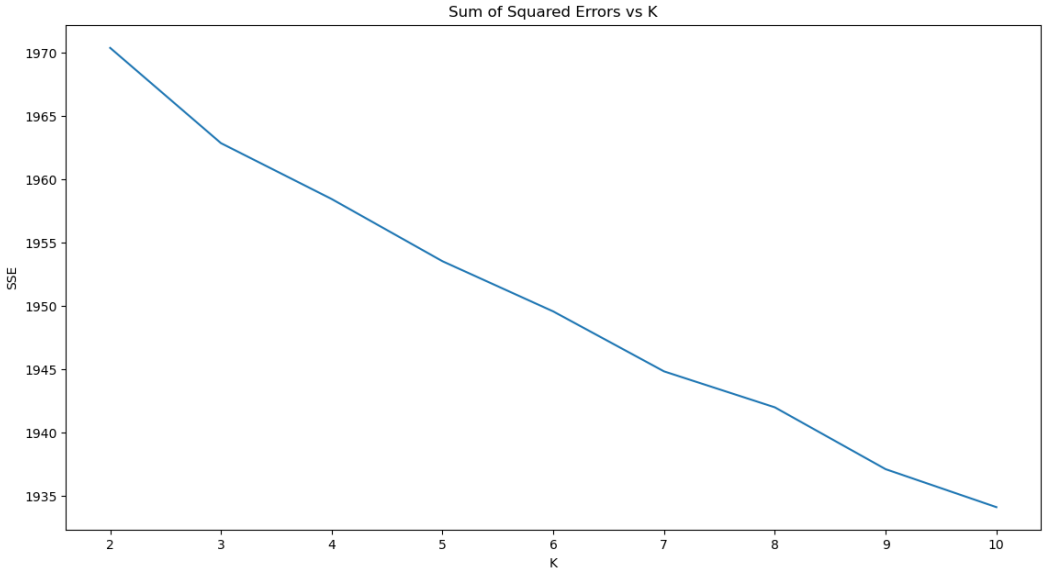

Figure 4.5: SSE plot against k when using bigrams

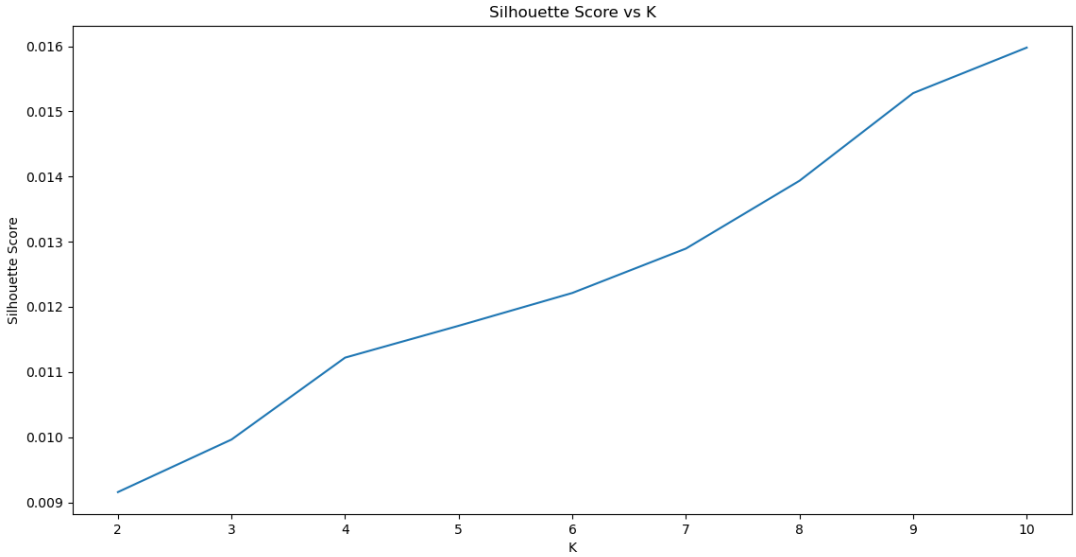

Figure 4.6: Silhouette score plot against k when using bigrams

4.3.4 Applying Singular Valued Decomposition

From the experimental results visualized up to now, we know that we get better scores for generated clusters when using tf-idf vectorizer with unigrams for our text corpus. Also, min_df and max_df were set to 3 and 1999 (number of input data items) respectively when obtaining these results. But still the SSE values are very high and Silhouette score values are very low for the generated clusters. The main reason for this could be the sparseness of our feature matrix which incurred a lot of noise in our clustering module. So, we used Singular Valued Decomposition(SVD) to reduce the dimensionality of our feature matrix.

So, experiment was continued by reducing the feature matrix into different dimensionalities while keeping the optimal parameters and methods identified in above tests as they are. Obtained results are depicted in table 4.3. n_components is the desired dimensionality of the output feature matrix (output feature matrix will have the dimensionality, *Number of Documents* x *n_components*). Below methods and parameter values were fixed (identified optimal methods and parameter values from our previous tests) in this test.

vectorizer = tf-idf **min_df** = 3 **max_df** = 1999 **ngrams** = unigrams

n_components	max_iter	n_init	Silhouette Score Range	SSE range	Elapsed time
5	100	10	0.1572 - 0.4092	26 - 87	00:14
7	100	10	0.2359 - 0.3248	40 - 120	00:13
10	100	10	0.2092 - 0.3196	60-160	00:14

Table 4.3: Test Results for feature extraction with SVD applied

From the above results we can see that the quality of generated clusters has improved a lot with the application of SVD. Also, the elapsed time of our model has reduced from ~15 minutes to few seconds hence a clear performance improvement. By analyzing the SSE plot and the Silhouette score plot for these values it was clearly visible that the number of clusters for our entire data set lies somewhere around 5 and 6. Graphs of SSE and Silhouette score plotted against k, when n_components = 7, are depicted in figure 4.7 and figure 4.8.

The entire data set was analyzed using our model for k = 6 and the generated clusters were visualized manually to give them a meaning. There were clearly visible categories like Football, Races, Fights, Olympics etc, according to the majority of news items that fell into that category. Of course, there were few clusters with mixed data where we cannot come up with a meaning for those.

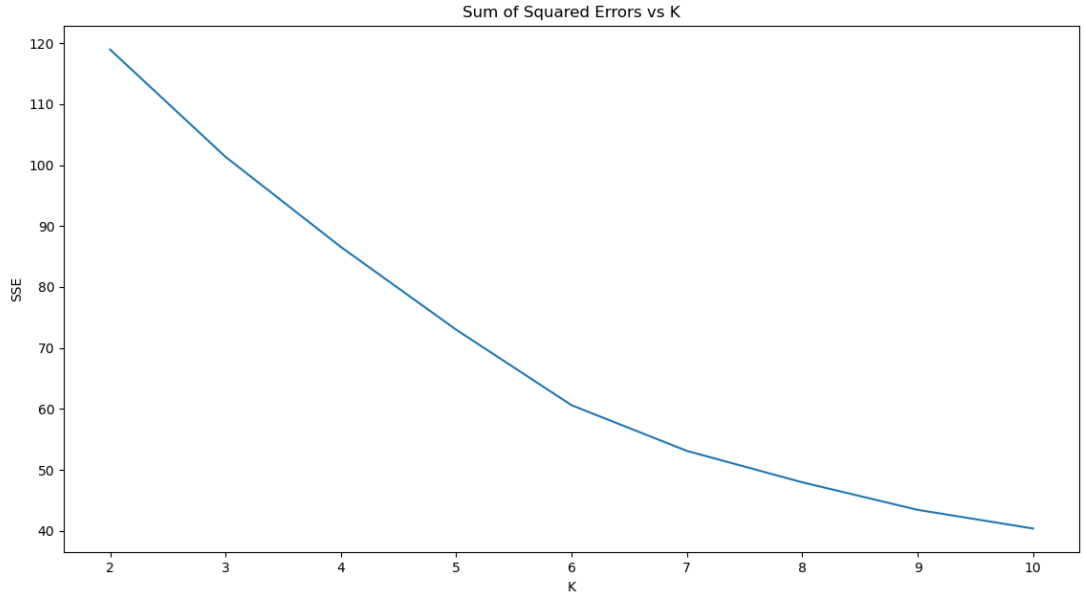

Figure 4.7: SSE plot against k with SVD applied

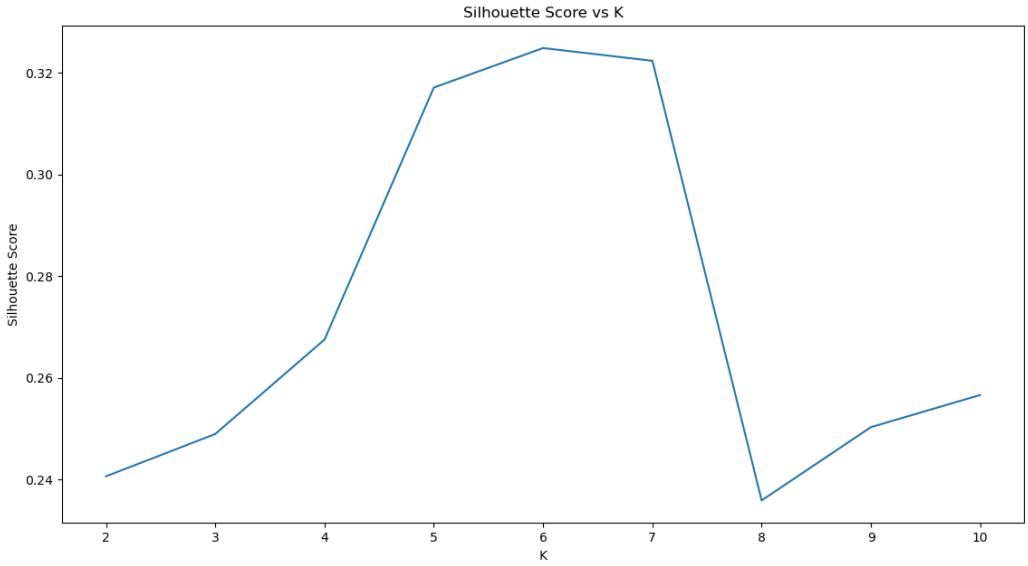

Figure 4.8: Silhouette score plot against k with SVD applied

4.3.5 Evaluating the Model Using a Sample Data Set with Known Ground Truth

A data set with 315 news items, randomly selected from our entire data set, was used to evaluate our model. The items were selected in a way that each item belongs to one of five predefined categories; Football, Cricket, Races, Fights, Olympics. This sample data set was evaluated using

our model without taking any available ground truth information for the execution of the model, and generated clusters were evaluated by comparing them with available ground truth values. Both intrinsic and extrinsic measures were used to evaluate the validity of generated clusters. Plots of SSE and Silhouette score against k are depicted in figure 4.9 and figure 4.10

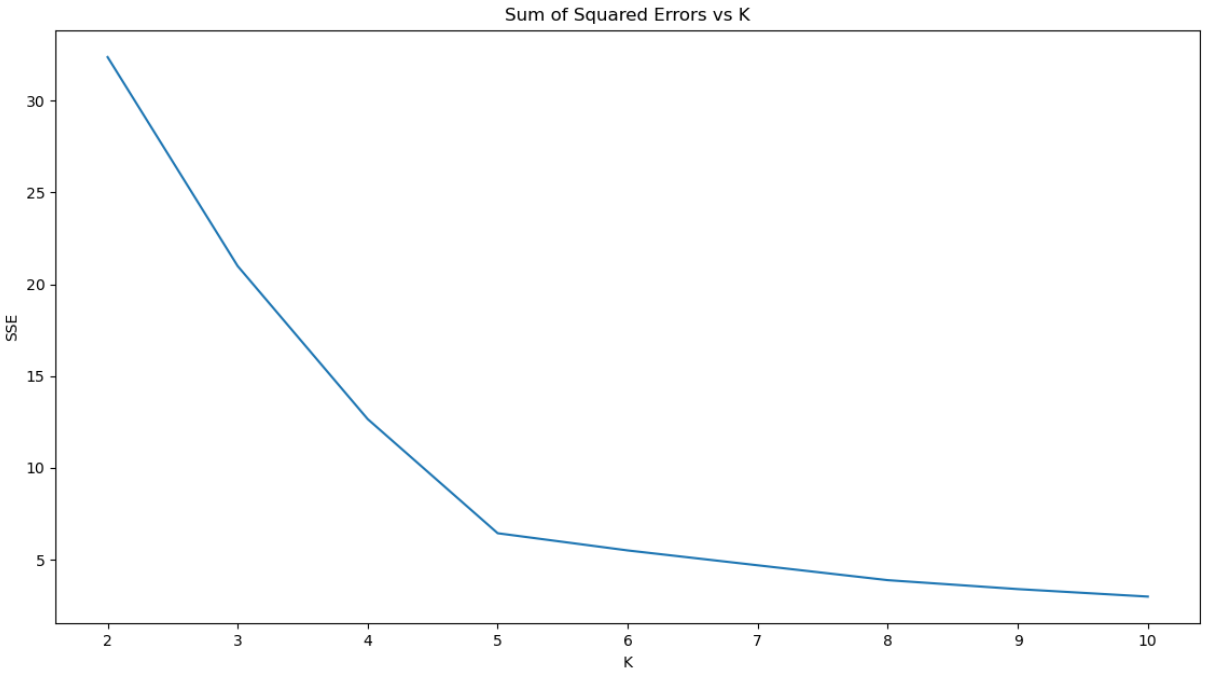

Figure 4.9: SSE plot against k for the sample with ground truth

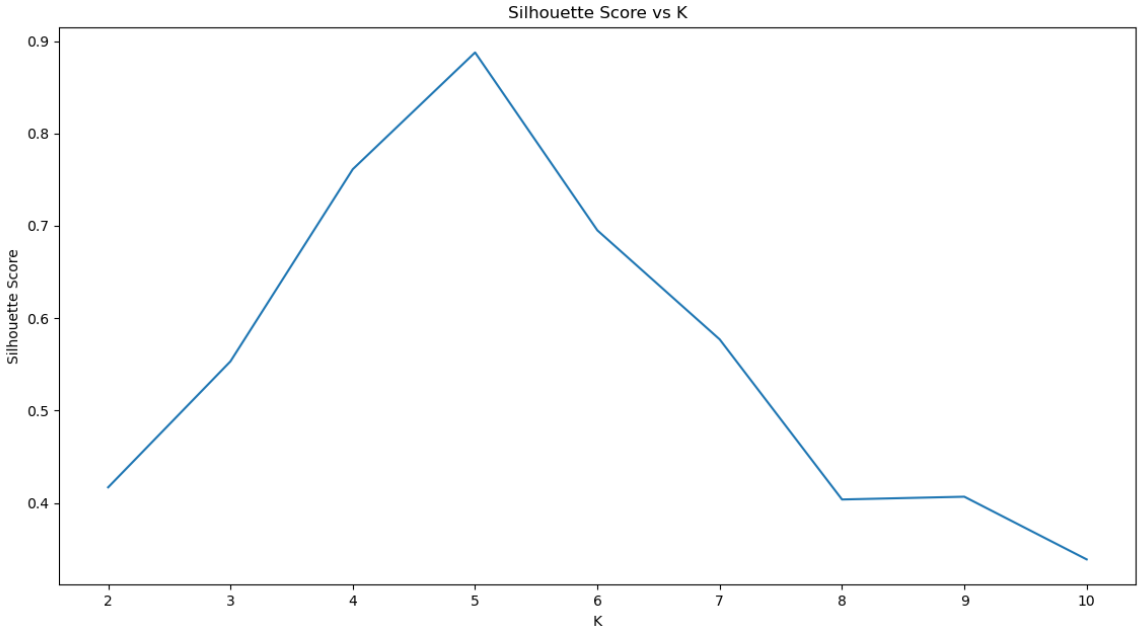

Figure 4.10: Silhouette Score plot against k for the sample with ground truth

From the above two diagrams it is very clear that the optimal number of clusters identified by our model is 5. Silhouette score value is very close to 1 (~0.9) and also SSE is less than 10 when k =5. Also, we calculate Adjusted Rand Index to check the validity of predicted cluster labels compared to ground truth labels. **Random index was 0.9694** which is very close to 1 and this shows almost all the data points are assigned to correct clusters by our model

This proves our model's ability to identify the optimal number of clusters and group them into correct clusters with the absence of predefined cluster labels or number of clusters with higher degree of accuracy.

Chapter 5 - Conclusion and Future Work

5.1 Conclusion

The focus of this study was to come up with an accurate and high-performance text clustering model to cluster online news data. The main objective was to find the optimal number of clusters for the data set, without any knowledge about the number of clusters or cluster labels beforehand. The biggest challenge we faced in this approach was to identify the optimal set of features when constructing the feature matrix. Since we used news bodies for clustering, each news item had lengthy content. So even after necessary and recommended pre-processing steps, the generated feature matrix was a sparse one, which degraded accuracy and performance of our model. By empirical results we observed that the tf-idf vectorizer with unigrams best suits our text corpus. But still the feature matrix was sparse and full of noise. Using Singular Valued Decomposition to reduce the number of features to a value less than 10 worked well and we observed a noticeable improvement in our results. By visualizing the Elbow curve and Silhouette score curve together we came up with candidate k values for optimal number of clusters for our corpus. Clusters were generated for those candidate k values and those clusters were interpreted manually. Though there were few overlapping clusters, we also had a set of meaningful clusters, represented by the majority of data items assigned into those clusters. We observed that our model's accuracy degrades with the presence of overlapping clusters. But it generated very accurate results when used with a data set which has non overlapping clusters. Results generated for our sample data set with a set of non-overlapping clusters verified that the accuracy of our model is more than 95%. We also observed a huge performance gain using our hybrid model, when compared with using k-means with only basic pre-processing steps.

5.2 Future work

Text clustering is a broad research area and this research will open up many doors to develop effective and improved versions of clustering models for the purpose of clustering dynamic text content. Even though the scope of this research was limited for developing a static model to identify an optimal set of clusters in a given data set, the model can be extended to a one which automatically adapts for dynamically changing text content. When the input data set changes and new clusters are introduced, a dynamic clustering model will automatically adapt to identify new clusters in those dynamic text domains. Also, our model produced less accurate results when

there is a lot of noise with overlapping clusters. Experiments can be performed to optimise the feature extraction process to identify an optimal set of features with the presence of overlapping clusters to distinguish them as different clusters. Different dimensionality reduction techniques can be experimented to improve the results when dealing with sparse feature matrices like ours.

References

[1] Wikipedia, “Natural language processing”, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Natural_language_processing

[2] Webopedia, “NLP - natural language processing”. [Online]. Available: https://www.webopedia.com/TERM/N/NLP.html

[3] Gaurav S. Chavan, Sagar Manjare, Parikshit Hegde, Amruta Sankhe “A Survey of Various Machine Learning Techniques for Text Classification”, International Journal of Engineering Trends and Technology 2014, 15(3), pp 288-292

[4] Computing for the Social Science, “Supervised classification with text data”. [Online]. Available: https://cfss.uchicago.edu/notes/supervised-text-classification/

[5] Mehdi Allahyari et al., “A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques”, “University of Georgia Athens, GA”, arXiv:1707.02919v2 [cs.CL] 28 July 2017

[6] Charu C. Aggarwal and ChengXiang Zhai, “Text Mining”, Boston MA: Springer US, 2012, Chapter 4, pp. 77-128

[7] Tao Peng, Wanli Zuo and Fengling He, “SVM based adaptive learning method for text classification from positive and unlabeled documents”, Knowledge and Information Systems 16(3), 281-301

[8] Laith Mohammad Abualigah and Ahamad Tajudin Khader, “Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering”, The Journal of Supercomputing, 76, pp. 4773-4795.

[9] Junyuan Xie, Ross Girshick and Ali Farhadi, “Unsupervised Deep Embedding for Clustering Analysis”, International Conference on Machine Learning 2016, 48, pp. 478-487

[10] Hutchison et al., “W-kmeans: Clustering News Articles Using WordNet”, Knowledge-Based and Intelligent Information and Engineering Systems 2010, 6278, 379-388

[11] Julian Sedding and Dimitar Kazakov, “WordNet-based Text Document Clustering”, Workshop on RObust Methods in Analysis of Natural Language Data - ROMAND '04, pp. 104-113

[12] Bisandu, D.B., Prasad, R. and Liman, M.M. (2018) ‘Clustering news articles using efficient similarity measure and N-grams’, *Int. J. Knowledge Engineering and Data Mining*, Vol. 5, No. 4, pp.333–348

[13] Ilya Blokh and Vassil Alexandrov, “News clustering based on similarity analysis”, Procedia Computer Science, Vo.l 122, pp. 715-719

[14] Dou Shen et al. “Thread Detection in Dynamic Text Message Streams”, Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '06, 35-42

[15] Irani et al. “Clustering Techniques and the Similarity Measures used in Clustering: A Survey”, International Journal of Computer Applications, Vol. 134, No.7, pp.9-14

[16] Professor Ashoka S. Karunananda, “HOW TO DO RESEARCH”, 2001, Author publication, Sri Lanka

[17] M. Oghenefejiro Winnie, “Assessing the credibility of online social network messages”, 2018. [online]. Available: https://derby.openrepository.com/bitstream/handle/10545/622367/Winnie%20E-Thesis%20Final%20copy%2017-10-2017.pdf?sequence=2&isAllowed=y

[18] Deborah Gabriel, “Inductive and deductive approaches to research”, 2013. [online]. Available: https://deborahgabriel.com/2013/03/17/inductive-and-deductive-approaches-to-research/

[19] Amigó Enrique et al. “A comparison of extrinsic clustering evaluation metrics based on formal constraints”,Information Retrieval, Vol 12, No.4, pp.461-486

[20] Science Direct, “Clustering Quality”. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/clustering-quality

[21] Towards Data Science, “Clustering Evaluation Strategies”, 2019. [Online]. Available: https://towardsdatascience.com/clustering-evaluation-strategies-98a4006fcfc

[22] SciPy.org, “scipy.cluster.hierarchy.linkage” [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html

[23] Science Direct, “Singular Value Decomposition”. [Online]. Available: https://www.sciencedirect.com/topics/mathematics/singular-value-decomposition

APPENDIX

Appendix A: Project Plan and Timeline

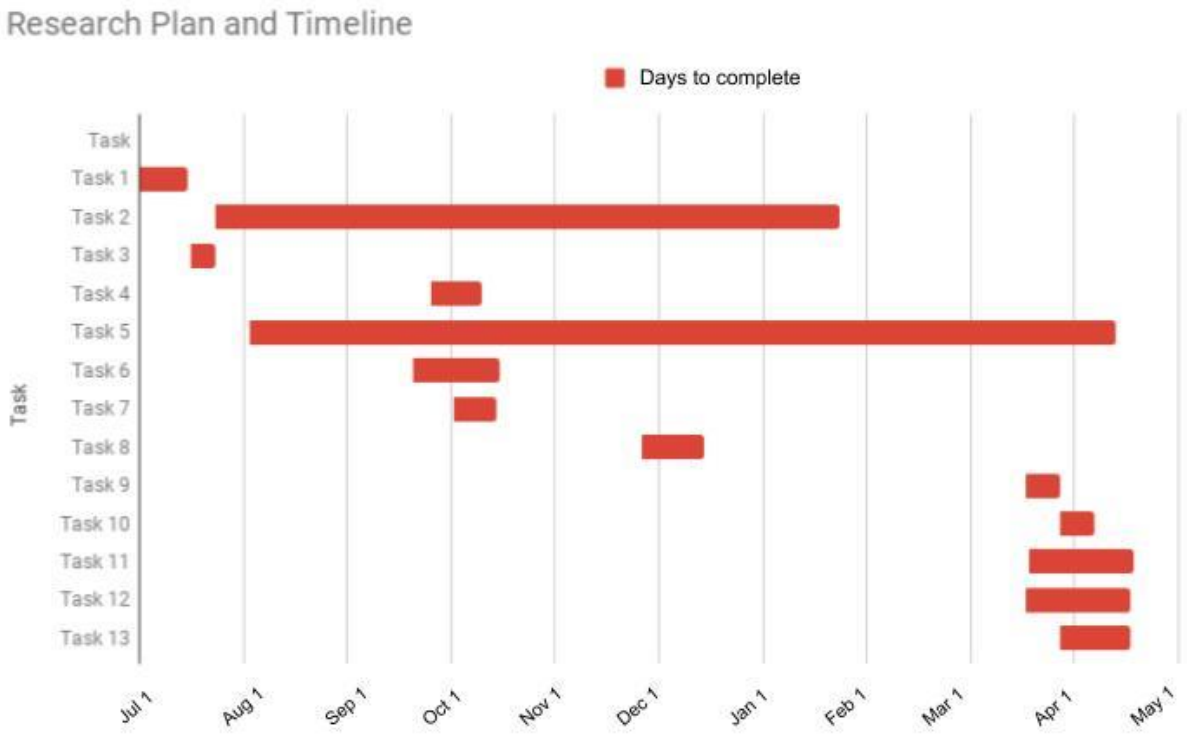

Figure: Research Plan and Timeline

Task	Description
Task 1	Selecting of Supervisor/ Project Title
Task 2	Literature review
Task 3	Prepare preliminary proposal
Task 4	Prepare final proposal
Task 5	Research and development of the proposed model
Task 6	Write Literature Review chapter
Task 7	Write Introduction chapter
Task 8	Prepare interim report
Task 9	Completing the Implementation chapter
Task 10	Write Evaluation Plan chapter
Task 11	Integrating system components
Task 12	Finalising the thesis
Task 13	Preparing final dissertation

Table: Task breakdown of the research plan and timeline