CLUSTERING BASED PERSONALITY PREDICTION ON TURKISH TWEETS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ESEN TUTAYSALGIR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JULY 2019

Approval of the thesis:

CLUSTERING BASED PERSONALITY PREDICTION ON TURKISH TWEETS

submitted by **ESEN TUTAYSALGIR** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit Oğuztüzün
Head of Department, **Computer Engineering** _____

Prof. Dr. İsmail Hakkı Toroslu
Supervisor, **Computer Engineering, METU** _____

Prof. Dr. Pınar Karagöz
Co-supervisor, **Computer Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Sinan Kalkan
Computer Engineering, METU _____

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering, METU _____

Assoc. Prof. Dr. Osman Abul
Computer Engineering, TOBB ETU _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Esen Tutaysalgır

Signature :

ABSTRACT

CLUSTERING BASED PERSONALITY PREDICTION ON TURKISH TWEETS

Tutaysalgır, Esen
M.S., Department of Computer Engineering
Supervisor: Prof. Dr. İsmail Hakkı Toroslu
Co-Supervisor : Prof. Dr. Pınar Karagöz

July 2019, 54 pages

In this thesis, we present a framework for predicting the personality traits of users using their tweets written in Turkish. The prediction model is constructed with a clustering based approach. We show how to extract linguistic features from tweet data and to adapt TF-IDF weighting and word embeddings to the Turkish tweets. Since the model is based on linguistic features, it is language specific. The prediction model uses features applicable to Turkish language and related to writing style of Turkish Twitter users. Our approach uses anonymous BIG5 questionnaire scores of volunteer participants as ground truth in order to generate personality model from Twitter posts. Experiment results show that constructed model can predict personality traits of Turkish Twitter users with relatively small errors.

Keywords: Personality analysis, Text mining, Clustering, Twitter

ÖZ

TÜRKÇE TWEETLERDEN KÜMELENMEYE DAYALI KİŞİLİK TAHMİNİ

Tutaysalgır, Esen
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. İsmail Hakkı Toroslu
Ortak Tez Yöneticisi : Prof. Dr. Pınar Karagöz

Temmuz 2019 , 54 sayfa

Bu tezde, Türkçe tweetler yazan kullanıcıların kişilik analizlerini tahmin etmek için yeni bir model sunuyoruz. Bu tahmin modeli kümelemeye dayalı bir yaklaşımla oluşturulmuştur. Dilbilimsel özelliklerin tweet verilerinden nasıl çıkarılacağını ve TF-IDF ağırlıklandırma ve kelime gömme işlemlerinin Türkçe tweetlere nasıl uyarlandığını gösteriyoruz. Model dilsel özelliklere dayandığından, dile özgüdür. Tahmin modeli, Türkçe'ye uygulanabilen ve Türkçe kullanan Twitter kullanıcılarının yazma stiliyle ilgili özellikleri kullanır. Bizim yaklaşımımız, Twitter gönderilerinden kişilik modeli oluşturmak için gönüllü katılımcıların anonim BIG5 anket puanlarını, temel dayanak olarak kullanıyor. Deney sonuçları, inşa edilen modelin, Türk Twitter kullanıcılarının kişilik özelliklerini nispeten küçük hatalarla tahmin edebileceğini göstermektedir.

Anahtar Kelimeler: Kişilik analizi, Metin madenciliği, Kümelenme, Twitter

To my beautiful family...

ACKNOWLEDGMENTS

Firstly, I would like to thank my supervisor Prof. Dr. Ismail Hakkı Toroslu and my co-supervisor Prof. Dr. Pınar Karagöz. They gave me a chance to proceed in this field and always guided, motivated and supported me in a way that a student may ever need.

I would also like to thank the other members of my thesis committee: Assoc. Prof. Dr. Sinan Kalkan and Assoc. Prof. Dr. Osman Abul for their criticism and suggestions.

I would also like to thank my colleagues in TUBITAK BILGEM ILTAREN for their understandings and patience throughout my thesis studies.

Finally, I would like to express my special thanks to my mom, my dad, my brother, and my dearest husband, Osman. They encouraged me every time when I fell in despair. They motivated and supported me during this thesis and my life. They were the source of inspiration for me. I am so lucky to have such a beautiful family. I know that I will achieve much more amazing things with them.

In addition, this work is supported by Ministry of Science, Industry and Technology of Turkey and Huawei Turkey with project number TEYDEB 1505-5180003.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ALGORITHMS

ALGORITHMS

LIST OF ABBREVIATIONS

NLP	Natural Language Processing
OCEAN	Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism
TF-IDF	Term frequency-Inverse document frequency
CBOW	Continuous Bag-of-Words
IQR	interquartile range
t-SNE	t-Distributed Stochastic Neighbor Embedding
PoS	Part of Speech
PCA	Principal Component Analysis
MBTI	Myers Briggs Type Indicator

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Social media platforms especially Twitter has been used as research platforms in recent years. There is a huge data to study and reflect the real world problems. According to latest Twitter statistics, there are almost 400 million active users and they are sending more than 500 million tweets each day[1]. Hashtags, topics mentioned, interactions between users, follower/following relationships and retweets mirror the real world.

Many companies are trying to analyze their customers' personalities and produce customer specific advertisement strategies based on their customers' interactions in social media [2]. Similarly, politicians are trying to find out voter characteristics, expectations and complaints [3]. However, there are too many users, tweets, videos and photos to process. Thus, data mining techniques are required to deal with these kind of problems, and, mining information from twitter became very popular research topic in recent years.

Researchers have studied on new methods, algorithms and tools to overcome the issues related to personality analysis and extracting personalities from big data. New clustering and classification algorithms were raised, also advanced mathematical and statistical models were developed. Distribution based methods [4], centroid [5] based methods, connectivity [6] based methods and density [7] based models can be given as examples to the clustering methods. Also, vectorization, feature selection and extraction, dimensionality reduction and measuring of cluster quality methods have

[1] http://twitter.com/twitterir

emerged.

In this thesis, the focus is personality analysis using Twitter texts. The study aims to detect personalities applying data mining and machine learning techniques. We have made surveys to the individuals which give us real Big5 personality scores called also as OCEAN which is an abbreviation for the following personality traits [8]:

- ***O****penness*,
- ***C****onscientiousness*,
- ***E****xtraversion*,
- ***A****greeableness*,
- ***N****euroticism* in full.

We applied a collection of data mining methods specifically selected for Turkish grammar features to successfully extract the personality of tweet owners. We used survey scores as ground truth to to assess the quality of our proposed method. We developed an algorithm to measure error rates between our results and the survey scores. In the following sections, we will give details of our approach and results.

1.2 Proposed Methods and Models

In this thesis, our main purpose is to determine the personality traits from Turkish tweets. We aimed to cluster twitter users according to their user characteristics. We adopted Big5 personality traits to our study and we have used surveys in order to find out big five scores of the individuals using Twitter data. Then, we tried to match the scores with the clusters generated using the features of the tweets. We have utilized feature extraction with Turkish part-of-speech tagger called as Zemberek [9], feature reduction with variance thresholds, TF-IDF vectorization and Mikolov's Word2Vec based Word Embedding [10] in this process. After clustering was done, we have calculated silhouette coefficients to measure the quality, applied a dimensionality reduction technique and proposed an algorithm to measure error rates according to real Big5 scores obtained from surveys.

1.3 Contributions and Novelties

Firstly, when we look through previous studies similar to our approach, we saw that almost all of them were done with English tweets. However, we used Turkish tweets to analyze. There were so many characteristic differences between Turkish and English languages. Therefore, we needed to pay attention Turkish language's features and differences.

Secondly, we searched for previous studies and looked for how we can improve the results of them and contribute new approach to the literature. While some studies only focused on features extracted from tweets by using NLP and data mining tools, some others focused on text vectorization techniques and cluster users according to most used words and phrases. In our approach, we used contributions of each and combined their result to reach final vector for each user. In addition, we applied word embedding technique which is Word2Vec model using vectorization results. Also, with different normalization techniques, we tried to see which one was more suitable for our data.

Finally, we clustered users with two different clustering algorithms. However, there was not any method to measure error rates between cluster labels and corresponding OCEAN scores. We proposed a new algorithm to calculate error rates. With this comparison, we managed to compare our results with ground truth values and analyzed results of our method.

1.4 The Outline of the Thesis

The rest of this paper is organized as follows:

- Firstly, we will mention about previous studies which are similar to our problem and how their approaches are.
- In the following section, which is Background section, we will refer to which methods and technologies we used in this thesis.
- After then, we will give a comprehensive detail of our proposed method in the

Proposed Method section and how we adapted the methods and technologies mentioned in Background section to our method.

- In the next section, which is Experiments section, we will present our experiments and their results with small discussions.
- Finally, we will discuss the results and future works in the Conclusion section.

CHAPTER 2

RELATED WORK

Twitter has a crucial effect on different fields such as psychology, marketing, analyzing disasters with event detection, health and politics [11, 12, 13, 14]. Also, it became a very important tool to analyze characteristics of individuals. Researchers try to bring into the open personality traits using tweets, videos, photos, for example. Therefore, some investigations and publications has been done until now.

Querci et al. [15] showed that there is a connection between twitter data and personality traits. They tried to reach which person has which OCEAN value. They claimed that following, followers, and listed counts are enough to guess a users personality. They supported their results with a small rootmean-squared errors.

Kwak et al. studied on Twitter conversations and user relationships [16]. Similarly, Zhang et al. used some features such as tweet texts, hashtags, urls, followings, retweeting relationships to investigate communities among users. They got more successful results compared to random clustering. They mentioned that using new features and clustering algorithms, they may improve their results [17].

Similarly, Ramage et al. [18] studied on characterization of twitter texts using hashtags, emotion and social hierarchies. They tried to classify the twitter users by looking at their most mentioned topics. Their approach was efficient and gave improved performance on user recommendation and users' timeline reranking.

Discovering the relationship in a set of data may be cumbersome. There are some hyper parameters to set before the clustering applied. One of the important hyper parameter is which features of the data is used in clustering algorithm. To this end, some studies [19, 20] shows ways of selecting the appropriate feature maps to suc-

cessfully cluster the dataset. Algorithm on [20] introduce expectation-maximization methods to select the irrelevant features. Although their approach differentiates the best features from the dataset successfully, this approach suffers from the complexity and slows down the computation.

TF-IDF vectorization is very popular approach in clustering. Petrovic et al. [21] used TF-IDF vectorization method to extract the topic of a document. In this study, they represents the frequency of certain words in a format what they called document-term matrix. Their approach significantly improves efficiency both in time and memory. It was one of the early studies that shows reasonable performance improvement using TF-IDF.

In document analysis, one of most common issue is to represent the paragraphs that have different lengths and contexts. This problem leads algorithms to cluster the documents with the same label even if they are completely different in context but has the same word frequencies. Word2Vec sort out these problems. Nalisnick et al. [22] used the model in their document ranking study. They did not directly use the Word2Vec model, they come up with new approach which they both input and output embeddings were holded, thus more accurate results were obtained in the document rankings. They proposed Dual Embedding Space Model which led to understand how similar query term and document.

In light of this information, we followed similar approaches with some additions. On one hand, we extracted features using Twitter texts and reduced them to increase dissimilarities. On the other hand, we applied TF-IDF vectorization to tweets, then we used Word2Vec model in order to get more accurate results. Finally, we combined these two vectors for each user. We will mention details in Background and Proposed Method sections.

CHAPTER 3

BACKGROUND

3.0.1 Big Five Personality Traits: OCEAN

There are many tools to find out personality traits but three of them is most popular such as Myers Briggs Type Indicator(MBTI), the PEN and OCEAN models [23]. We chose OCEAN model to get personalities since its strength and convenience. Also, other models are giving wrong and changeable results [23]. The Big Five personality traits is a classification for different personalities [24]. It is widely accepted and used within academic world of psychology [25]. Personality surveys are applied to people and statistical approaches are used to disclose the traits of people. The OCEAN values correspond to ***O****penness*, ***C****onscientiousness*, ***E****xtraversion*, ***A****greeableness* and ***N****euroticism*.

Individuals who shows Openness are most probably imaginative and instinctive people and take risky actions easily. Conscientiousness are mostly related with work ethics and appears in individuals who are determined, capable and persistent. Extraversion [26] exists in sociable and adventurous human beings. Agreeableness is relevant with naive, trustworthy, altruistic personalities and tends to maintain positive and long term relationship with others [27]. Neuroticism is known to be a emotionally liable and impulsive personality [28].

3.1 Zemberek-NLP

Zemberek-NLP is an open source Turkish natural language processing (NLP) library [9]. It helps us to do basic NLP operations in Turkish such as morphological analysis,

stemming, lemmatization, ambiguity resolution, tokenization, deasciification, spell checker, word suggestion and noisy text normalization etc. Zemberek tokenizator splits the words and punctuations with their corresponding types. It has a feature to convert Turkish text written with only ASCII characters into text with Turkish specific characters. It provides methods for checking if a word is correctly written and can give suggestions for a word. It analyzes given words and sentences, then it gives morphemes of the words and sentences. Beside this, it gives lemmas and stems of the words. It uses some machine learning approaches to detect the disambiguities in the sentences and try to make corrections on them by using best results. Zemberek-NLP's tokenizator splits the words and punctuations according to their types. It has a feature which is deasciification in order to convert Turkish text written with only ASCII characters into text with Turkish specific characters. Moreover, there are some normalization features which are spell checking, spelling suggestion if written wrong and noisy text normalization to detect and correct problematic texts.

3.2 TF-IDF

Term frequency-Inverse document frequency (TF-IDF) is a measure to determine what words in a document is more important than others [29]. It is used to prepare a search query and put these words to the query for data mining, text mining or information retrieval applications. For each word in a document, TF-IDF scores are calculated. Term- frequency (TF) value corresponds to percentage of a word that appears in a document. If a word is more frequent in a document, it is more relevant also. Inverse document frequency (IDF) value is equal to the inverse proportion of the documents that contain the word. The word is more discriminative if it is less frequent among documents.

$$TF(w_i, d) = \frac{t_{f,d}}{n} \tag{31}$$

where

- w_i: the word w_i we are calculating the score for
- $t_{f,d}$: word frequency in document d

- n: total word count in document d

$$IDF(w_i) = log\frac{N}{1+k} \tag{32}$$

where

- w_i: the word w_i, we are calculating the score for
- N: total number of documents
- k: total number of documents where word w_i appears. 1 is added to denominator in order to avoid division-by-zero exception if the k is equal to 0.

$$TFIDF(w_i, d) = TF(w_i, d) * IDF(w_i) \tag{33}$$

3.3 Word2Vec based Word Embedding

Word vectors is high dimensional vectors in order to represent the context of a word. These vectors include numbers which correspond to weights. There are some classical NLP approaches like one-hot encoding and bag-of-words [30]. Their purpose is to find the absence or the presence of the word. However, they do not give any clue related to context of the word. So, we do not get any relationship by using classical approaches. For example, we need a relationship between "Uncle" and "Aunt". In one-hot encoding since they do not have exact vectors, any relationship will not arise, but they are similar with respect to line of descent, so we need similar vectors for these words.

Thomas Mikolov et al. [10] proposed word2Vec algorithm is used to map words to high dimensional vectors. For example, algorithms gives similar vectors for $vec("Windows")$ and $vec(Linux) - vec(IBM) + vec(Microsoft)$.

Suppose we have a dictionary and there are five words: "Uncle", "Aunt", "Man", "Woman", "Child". We get vectors as shown in Table 3.1.

Uncle	1	0	0	0	0
Aunt	0	1	0	0	0
Man	0	0	1	0	0
Woman	0	0	0	1	0
Child	0	0	0	0	1

Table 3.1: One-Hot Encoding

However, if we use Word2Vec, we will get relationship between words as in Table 3.2.

	Uncle	***Aunt***	***Man***	***Woman***	***Child***
Affinity	0.99	0.99	0.02	0.02	0.02
Masculinity	0.99	0.05	0.99	0.02	0.5
Femininity	0.05	0.99	0.05	0.99	0.5
Age	0.6	0.6	0.5	0.5	0.1

Table 3.2: Word2Vec Sample Vectors

As you see from the table, each dimension give a clue about the context of the word. In order to calculate these dimensions and weights, there are two architectures: Continuous Bag-of-Words Model(CBOW) and Continuous Skip-gram Model. Both of them have input layer, hidden layer and output layer. However, they become different according to what they need to find whether it is context of the word or word itself.

For example,

process of finding out patterns in large data sets

focus word: *patterns*
context of the word: *process of finding out* and *in large data sets*

3.3.1 Continuous Bag-of-Words Model

In the CBOW architecture, as shown in 3.1a, the input layer includes the context words and uses single hidden layer and output layer. If we apply CBOW to our example with the context words, “finding”, “out”, “in”, “large”, the model will return us “patterns” word.

3.3.2 Continuous Skip-gram Model

In the Skip-gram architecture, as shown in 3.1b, the input layer includes the word and uses single hidden layer and output layer. Output layer includes the context of the word. For example, if we apply skip-gram to our example with the word, “patterns”, the model will return us, “finding”, “out”, “in”, “large”, words.

3.4 Normalization Methods

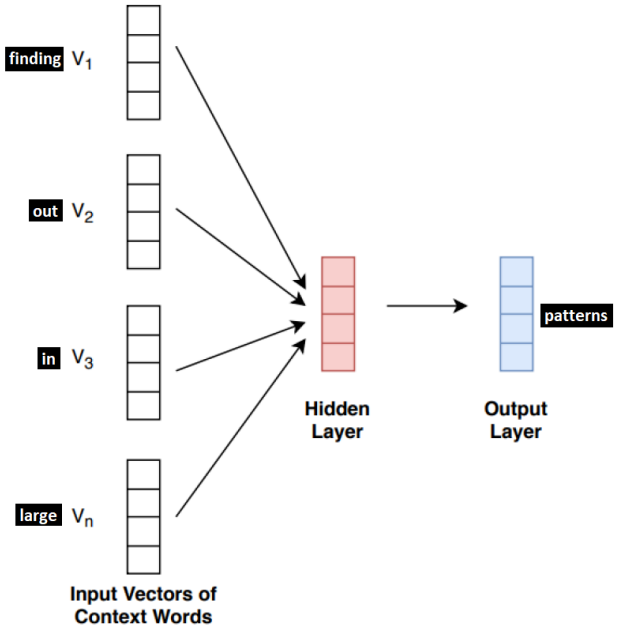
3.4.1 Standardization Scaling

For each column of the data, means and standard deviations are calculated. Then, z-score of each sample x is calculated by subtracting the means and dividing by the standard deviation. Finally, we get 0 as new mean and 1 as new standard deviation for each columns.

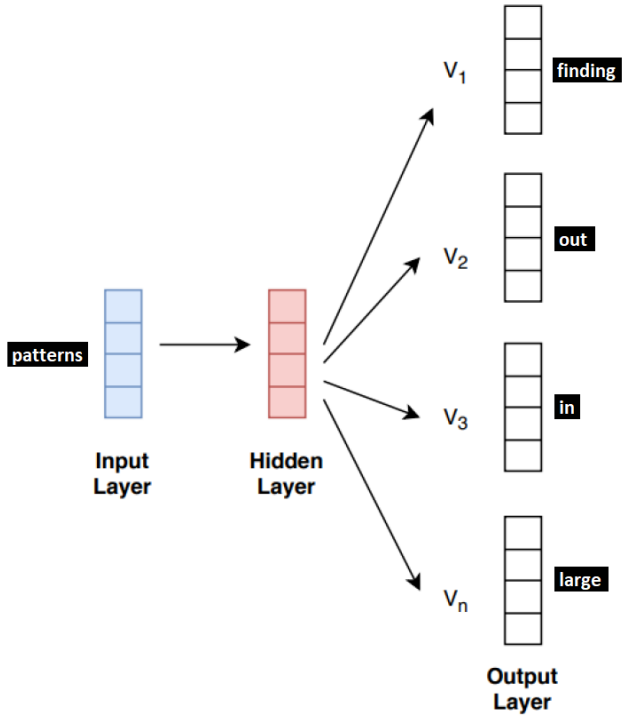
$$z\_score(x) = \frac{x - \mu}{\sigma} \tag{34}$$

where

- μ: the mean of the column where x is located
- σ: the standard deviation of the column where x is located



(a) CBOW



(b) Skip-Gram

Figure 3.1: Word2Vec Architectures. This figure is adapted from [1]

3.4.2 Robust Scaling

In the robust scaling method, interquartile range(IQR) is used for scaling [31]. Also, it is robust against outliers. For each column of data, median and first and third quartiles are calculated. Then, scaling score is calculated by removing median and scaling the data between first and third quartiles. Contrary to standard scaler, outliers does not affect much the resulting data.

3.4.3 Discretization

In the discretization, data is divided into sub-intervals according to predefined threshold values [32]. New and constant values are assigned to these intervals. For example, if we want data to discretized by a value, a sample choice for a threshold may be the mean of the data values for convenience. After that, anything below the threshold set to zero and others set to one for discretization.

3.5 Clustering Algorithms

3.5.1 K-means Clustering

K-means clustering is one of the unsupervised clustering algorithms to assign labels to the uncategorized data [33]. The algorithm needs predefined k value in order to specify the resulting number of groups in the data. The algorithm uses iterative approach to assign labels. Object similarity metrics are used while clustering the data. There are different feature similarity metrics such as Euclidean distance, Cosine distance, Manhattan distance etc.

$$EuclideanDistance(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2} \quad (35)$$

$$ManhattanDistance(x, y) = \sum_{i=1}^{n} \mid x_i - y_i \mid \quad (36)$$

$$CosineDistance(x, y) = \frac{\sum_{i=1}^{n} x_i * y_i}{\sum_{i=1}^{n} x_i^2 * \sum_{i=1}^{n} y_i^2} \tag{37}$$

According to the need, one of these distance functions is used to calculate cluster centroids and assign labels according to centroids. Algorithms iterates until there is no change in the k centroids.

3.5.2 Agglomerative Clustering

Agglomerative clustering is a bottom-up hierarchical clustering method [34]. Initially, each sample is treated as bottom-up which means each sample treated as a single cluster then in each step clusters more similar to each other are merged until one big cluster created. The process is represented as a tree which has a special name called as dendrogram.

There are different linkage criterions which are single, average and complete. With these criterions, the algorithm decides how it will merge the clusters by looking the distance between clusters. Two clusters are merged, if selected criterion minimizes. Single linkage looks for minimum distance between clusters. Complete linkage looks for maximum distance between clusters. Average linkage looks for average distance between clusters.

3.6 Data Visualization: t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a dimensionality reduction technique which aims to visualize and map similar data points in a concise manner [35]. t-SNE algorithm is achieved this by transforming high dimensional representation of the target data to a low dimensional easy to visualize vector space. This calculation is performed by finding the probability of similarities between target and input spaces. For each input point in the high dimensional data, one candidate point would be chosen in the low dimensional representation under Gaussian distribution assumption. Differences between input and output points needs to be minimized to reach a certain level. t-SNE minimizes these sum of differences using gradient decent

method.

In simpler terms, t-SNE unfolds the implicit structure of the high dimensional data, group the related features and transfer to a set of under-sampled data structure without losing the general structure.

CHAPTER 4

PROPOSED METHOD

In this thesis, our main purpose was to predict users' personalities using their twitter data. We proposed a method as it is depicted in Fig.4.1. Our proposed method includes data collection, data preprocessing, data cleaning, feature extraction, feature reduction, normalization, vectorization and clustering phases, which are explained in more detail below.

4.1 Data Collection

During data collection, firstly, we made a survey to get OCEAN values and tweets. Almost 40 volunteers participated our survey and we used their OCEAN results as a ground truth in our study. With the survey, we collected individuals' tweets and big five personality scores between 0 and 100. The tweets included time, date and text information with hashtags, emotions, punctuation etc. Afterwards, we collected tweets of 2000 random users writing in Turkish using tweepy[1] API. We filtered results by time such that we identified the users who tweeted between 2014-2019. Then, we pulled their tweets with tweepy. Then, we merged data from survey users with data from random collected users to follow further processing.

4.2 Data Cleaning and Preprocessing

Almost all Twitter texts include noisy data such as informal, misspelled and slang words, URLs, mentions, meaningless texts, multiple spaces etc. Therefore, all data

[1] https://www.tweepy.org/

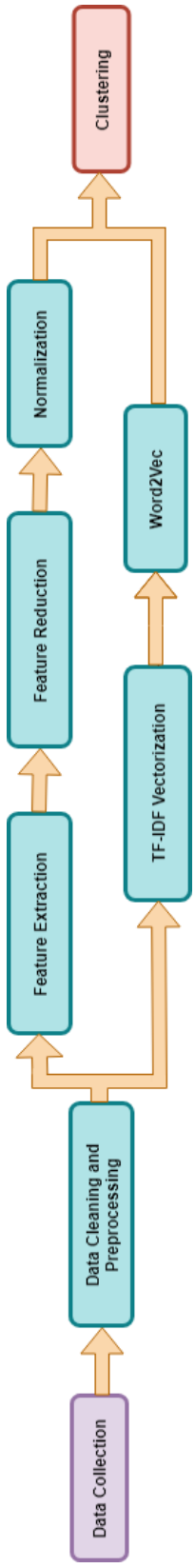


Figure 4.1: Flow Chart of the Proposed Method

needs to be preprocessed before further processing. Applied preprocessing steps can be summarized as follows:

Removing Multiple Spaces, Newlines, Meaningless Texts and Stopwords: Almost all tweets include more than one spaces and newlines. We removed them to get a straight sentence. Thus, we was able to implement further preprocessing steps correctly and eliminate any programming errors while reading the data. Tweets that have leading or trailing white spaces are trimmed in order to have straight text body. Some tweets include meaningless data. Since meaningless data may lead to incorrect results, we need to remove them. Mentions, hashtag symbol "#", *RT* keyword, URLs are useless, so, they were removed from the texts as well. Hashtag keyword was kept since it may include a meaningful phrase related to the tweet or to the user. Also, we neglected the users if all their tweets include *RT* keyword or URLs only. According to our assumption, these users does not provide us any information related to personality. When we looked at them in detail, we saw that generally they included advertisements, news etc.

Additionally, We cannot extract any meaningful results with the common word syntaxes as they occur mostly in every sentences, so Turkish stopwords[2] were removed from tweets in order to improve accuracy for TF-IDF weighting and Word2Vec embedding.

Noisy Text Normalization: Twitter users do not pay attention to grammar and formality in their tweets. Almost all of tweets have incorrect, slang and informal words. Therefore, before applying NLP operations, we needed to normalize the tweets by removing or replacing incorrect words with corrected ones. For example, user can write a tweet as "*Yrn okula gidicem*". In this Turkish tweet, all words are written incorrectly and informally. If we try to resolve the sentence into its elements without normalization, Zemberek NLP may not find correct tags. To this aim, we have used Zemberek's SpellChecker tool such that incorrect words were detected and replaced by the best alternatives. For example, if we give the tweet to Zemberek normalization tool, we get "*Yarın okula gideceğim*". During this normalization process, deasciification is also done. If a tweet is written with non-Turkish characters, words are corrected

[2] https://www.ranks.nl/stopwords/turkish

with corresponding Turkish characters again by Zemberek's deasciification tool. For example, "*Bilgisayarim cok isiniyor.*" is written with non-Turkish characters. Zemberek NLP deasciified and converts it into "*Bilgisayarım çok ısınıyor.*". Thus, we got grammarly corrected tweets and part of speech tagging operation gave more accurate results to us.

Afterwards, we applied other preprocessing techniques for different purposes such that both creation of feature and word vectors required a few more preprocessing steps to get more specific outcomes.

4.3 Vector Construction

As depicted in Figure 4.1, after data cleaning and preprocessing, we applied two different paths for vector construction, and then combined the resulting features. As the first path, we extracted features from the use of language and timestamp of the tweet. As the second path, terms within the texts are used as features. The collection of these features enable us to extract more information that the plain text can not unveil.

4.3.1 Feature Extraction

As listed in Table 4.1, we have constructed 48 features. These features are related to use of languages and timestamp of the tweet. Some of them include values related to the non-Turkish words in tweets, emoticons, numerals and punctuations etc. In this step, we count the number of occurrences of them in tweets and calculate the averages for each user. These were the temporal features, we removed them from tweets later.

Some features require extra treatment:

- Tweet timestamp is one of them. Tweet times are categorized as *Morning*, *Afternoon*, *Evening* and *Night* and can be represented with 4 columns and using this, we needed to calculate average tweet time for each user. Since these times are circular, one-hot encoding is not suitable to represent. The one hot encoding

does not account circularity of the data such that night to morning distance in 4 column representation is much greater than morning to evening differences. This leads clustering algorithms to extract completely wrong relationships. We need to take account of the circular nature of the time in order to cluster the users correctly. So, we employed a two hot encoder as follows:

  - Tweet time is *Morning* →Add 1 to *Morning* and *Night* Columns
  - Tweet time is *Afternoon* →Add 1 to *Morning* and *Afternoon* Columns
  - Tweet time is *Evening* →Add 1 to *Afternoon* and *Evening* Columns
  - Tweet time is *Night* →Add 1 to *Evening* and *Night* Columns

- After, the other extra treatment was done for emoticons since tweepy returned us emoticons with their hexadecimal representations. Firstly, we replaced them with their real emoticon characters. Afterwards, we looked for all emoticons which can be useful in personality analysis. We used Unicode,Inc. emotion table[3] and we selected most popular ones and created a dictionary to group related emoticons together. Finally, We separated them as smiling, affection, tongue, neutral, unwell, negative, romantic, fingers, activity, sport and plant emoticons. Each group is refer to corresponding column in user vector. If a tweet includes an emoticon, then we look up our dictionary, obtain its group and add one to corresponding emoticon column. Thus, we get how much and which emoticons are used by the user.

Then, the average values for each temporal feature is calculated. In this way, Euclidean distance can be used for as the distance measure between temporal features.

Part of Speech(PoS) Tagging: Another important set of features is obtained through Part of Speech (PoS) tags used within the texts. To this aim, we determine PoS of the words in the tweets through Zemberek-NLP, and average frequency of each PoS tag is recorded as the weight of the feature.

[3] http://unicode.org/emoji/charts/full-emoji-list.html

4.3.2 Feature Reduction

Before clustering, we have applied feature reduction as follows: As given in Figure 4.2, we have plotted the sorted distribution of the values for each feature for 1000 users. As seen in the plots, values for some of the features are nearly the same for most of the users, hinting that these features are not discriminative for clustering and lead to almost all users drop the same cluster(s). Hence, we filtered the features according to a variance threshold with scikit-learn's VarianceThreshold implementation[4]. We calculated variances for all features and selected our threshold value as 0.01. Hence, the features having a low-variance (i.e., lower than 0.01) are removed. After feature reduction, the following 20 distinguishing features are left: *Morning*, *Afternoon*, *Evening*, *Night*, *Word*, *Verb*, *Noun*, *Punctuation*, *Adjective*, *Adverb*, *Negative*, *Numeral*, *Determiner*, *Conjunction*, *Pronoun*, *Incorrect*, *Plural*, *Full Stop*, *Smiling Emoji*, *Negative Emoji*.

4.3.3 Normalization

We have applied different normalization techniques on our feature vectors, which are standard scaling, robust scaling and discretization. The effect of each normalization technique on the clustering performance was analyzed in order to determine the best one for our problem. We applied standart and robust scaling as it is in scikit-learn library[5]. In case of discretization, we discretized our data to 4 using percentile values in statistics as shown in Algorithm 1. For all feature columns, we calculated first

[4] http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.VarianceThreshold.html
[5] http://scikit-learn.org/stable/modules/preprocessing.html

Table 4.1: Range of Features

Feature	***Range***
Morning	0.0 - 1.0
Afternoon	0.0 - 1.0
Evening	0.0 - 1.0
Night	0.0 - 1.0
Case Ratio	0.58 - 1.0
Word	3.4 - 20.0
Verb	0.0 - 3.0
Noun	1.2 - 11.0
Punctuation	0.0 - 5.38
Adjective	0.0 - 2.5
Adverb	0.0 - 1.5
Numeral	0.0 - 2.06
Determiner	0.0 - 1.0
Post Positive	0.0 - 0.726
Duplicator	0.0 - 0.059
Conjunction	0.0 - 1.0
Interjection	0.0 - 1.0
Pronoun	0.0 - 1.23
Question	0.0 - 1.0
Incorrect	0.0 - 6.58
Negative	0.0 - 1.0
Plural	0.0 - 3.52
Present Time	0.0 - 1.0
Future Time	0.0 - 0.5

Feature	***Range***
Past Time	0.0 - 0.88
Narrative Time	0.0 - 0.55
Progressive Time	0.0 - 1.5
Condition	0.0 - 0.5
Imperative	0.0 - 1.0
Necessity	0.0 - 0.42
Ability	0.0 - 0.5
Negative Ability	0.0 - 0.33
Question	0.0 - 1.0
Exclamation	0.0 - 0.67
Ellipsis	0.0 - 0.84
Full Stop	0.0 - 0.91
Non-Turkish Words	0.0 - 1.0
Smiling Emoji	0.0 - 1.0
Affected Emoji	0.0 - 0.32
Tongue Emoji	0.0 - 0.16
Neutral Emoji	0.0 - 0.32
Unwell Emoji	0.0 - 0.2
Negative Emoji	0.0 - 0.06
Romantic Emoji	0.0 - 0.3
Fingers Emoji	0.0 - 0.56
Activity Emoji	0.0 - 0.036
Sport Emoji	0.0 - 0.09
Plant Emoji	0.0 - 0.14

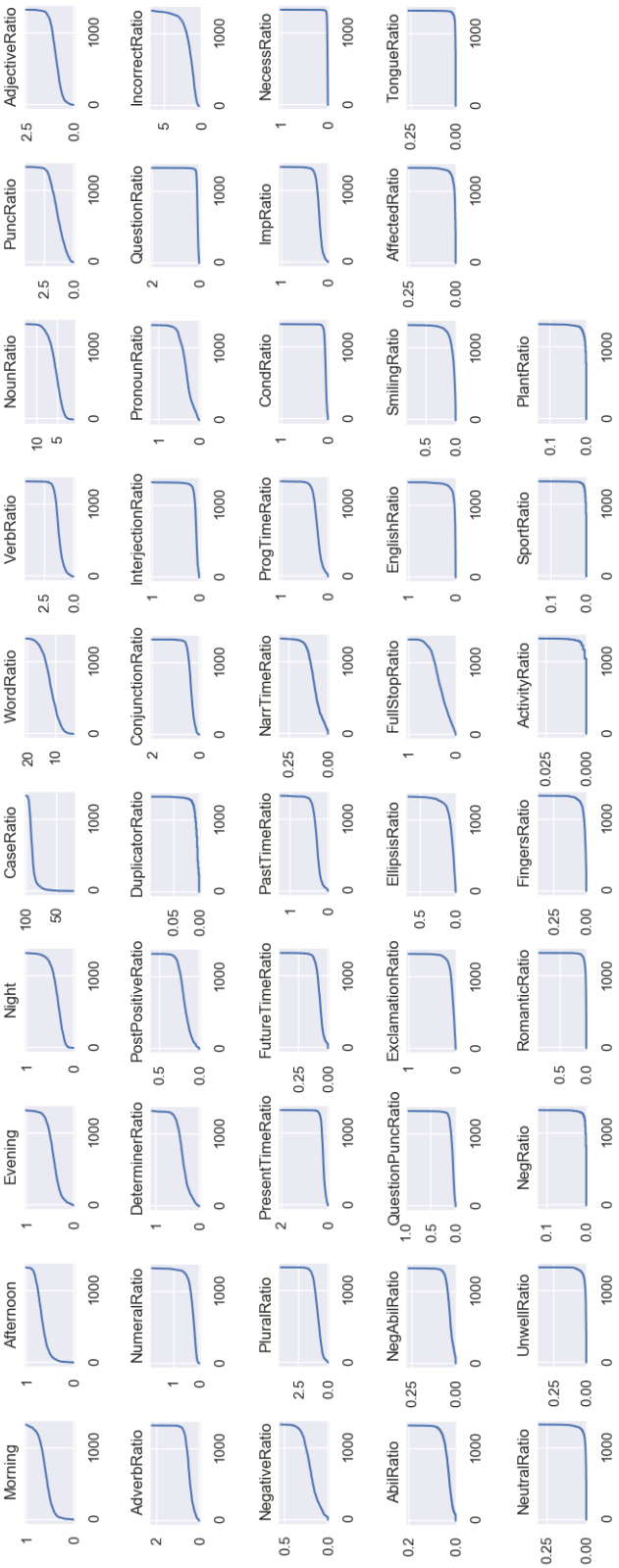


Figure 4.2: All Sorted Linguistic Feature Values for All Users

quartile, mean and third percentiles using numpy library in python [36].

Algorithm 1: Our Discretization Algorithm

Input: $FeatureColumn$

Output: $DiscretizatedColumn$

$firstQuartile$ = $CalculateFirstQuartile(FeatureColumn)$;
$mean$ = $CalculateSecondQuartile(FeatureColumn)$;
$thirdQuartile$ = $CalculateThirdQuartile(FeatureColumn)$;
$current$ = 0;
for $data$ *in* $FeatureColumn$ **do**
    if $data$ <= $firstQuartile$ **then**
        $DiscretizatedColumn$[$current$] = 0.0;
    else if $data$ > $firstQuartile$ *and* $data$ <= $mean$ **then**
        $DiscretizatedColumn$[$current$] = 0.25;
    else if $data$ > $mean$ *and* $data$ <= $thirdQuartile$ **then**
        $DiscretizatedColumn$[$current$] = 0.75;
    else
        $DiscretizatedColumn$[$current$] = 1.0;
    end
    ++$current$;
end

4.3.4 TF-IDF Weighting

The second path, we used the terms with the tweets of a user. Before applying vectorization algorithms, we needed extra preprocessing steps. Firstly we did lemmatization and tokenization on tweets.

Lemmatization: Using Zemberek-NLP, we replaced the words with their lemmas in the tweets so the words with different forms are represented in the same manner for word vectors. We used lemmas of words instead of stems because we required their real dictionary representations [37]. If we look at the "*kitabımız*" word, stemming leads to "*kitab*", but lemmatization results in "*kitap*" which is more suitable than the stems of the words.

Tokenization: Another critical step of preprocessing was splitting the sentences into tokens. Tokens might be words, phrases or even whole sentences or paragraphs. Firstly, we splited the words by using white spaces. Another tokenization was applied during TF-IDF weighting with n-grams to detect important phrases used in tweets.

TF-IDF vectorization is used to find out more important words or phrases in documents [38]. In our case, documents for each user were created by merging all tweets of the user. As the first step, in order to find the most important words or phrases used within a users all crawled tweets, we calculate TF-IDF values for 1-grams, 2-grams and 3-grams. Secondly, for each user, top used words or phrases is extracted from the resulting matrix created by vectorization as given in Table 4.2. As you see, applying TF-IDF to the tweets, we can have an idea about which topics the user was used. There are topics such as sports, politic, religion, economy etc.

4.3.5 Word2Vec based Word Embedding

Finally, Word2Vec embedding of the top words (i.e., with the highest TF-IDF values) was constructed using Word2Vec[6] implementation of Gensim, which is free python library for topic modelling [39]. We constructed Word2Vec model with 38 dimensionalities. The dimension number was found as dividing 1923 user number by 50 which is interval. Most frequently used interval is between 100-300, but it depends on the particular task [40]. In our case, we tried different interval values, but we obtained best results with interval 50. Also, window size was chosen 7 which is the distance between the present and guessed word within the tweet. Window size was tried with different settings and 7 gave the best results in our evaluations. Minimum word count which is used to ignore words with their frequency below the specified value [41] is sat to 3. We used TF-IDF output words as input while constructing the model. The model includes a vector for each word in the corpus. Vectors for top words are added and final word vector is obtained for each user. In other words, each user is represented as a word vector. Finally, user vectors with 38 dimensions were constructed through concatenation of Word2Vec representations of the top words.

[6] http://radimrehurek.com/gensim/models/word2vec.html

Table 4.2: N-Grams with Highest TF-IDF scores

User	***Important Words***
1	galatasaray, maç, fatih terim, terim, futbol, fenerbahçe, taraftar, forvet, ...
2	asgari, asgari ücret, ücret, oğul, baba, sokak, istanbul, genç, ...
3	vatan, parti, abd, doğu, pkk, devrim, ulusal, atatürk, savaş, cumhuriyet, ...
4	trabzonspor, maç, gol, trabzon, fenerbahçe, şampiyon, taraftar, puan, futbol, ...
5	abd, opec, faiz, fed, piyasa, form, anlama, banka, enflasyon, dolar, ticaret, ...
6	günaydın, tertemiz, gör kadar, türkan, yeniden, etkileşim, böyle güzel, istem, dolan, ...
7	müslüman, islam, hesap, yerli milli, milli, kuran, abdülhamid, rabbi, cami, ...
8	hasta, depo, hastalık, ilaç, sağlık, tedavi, şeker, hücre, mücadele, yardım, doktor, ...
9	hisset, mutlu, hayal kırık, ağla, kalp, hayal, kötü, sevgi, boşluk, acı, yalnız, duygu, ...
10	google, site, dijital, içerik, form, sunum, link, test, mobil, reklam, web, sonuç ...
11	kitap, konferans, öğretmen, form, eğitim, okul, başarı, ingilizce, dijital, öğrenci, ...
12	cuma, aşk, millet, ihanet, yürek, şiir, gül, namaz, bereket, hayır cuma, mümin, ...
13	robot, pro, teknoloji, kdv, vergi, hız, ekran, harika, phone, performans, vodafone, ...
14	pil, yeni nesil, incele, telefon, kutu, android, nesil, apple, video, detay, kamera, ...
15	son dakika, ekonomi, açıkla, müjde, bakan, kredi, banka, indirim, sektör, ihracat, ...
16	köpek, papağan, kadın, sanat, istanbul, kedi, hayvan, görüntü, poşet, şiddet, araştır, ...
17	maden, kaza, işçi, fuar, deprem, geçit, yaya, tohum, dost, istismar, soma, tren, ...
18	vatandaş, belediye, ziyaret, esnaf, mahallesi, merkez, hikmet, ilçe, kültür, halk, ...
19	lira, döviz, şekil, ekonomi, ekmek, banka, türk, merkez banka, öde, borç, ...
20	eğitim, önem, toplum, geliş, sağlık, ekonomik, nüfus, öğrenci, üniversite, başarı, ...
⋮	⋮

4.3.6 Composition of Feature and Word Vectors

After obtaining features on both paths, i.e., extracted features and Word2Vec representations of the top terms in the tweets, the two sets of features are concatenated to construct the final vector. As the result of this step, we have obtained a matrix with 58 columns for 1923 users.

$$FinalVector = \left[FeatureVector \ WordVector \right] \tag{41}$$

In this point, what stages the user tweets have gone through can be shown as below:

Firstly, we get the average reduced feature counts from the sample user. Then, we applied different normalization techniques as shown is Table 4.3.

Feature Vector (with Discretization) = [0, 0.25, 1, 1, 0.75, 0, 0.75, 1, 0.75, 0, 0, 0.25, 0, 0, 0, 0.25, 0.25, 0, 0, 0]

Secondly, TF-IDF weighting was applied to the users' tweets.

Top words of the sample user extracted with TF-IDF:
hukuk, *karar*, *tazminat*, *dava*, *mahkeme*, *avukat*, *daire*, *ceza*, *öde*, *sözleşme*, *kanun*, *süre*, *icra*, *savcı*, *tarih*, *dolandırıcı*, *hakim*, *imar*, *hüküm*, *son dakika*, *yargı*, *esas*, *başvuru*, *teklif*, *adliye*

Thirdly, we got the corresponding vector for each top word from Word2Vec model then we added them. Thus, we obtained word vector of the sample user extracted with Word2Vec.

Word Vector = [5.57, 12.84, -16.87, 12.09, 9.84, -11.29, 8.66, -29.17, -7.34, 13.40, -1.09, -5.78, 2.39, 16.35, 10.55, 12.31, -1.32, -7.41, 4.53, -10.84, -4.24, 8.05, -6.87, 20.16, 2.34, 5.02, -0.96, 1.21, -15.80, -2.77, 9.42, -12.58, -1.77, -11.92, 3.76, 1.31, -9.47, -11.09]

	Initial	*Standard Scaler*	*Robust Scaler*	*Discretization*
Morning	0.46	-0.76	-0.64	0
Afternoon	0.58	-0.58	-0.54	0.25
Evening	0.53	0.76	0.64	1
Night	0.41	0.58	0.54	1
Word Ratio	12.13	0.18	0.06	0.75
Verb Ratio	0.66	-1.73	-1.47	0
Noun Ratio	6.17	0.56	0.39	0.75
Punctuation Ratio	2.55	1.20	1.04	1
Adjective Ratio	0.98	0.06	0.03	0.75
Adverb Ratio	0.21	-1.57	-1.40	0
Determiner Ratio	0.12	-1.93	-1.62	0
Post Positive Ratio	0.18	-0.22	-0.19	0.25
Conjunction Ratio	0.24	-0.90	-0.67	0
Pronoun Ratio	0.05	-1.85	-1.51	0
Incorrect Ratio	0.52	-1.02	-0.87	0
Negative Ratio	0.15	-0.58	-0.49	0.25
Past Time Ratio	0.26	-0.27	-0.24	0.25
Full Stop Ratio	0.09	-1.31	-0.95	0
Smiling Ratio	0.04	-0.57	-0.33	0
Negative Ratio	0	-0.35	0	0

Table 4.3: Feature Values of Sample User

Final Vector = [0, 0.25, 1, 1, 0.75, 0, 0.75, 1, 0.75, 0, 0, 0.25, 0, 0, 0, 0.25, 0.25, 0, 0, 0, 5.57, 12.84, -16.87, 12.09, 9.84, -11.29, 8.66, -29.17, -7.34, 13.40, -1.09, -5.78, 2.39, 16.35, 10.55, 12.31, -1.32, -7.41, 4.53, -10.84, -4.24, 8.05, -6.87, 20.16, 2.34, 5.02, -0.96, 1.21, -15.80, -2.77, 9.42, -12.58, -1.77, -11.92, 3.76, 1.31, -9.47, -11.09]

Finally, the sample user is represented with final vector which is created with concatenating feature and word vectors.

4.4 Clustering

As we mention before, we have applied a standard survey to obtain OCEAN scores for users. The survey gave five percentage scores of each one of the OCEAN dimensions. For each dimension, we have discretized each score into 4 blocks as 0-25%, 25-50%, 50-75% and 75-100%. Thus, for 5 OCEAN scores there are 20 maximum number of possibilities in order to cluster people.

In order to find groupings, we applied two clustering algorithms, K-means and Agglomerative clustering. Clustering quality is measured with silhouette-coefficient. We used scikit-learn[7] library for the implementation of the methods. In K-means algorithm, k-values were selected as 4, 8, 12, 16 and 20 to determine how silhouette coefficient is changing with the k value. In order to accelarate the convergence, kmeans++ initialization method in scikit-learn library was used. k-means++ chooses initial centroids in a smart way, instead of the random choosing [42]. For similarity calculation, euclidean distance metric is used. Agglomerative clustering is applied with 4, 8, 12, 16 and 20 number of clusters. The effect of different linkage options, including single, average and complete linkage, is tested, and the best result is obtained with average linkage. In addition, we used euclidean distance as the distance metric.

[7] http://scikit-learn.org/stable/modules/clustering.html

CHAPTER 5

EXPERIMENTS

5.1 Error Rate Calculation

As stated previously, we have about 40 users with OCEAN scores obtained from surveys. Their user vectors and discretized OCEAN scores were used as the ground truth in order to calculate the error rates of the clustering results. In the rest of the thesis, we call these survey users as the *base users*. To this aim, we have used the following simple approach (Algorithm 2): Firstly, we followed our base users and looked at which one is in which cluster and we found the highest k value (i.e., the number of clusters) such that each base user has a base neighbor in the same cluster. In other words, if there is only one base user in a cluster than we backtracked to the previous k value and finished clustering. Thus, we had a dictionary for our base users with their cluster labels.

Then, we calculated the error rates for each OCEAN dimension separately as follows: we consider the base users in the same cluster, and for each user, we calculate the average score of its neighbors. Then, the average value is subtracted from the selected users score. The result is the error rate for that base user. Then, we have repeated the process for all base users in the same cluster. Base error rate of a cluster, for the corresponding OCEAN dimension, is calculated as the average error rate of all base

users in the cluster.

Algorithm 2: Calculating Error Rates for each OCEAN dimension

Input: $Dictionary, Dimensions, Users$

Output: $ErrorRates$

$ErrorRates$ = {};
for *each* $oceanDimension$ ***in*** $dimensions$ **do**
    $dimensionError = 0$;
    for *each* $(clusterLabel, users)$ ***in*** $dictionary$ **do**
        $localErrorRate$ = 0;
        $totalValue$ = $sum(surveyResults[all, oceanDimension])$;
        $localUserSize$ = $len(users)$;
        for *each* $user$ ***in*** $Users$ **do**
            $userOceanScore$ = $surveyResults[user, oceanDimension]$;
            $difference$ = $totalValue - userOceanScore$;
            $localErrorRate$ = $abs(\frac{difference}{localUserSize-1} - columnValue)$;
        end
        $dimensionError$ += $localErrorRate$;
    end
    $dimensionError$ = $\frac{dimensionError}{totalNumberOfClusters}$;
    $ErrorRates[oceanDimension]$ = $dimensionError$;
end

5.2 Experiments

We have performed 6 different experiments. In all experiments, TF-IDF weighting was applied to the crawled tweets to be able to study on important words. Then, we tried to find out the effects of feature reduction, different normalization techniques and Word2Vec algorithm on the results. We wanted to see results with contributions of all features, then we applied feature reduction with VarianceThreshold. We have applied Standard, Robust and Dicscretization Scaling With normalization techniques. Also, after TF-IDF weighting is applied, we looked at how Word2Vec affected the silhouette scores. Then, two different clustering algorithms were applied to the pre-

processed data. We will give all details about experiments as follows:

5.2.1 Experiment 1: Analysis on the Effect of Normalization and K-means Clustering on TF-IDF Weighting

In the first experiment, we wanted to see the contributions of all features. Firstly, we applied different normalization techniques to our feature vectors and TF-IDF weighting to our crawled tweets. Secondly, We chose K-means as clustering algorithm and calculated silhouette scores under this circumstances. As you see in Figure 5.1, silhouette scores are near 0, and the results are not satisfying. Finally, when we looked at the t-SNE projection of this experiment shown in Figure 5.7, we saw that there is overlapping between clusters and there is no separation among them.

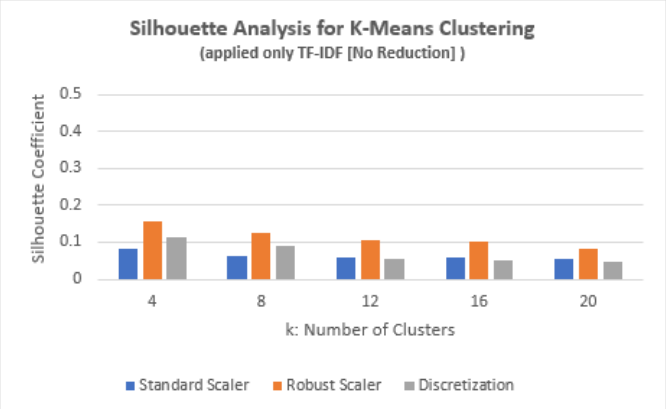


Figure 5.1: Experiment One: Silhouette Scores Using K-means Clustering with TF-IDF weighting on the tweets without Feature Reduction and Word2Vec embedding

5.2.2 Experiment 2: Analysis on the Effect of Normalization and K-means Clustering on TF-IDF Weighting and Word2Vec based Word Embedding

In the second experiment, we evaluated the silhouette scores with TF-IDF and Word2Vec vectorization but without feature reduction again. Firstly, different normalization techniques were applied to the feature vectors. Also, we applied TDF-IDF weighting and Word2Vec to the crawled tweets, then we combined feature and word vectors. Secondy, K-means was used as clustering algorithm, we got the scores as shown

in Figure 5.2. As you see, silhouette scores are much higher than the first experiment scores. We understand that Word2Vec on clustering makes a positive difference. Finally, we reduced dimensionality to 5 using t-SNE projection as shown in 5.8. Although, silhouette scores increased, clusters were still overlapped. Nevertheless, when we compared the t-SNE results between Experiment 1 and 2, there was an huge improvement due to the Word2Vec.

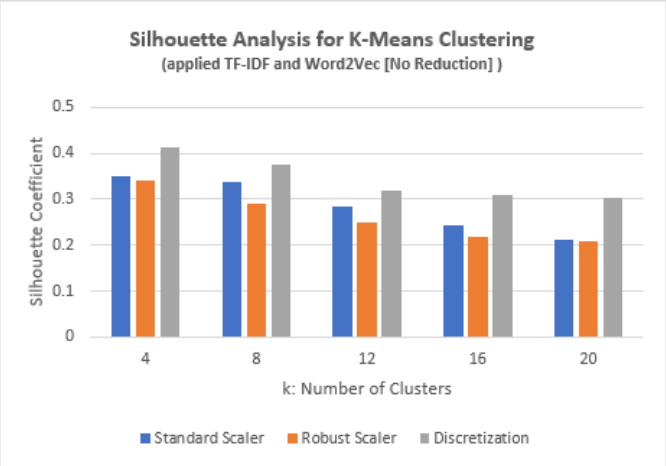


Figure 5.2: Experiment Two: Silhouette Scores Using K-means Clustering with TF-IDF Weighting and Word2Vec based Word embedding on the tweets without Feature Reduction

5.2.3 Experiment 3: Analysis on the Effect of Normalization and K-means Clustering on TF-IDF Weighting and Feature Reduction

In the third experiment, only TF-IDF vectorization was applied to the crawled tweets of users. Besides, we have applied different normalization techniques to our reduced feature vectors and combined them with the TF-IDF vectors. Then, K-means was used as clustering algorithm. As a result of this experiment, silhouette coefficient around 0 has been obtained, whose results are shown in Figure 5.3. If we look at the t-SNE projection (Figure 5.9) for this experiment, there will be so many overlapping clusters. On one hand, we can see that compared to Experiment 1 there is a small improvement due to the Feature Reduction. On the other hand, due to absence of Word2Vec t-SNE results are not satifying compared to Experiment 2.

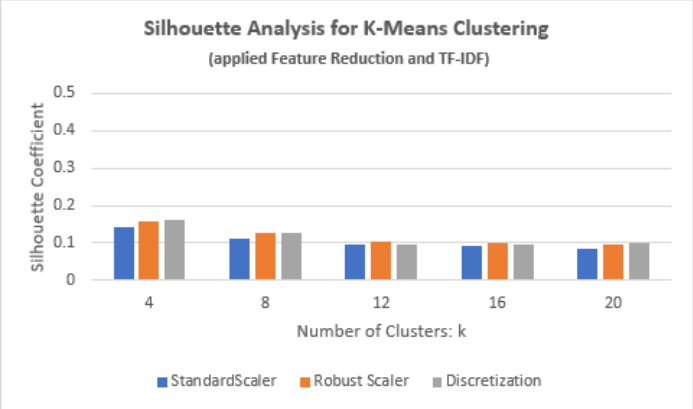


Figure 5.3: Experiment Three: Silhouette Scores Using K-means Clustering with TF-IDF Weighting and Feature Reduction on linguistic feature vectors without Word2Vec embedding

5.2.4 Experiment 4: Analysis on the Effect of Normalization and K-means Clustering on TF-IDF Weighting, Feature Reduction and Word2Vec based Word Embedding

In the fourth experiment, firstly, TF-IDF vectorization was applied to crawled tweets, and then, resulting vectors were passed to Word2Vec as an input. Secondly, feature reduction was applied to feature vectors. After that, K-means is used as clustering algorithm. We have obtained higher silhouette scores than the third experiment, and the results are shown in (Fig.5.4). When we analyzed the results of t-SNE projection as shown in Figure 5.10, we concluded that intra-cluster distances have decreased and inter-cluster distance have increased. Besides, there are so small overlapping areas between clusters.

5.2.5 Experiment 5: Analysis on the Effect of Normalization and Agglomerative Clustering on TF-IDF Weighting and Feature Reduction

In the fifth experiment, we applied feature reduction to our feature vectors and only TF-IDF weighting was applied to the crawled tweets. Unlike the previous experiment, agglomerative clustering was applied as clustering algorithm. For this experiment, small silhouette scores were obtained as plotted in Figure 5.5). Also, we plotted the

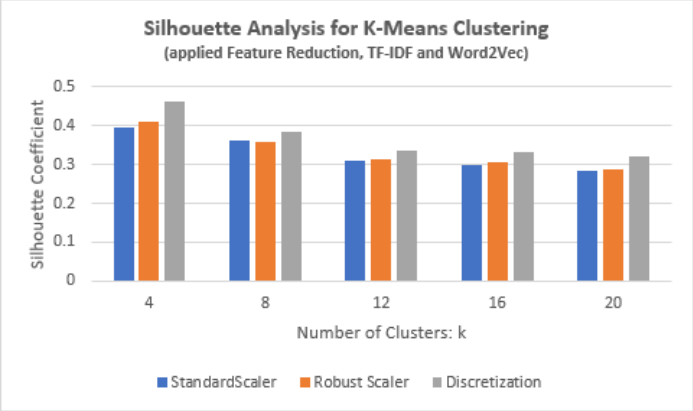


Figure 5.4: Experiment Four: Silhouette Scores Using K-means Clustering with TF-IDF Weighting and Word2Vec embedding on the tweets and Feature Reduction on the linguistic feature vectors

t-SNE projection results as shown in Figure 5.11, and we concluded that there is again overlapping between clusters and the identified clusters has not been separated from each other.

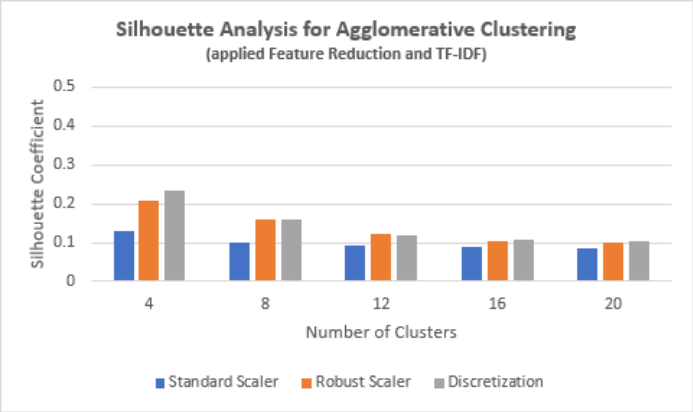


Figure 5.5: Experiment Five: Silhouette Scores Using Agglomerative Clustering with TF-IDF Weighting and Feature Reduction on the linguistic feature vectors without Word2Vec embedding

5.2.6 Experiment 6: Analysis on the Effect of Normalization and Agglomerative Clustering on TF-IDF Weighting, Feature Reduction and Word2Vec based Word Embedding

Finally, in the sixth experiment, feature reduction was applied to the feature vectors and Word2Vec based word embedding was applied to resulting vectors due to TF-IDF weighting. Similar to previous experiment, agglomerative clustering was used instead of K-means. Silhouette scores was obtained as Figure 5.6. If we look at the differences, K-means gives a bit better scores with small changes. Then, t-SNE projection was applied to the resulting clusters with 5 as a number of clusters as plotted in Figure 5.12. As a result, compared to previous experiment, reasonable clusters were obtained. When we looked at in detail, agglomerative clustering led to one huge cluster and 4 smaller clusters. However, our aim is divide users as much as possible. In other words, we do not want to attach users to the each other since there is so many personality possibilities to find out.

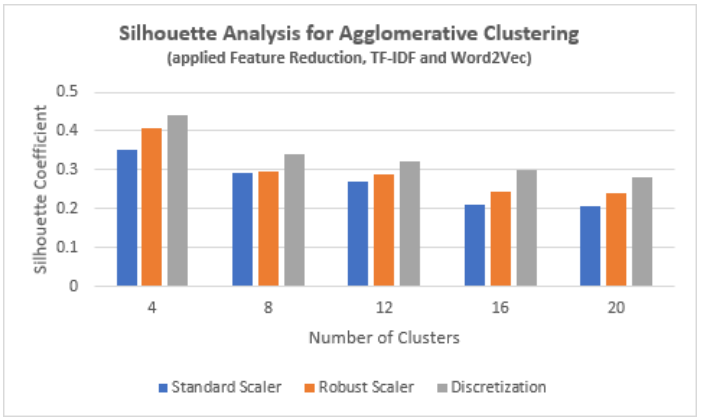


Figure 5.6: Experiment Six: Silhouette Scores Using Agglomerative Clustering with TF-IDF Weighting and Word2Vec embbeding on the tweets and Feature Reduction on the linguistic feature vectors

5.3 t-SNE Projection of Clustered Users

In addition to silhouette scores, dimensionality reduction using resulting cluster labels was applied to analyze how data was spread. We chose number of clusters as 5 to analyze the clusters more clearly. We used t-SNE as projection method. There are two reasons why we selected it as dimensionality reduction method over its alternatives such as Principal Component Analysis(PCA):

- It provides us non-linear projection when there is a non-linear relationships between features[43].
- Global and local structure of the data in high dimensional space are preserved. In other words, if tha data point are close to each other in high dimensional space, it also put them close in the low dimensional space.
- It avoids crowding by spreading out the medium points [44].

We used scikit-learn implementation of t-SNE in this thesis, unfortunately it is slow due to its complexity. There is alternative implementations like Multicore-TSNE[1] which is multicore(parallel ie.) modification of t-SNE. However, our data gives results almost 2 seconds while running t-SNE. So, we do not need to accelerate more.

We showed our dimensional reduction results in the Figures 5.7, 5.8, 5.9, 5.10, 5.11, 5.12. As it is seen in the figures, the projection provides an insight about which experiment gives more reasonable results for our problem. We can see how many different users resembles or differentiates from each other according to the tweets.

[1] https://pypi.org/project/MulticoreTSNE/

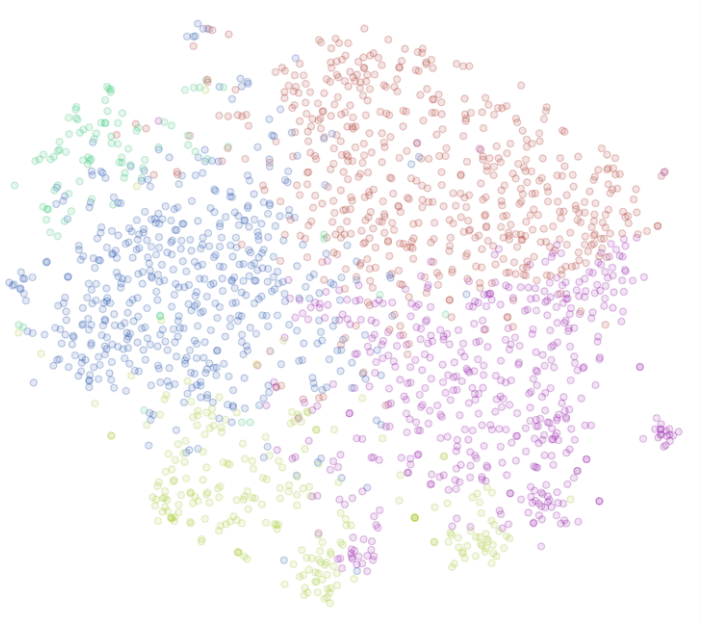


Figure 5.7: t-SNE Projection of Clustered Users Using K-means Clustering with TF-IDF weighting on the tweets without Feature Reduction and Word2Vec embedding

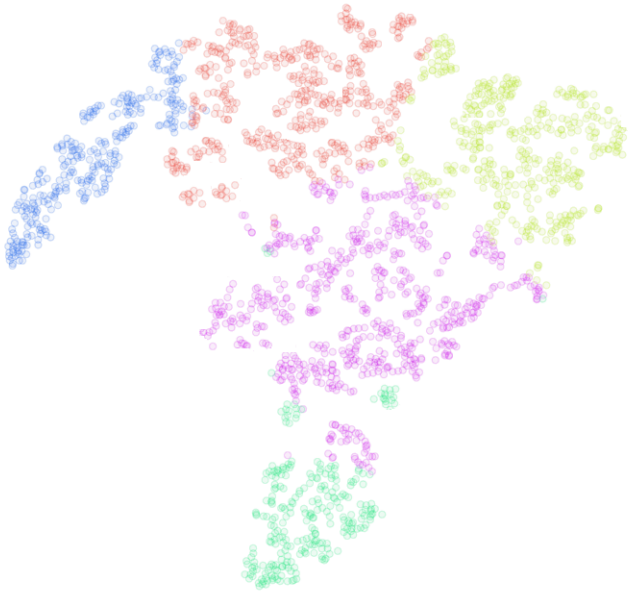


Figure 5.8: t-SNE Projection of Clustered Users Using K-means Clustering with TF-IDF weighting and Word2Vec embedding on the tweets without Feature Reduction

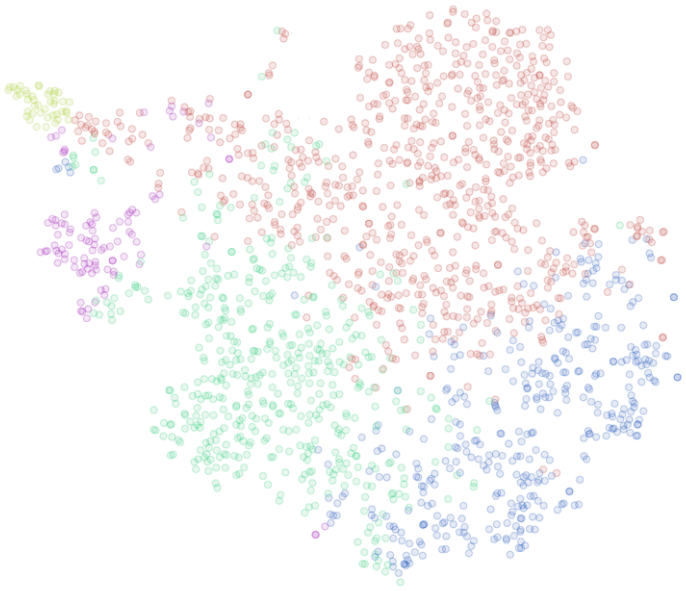


Figure 5.9: t-SNE Projection of Clustered Users Using K-means Clustering with TF-IDF weighting and Feature Reduction on the linguistic feature vectors without Word2Vec embedding

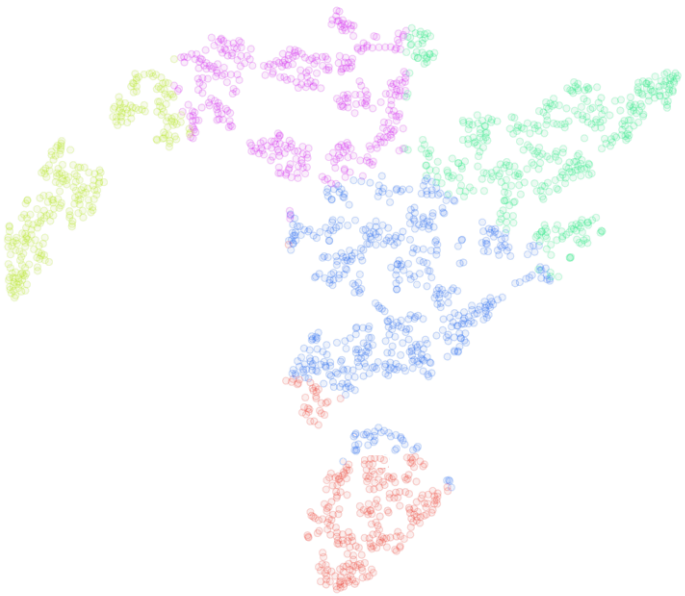


Figure 5.10: t-SNE Projection of Clustered Users Using K-means Clustering with TF-IDF weighting and Word2Vec embedding and Feature Reduction on the linguistic feature vectors

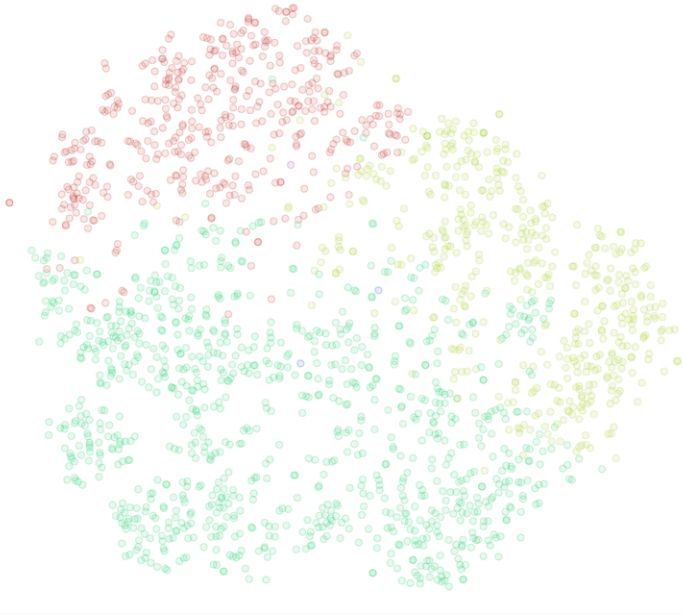


Figure 5.11: t-SNE Projection of Clustered Users Using Agglomerative Clustering with TF-IDF weighting on the tweets and Feature Reduction on the linguistic feature vectors without Word2Vec embedding

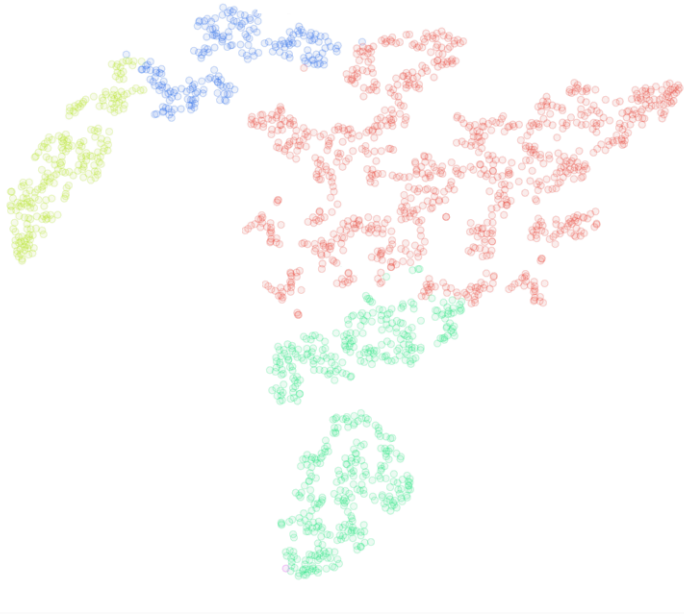


Figure 5.12: t-SNE Projection of Clustered Users Using Agglomerative Clustering with TF-IDF weighting and Word2Vec embedding on the tweets and Feature Reduction on the linguistic feature vectors

5.4 Error Rates

In consideration of these experiments and their results, we realized that agglomerative and K-means clustering produces similar results. Also, we saw that Word2Vec and feature reduction on the data have positive effect on silhouette scores, and also there is a little difference in favor of K-means. When we compare the normalization methods, we concluded that discretization always won with respect to silhouette scores. In the light of this information, we decided to calculate error rates with the approach which includes feature reduction, discretization as normalization method, TF-IDF weighting, Word2Vec and K-means as clustering algorithm.

Since the experiment which uses feature reduction on feature vectors, applies TF-IDF vectorization and Word2Vec based word embedding on crawled tweets and finally clusters users with K-means algorithm produces the best silhouette scores, we have calculated the error rates for each dimension of OCEAN, which is shown in Figure 5.13. We used base users' OCEAN scores as a ground truth, then measured with our proposed algorithm which we gave its details in the Section 5.1. Error rates for other experiments is added to Appendix A with Figure A.1, A.2, A.3, A.4 and A.5.

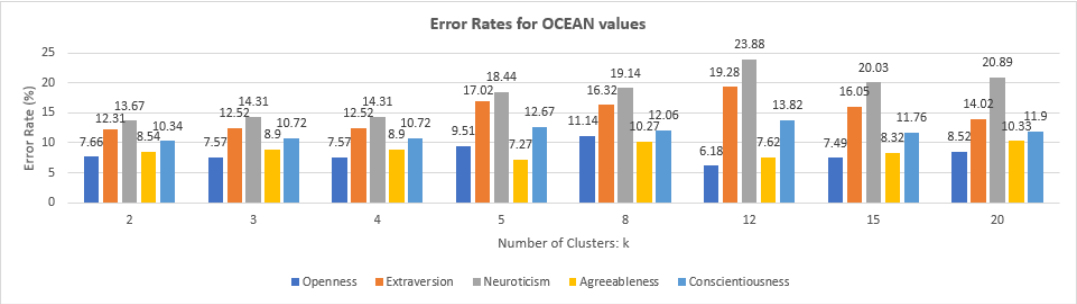


Figure 5.13: Error Rates

Error rates measured with different k values in order to see how to change compared to the number of clusters. As can be seen, error rates varies according to corresponding OCEAN value, but not change much with k. "Openness", "Aggreableness" and "Conscientiousness" have smaller error rates than "Extraversion" and "Neuroticism". Thus, we concluded that our approach gives accurate results for "Openness", "Aggreableness" and "Conscientiousness". However, the approach need to be improved for other OCEAN features in the future.

5.5 Discussions

In this section, we will compare our clustering results with respect to normalization methods, silhouette coefficients, t-SNE projections and error rates. There are many findings and inferences related to our problem:

- We used two different clustering algorithms as K-means and agglomerative clustering. With respect to silhouette coefficients, we obtained minor differences in scores. Nevertheless, K-means gave us better results in all experiments. If we analyze the t-SNE projections further, agglomerative clustering unified the users as one big cluster and other small sized clusters. However, we required similar sized clusters. Our purpose is not to get resembling clusters by making them one big clusters, instead, we aimed separate clusters from each other with the equal sizes as much as possible.
- With respect to different normalization methods, discretization had always positive effect on silhouette scores compared to standard and robust scalers.
- We measured the effect of feature reduction on the results. If clustering is applied after the applied the feature reduction on feature vectors, we had higher silhouette scores. Similarly, we analyzed the t-SNE projection results, feature reduction lead to differentiate clusters from each other.
- Word2Vec based word embeddings were very important method to apply in this thesis. In all experiments with Word2Vec, we attained significant improvements with respect to silhouette scores, t-SNE projection and error rates.
- After we obtained best results with feature reduction, Word2Vec and applied K-means algorithm, we calculated error rates. As a result, we achieved error rates below 10% in “Openness”, “Aggreableness” and “Conscientiousness” scores.

CHAPTER 6

CONCLUSIONS

In this thesis, we have presented a framework for predicting personality traits of people from their tweets. The framework has been designed especially for Turkish tweets. The whole process starts with data cleaning and preparation step, followed by determining tweet related and linguistic features from the tweets, and then, after eliminating less useful features thresholding using feature variances and creating word vectors with weighting methods like TF-IDF, applying Word2Vec based word embedding to the crawled tweets before the final step of clustering. Since we had relatively small number of ground truth personality trait values, we had to develop a suitable error mechanism to measure the quality of our results. Our experiment show that the overall approach produces acceptable results, and open to further improvements.

This thesis has produced important results with small error rates. We tried to reach out individuals' OCEAN scores by using their twitter interactions. Firstly, we defined and extracted tweet features such as average word count, emoticon count, negative word count etc. To increase dissimilarity among users we applied feature selection which was variance threshold. In other words, we removed features from our data which had low variance. Secondly, we applied lemmatization and removed stop words from tweets. Then, all tweets were merged belonging to same users and user documents were created. TF-IDF weighting was applied (with 1-gram, 2-grams and 3-grams) to the documents to get most important words or phrases. We adopted Word2Vec model with TF-IDF results. Afterwards, We combined extracted feature values with Word2Vec results. Thus, final data has been created before clustering. For our main purpose, clustering methods K-means and agglomerative clustering were applied to our final data. We tried these clustering algorithms with different cluster counts to

achieve the best results. By the way, when we have clustered users with either only TF-IDF or Word2Vec with TF-IDF vectors, we saw that Word2Vec had a very positive effect on the clustering results. Finally, K-means and agglomerative clustering gave us similar results with minor differences in favor of K-means. Thus, we used K-means clustering results while calculating error rates. We concluded that, our error rates were small and acceptable based on Openness, Agreeableness, Conscientiousness scores. In the future, there can be done an improvement for achieving good results for Neuroticism and Extraversion scores.

REFERENCES

[1] "The amazing power of word vectors." http://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/. Accessed: 2019-01-13.

[2] "The new marketing is people centric: Know your customer personality!." http://www.neosperience.com/blog/. Accessed: 2019-05-28.

[3] D. D. C. M. D. Alan S. Gerber, Gregory A. Huber and S. E. Ha, "Personality and political attitudes: Relationships across issue domains and political contexts," *American Political Science Review*, vol. 104, no. 1, p. 111133, 2010.

[4] H.-P. K. Xiaowei Xu, Martin Ester and J. Sander, "A distribution-based clustering algorithm for mining in large spatial databases," *f 14th International Conference on Data Engineering*, 1998.

[5] S. B. Venkatesan M*, Doshi Aditya Ashvin and R. Thomas, "A centroid-based clustering approach to analyze examinations for diabetic patients," vol. 8, pp. 75–79, 01 2016.

[6] J. Li and L. Behjat, "A connectivity based clustering algorithm with application to vlsi circuit partitioning," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 53, pp. 384 – 388, 2006.

[7] H.-P. K. Jörg Sander, Martin Ester and X. Xu, "Density-based clustering in spatial databases: The algorithm gdbscan and its applications," *Data Mining and Knowledge Discovery*, vol. 2, pp. 169 – 194, 1998.

[8] "The big five personality traits." https://www.verywellmind.com/the-big-five-personality-dimensions-2795422. Accessed: 2019-05-29.

[9] A. Akın, "Zemberek-nlp." https://github.com/ahmetaa/zemberek-nlp, 2019.

[10] G. C. Tomas Mikolov, Kai Chen and J. Dean, "Efficient estimation of word representations in vector space," *Proceedings of the International Conference on Learning Representations*, 2013.

[11] H. M. Johan Bollen and A. Pepe, "Determining the public mood state by analysis of microblogging posts," *In Procceedings Of the Alife XII Conf. MIT Press*, 2010.

[12] P. G. S. Andranik Tumasjan, Timm O. Sprenger and I. M. Welpe, "Predicting elections with twitter: What 140 characters reveal about political sentiment," *Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[13] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes twitter users: Real-time event detection by social sensors," in *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, (New York, NY, USA), pp. 851–860, ACM, 2010.

[14] M. J. Paul and M. Dredze, "You are what your tweet: Analyzing twitter for public health," *Artificial Intelligence*, vol. 38, pp. 265–272, 01 2011.

[15] D. S. Daniele Quercia, Michal Kosinski and J. Crowcroft, "Our twitter profiles, our selves: Predicting personality with twitter," *IEEE Third International Conference on Privacy, Security, Risk and Trust and IEEE Third International Conference on Social Computing*, 2011.

[16] H. P. Haewoon Kwak, Changhyun Lee and S. Moon, "What is twitter, a social network or a news media?," *19th International World Wide Web Conference*, 2010.

[17] Y. W. Yang Zhang and Q. Yang, "Community discovery in twitter based on user interests," *Journal of Computational Information Systems*, 2012.

[18] S. D. Daniel Ramage and D. Liebling, "Characterizing microblogs with topic models," *Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[19] V. Roth and T. Lange, "Feature selection in clustering problems," *Advances in Neural Information Processing Systems*, 2004.

[20] M. A. F. Martin H.C. Law and A. K. Jain, "Simultaneous feature selection and clustering using mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[21] M. O. Saa Petrovi and V. Lavrenko, "Streaming first story detection with application to twitter," *In Human Language Technologies: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2010.

[22] N. C. Eric Nalisnick, Bhaskar Mitra and R. Caruana, "Improving document ranking with dual word embeddings," in *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, (Republic and Canton of Geneva, Switzerland), pp. 83–84, International World Wide Web Conferences Steering Committee, 2016.

[23] "Personality character traits: The good, the bad and the ugly." `http://positivepsychology.com/character-traits/`. Accessed: 2019-07-03.

[24] M. R. Barrick and M. K. Mount, "The big five personality dimensions and job performance: A metaanalysis," *Personnel Psychology*, 1991.

[25] H. W. E. R. H. M. C. A. C. R. C. Lewis R. Goldberg, John A. Johnson and H. G. Gough, "The international personality item pool and the future of public-domain personality measures," *Journal of Research in Personality*, 2006.

[26] D. K. Cameron Anderson, Oliver P. John and A. M. Kring, "Who attains social status? effects of personality and physical attractiveness in social groups," *Journal of Personality and Social Psychology*, 2001.

[27] L. A. JensenCampbell and W. G. Graziano, "Agreeableness as a moderator of interpersonal conflict," *Journal of Personality*, 2004.

[28] B. R. Karney and T. Bradbury, "The longitudinal course of marital quality and stability: A review of theory, method, and research," *Psychological Bulletin*, 1995.

[29] J. Ramos, "Using tf-idf to determine word relevance in document queries," 2003.

[30] "Text vectorization and transformation pipelines." https://www.oreilly.com/library/view/applied-text-analysis/9781491963036/ch04.html. Accessed: 2019-05-29.

[31] "Robust measures of scale." http://en.wikipedia.org/wiki/Robust_measures_of_scale. Accessed: 2019-05-28.

[32] "Discretization." http://en.wikipedia.org/wiki/Discretization. Accessed: 2019-05-28.

[33] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[34] "Agglomerative hierarchical clustering." http://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/. Accessed: 2019-05-28.

[35] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Machine Learning Research 9 (2008) 2579-2605*, 2008.

[36] "Numpy percentile." http://docs.scipy.org/doc/numpy/reference/generated/numpy.percentile.html. Accessed: 2019-05-26.

[37] "Stemming and lemmatization." http://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html. Accessed: 2019-05-25.

[38] R. N. M. M. A. P. A. A. Kathy Lee, Diana Palsetia and A. Choudhary, "Twitter trending topic classification," in *2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 251–258, Dec 2011.

[39] "Visualizing tweets with word2vec and t-sne, in python." http://leightley.com/visualizing-tweets-with-word2vec-and-t-sne-in-python/. Accessed: 2019-05-27.

[40] P. M. Janus Wawrzinek, José María González Pinto and W. T. Balke, "Do scaling algorithms preserve word2vec semantics? a case study for medical entities," *International Conference on Data Integration in the Life Sciences*, 2018.

[41] "Gensim word2vec tutorial." http://www.kaggle.com/pierremegret/gensim-word2vec-tutorial. Accessed: 2019-07-05.

[42] "Kmeans clustering." http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html. Accessed: 2019-05-25.

[43] "Advantages and disadvantages of t-sne over pca (pca vs t-sne)." http://theprofessionalspoint.blogspot.com/2019/03/advantages-and-disadvantages-of-t-sne.html. Accessed: 2019-05-27.

[44] "Paper dissected: visualizing data using t-sne explained." http://mlexplained.com/2018/09/14/paper-dissected-visualizing-data-using-t-sne-explained/. Accessed: 2019-05-27.

APPENDIX A

APPENDIX I

A.1 Error Rates for Other Clustering Results

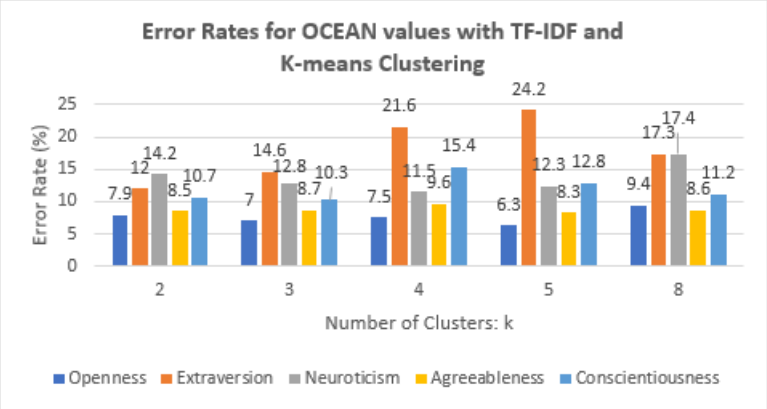


Figure A.1: Error Rates of K-means Clustering on TF-IDF Weighting

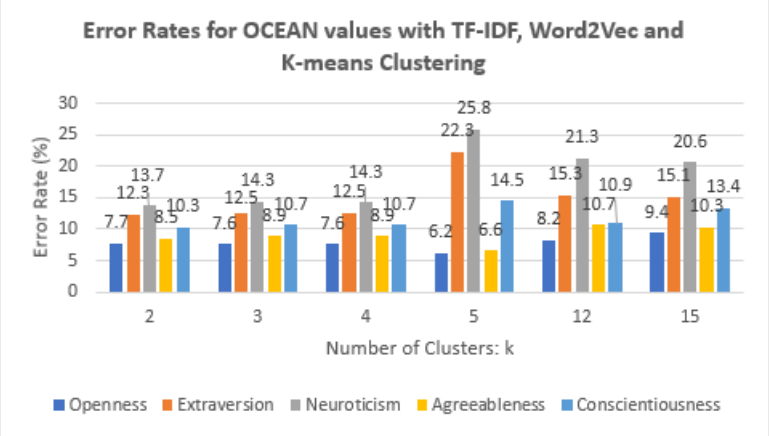


Figure A.2: Error Rates of K-means Clustering on TF-IDF Weighting and Word2Vec based Word Embedding

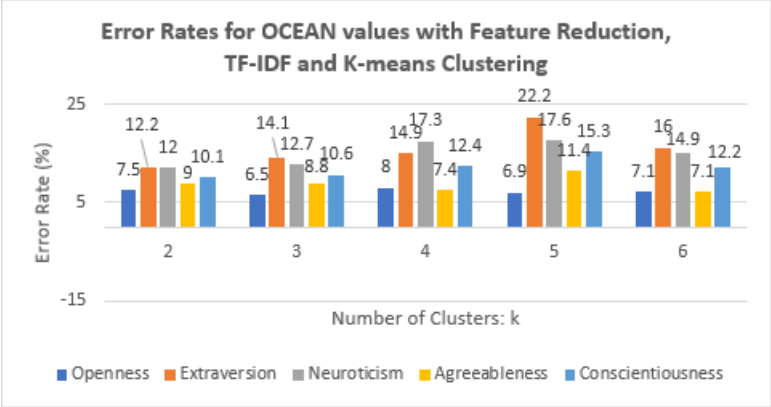


Figure A.3: Error Rates of K-means Clustering on TF-IDF Weighting and Feature Reduction

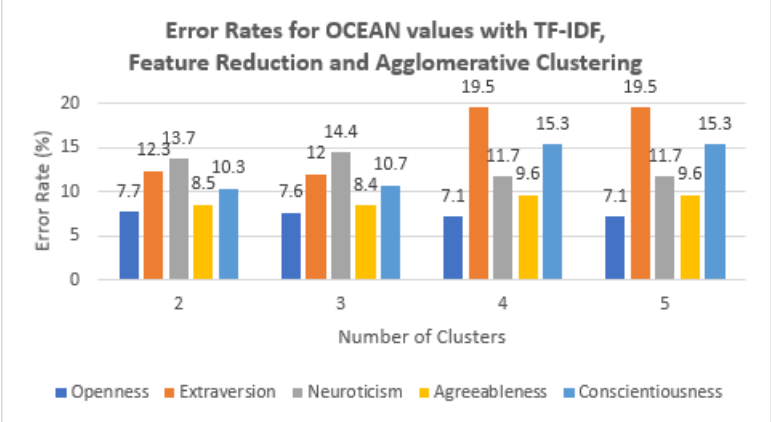


Figure A.4: Error Rates of Agglomerative Clustering on TF-IDF Weighting and Feature Reduction

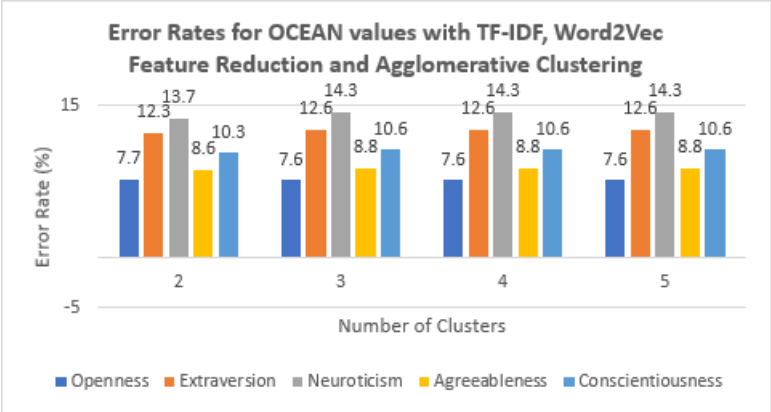


Figure A.5: Error Rates of Agglomerative Clustering on TF-IDF Weighting, Feature Reduction and Word2Vec based Word Embedding