A NOVEL PRE-PROCESSING WORKFLOW FOR POPULARITY PREDICTION IN SOCIAL MEDIA

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

HÜSEYİN BUĞRA YILDIRIM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

SEPTEMBER 2021

A NOVEL PRE-PROCESSING WORKFLOW FOR POPULARITY PREDICTION IN SOCIAL MEDIA

submitted by **HÜSEYİN BUĞRA YILDIRIM** in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems Department, Middle East Technical University** by,

Date: 10.09.2021

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, Last name: Hüseyin Buğra Yıldırım

Signature : _____

ABSTRACT

A NOVEL PRE-PROCESSING WORKFLOW FOR POPULARITY PREDICTION IN SOCIAL MEDIA

Yıldırım, Hüseyin Buğra

M.S., Department of Information Systems

Supervisor: Assoc. Prof. Dr. Tuğba Taşkaya Temizel

September 2021, 74 pages

Users in Twitter are in continuous interaction with each other through posts and reactions such as likes and retweets. Tweets often get a little reaction from people, with only a few of them receiving a prominent response. Thus, reaction numbers result in having a heavy right-skewed distribution. Furthermore, some tweets show unexpected response performance that cannot be depicted by standard features and are often dependent on extraordinary situations such as being the first reporter and mass reaction. Heavily skewed distribution of social media dataset and variation between expected and the observed reactions are mainly two distorting factors for model prediction. This thesis initially addresses the concept of outliers and uncertainty in reaction numbers in social media datasets. A method for identifying social media outliers is proposed, and the adverse effects of outliers on modeling are presented. Finally, a SMOTE-based data augmentation method, where a discretization is applied and synthetic data is generated predominantly from the clusters with fewer instances, is presented. The results show that the models where outlier removal and data augmentation are applied achieve slightly better prediction performance than those constructed without them. This research presents practical implications for studies that aim to predict the popularity of tweets.

Keywords: Social Media, Popularity Prediction, Pre-processing, Outlier Detection, Data Augmentation

ÖZ

SOSYAL MEDYADA POPÜLERLİK TAHMİNİ İÇİN YENİ BİR ÖN İŞLEME İŞ AKIŞI

Yıldırım, Hüseyin Buğra

Yüksek Lisans, Bilişim Sistemleri Bölümü

Tez Yöneticisi: Doç. Dr. Tuğba Taşkaya Temizel

Eylül 2021, 74 sayfa

Twitterdaki kullanıcılar, gönderiler ve beğeni, retweet gibi reaksiyonlar aracılığıyla birbirleriyle sürekli etkileşim içerisindedirler. Tweetler genellikle insanlardan az sayıda tepkiler alırken, onlardan sadece birkaçı öne çıkan bir tepki alır. Bu nedenle, reaksiyon sayıları yoğun sağa çarpık bir dağılıma sebep olurlar. Ayrıca, bazı tweetler, olağan özellikler ile tasvir edilemeyen ve genellikle bir konunun ilk habercisi olma veya kitlesel reaksiyonlar gibi olağan üstü durumlara bağlı olan beklenmedik bir etkileşim performansı gösterir. Sosyal medya veri setinin aşırı derecede çarpık dağılımı ile varsayılan ve gözlemlenen reaksiyon sayıları arasındaki farklılık, temel olarak model tahmin sürecindeki iki bozucu unsurdur. Bu tez, ilk olarak sosyal medya veri setlerindeki aykırı değerler ve reaksiyon sayılarındaki belirsizlik kavramlarını ele almaktadır. Sosyal medyadaki aykırı değerlerin belirlenmesi için bir yöntem önerilmekte ve aykırı değerlerin modelleme üzerindeki olumsuz etkileri sunulmaktadır. Son olarak, ayrıklaştırmanın uygulandığı ve ağırlıklı olarak daha az sayıda örnek içeren kümelerden sentetik verilerin üretildiği SMOTE tabanlı bir veri arttırma yöntemi sunulmaktadır. Sonuçlar, aykırı değerlerin temizlendiği ve veri arttırmanın uygulandığı modellerin, bunlar olmadan oluşturulanlardan biraz daha iyi bir tahmin performansı sağladığını göstermektedir. Bu araştırma, tweetlerin popülerliğini tahmin etmeyi amaçlayan çalışmalar için pratik çıkarımlar sunmaktadır.

Anahtar Kelimeler: Sosyal Medya, Popülerite Tahmini, Ön işleme, Aykırı Değer Tespiti, Veri Arttırma

To My Family

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my supervisor, Assoc. Prof. Dr. Tuğba Taşkaya Temizel for her endless support and wise guidance during this process. Her passion for research and her multidimensional approach to subjects shed light on me identifying the methods and steps that I followed in my dissertation.

I also would like to thank the examining committee members for their valuable feedback and suggestions.

I offer my deepest love to my wife, who always encourages me, and to my beloved daughter, joined to our lives in this process. They have been and will always be my motivation in every respect.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

AIC	America Iran Conflict dataset
SBR	Step back Royal dataset
SMOTE	Synthetic Minority Oversampling Technique
KNN	K-Nearest Neighbor
SVM	Support Vector Machine
XGBoost	eXtreme Gradient Boosting
API	Application Programming Interface
NLP	Natural Language Processing
CNN	Convolutional Neural Network
PCA	Principal Component Analysis
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
ML	Machine Learning
OutAug	Proposed pre-processing workflow
ClusOut	Proposed cluster-based outlier detection mechanism
ClusAug	Proposed cluster-based data augmentation mechanism

CHAPTER 1

INTRODUCTION

Social media is the general name of digital platforms that enable users to communicate and interact online. Along with the process starting with Web 2.0 applications' development and reaching a peak with the prevalence of smart devices such as smartphones and tablets, it has attracted many users for various purposes like social networking, sharing ideas, self-expression, marketing, dating, or politics. Social media mainly comprises social networking apps like Facebook, LinkedIn, microblogging apps like Twitter, Tumblr, photo sharing apps like Instagram, Pinterest, and video sharing apps like YouTube and TikTok. A microblogging service is a platform in which users share their experiences, feelings, thoughts, or information in short-verbal or visual forms. Twitter, a microblogging service, also enables information to be created, transmitted, transformed, and consumed in its cyberspace.

Alongside being actively used by individuals who aim to self-express and gain popularity among friends, the companies, organizations, and politicians show deep interest in Twitter to declare their standpoints and increase their recognition or popularity on crowds. Furthermore, Twitter has a preponderance for the interest of researchers [1]. Reasoned to pre-mentioned use cases of users, tweets are primarily visible to public internet that maps to lots of data for researchers.

Users are both the producer and consumers of content, and they express their interests in different ways. For instance, retweet means to forward one's root tweet to own followers by contributing to information propagation. Users generally retweet for a communication purpose, social relationship maintenance, self-expression, obtaining/updating information, or prosocial motivations [2]. Also, the meaning of retweeting may not be only affirmation or approval, but also démenti or disparagement.

Popularity means the state of being liked, supported, or appreciated by many others. Predicting the popularity (reaction number) in social media is a trending and challenging topic for researchers. Popularity can be measured for numerous items such as online news [3–9], a tweet [10–14], a Facebook post [15, 16], or an Instagram photo [17].

In Twitter, popularity is related to a user, hashtag, or a single tweet. As a reflection of people's fame, a user's popularity is positively correlated with the number of followers. Nevertheless, popularity and influence dissociate each other, implying that the popular users do not always receive more reactions for their tweets [18–20]. Tweets posted

from an account with fewer followers may have more retweeted than a popular account. The number of followers only affects the number of users reached in the first stage; however, having been retweeted via nodes creates a cascading effect by promoting the diffusion of information among crowds. Also, the denomination of tweets' popularity could range a wide variety of spectrum according to their usage purposes, such as promotional effectiveness for marketing, attracting supporters for politics.

Popularity prediction methods are generally characterized by two types of approaches; feature-driven and generative models. The feature-driven method extracts a comprehensive list of related features, including obvious ones such as number of followers, number of hashtags, and latent ones such as sentiment, readability. It then employs different learning models for prediction such as Regression [11], Support Vector Machines [3], Random Forest [21], Neural Networks [22, 23]. Also, some studies employed deep-learning models to extract new features from posts [24]. The feature-driven method requires accomplished feature engineering. Generative models describe a set of phenomena with a parametric model and then fit the parameters to observed data. Point-processed generative models can account for popularity burst and decays [25], online diffusion [26], or a final number of reshares [27]. Enrichment of social phonemes creates a scalability problem for these models. Another characterization of popularity prediction is related to time, before and after post-publication; another denomination is ex-ante prediction and peeking prediction, respectively. The primary difference between these approaches is taking the early popularity measurements into account in peeking prediction. The ex-ante prediction, also known as cold-start, is highly sensitive to selected features and a challenging task because of the highly skewed distribution of popularity metrics [3, 28]. My study brings a novel pre-processing workflow to struggle with these challenges of cold-start prediction problems; what is more, with minor adaptations and parameter changes, the proposed pre-processing workflow can be applied to all popularity prediction problems in social media.

Popularity scores are usually discrete variables. In the literature, the studies either attempt to predict the original values or discretize them to treat the problem as a classification study. For the former case, as the range of the scores is too broad and the distribution is highly skewed, simple normalization of the scores results in many data points being transformed into a smaller range for the prediction model. Therefore, it is often challenging to select the correct transformation. For the latter case, the standard approach is to discretize the values into two groups, where the threshold is set to the mean of the popularity score distribution, i.e., the ones that are higher than the mean is labeled as "popular" and the lower ones as "unpopular" [4]. Some other studies are treated as a multi-class prediction problem [28, 29] with user-defined thresholds. Such approaches, however, hardly classify the posts around the threshold values—for instance, several data points lie around the threshold values with a difference of one or two scores. In addition, the mean or median threshold value often corresponds to a small value due to the highly skewed nature of the popularity score distribution, where the "popular" and "unpopular" labels can be misleading. Consequently, such discretization strategies often cause classifiers not to learn classes correctly and decrease model performance.

Uncertainty is defined as a situation in which something is not known or something that is not known or certain[1]. In machine learning (ML), there are two significant types of uncertainty; epistemic and aleatoric. Epistemic uncertainty is related to the noise in the model's parameters, which can be reduced by increasing the dataset size. Aleatoric uncertainty corresponds to observed data noise, resulting from measurement errors, homoscedastic, and dependent uncertainty according to input data, heteroscedastic. While estimating uncertainty is generally studied in deep learning models [30, 31], none of the studies describes the source of uncertainty in social media and aims to alleviate uncertainty in reaction numbers to prepare the dataset for standard machine learning models.

Three attributes create uncertainty in reaction numbers; time, user choices, and missing denominator. Firstly, the reaction number of a tweet depends on its posting time [32] and user activity levels [33]. Some tweets are shown to be shared or seen more on specific dates or times [34]. Furthermore, widely fluctuating user activity levels make figuring out a precise selection of time impossible. Secondly, the reaction is also affected by user choices like tweet content or using hashtags. Due to the fuzziness of human language, leveraged linguistic attributes may have noise. For example, content in glowing terms with lots of allusions may be misleading for sentiment analysis. Also, hashtags can attract attention to a particular topic, but they are redundant if everyone knows the topic [1]. Lastly, social media platforms that offer a flow of posts and do not navigate via clicks between posts do not provide information about who see or read. The solution might be to model the likely estimation of possible sighters, but it does not help forecast the specific follower's behaviors. As an example, a user has some royal followers who mostly like and retweet his posts. Some of them might not see his post due to their unavailability at the time of the post or new posts from other people that could hinder his posts, resulting in the loss of possible retweets. As a result, there is always an uncertainty in the number of reactions in social media. Although this uncertainty may not change the numbers dramatically from their expected value, they may fluctuate in small margins.

While many tweets cannot get any reactions, only a handful of them goes to thousands. It creates a highly unbalanced dataset and causes machine learning models to learn the most frequent ones. Having been highly retweeted is a rare event whose detection is of utmost importance. In classification, imbalanced learning can be solved with under-sampling, over-sampling, or hybrid models. One of the hybrid models, SMOTE, selects examples close in the feature space and synthesizes new ones using this distance among selected examples [35]. It works well for classification problems, and many advanced SMOTE methods have been presented to the literature [36–38]. Also, Torgo et al. [39] conducted a study to apply SMOTE to regression modeling by discretizing a dataset. My study offers a novel instance-based SMOTE for regression problems. It synthesizes new instances for balancing datasets and adds systematic noise for alleviating uncertainty in reaction numbers.

Some tweets show unexpected reaction performance compared to similar ones. Uncertainty in reaction numbers cannot simply explain this high variation. Many

[1] https://dictionary.cambridge.org/dictionary/english/uncertainty, the definition of uncertainty in online Cambridge dictionary.

internal and external factors affect the reaction number of tweets, and features in a dataset cannot comprise all of these factors. Furthermore, linguistic analyses might not be sufficient to identify the interestingness and crowd-pulling of content. As a result, these tweets are extreme points in the dataset that distort a model's pattern learning. In my study, tweets that show under- and over-performance are detected via defined thresholds and removed to ensure a smoother dataset.

In regression, randomization is an essential factor while splitting the dataset as train and test. A test dataset which only contains some specific pattern observations would cause an overfitting problem for a model. Researchers adopt a cross-validation method to ensure that each fold is randomized and represents the whole dataset well. Following a training process of models, they employ evaluation metrics like RMSE or MAE to evaluate the performance of machine learning models. Nevertheless, classical train-test data splitting methods like cross-validation or simple random sampling are not convenient for highly skewed datasets. They reflect the overall distribution into the test data distribution, causing only a few valuable observations to appear. So, researchers should employ further investigation and plotting to observe the improvement in rare instances. This study provides a novel splitting method for highly skewed datasets, directly observing the model improvements via standard evaluation metrics.

The main objective of this thesis is to develop a complete and seamless approach for popularity prediction where:

- the problem is handled as a regression problem rather than a classification problem,
- outliers are conceptually defined and identified prior to analysis,
- a new data augmentation approach is introduced to cope with the uncertainty in reaction numbers by adding a systematic noise to the observations,
- oversampling is applied to balance towards the tail of the distribution in a highly right-skewed dataset,
- a new dataset splitting method for highly skewed datasets is developed to measure the model's performance for the data points by reflecting different popularity ranges as equally as possible.

1.1 Research Questions

The following research questions are posed concerning the objectives of the study:

Q.1. What is an outlier in the Social Media domain? Which observations may be considered as an outlier? How could an approach manage to detect them?

Q.2. What are the reasons for uncertainty in reaction numbers? How can this uncertainty be alleviated in a dataset?

Q.3. How to handle the fat-tailed skewed data of social media in modeling? How can SMOTE augment the observations in a regression problem?

Q.4. What is the proper splitting method for a highly skewed dataset to facilitate the studies aiming to observe the improved performance of valuable rare target observations without further plotting?

In this thesis, XGBoost, a gradient boosting model, and SVM, support vector machine for regression, are trained to predict the retweet number of tweets in a highly right-skewed dataset in which zeros account for about %70 of observations. The data acquisition process is carried out via Twitter Application Programming Interface (API), considering API limitations and the other studies in literature for the feature selection process. The proposed model predicts the retweet number of tweets as numeric; subsequently, the higher results imply the popularity of observation. To analyze the first research question, with a conceptual point of view, observations showing under- or over-performance compared with similar instances in the dataset are detected with the k-nearest neighbor (KNN) algorithm, and they are labeled as outliers. For the second and third research questions, each observation in the tail is individually augmented in high numbers via SMOTE following creating small temporary datasets with its closest neighbors. New synthetic instances are generated following the labeling of the observations in these temporary datasets according to optimal ***k*** value. Synthesized instances are pretty close to their root instances in multidimensional space and have similar values, independent and dependent features. Thus, with adding noise to the dataset, uncertainty in retweet numbers is aimed to be alleviated. When it comes to the fourth research question, frequent and rare observations are partially balanced with an algorithm that starts with the random sampling from clusters in the tail and stepwise shifting to initial clusters while considering the partition of observations according to their clusters.

In order to validate the models and the framework proposed in the thesis, two datasets were collected using Twitter Search API. Two different datasets about popular topics during data collection ("Iran, Crown") were obtained using their related keywords over seven days. For joining, manipulating, and organizing data, a unique tweet_id provided by Twitter was used. By writing a tweet_id in HTML address, independently from screen_name, the original tweet can be reachable.

Essential pre-processing steps are carried out to interpret and analyze the data, and one-hot encoding is applied to transform factor variables to numeric ones as a must of XGBoost modeling. Datasets are split into two samples regarding the clusters of retweet numbers detected with the Natural Breaks Algorithm [69] running on retweet numbers. Neighbors of instances are found with the KNN algorithm using the Gower distance to determine the proximity, and the importance of features obtained from the base XGBoost model is included in the KNN algorithm as weights. MAE metric is employed to find the extreme instances compared to the others in their cluster, and these instances are removed from the dataset. Optimal ***k*** value derived from the KNN algorithm in the outlier detection process guides to label the instances in the generated small samples for each instance in tail clusters. Effects of the proposed workflow are compared on a framework, includes the baseline and state-of-art models by employing MAE and RMSE metrics.

1.2 Contributions of the Study

The main contributions of the thesis are listed below:

- A novel outlier detection methodology is proposed for the social media domain, and a comparison of this method with the three outlier detection methods from the literature shows that the proposed method works slightly better for social media domain.
- It is one of the few studies that detect outliers, and the first study defines uncertainty in reaction numbers in the social media domain to best my knowledge.
- The outlier detection mechanism could be used for testing the representativeness of features in popularity prediction.
- Running the proposed workflow on the base ML models (XGBoost and SVM) demonstrates that the models' sensitiveness is in sundry amounts against outliers and generated data points.
- The novel splitting methodology of datasets as train and test for highly skewed datasets facilitates researchers' work to evaluate their models' success pursuing rare target cases.

CHAPTER 2

RELATED WORK

This chapter provides a systematic literature review to analyze the related studies and explain how this study fits the literature. Related work about popularity prediction and pre-processing steps in related studies are explained in this section. These topics are discussed in the following subsections.

2.1 Popularity Prediction

Popularity prediction in social media has been a prevalent and challenging task in the last decade, and many prediction methods have been proposed in the literature [41]. Due to being conscious of popular posts' profound social impact or commercial profitability, researchers have tried to predict popular ones or life cycles of topics in social media. This task is not a mere prediction process; it is the prehension of what makes a post more popular than others and observes the popularity and social dynamics, making the whole process a challenging task.

While the vast majority of prediction methods relies entirely on the information from a single domain where a post is published, like Twitter [19, 34, 42, 43], Facebook [15, 16], YouTube [44, 45] and others [5, 46], some studies use information from multiple domains to predict the popularity of content [47, 48]. This process is based on extracting information from a domain and transforming it into knowledge for popularity prediction in another domain. In [47], information from Twitter is extracted to detect the videos that will experience a sudden burst of popularity on YouTube. Oghina et al. [48] use tweets from Twitter and comments from YouTube to predict the movie ratings on IMDB. These studies imply the plethora and variety of information/features which can be used for popularity prediction.

As the interest and number of users increase, real-life implementations of predictions have become a large-scale problem. General characteristics of Twitter, a microblogging service, like availability of data, tools, and ease of use, make it take the lion’s share of the interest, throwing it up as the hub for detecting the distribution of information. An overarching and highly cited study [18], conducted in earlier times in Twitter by crawling the entire Twitter site, has signified the fast diffusion of information following being retweeted. Moreover, Tufekci [1] puts forward the

datasets in almost half of the papers presented in ICWSM in 2013 have been drawn from Twitter.

In literature, existing studies have attempted to approach the popularity prediction problem in two main ways, (1) before publication (i.e., ex-ante) and (2) after publication (i.e., peeking). The ex-ante prediction approach aims to measure the popularity of content before its spreading over the network by employing structural features, content features, and others. In [3], a large of related features are engineered by using the content of feature and external sources. Due to the imbalanced class proportion and little association between the popularity and inherent structural features, they state the difficulty of the popularity prediction at cold-start problems. Bandari et al. [28] handle the problem of both classification and regression, taking into account four derived features –the source, the category, subjectivity, the named entities mentioned- related to the tweet and its content. Results show that although these features are not sufficient for the exact number of the prediction, they can effectively provide a range for the article's popularity on Twitter. To the contrary of practicableness and apprehensibility of ex-ante prediction approaches, new approaches are proposed because the success of prediction models heavily depends on the correct feature selection and the data structures in social media.

Peeking prediction approaches employ the early popularity metrics in modeling by observing the initial part of the cascade. This approach mainly falls into two categories: feature-driven and generative models. Feature-driven approaches are based on training a classification or regression prediction model following the extraction of a set of related temporal features, structural features, content features, or features defined on early adopters. Weng et al. [43] try to predict the future popularity of a meme given its early spreading patterns on Twitter. All memes are partitioned into classes based on the order of magnitude of the popularity, and in the prediction task, results show that early popularity of a meme is not a good predictor for future popularity and features based on community structure outperform the future success of a meme. In [49], the researchers systematically compare the performance of previously proposed learning models and features in literature. They find the Random Forest, a decision tree model, as the most successful model among linear and non-linear models, and user features are the most contributing features in model performance. Besides, PCA is applied to extract several features from the Twitter dataset to create predictive models for predicting the degree of retweeting of tweets in [50]. In [10], a set of related features is analyzed with the retweet numbers, and these features are later employed in developing different probabilistic models to predict whether a tweet will be more frequently retweeted than a certain threshold. Due to the lack of structural features in social media, content-related features are often employed in predictions. Sentiment analysis, one of the text-related operations, is discussed in [68] and reveals that most studies only implement less complex and noise removal steps in sentiment analysis. Beyond traditional machine learning models, in [51], a hybrid learning model employing CNN is proposed to learn the high-level representation of data, and the outcome of feature representations is used as input for the XGBoost model. Studies are of feature-driven approach, generally, focus on the suitable model and feature selection. Nevertheless, feature selection is generally a hand-crafted process based on

the user's previous domain knowledge and has a lack of understanding of whether sufficiently reflective for the process or not.

Generative models aim to formulate the general phenomena in social media, basing on the temporal or structural features, or explaining the observed event history or regarding the popularity cumulation of online contents. To model and predict the popularity dynamics, Shen et al. [52] present a general framework based on a reinforced Poisson process, incorporating three phenomena: the attractiveness of an item, a temporal relaxation for aging effect, and "rich-get-richer" mechanism. In [53], Hawkes's self-exciting point process [54] is employed to model the contribution of each tweet separately. In [55], the proposed model, SEISMIC, develops a statistical model on the theory of self-exciting points of each tweet, where the number of supporters is used as the influence of the user. SEISMIC manages to predict the final size of cascades with 15% error by only observing them for just 1 hour. DeepHawkes is proposed to understand the information cascades in [22] because generative models are not properly optimized for popularity predictions. Strong assumptions and oversimplification of reality in generative models are resulting in underperforming of real-world tasks.

2.2 Pre-processing in Social Media

While basic pre-processing steps are applied in many studies, social media data is less examined and analyzed before running the prediction model. In popularity prediction problems, pre-processing is mainly related to natural language processing (NLP) in text data of social media [56, 57], simple transformations of variables [4, 42], or feature-related analysis [50, 58] (i.e., selection, extraction).

In [56], researchers propose a pre-processing model for a clean dataset purified from unstructured user syntax and grammar in Twitter, implementing a text-mining process to clean the text data. Sequential text pre-processing steps are proposed to evaluate sentiment mapping of emojis in [57]. In [42], some features are log-transformed to account for their skewed distribution. Alike, a logarithmic transformation is applied to scale the unbounded numeric features in [4]. Morchid et al. [58] employ PCA to analyze the specific tweets to understand the behavioral difference between the highly retweeted tweets and opposites, retweeted only a few times. In [50], following PCA, the feature selection process is employed by sorting the features according to impact on the total variability.

For the outlier detection process in the social media data set, in [59], researchers employ the basic IQR (interquartile range) outlier detection method for detecting the extreme points in the social media usage among college students. Not for popularity prediction, but outliers in social media data sets are studied in the spatial analysis [60] and for social networks [61, 62]. In [61], extracted graph features defined with the social network are employed in two different algorithms (i.e., Expected-Maximization and Fuzzy logic) for detecting an anomaly. In [62], a distance-based outlier detection

method is used to transform Twitter data into regions; later on, these regions are employed for outbreak detection of influenza.

When it comes to data augmentation, in [63], researchers conduct a study for aggression detection in social media by using text data. Inspired by label-preserving transformations, primarily used to enrich images, they translate their tweets into four different intermediate languages and back translate to English to augment their dataset.

CHAPTER 3

METHODOLOGY AND RESULTS

This chapter proposes a novel pre-processing workflow called OutAug to predict the future popularity in social media at their publishing time, named cold-start. It uses a structured dataset obtained by combining the basic features and processed text data in which the lexicon approach is utilized to extract a range of linguistic features that points to the significance of the content. It has two main components, (1) detecting outliers to eliminate the misleading observations (e.g., tweets) and (2) augmenting data highly in favor of rare target cases to balance the dataset and to alleviate the uncertainty in reaction numbers. For this purpose, a pre-processing workflow capable of interpreting the significance of features, detecting the under- and over-performance tweets, adding systematic noise to observations, and increasing the volume of datasets in regression problems, is developed.

The proposed workflow is evaluated on social media datasets, where the purpose is to predict the popularity of a single tweet at their publishing time. Twitter has been central in disseminating information, imposing on crowds, and triggering social or political events. OutAug is validated on two Twitter datasets through extensive experiments. Also, comparisons across several baselines and state-of-art methods are performed.

The chapter first defines the predicting problem on a specific domain and situation (i.e., single tweet popularity at cold-start on Twitter), where the proposed workflow is applied and evaluated. Next, the steps of the proposed workflow are presented in detail. After that, the experiment details like scraping datasets, extracting features, parameter-tuning approaches, and comparison methods are described. Finally, the experiment results and their comparisons are presented.

3.1 Problem Definition

Suppose there are $\boldsymbol{X}$ observations (e.g., tweets) of interest, and each tweet $\boldsymbol{x}$ can be represented by a collection of static and dynamic features. While dynamic features (e.g., #retweets, #comments) are updated for each time interval $\boldsymbol{r}$ (e.g., minute, hour), static features (e.g., publishing time, content, #hashtags) remain as same. User features of the tweet owner (e.g., #followers, #alltweets) may change slowly throughout prediction time, $\boldsymbol{t^*}$**.** However, the cold start prediction problem aims to predict the

future popularity at t^* by employing the features at their publishing time (t_p or t_0), so user features would be considered constant.

Let S be the set of static features, and D_t is the set of dynamic features at time t. The collection of k number of static features can be represented as $S = \{s_1, \ldots, s_k\}$. The collection of dynamic features within an observing time window with size r up to time t can be represented as $D_{p:t} = \{D_p, D_{p+r}, \ldots, D_t\}$, and for m number of dynamic features $D_t = \{d_{1,t}, \ldots, d_{m,t}\}$. A single tweet x is the collection of them, $x_{<S, D>}$, and for n number of x, the dataset is $X = \{x_1, \ldots, x_n\}$

While predicting the popularity in regression, the dynamic features of a tweet are the target features. The purpose is to predict the $d_{i, t^*} \in Z^+ + \{0\}$ for a specific dynamic feature i (e.g., #retweet, #like, #comment) at a future time $t^* = t_p + \delta t$, where δt is called the lead time for predicting and formulated as $\delta t = lr$, r is the time interval and l is the number of time intervals. The prediction is based on the static features of the tweet itself. Therefore, this problem can be formulated as a learning function $f(S) = d_{i, t^*}$ that maps the static or dependent features to a countable popularity number at the future time t^* for a target feature i.

3.2 Proposed Workflow: OutAug

The proposed workflow starts with the reading dataset from the source and ends with evaluating final ML models. There are four main processes in the flow; (1) discretization process of the dataset by clustering the target feature, (2) train-test splitting process for a partially balanced test data in highly skewed datasets, (3) outlier detection process for eliminating under- and over-performance observations, and (4) data augmentation process for creating new synthetic observations to alleviate uncertainty in reaction numbers and to balance the dataset by synthesizing highly from tails.

The dataset includes the static features S and target feature d_{i, t^*}. The discretization process employs Jenk's Natural Break optimization method [69] in which the optimum cluster number is determined with the goodness of variance fit calculation. Clusters will later be employed in splitting data, outlier detection, and data augmentation processes. Base models provide the feature's importance to represent the weights in the KNN model for the outlier detection process. The outputs of this process are the extracted observations and k, the number of optimal nearest neighbors. The data augmentation process applies the differentiated SMOTE model on relatively clean data. It synthesizes new observations from virtual datasets constructed for each observation and their first N neighbors, N is a user-defined number. Also, a higher number of synthetic observations are created for the ones in tails. Thanks to synthetic data, ML models are developed with an enriched and balanced dataset.

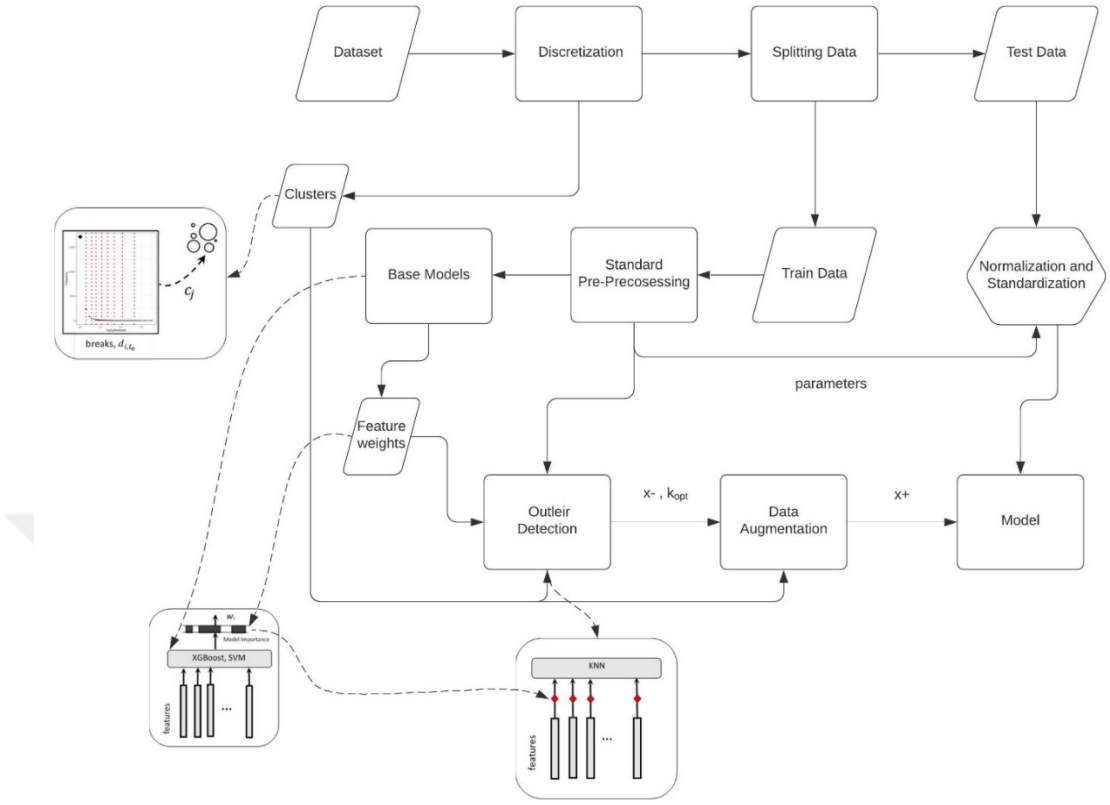


Figure 1: Overview of proposed pre-processing workflow

3.2.1 Discretization

The reaction value of a tweet cannot be a negative or floating number. It follows a similar distribution to Tweedie, containing many 0's and discrete continuous values. The dataset is discretized via Jenk's natural breaks algorithm running on the target feature. It tries to provide the best arrangement of retweet numbers into different classes by minimizing the variation within each range and maximizing the variance between ranges. The algorithm requires ***K***, the number of groups, to assign observations to one of the groups that the within-group distances are minimized. The ideal ***K*** number is determined by plotting the goodness of variance fit (GVF) for a range of numbers. Higher GVF value points to a better fit.

$$\bar{x} = \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right) \quad (1)$$

$$\sigma_A = \sum_{i=1}^{n} (x_i - \bar{x})^2 \quad (2)$$

$$\sigma_C = \sum_{i=1}^{n} (x_i - \bar{y}_j)^2, \quad x_i \in y_j \quad (3)$$

$$GVF = \frac{\sigma_A - \sigma_C}{\sigma_A} \quad (4)$$

$\bar{x}$ is mean of data, n is the sample size and σ_A is the sum of squared deviations for array mean. Calculating σ_C, the sum of squared deviations for class means, is an iterative process repeated for all break combinations. $\bar{y}_j$ is the mean of a cluster, where $j \in (1, \ldots, K)$ and y is one of the possible clusters.

3.2.2 Splitting Data

Randomization is an essential factor for train/test split by providing the same distribution for both datasets, train and test. The user determines the split rate (e.g., %80 train, %20 test). Although random sampling provides the same distribution for both datasets, it is not a precise split of factor variables. Stratified random sampling may be an optimal solution if the response attribute is a factor or the dataset contains important factor variables. These small amounts of difference could be significant at model evaluation. While using a stratified sampling method, correct and accurate splitting of factor variables is guaranteed.

In the social media domain, the datasets are generally highly right-skewed. While valuable ones are sparse among a wide range, many observations are stuck in a narrow range. There are some challenges to split a highly skewed dataset with random sampling. Following a random split, the distribution of the target feature would be the same for train and test datasets. A skewed test data complicates evaluating the model performance objectively with well-known metrics (e.g., RMSE, MAE). Errors of the instances falling into the highest parts of the cumulative distribution will dominate the model's overall performance so much that improvements in errors of the rare observations, the most valuable ones, do not fully reflect on the modeling performance. Therefore, it needs further plotting to observe the improvement of observations in tails.

A cluster-based sampling framework is proposed for highly skewed social media domain datasets to have a partially balanced test dataset. It creates two different folds; per fold comprises train/test data and covers all observations. The algorithm starts with the random sampling from clusters in the tail and stepwise shifting to the initial clusters while considering the partition of observations according to their clusters.

Algorithm 1 Train/Test split

function ClusterBasedSplit ($\mathcal{D}$, n)
// $\mathcal{D}$ – Dataset
// p – Desired percentage of test dataset
// y – Instances
// c – Clusters
// n – Desired the number of folds = 2

N ← #rows of $\mathcal{D}$ // number of instances
p ← user_input()
test_smp ← **Round** (N / p) // test dataset sample number
avg_size ← **Round** (test_samp / #number of c) // average sample number per cluster
all_test$\mathcal{D}$ ← {}
for i ← 1 to n **do**
for all c ∈ ($c_{\#number\ of\ c}$ *to* c_1) **do** // all clusters from last to first
clus$\mathcal{D}$ ← { $\langle x, y \rangle \in \mathcal{D}$: c = $\mathcal{D}$[clus] } // select the observations in the cluster
if size(clus$\mathcal{D}$) <= (2 * avg_size) **then**
smp_size ← **Round** (1/n * size(clus$\mathcal{D}$)) // sample size for the cluster
test$\mathcal{D}$ ← **Sample**({ $\langle x, y \rangle \in \mathcal{D}$: q = $\mathcal{D}$[clus] ∧ ∉ all_test$\mathcal{D}$ }, smp_size)
resid ← avg_size – smp_size // incomplete sample number from average cluster's sample number
supp ← **Round** (resid / (#number of c – a)) // allocate the residual number to remaining clusters
avg_size ← avg_size + supp // new avg. sample numb. for remaining clusters
else
smp_size ← avg_size
test$\mathcal{D}$ ← **Sample** ({ $\langle x, y \rangle \in \mathcal{D}$: q = $\mathcal{D}$[clus] ∧ ∉ all_test$\mathcal{D}$ }, smp_size)
end if
increase a by 1
all_test$\mathcal{D}$ ← all_test$\mathcal{D}$ ⁀ test$\mathcal{D}$
end for
all_train$\mathcal{D}$ ← $\mathcal{D}$ \ all_test$\mathcal{D}$
end for
end function

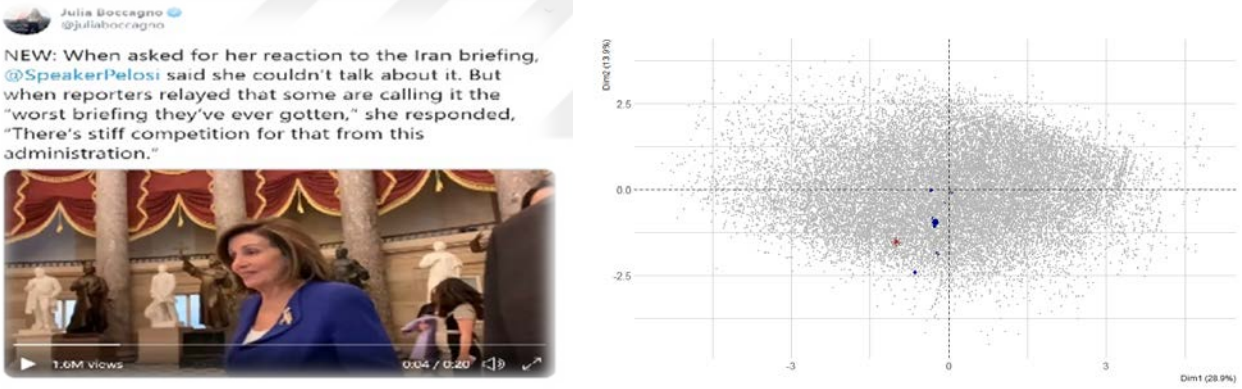
3.2.3 Outlier Detection

Many factors are influential in the number of reactions (e.g., comment, retweet, like) of tweets, and they can be grouped as internal and external factors. Internal ones are the primary indicators of a tweet, mainly represented as features (e.g., text content, number of followers, publishing time) and easily accessible via API. Nevertheless, some internal situations like being retweeted by a popular account boost the accessibility of a tweet, thereby the chance of being retweeted more. A different

endeavor such as following the graph or collecting extra information about nodes and links is needed to measure or detect these cases. Similarly, counting external factors in the modeling increases the complexity.

Intending to simplify the dataset from highly affected observations, we label tweets as outliers that show significant deviation from the tweets sharing similar features. For instance, consider the following tweet in Figure *2.a.* The features of that tweet and the neighborhood tweets are displayed in Figure *2.b.* While the particular tweet gets 5828 retweets at t_1(24 hours after tweeting), its first seven neighbors, except for the fifth (it has 12 retweets), get 0 retweets at t_1.

This tweet appears that it overperformed compared to the rest significantly. It may be due to several reasons (1) The collected features in the dataset are not enough to explain the overperformance. There could be another important feature that needs to be considered. (2) The content of the tweet is much appealing. The tweet owner could be the eye-witness to an incident or supports his claims with visual arguments. (3) A popular account interacts with someone's tweet. It could provide a remarkable diffusion over an extensive network.



(a) (b)

Figure 2: The global tweet id is 1215036082229784576, illustrated in (a). A video journalist is having a quick word with the Speaker of the United States House of Representatives. (b) This tweet is visualized in 2d dimension with a red shape. As demonstrated with blue shapes, neighborhood tweets are detected with the KNN algorithm proposed in my methodology. Note that there is a sparse among colored tweets due to the low proportion of variance (42.8%) of first and second components in PCA.

Furthermore, reaction numbers of tweets may show variability in observations due to uncertainty. The reasons could be numerous (1) Each tweet has different peak times on Twitter, making it impossible to define a specific crawling time. For example, tweets related to a specific topic may be crawled after 1 hour from their publishing times(t_0), and till this time, some of them already got their total reactions. After 3 hours

from t_0, some keeps getting reactions while some cut out at 2nd hour. This example may also be denoted as minutes or days. Also, some external situations may make old tweets popular again. The exact question is that what is the ideal time to measure the popularity of a tweet? There is no particular answer to that, so the noise is always available in the social media domain crawled datasets. (2) Tweets with a trending hashtag may not be noticeable due to the dynamic timeline, and users are usually only interested in recent ones. Sometimes, uncertainty in retweet numbers also implies the under-performance of tweets.

Removal of performance-related outliers from the training dataset aims to increase the model performance by facilitating model learnings. Another solution could be to seek a new alternative feature that will be able to explain this variance of the popularity of tweets better. However, to do that, a second data crawling process should be handled that causes more processing time in data acquisition and is also possible to face some limitations in API credentials. A novel approach to bypass those is to indicate the nearest neighbors of observations in the training dataset and find the cluster-based outliers according to ***k*** neighbors by minimizing MAE (mean absolute error) in the overall dataset.

K-nearest neighbor algorithm uses mathematical similarity of cases while finding the clusters and neighbors. Each observation is placed on an n-dimension surface according to their independent attributes; later, similarities can be calculated with Euclidian, Manhattan, or different hybrid distance functions (e.g., Gower). If all variables are interval scaled in KNN algorithm, Euclidian metric in *Equation (*5*)* or Manhattan metric in *Equation (*6*)* may be used to calculate distances between observations, and the input x gets assigned to the class with the largest probability for k number of neighbors in *Equation (*7*).*

$$d(x, x`) = \sqrt{(x_1 - x_1`)^2 + \cdots + (x_n - x_n`)^2} \tag{5}$$

$$d(x, x`) = |x_1 - x_1`| + \cdots + |x_n - x_n`| \tag{6}$$

$$(y = j|X = x) = \frac{1}{k}\sum_{i \in A} I\left(y^{(i)} = j\right) \tag{7}$$

In the Twitter dataset, some variables like #DayofTweet (the publishing day of tweet) or #VerifiedAccount (the account is verified or not) are non-numeric features (e.g., ordinal, binary). So, Gower's coefficient ensures standardization of each variable and calculates distances between two units as the sum of all the variable-specific distances is employed.

$$d_{ij} = d(i,j) = \frac{\sum_{k=1}^{p} w_k \delta_{ij}^{(k)} d_{ij}^{(k)}}{\sum_{k=1}^{p} w_k \delta_{ij}^{(k)}} \quad (8)$$

*Equation (*8*)* is the weighted mean of $d_{ij}^{(k)}$ with weights $w_k \delta_{ij}^{(k)}$. $d_{ij}^{(k)}$ is the distance between $x[i,k]$ and $x[j,k]$. $w_k \delta_{ij}^{(k)}$, where w_k equals weights[k] and $\delta_{i,j}^{(k)}$is in [0, 1], is the weight in the equation.

Each independent attribute should be appropriately weighted due to the strong effect of an observation's position on the n-dimension surface. Following standard pre-processing steps on the training dataset, ML models with hyperparameter tuning and fivefold cross-validation run on data to determine the feature weights. These models are employed as the baseline models for further comparisons. After using XGBoost, the model calculates the importance of features with different importance metrics (e.g., gain, coverage, frequency). While SVM with linear kernel enables to obtain feature's importance; other kernels cannot, due to transforming the data to another space. In my methodology, hyperparameter tuning for SVM includes different kernels (linear, polynomial, radial basis). So, the feature's importance of the XGBoost model is used to weigh the features in KNN.

The relative contribution of the corresponding feature to the model is implied by gain. In XGBoost, it is calculated by taking each feature's contribution to each tree in the model. In tree-based models, if there is an incorrectly labeled element in a branch, adding a new split on the feature to the branch creates two new branches, and each of these branches will become more accurate. The gain is the improvement in this accuracy and directly relevant with the importance of features.

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_R + H_L + \lambda}\right] - \gamma \quad (9)$$

Equation (9) can be decomposed as the scores on the new left-right leaf, the score on the original leaf, and regularization on the additional leaf. γ is used for pruning of trees. Here, G and H are the first and second derivatives of a loss function of the previous prediction for the set of examples ending in the given leaf. λ is the L2 regularization term on weights.

The relative importance of each feature via Gain metric transformed with $\log(1+x)\, for\, x \leq |1|$ for aiming a smoother weight is an input of the KNN algorithm to find the similarities among observations in *Equation (*8*).*

The KNN algorithm returns a large dissimilarity matrix. For each observation, the first 15 nearest neighbors are listed with their ids, and target feature (d_{i,t^*}) values of

neighbors are linked in a table, as demonstrated in *Table 1*. $x_{i,Id}$ represents the id value of ith observations, and $N_{i,k}$ points the kth neighbor of ith observation.

Table 1: Dissimilarity Matrix of KNN function

$RowId$	$d_{i,t*}$	$N_{i,15}$	…	$N_{i,1}$	$d_{N_{i,1},t*}$	…	$d_{N_{i,15},t*}$
$x_{1,Id}$	$d_{1,t*}$	$x_{N_{1,15},Id}$	…	$x_{N_{1,1},Id}$	$d_{N_{1,1},t*}$	…	$d_{N_{1,15},t*}$
$x_{2,Id}$	$d_{2,t*}$	$x_{N_{2,15},Id}$	…	$x_{N_{2,1},Id}$	$d_{N_{2,1},t*}$	…	$d_{N_{2,15},t*}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
$x_{n,Id}$	$d_{n,t*}$	$x_{N_{n,15},Id}$	…	$x_{N_{n,1},Id}$	$d_{N_{n,1},t*}$	…	$d_{N_{n,15},t*}$

$Median(d_{N_{i,1:j},t*})$ is calculated for each observation, $i \in (1, 2, \dots, n)$, n is the sample size, $j \in (1, 3, 5, \dots, 15)$. It is a step to determine the ideal k value, the number of neighbors, which provides minimum error (MAE metric) for further calculation. Compared to RMSE, where the deviation of errors is squared, MAE is a more appropriate metric for highly skewed datasets.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - \hat{x}_i| \tag{10}$$

In this stage, rather than calculating MAE between response value and predicted value of the model as per usual as demonstrated in *Equation (10)*, MAE is calculated by averaging the total MAE between the response value of each observation and each of their first kth neighbors' median value, by considering first kth neighbors' median values as predicted values. Also, this process separately calculates the MAE for each cluster to be a threshold for detecting outliers. k value, which minimizes the total error for all datasets, is selected as an ideal neighbor number.

$$x_{i,outlier} = \begin{Bmatrix} 1 & \rightarrow & MAE\left(d_{i,t*}, median\left(d_{N_{i,1:k},t*}\right)\right) > 1.5 \times MAE_{clus} \\ 0 & \rightarrow & Otherwise \end{Bmatrix} \tag{11}$$

If the MAE value, computed for each observation according to their kth neighbors' median value, is 1.5 times higher than the cluster-based MAE value of kth neighbors, this observation would be evaluated as an outlier. Outlier Detection Mechanism conveys the information of removed outliers and information about neighbors of each observation to Data Augmentation Mechanism.

3.2.4 Data Augmentation

When conducting a supervised classification with machine learning algorithms such as tree-based models, working with an imbalanced dataset may lead to misleading accuracies. An imbalanced dataset refers to the situation where the proportion of classes are not equal to each other. It is prevalent in some domains like social media, fraud detection, or disease diagnosis. The minority is the most valuable one for these domains; naturally, there is an inverse relationship between the number and the importance of observations. Machine learning models are generally sensitive to the proportion of classes and tend to favor the class with the most significant proportion of observations. This problem may be surpassed by employing over-sampling, under-sampling, or hybridizing them to balance the classes. Data augmentation is an over-sampling method that aims to increase data by adding similar copies of existing data or creating new synthetic data from existing data. It is a well-known problem, and solutions are already implemented many times in classification modeling.

An imbalanced dataset in classification corresponds to the highly skewed dataset in regression. A primary solution to overcome the skewed distribution of the target feature is to employ a numeric threshold value to divide it into classes. Even though this method has been applied many times in the social media domain [4, 28, 29], it transforms the problem from regression to classification by eliminating numeric sensitivity and roughly labeled observations in the intersection of classes.

The data augmentation process aims to increase observations by creating new synthetic data from existing observations. It uses SMOTE algorithm to generate synthetic observations. It creates virtual datasets per number of observations. Each virtual dataset has 51 observations, which include a specific data point and its closest 50 neighbors.

The ideal k number was obtained in the previous component. The dataset is divided into two classes using k number; first class (A) that will be augmented, consisting of that specific observation and its closest k number of neighbors, while others are in the second class (B). SMOTE algorithm runs on each virtual dataset and generates k synthetic observations for the minority class in each dataset within the first half of clusters that do not have enough samples to correspond in the train/test split process. For the second half of clusters, the number of synthetic observations is doubled ($2k$). This way aims to synthesize more data from clusters that fall within the high tail of the distribution. For instance, the number of reposts in the temporary dataset is 51 and the k value is 11; the algorithm generates 22 synthetic data points closest to this observation in clusters from the high tail of distribution using SMOTE. Forwhy such numbers are sporadic in social media datasets, and the number of zeros and ones is quite frequent.

Besides augmenting the dataset, this method aims to smooth the dataset by alleviating the uncertainty that may fluctuate in small margins in retweet numbers sourced from time-manner scraping, user behaviors, or external factors.

Objective Function. The objective function of XGBoost comprises three terms, loss function, pruning term, and a regularization term. It indicates the difference between actual values and predicted values. It is formulated as follows:

$$\mathcal{L}_{overall} = \mathcal{L}_{predict} + \gamma T + \frac{1}{2}\lambda O_{value}^2 \quad (12)$$

where λ is the regularization coefficient for scaling output value, O_{value} is the output value for the leaf that minimizes the whole equation. For pruning term (γT), T is the number of terminal nodes or leaves in a tree, and γ is a user-definable penalty to encourage pruning. $\mathcal{L}_{predict}$ is the entropy loss which penalizes the prediction of the model for future tweets as follows:

$$\mathcal{L}_{predict} = \sum_{i=1}^{n} \mathcal{L}(y_i, p_i) \quad (13)$$

$$\mathcal{L}_{(y_i,p_i)} = \frac{1}{2}(y_i, p_i)^2 \quad (14)$$

n is the number of samples, y_i stands for the y-axis value from one of the observed values, and p_i stands for a prediction. The model optimizes the output value from the first tree, and prediction (p_i) can be replaced $p_i^0 + O_{value}$, p_i^0 is the initial prediction ($mean(y_i)$) and O_{value} is the output value from the new tree.

Pruning tree means to stop splitting a node when the model encounters a negative loss in the split. In the XGBoost model, pruning occurs after the whole tree is built, working backward by removing splits that are no positive gain. So, it plays no role in deriving output values. The overall objective function becomes:

$$\mathcal{L}_{overall} = \left[\sum_{i=1}^{n} \mathcal{L}(y_i, p_i^0 + O_{value})\right] + \frac{1}{2}\lambda O_{value}^2 \quad (15)$$

SVM for regression aims to find an appropriate line or hyperplane in higher dimensions to fit the data by defining the error tolerance of the model. The objective function aims to minimize the coefficients. The coefficient is the l2-norm of the coefficient vector. In *Equation (16)* and *(17)*, ε $(epsilon)$ is the maximum error, ξ is the deviation of a point from the margin, C is the cost that points the tolerance for the points outside of ε.

$$MIN\frac{1}{2}\vec{w}^2 + C\sum_{i=1}^{n}|\xi_i| \quad (16)$$

$$|y_i - w_i x_i| \leq \varepsilon + |\xi_i| \quad (17)$$

3.3 Experiments

The process, from obtaining the datasets used in the experiments to evaluating the proposed workflow, is explained here. This section also provides information about standard pre-processing steps, including normalization and feature selection, the comparison methods for the evaluation, experimental settings, and the computational cost of the proposed workflow. Pre-processing is explained in detail as it is the main objective of this thesis.

3.3.1 Datasets

The experiments were performed on two different cause célèbre; state parties' counter and preemptive maneuvers, starting with the assassination of a famous Iranian general by the USA, and Prince Harry and Megan stepping back as senior members of the Royal Family. The same procedure was applied in collecting datasets for both events. These cases were chosen to respect their social significance in the eye of the public. They happened suddenly, appealed to every segment of society by encouraging them to tweet or retweet about the issue, and were active events indicating that new following statements could have been made later. Eventually, two cause célèbre were selected to create America-Iran Conflict (AIC) and Step Back Royal (SBR) datasets.

3.3.1.1 Data Scraping

Twitter Inc. offers different application programming interfaces (API) for developers. Search API is used to acquire tweets and their associated features in this scraping process, which has three types of subscription methods; Standard, Enterprise, and Premium Search. Enterprise and Premium Search requires a high-cost subscription to use advanced searching features. The Standard Search is favorable for developers because of providing enough data free of charge. The limitations are:

- Developers can make 180 requests in 15-min. intervals, and a maximum of 100 tweets can be acquired in each request.
- It searches against a sampling of recent tweets published in the past seven days by only focusing on relevance, not completeness.

Developers need keys and tokens to access the data via API. A developer account on Twitter has the right to start a project and get Consumer API Keys and access tokens. Twitter supports many libraries for different programming languages. Datasets were collected through the Twitter Streaming API by related keywords in Python. Retweets were not included. Tweets in English were included to employ further content analysis. The geolocation-based search was preferred to avoid the adverse effects of the time zone. The width of locations was determined by being in the same time zone because the data was only gathered in 2 hours window. When conducting a geo-search in Twitter, it first attempts to find the tweets, including the queried geocode in a tweet; otherwise, it tries to find tweets where the owner's location is used as the queried geocode. It means that it is possible to have a dataset with these two kinds of tweets.

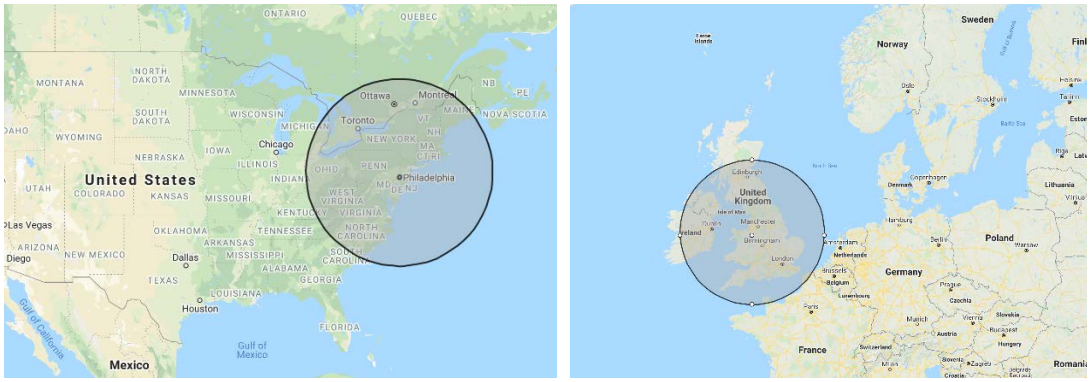
3.3.1.2 Strategies

For the AIC dataset, tweets containing the 'iran' keyword were scraped between 07.01.2020 and 13.01.2020 in North-East America and big cities of Canada (Latitude=39.959796, Longitude= -75.181894, Radius= 500 miles) shown in Figure *3.a*. Tweets which were between 21:30 and 23:30 were scraped. This process was repeated three times for the same collected tweets to retrieve the retweet number and share: 23-25 hours, 71-73 hours, and 119-121 hours after each tweet's publishing time.

During the week of the scraping process, issues about Iran were trendy topics.

Timeline of issues are:

- 03.01.2020 - USA killed top Iranian general in Baghdad airstrike,
- 06.01.2020 - Stampede killed 50 mourners at the burial in Iran,
- 07.01.2020 - Iran launched a series of missiles to US bases in Iraq,
- 07.01.2020 - A plane belongs to Ukraine Airlines was crashed in Iran, many Canadian citizens have died in the plane crash,
- 08.01.2020 - Trump made a statement after Iran missile attacking,
- 10.01.2020 - Iran admitted to hitting down the commercial plane wrongly,
- Also, many bilateral statements were performed.



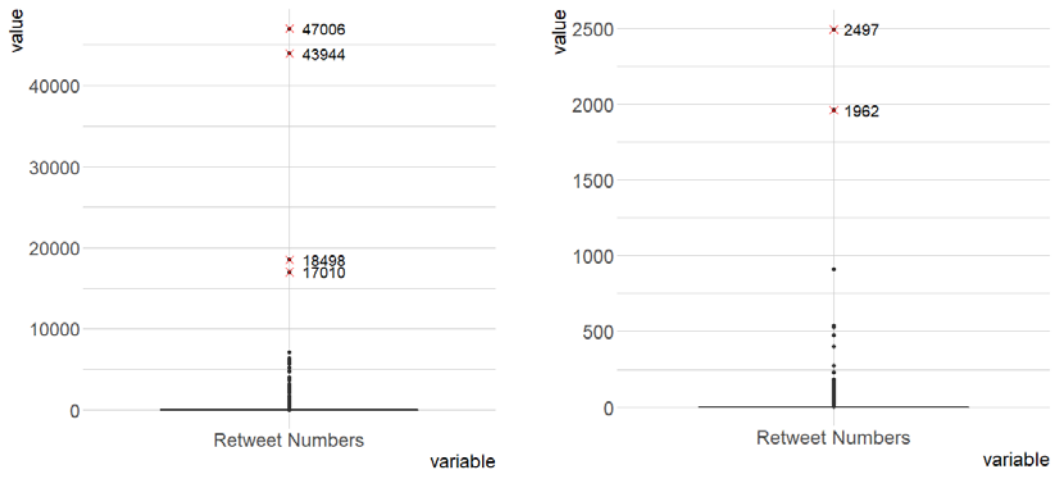
(a) (b)

Figure 3: Queried locations for data scraping process

For the SBR dataset, tweets that include one of the keywords, 'Meghan', 'Royal Family', or 'Harry' were scrapped between 08.01.2020 and 14.01.2020 in England and Ireland (Latitude = 53.417825, Longitude = -3.032107, Radius = 260 miles) shown in Figure *3.b.* Tweets that were published between 21:30 and 23:30 were scraped. This process was repeated three times for the same collected tweets: 23-25 hours, 71-73 hours, and 119-121 hours after each tweet's publishing time.

Meghan and Harry stepped back from Royal duties on 08.01.2020 by leading to public indignation.

After the first investigation of the distribution of the target feature, some instances were considered as outliers due to the lack of information indicating this apparent deviation, and they were removed from the datasets. Otherwise, these observations' measurement errors would highly affect ML models' overall performance, directing the generalization of study to higher observations rather than higher tails. Also, they would distort the algorithm of Natural Breaks, leading some clusters into having only a few observations.



(a) AIC dataset (b) SBR dataset

Figure 4: Apparent outliers in AIC and SBR datasets

After apparent outliers are removed from datasets, Table *2* shows the basic statistics of datasets, number of tweets, and the total number of unique users who tweet in datasets.

Table 2: Basic statistics of datasets

Dataset	Duration	#Tweets	#Users
AIC	Jan 07 – Jan 13 (2020)	23184	14683
SBR	Jan 08 – Jan 14 (2020)	4069	3280

3.3.1.3 Features

There are twenty attributes in datasets grouped under four general characteristics; id, single tweet features, user features, and popularity indicators. Single tweet and user features are expected to represent enough information to predict the popularity indicators in datasets.

Table 3: Details of the features in the datasets

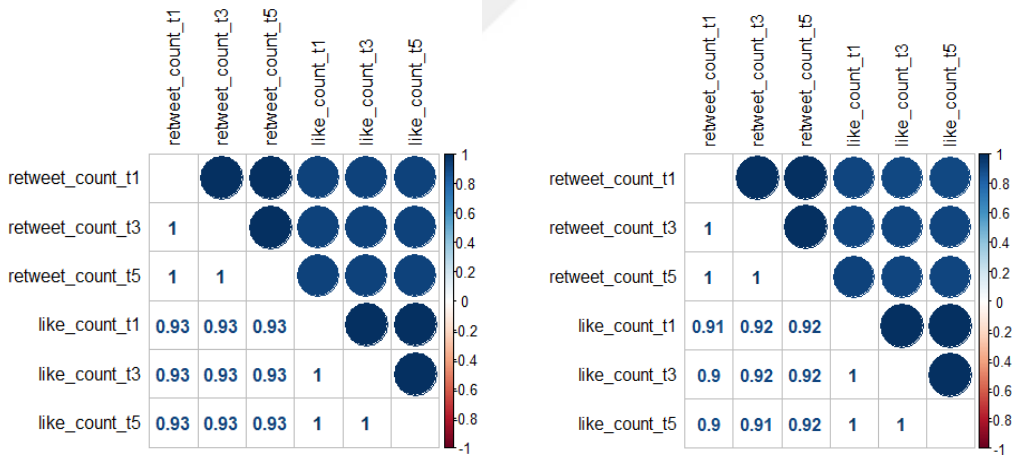
Feature	Attribute Name	Remarks	Type
Id	id	Unique numbers for identification of tweets in the dataset	String
Single Tweet Features	timedelta	Number of days between the tweet posted and account created	Integer
	day_of_tweet	Shows the publishing day of the tweet	Categorical
	num_of_hashtag	Number of hashtags	Integer
	num_of_mentions	Number of mentions	Integer
	num_of_url	Number of URLs	Integer
	num_tokens_content	Number of words in the tweet	Integer
	polarity	Text sentiment polarity	Float
	subjectivity	Text sentiment subjectivity	Float
	readability	Text readability score	Float
User Features	followers_count	Number of accounts that follow the owner of the post	Integer
	followees_count	Number of accounts which the owner of the post follows	Integer
	num_of_alltweets	Total number of tweets posted by the user	Integer
	verified_user	Is the user verified?	Boolean {0,1}
Popularity Indicators	retweet_count_t1	Number of retweets that a tweet has received in 24 hours after its posting time	Integer
	retweet_count_t3	Number of retweets that a tweet has received in 72 hours after its posting time	Integer
	retweet_count_t5	Number of retweets that a tweet has received in 120 hours after its posting time	Integer
	like_count_t1	Number of likes that a tweet has received in 24 hours after its posting time	Integer
	like_count_t3	Number of likes that a tweet has received in 72 hours after its posting time	Integer
	like_count_t5	Number of likes that a tweet has received in 120 hours after its posting time	Integer

Tweet contents were converted into structured features. Firstly, symbols of hashtag(#) and mention(@), URLs, smiles, and other meaningless characters that are not valid in

dictionaries were removed from the full text of the tweet. Following the split of sentences into tokens, words that are listed in the stop-word dictionary[2] were extracted.

Sentiment analysis[3] carried out on clean text. Polarity shows positive and negative statements. Subjectivity is defined in the range of [0,1], and the higher values indicate more personal opinion, emotion, and judgment. The readability feature shows the ease of readability. Flesch-Kincaid Grade Level [70] is a readability formula that assesses a text's approximate reading grade level. In this formula, a score of 9.3 means that a ninth-grader would be able to read the document.

Two types of features were incorporated into the proposed method: static and dynamic features. User features and single tweet features are defined as static features because the data was downloaded in a short period, and hence, the change in user features is negligible. Popularity indicators are dynamic features besides being a target feature. The number of comments that a tweet has received is not directly reachable via Twitter API. Retweets number indicates the tweet's popularity. Also, retweet numbers are positively correlated among themselves and the number of likes. Note that there is no significant change in retweet numbers for the majority of the tweets. Some tweets face a significant change, but it does not affect the overall tendency. When the correlation between retweet numbers and the short life-span of a content [40] are considered together, retweet numbers at t_{24} was selected as the target feature for the modeling.



(a) AIC dataset (b) SBR dataset

Figure 5: Correlation plot of reaction features

[2] Stopwords dictionary from NLTK library is employed, detailed information about NLTK package is in https://www.nltk.org/api/nltk.html.
[3] Textblob library is used for sentiment analysis in Python, detailed information about functions can be found in https://textblob.readthedocs.io/en/dev/quickstart.html#sentiment-analysis.

3.3.1.4 Data Overview

In order to avoid complexity, descriptive statistics and pre-processing steps will be explained through AIC dataset. The detailed information for SBR dataset can be found in the APPENDIX.

Table 4: Descriptive Statistics of AIC Dataset

Attribute Name	Min. Value	Max. Value	Median	Mean	SD	Skewness
id	-	-	-	-	-	-
timedelta (unit: day)	0.0	4826.0	2657.0	2423.00	1334.01	-0.26
num_of_hashtag	0.0	30.0	0.0	0.37	1.23	6.57
num_of_mention	0.0	50.0	1.0	1.13	2.78	11.68
num_of_url	0.0	4.0	0.0	0.50	0.53	0.36
num_tokens_content	0.0	64.0	13.0	14.22	7.48	0.50
polarity	-1.0	1.0	0.0	0.01	0.25	-0.26
subjectivity	0.0	1.0	0.4	0.35	0.30	0.39
readability	-3.5	32.8	10.3	10.75	6.01	0.06
followers_count	0.0	44883986.0	909.0	72478.00	1031511.3	34.55
followees_count	0.0	370905.0	906.0	3502.00	10628.26	12.58
num_of_alltweet	1.0	1894208.0	13967.0	55281.00	135348.33	6.28
retweet_count_t1	0.0	7161.0	0.0	10.52	146.79	30.35
retweet_count_t3	0.0	7446.0	0.0	10.86	150.92	30.30
retweet_count_t5	0.0	7486.0	0.0	10.88	151.14	30.32
like_count_t1	0.0	24267.0	0.0	30.44	484.60	34.56
like_count_t3	0.0	25160.0	0.0	31.51	500.73	34.49
like_count_t5	0.0	25292.0	0.0	31.58	501.82	34.50

Timedelta feature shows the age of the account in the unit of days. Having a small skewness and a high standard deviation value presents the similar interest of young and old accounts for the topic. When descriptive statistics of follow-related features are considered together, users generally tend to have more followees than followers. The wide range of spread in the number of all tweets implies the big difference in user habits. While some users are mainly viewers of tweets, some users continually tweet. Skewness, a measure of the symmetry in distribution, is theoretically expected on [-1, +1] range to have a Gaussian normal distribution. The reaction numbers' features, foloowers_count, and followees_count have high skewness values, implying a long distribution tail.

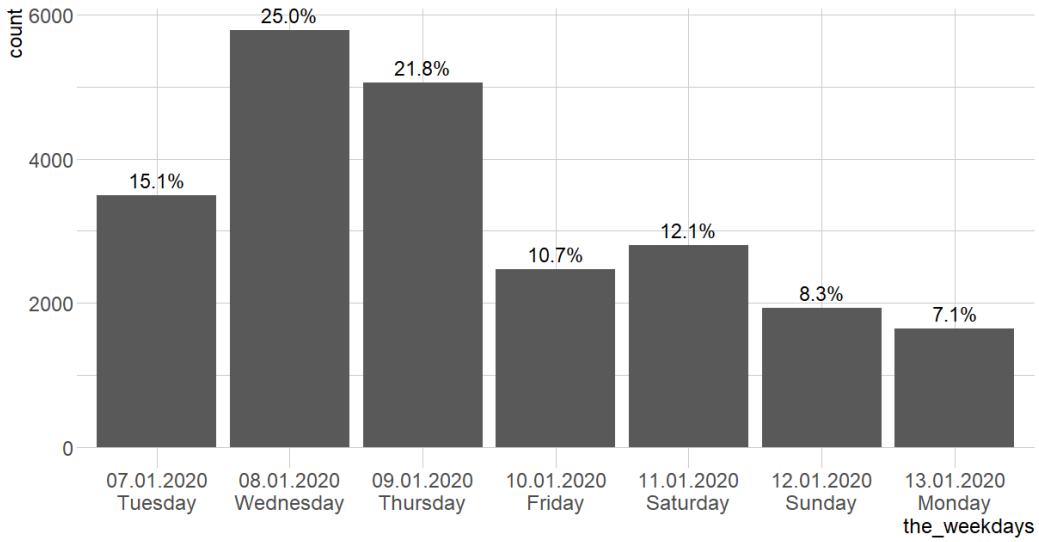


Figure 6: Timeline of tweets. The histogram shows the number and percentages of the collected tweets for each day in the week

The blue verified badge on Twitter lets people know that an account of public interest is authentic. 10.41% of the users in the dataset are verified users.

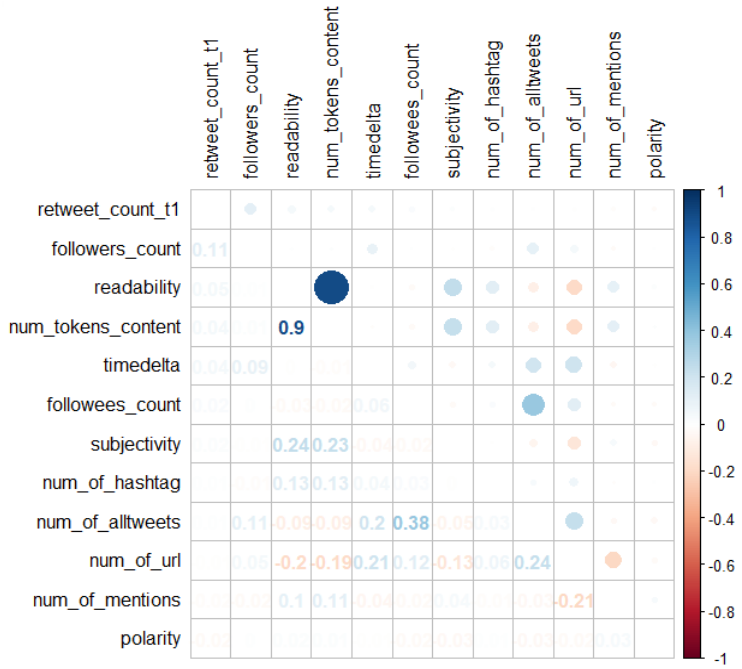
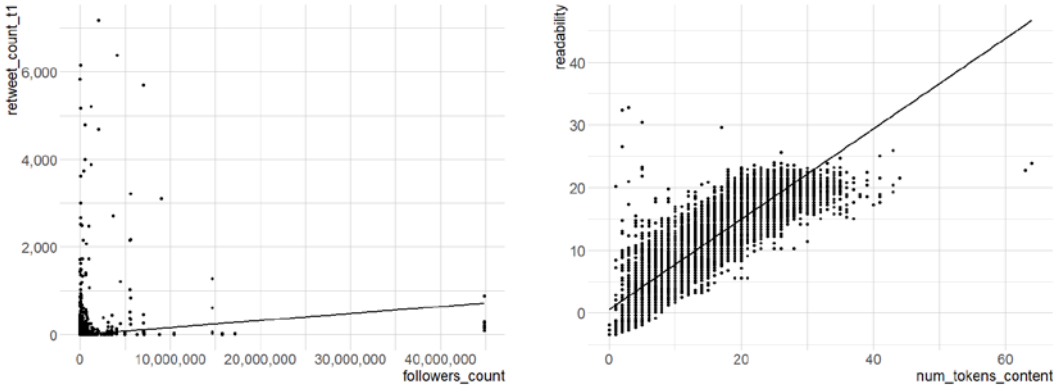


Figure 7: Correlation matrix between the numeric variables and the target feature

The retweet number is lowly correlated with the number of followers. An account creates a network with its followers. The width of the cluster increases the chance of receiving more retweets because of directly reaching more users. On the other hand, tweets from users having a relatively low number of followers may leap to other networks to reach more people thanks to the retweet mechanism. As the Flesch-Kincaid Grade Level Readability score is calculated according to word and sentence lengths, a high correlation is expected between the readability score and the number of tokens in content.



(a) (b)

Figure 8: Bilateral plots of correlated features

Figure *9* points that verified accounts are more likely to being retweeted. Due to the non-normal distribution of retweet numbers and different sample numbers for each group, a non-parametric and unpaired statistical test is employed in 95 percent confidence interval with Wilcoxon rank-sum test. Null hypothesis (H_0) is there was no difference in retweet numbers of verified and unverified users' tweets, and alternative hypothesis (H_a) is that verified users' tweets were more retweeted than unverified users' tweets. A significantly smaller p-value (2.2e-16 < 0.05) implies stronger evidence in favor of the alternative hypothesis.

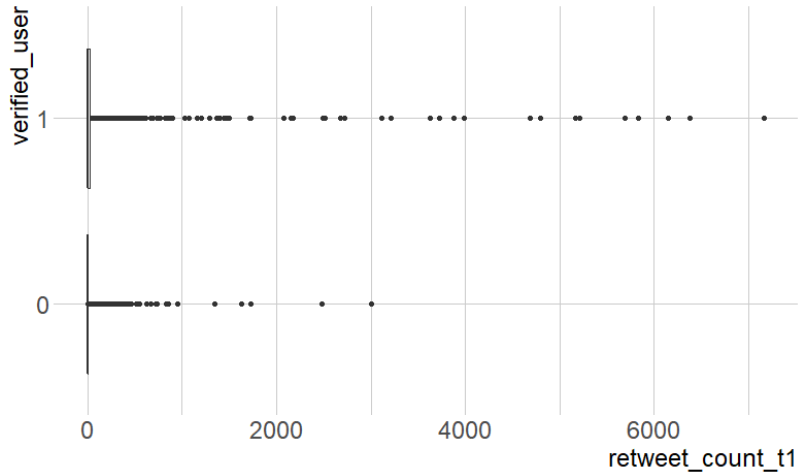


Figure 9: Verified user-retweet number plot

As illustrated in Figure *10*, only a few users have more than 25 tweets in the dataset. Most users are represented with only a single tweet. While many tweets have a few retweets, a handful of them goes to thousands. Table *5* and Figure *11* demonstrate the fat-tailed right-skewed distribution of the target feature.

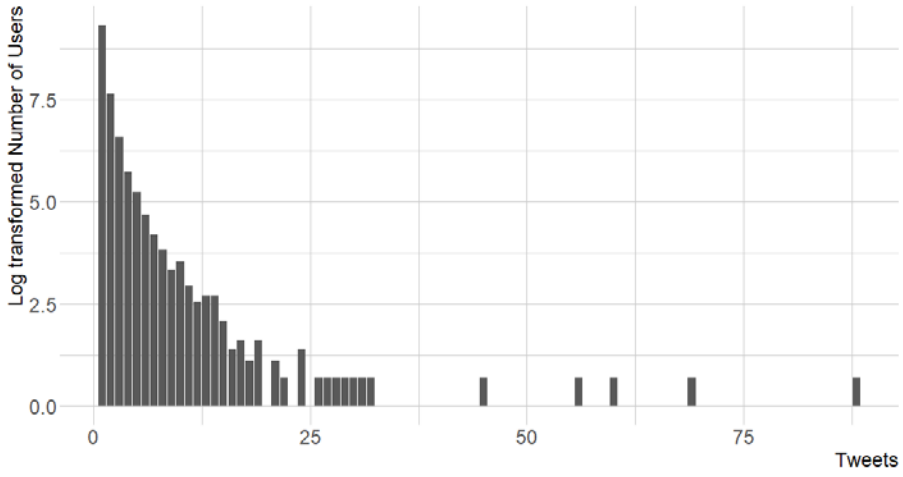
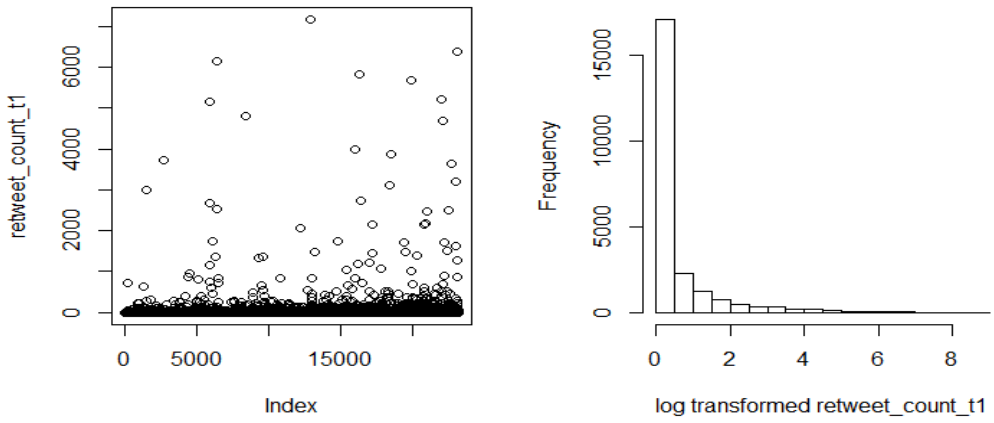


Figure 10: User-Tweet histogram

Table 5: Descriptive statistics of retweet_count_t1

Min.	**1st Qu.**	**Median**	**Mean**	**3rd Qu.**	**Max.**
0.00	0.00	0.00	10.52	1.00	7161.00



(a) Plot (b) Histogram

Figure 11: Fat-tailed distribution of retweet_count_t1

3.3.2 Comparison Methods

The proposed flow, OutAug, was compared with several baselines and state-of-art approaches. In order to evaluate the contribution of proposed architecture to prediction effectiveness, the same machine learning models were run in three different samples of the same dataset: (S1) complete dataset (S2) dataset with outliers removed startup (S3) dataset enriched with data augmentation after outlier removal, see Table *6*.

XGBoost and Support Vector Machines for Regression (SVM) are the chosen machine learning models for evaluation. While SVM considers the points within the decision boundary lines, gradient boosting mechanisms minimize the loss when adding new trees to decision tree models. RMSE and MAE metrics are used for performance measurement.

In-degree for observations in a *k*-nearest neighbors graph (KNN_IN), aggregated k-nearest neighbors distance over different *k*'s (KNN_AGG) and local outlier factor (LOF) algorithm are handy algorithms for outlier detection in clustering and other multidimensional domains, referenced by [64–66] respectively.

Table 6: Description of employed models

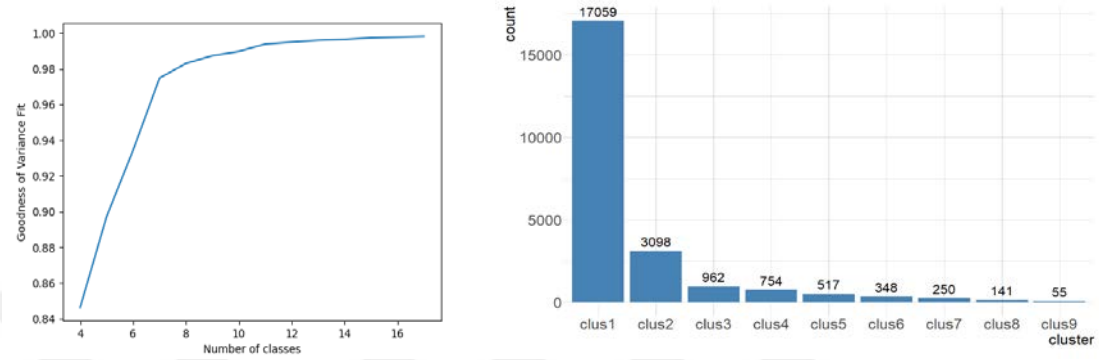
Sample	Model	Description
S1	Mean	calculates the mean value of the train dataset's response attribute and uses this constant prediction value on the test dataset.
	ML_base	refers to a hyper-parameter tuned XGBoost and SVM models on the training dataset, and runs that model on the test dataset.
S2	ClusOut	refers to a hyper-parameter tuned model on a training dataset where outliers were discarded by the cluster-based outlier detection method described in Section 3.2.3 and runs that model on the test dataset.
	KNNin.k5, KNNin.k9, KNNin.k13	refers to hyper-parameter tuned models on train dataset where outliers were discarded using KNN_IN function with k = 5, k = 9, k = 13, respectively, and runs that model on the test dataset.
	KNNAGG.k5-13	refers to a hyper-parameter tuned models on train dataset where outliers were discarded using KNN_AGG function with min.k = 5 and max.k = 13 parameters and runs that model on the test dataset.
	LOF.k5, LOF.k9, LOF.k13	refers to hyper-parameter tuned models on train dataset that outliers were discarded using LOF function with k = 5, k = 9, k = 13, respectively and runs that model on the test dataset.
S3	ClusAug	refers to hyper-parameter tuned models on train dataset where proposed data augmentation is applied in Section 3.2.4 after outlier removal as described in Section 3.2.3 and runs that model on the test dataset.

3.3.3 Experimental Settings

The k-fold cross-validation is a handy resampling process that splits data samples into the number of folds referred by k. For k number of loops, it provides unseen data for the processes like performance evaluation of ML models or optimization of model parameters. However, using the same cross-validation process and data for tuning and evaluating a model will likely lead to an optimistically biased model performance evaluation. To overcome it, hyper-parameter tuning of the model is nested under the evaluation of the model, known as the nested cross-validation procedure. In this method, two loops are running on a dataset. While the outer loop is splitting data as train and test folds, the inner loop splits the training fold as training and validation folds.

My thesis used *Equation (1)* to *(4)* to assign the observations to clusters by minimizing error within clusters and maximizing error between clusters. Figure *12.a* is employed to determine the ideal cluster number as 11 ($k = 11$). Jenk's algorithm breaks the dataset two times at 0 value because of the dataset's dominant proportion of non-

retweeted tweets. Henceforth, there are 9 clusters in the dataset. $clus_1$, having only the observations whom reaction number is 0 accounts for %73.6 of all dataset. Clusters' break numbers are 1, 3, 6, 13, 28, 64, 175, and 762. Throughout the tail clusters, observation numbers are explicitly decreasing, as shown in Figure *12.b.*



(a) (b)

Figure 12: (a) Plotting of goodness of variance fit according to the different number of classes. (b) Histogram of clusters in the dataset

Algorithm 1 was employed to provide the outer loops of nested cross-validation procedures and have relatively balanced test datasets to evaluate model performance with basic evaluation metrics like RMSE and MAE. More verbally, due to the dominance of frequent observations in the test dataset, improvements in the prediction of rare observations do not equally reflect basic evaluation metrics.

The pre-mentioned algorithm splits the dataset into two folds. The proportion of test data to the dataset is selected as 10%, and the optimal number per cluster is calculated by dividing the sample number of test data by the number of clusters. For an ultimately balanced dataset, the algorithm shifts from up tail cluster to leftmost one, trying to fulfill the optimal number by taking 50% random samples from clusters. If a cluster cannot provide enough samples, the lacking samples are shared among clusters. Clus2 and Clus3 do not lack samples due to enough samples to compensate for the required sample number, as shown in Figure *13*. This process created test datasets where frequent and rare observations are partially balanced. The distribution of train and test datasets are graphed in Figure *14*.

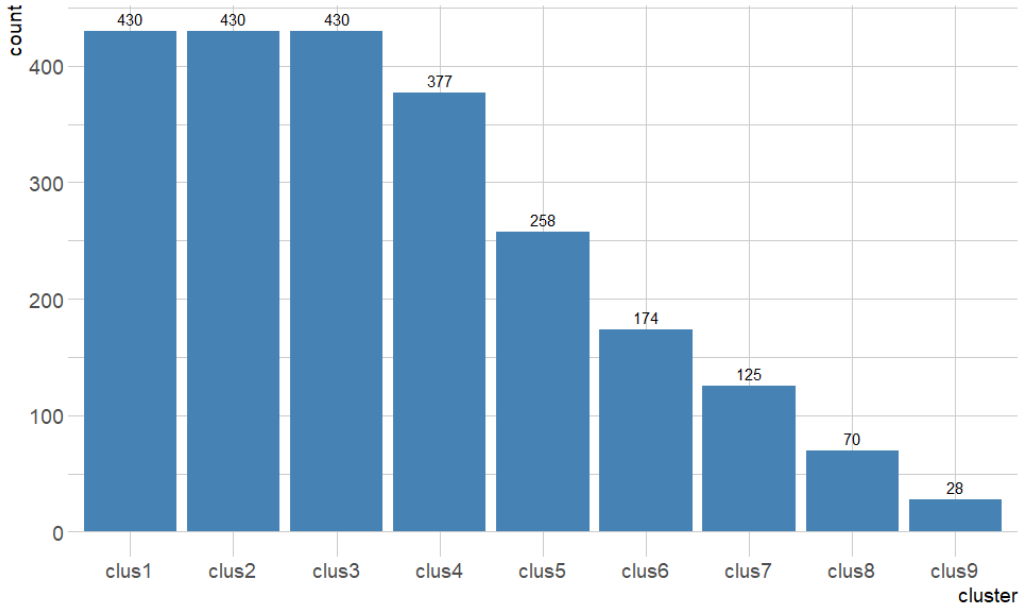
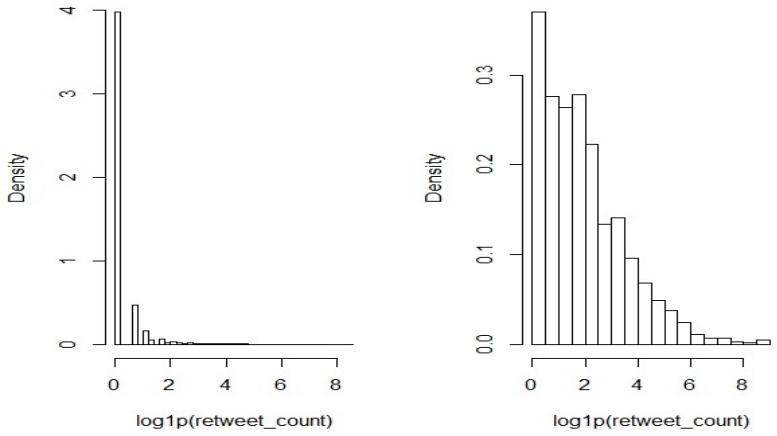


Figure 13: Histogram of clusters in test data



(a) Train data (b) Test data

Figure 14: Distribution of train and test datasets

In the experiments, each model was hyperparameter tuned with a random search method. One hundred random sets of parameters were selected within the specified range in Table *7* and Table *8*. The specified ranges were estimated with numerous attempts, handled with grid search. The explanation of abbreviated or domain-specific technical terms can be found below in Table *8*.

Table 7: Tuned parameters and intervals of XGBoost model

Parameter	Type	Constant	Min. value	Max. value
learner	-	regr.xgboost	-	-
predict_type	-	response	-	-
objective	untyped	reg:squarederror	-	-
eval_metric	untyped	rmse	-	-
nrounds	integer	2000	-	-
early_stopping_rounds	integer	10	-	-
booster	discrete	gbtree	-	-
max_depth	integer	-	5	10
min_child_weight	numeric	-	0.2	2
subsample	numeric	-	0.3	1
colsample_bytree	numeric	-	0.5	1
eta	numeric	-	0.01	0.4
lambda	numeric	-	0	2
gamma	numeric	-	0	1

Table 8: Tuned parameters and intervals of SVM model

Parameter	Type	Constant	Min. value	Max. value
learner	-	regr.ksvm	-	-
predict_type	-	response	-	-
type	untyped	epsilon-regression		
eval_metric	untyped	rmse	-	-
nrounds	integer	2000	-	-
kernel	discrete	rbfdot, polydot, vanilladot	-	-
sigma	numeric	automatic calculation		
epsilon	numeric	-	0	1
cost	numeric	-	0	5

Eval_metric Evaluation metric
nround Number of rounds
rbfdot Radial basis kernel “Gaussian”
polydot Polynomial kernel
vanilladot Linear kernel

3.3.4 Computational Cost

The complexity of a model/algorithm is often expressed with Big O notation, which defines an upper bound. It measures how fast or slow an algorithm will perform for input size, n. If the computational cost will be measured for a flow, the total cost is the aggregation of individual methods.

The proposed workflow starts with a Natural Breaks algorithm for detecting clusters in the target feature. It uses the Jenks Natural Breaks Algorithm, which has time complexity $O(k \times n^2)$, k is the number of classes and smaller than n. Due to being an iterative process repeated inner loops, it is a quadratic complexity time algorithm. Following the discretization process, train/test splitting is employed. This process runs on the clusters and is only interested in the sample number within the clusters. The complexity is $O(n)$, n is the number of clusters, and n<10 for both datasets. So, the complexity of the algorithm is negligible.

Each ML model in the literature has different complexity of time. XGBoost and SVM, employed in my dissertation, have time complexities $O(nlog(n))$ and $O(n^2)$, respectively. In ClusOut, the KNN algorithm runs on the training dataset to determine the neighbors. There are many optimized versions of KNN to reduce the time complexity for massive datasets. Typical KNN model has time complexity $O(nd)$, d is the dimension of each sample.

The ClusAug component comprises a differentiated SMOTE model. SMOTE algorithm runs on the tiny virtual dataset that is created for each observation. It has the time complexity $O(mn)$, m is the number of observations in the tail and shows how many times the algorithm will run. n, the cardinality of virtual datasets, is a fixed number at 51 (observation and its closest 50 neighbors) in the proposed methodology.

3.4 Standard Pre-processing

This section plots and interprets descriptive statics and distributions to identify any missing data and noisy observations. Then dataset is normalized and pre-processed appropriately. Also, correlations among features are calculated to prevent redundancy.

Normalization and transformation. Normalization means adjusting values with a different range of scales to a notionally standard scale. Kolmogorov-Smirnov normalization test is applied to numeric features following a singularization of repetitive values. It returns an approximate p value indicating the normalization of the feature. ($p < 0{,}05$) is the threshold to accept the null hypothesis, demonstrating no difference in data distribution from the normal distribution. None of the features in the dataset show normal distribution.

Transformation methods may be applied to obtain a near-normal distribution. Popular transformation types are log, square-root, and box-cox transformation in the literature.

Log transformation is used in high skewed datasets. It is not applicable for 0 value, and the solution is to add small numbers to the feature (Laplacian correction). Square-Root transformation is primarily used in datasets, which is a count of something. It does not work on negative values; the solution is to use the absolute value of the feature. Box-Cox transformation is configurable via lambda (λ) by supporting both square root and log transform. All three methods are applied to the features. For log transformation, if a variable has some values in the [0-1] range, $\log(x + 1)$ is used; for square-root transformation, the absolute value function is applied to negative ones, and for Box-Cox transformation, if lambda is equal to 0, the transformation process is interrupted. Although there is a decrease in the feature's skewness values after transformations, none can pass the confidence interval to accept the null hypothesis that points to the normal distribution ($p < 0.05$).

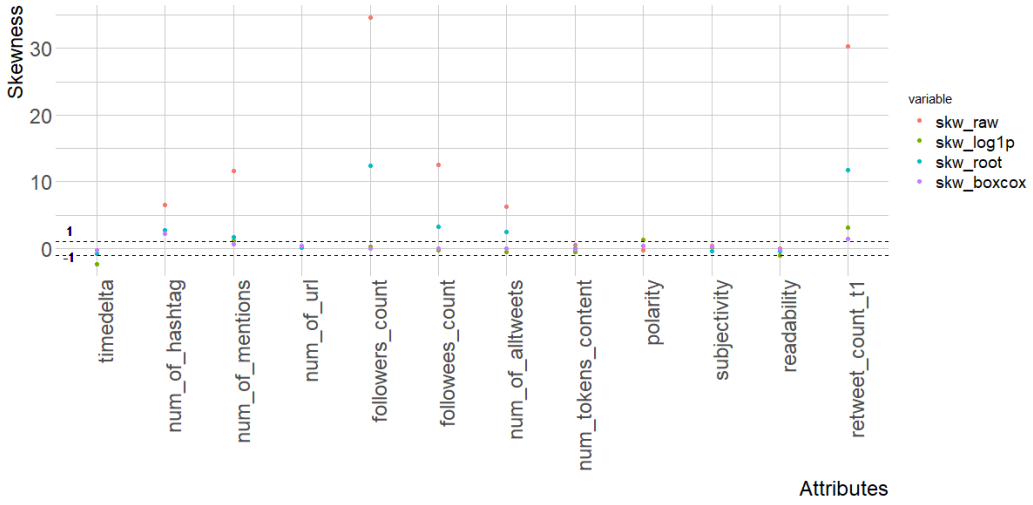


Figure 15: The plot of skewness of attributes with different transformations

Attributes with a high skewness value are transformed to an acceptable skewness value in the range [-1, +1]. As seen in Figure *15*, Box-cox transformation shows much convergence to this range. The distributions of the transformed followers_count feature are illustrated as an example in Figure *16*.

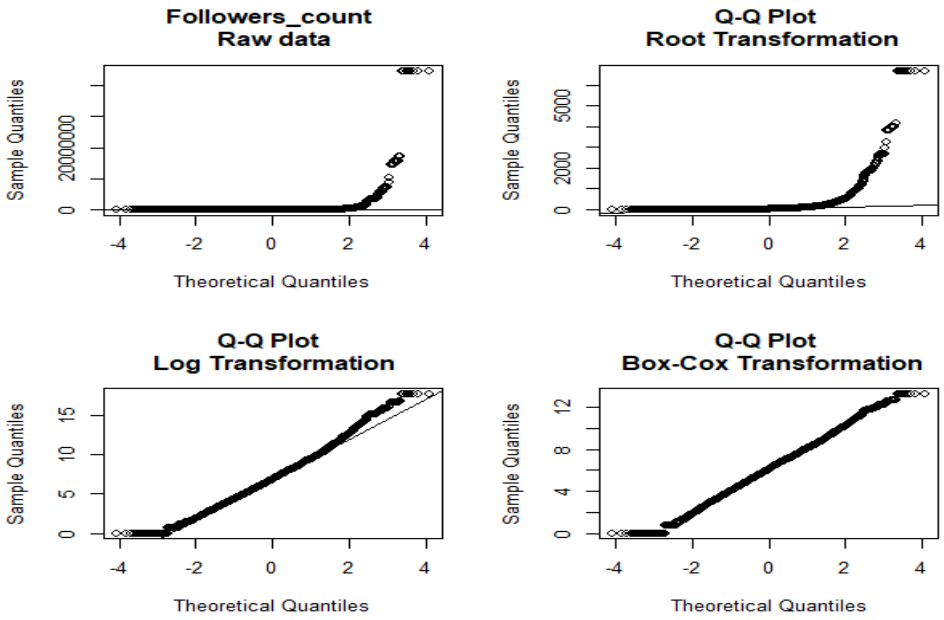


Figure 16: Distribution of followers_count according to different transformations

Retweet_count_t1 feature shows a fat-tailed distribution. Although box-cox transformation provided the minimum skewness value for retweet_count1, log transformation is applied due to providing continuity in numbers when distributions are plotted.

Table 9: Skewness and kurtosis values of retweet_count_t1

Popularity_indicator	Data	Skewness	Kurtosis
retweet_count_t1	Raw data	30.35	1093.27
	Square-Root	11.82	208.85
	Log transformation	3.09	11.28
	Box-cox	1.46	0.61

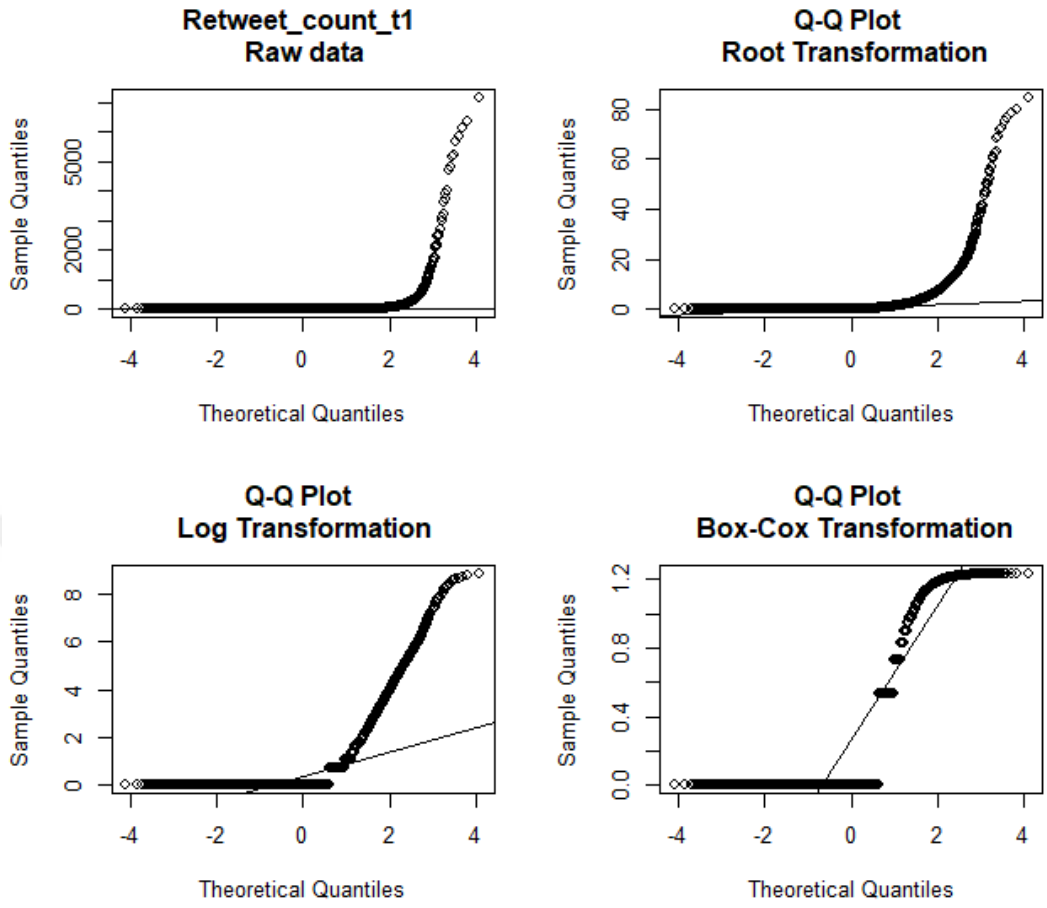


Figure 17: Distribution of retweet_count_t1 according to different transformations

Feature Selection. Principal component analysis (PCA) preserves as much information as present in the dataset by transforming to principal components and selecting them according to their loadings' magnitude.

Dataset was prepared by discarding factor variables, id (unique identifier), num_tokens_content (highly correlated with “readability”) and retweet_count_t1 (target feature). Box-cox transformation is applied to the train dataset variables whose skewness value is not in the range [-1,1]. The pre-processing transformation is applied to scale and center the variables. Later, PCA runs on the ten attributes of train datasets. The scree plot of PCA components is illustrated in Figure *18*.

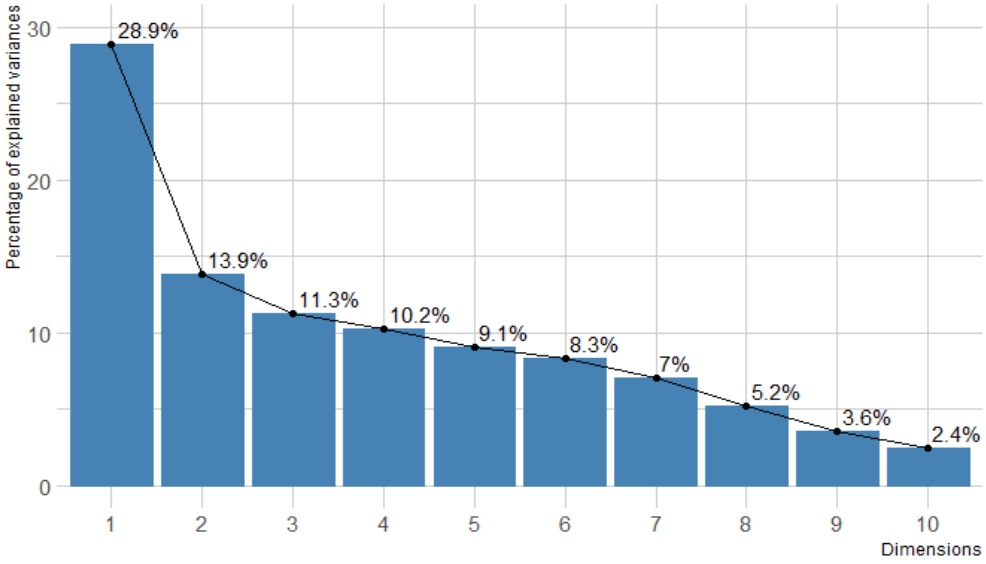


Figure 18: Scree plot of PCA components and their variances

Table 10: Importance of PCA components

Component (factor f_i)	Eigenvalues e_{vi}	Eigenvalues % Variance $v(f_i)$	Cumulative % variance
1	1.70	28.93	28.93
2	1.18	13.87	42.81
3	1.06	11.26	54.07
4	1.01	10.24 ~ (1/10)	64.30
5	0.95	9.08	73.39
6	0.91	8.32	81.71
7	0.84	7.04	88.75
8	0.72	5.24	93.99
9	0.60	3.56	97.55
10	0.49	2.24	1.00

The Kaiser criterion is retaining factors with eigenvalues equal to or greater than 1. From this point of view, the fifth eigenvector is selected as a cutout vector. According to Figure *18* and Table *10*, the first four eigenvectors represent 64% of the dataset.

Orthogonal rotation (VARIMAX) is applied to the first four eigenvectors, and their loadings are calculated. Each factor is independent, and correlations between variables and factor loadings are presented in Figure *19*. It points to the effect of features on the four eigenvectors. It also implies the most important features to represent the dataset.

Following the feature selection methodology in [58], it is possible to determine a threshold to discard some features for the final dataset. However, the cumulative variance of four eigenvectors is not high enough, and also, varimax loadings of any feature are not too low. So, none of the features are discarded with PCA.

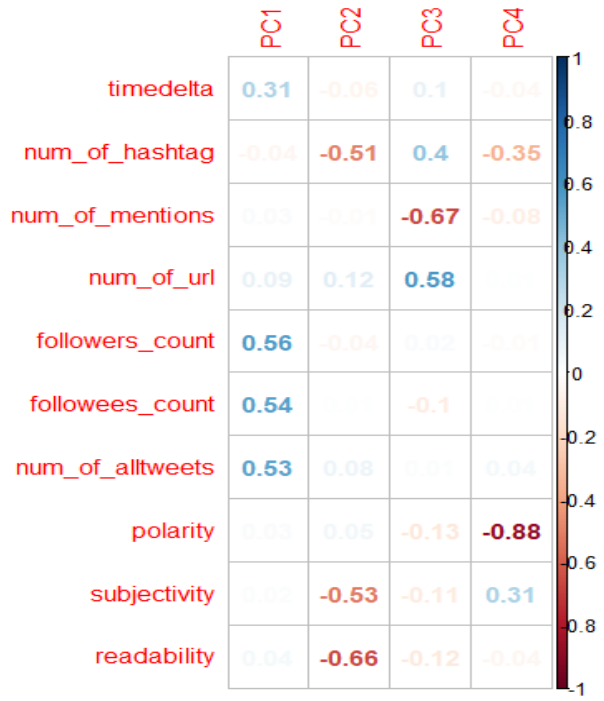


Figure 19: VARIMAX loadings of the first four eigenvectors according to features

Final pre-processing includes these steps; (1) discarding num_of _tokens, (2) applying the box-cox transformation to the variables; the skewness values of which are not in the range [-1, 1] (num_of_hashtag, num_of_mentions, followers_count, followees_count, num_of_alltweets), (3) scaling and centering all numeric features, (4) one-hot-encoding of the categorical variables, (5) log transformation of the target variable, scaling and centering and (6) applying the same procedure for the test dataset.

3.5 Results

A comprehensive set of results is presented in this section. Based on the experiment settings mentioned earlier, the effectiveness of the proposed workflow on the prediction is demonstrated by comparing with several baselines and state-of-art methods.

3.5.1 Performance Comparison

The predicting performance of ML models on datasets employing OutAug and other outlier detection methods are compared within each other. Note that the comparison does not aim to analogize different ML models; it only shows the improvement of prediction performance of a particular ML model by following the proposed pre-processing workflow. The results are organized to answer the following questions:

1. Overall, how well could OutAug help to predict popularity at cold-start compared with the baseline methods?

2. How does the discarding outlier in train data affect the performance of ML models?

3. Does the data augmentation process provide an improvement compared to ML models running on outlier discarded datasets?

4. How much the overall improvement in predictions reflects the clusters in the tail?

3.5.2 Overall Performance

Mean and ML_base are the starting models and applied to the dataset following a standard pre-processing. The ML models' performance was compared between three models (KNNin, KNNAGG, LOF) from the literature and the ClusOut component from the proposed workflow for outlier detection. The ClusOut model can determine the ideal k number within a specified range value of k. However, k value is an input for models from the literature to be used for comparison. $k\ in\ (5, 9, 13)$ is provided for comparison KNN algorithms to measure the effect of k to the performance, and ClusOut employs the same range of k for an equal basis. ClusOut manages to improve the learning capacity of ML models higher than the other three methods. Also, ClusAug, the proposed data augmentation method, manages to improve the prediction performance of ML models. Table *11* indicates that OutAug provides the lowest MAE and RMSE value for both datasets.

Table 11: Performance comparison of methods. Detailed information for abbreviated model names is listed in *Table 6*

Performance Results on the Test Dataset.								
	AIC dataset				SBR dataset			
	XGBoost		SVM		XGBoost		SVM	
MODELS	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
Mean	2.74	2.08	2.74	2.08	2.77	2.01	2.77	2.01
ML_base	1.68	1.20	2.09	1.53	1.83	1.25	2.33	1.63
KNNin.k5	1.68	1.20	2.08	1.53	1.84	1.26	2.28	1.61
KNNin.k9	1.69	1.21	2.01	1.47	1.87	1.27	2.32	1.62
KNNin.k13	1.68	1.20	2.06	1.51	1.84	1.26	2.35	1.64
KNNAGG.k5-13	1.69	1.20	**1.99**	1.46	1.85	1.26	2.34	1.63
LOF.k5	1.69	1.20	2.07	1.51	1.87	1.27	2.27	1.59
LOF.k9	1.68	1.20	2.16	1.58	1.88	1.28	2.29	1.61
LOF.k13	1.70	1.21	2.10	1.54	1.88	1.28	2.34	1.63
ClusOut	**1.64**	**1.17**	2.00	**1.45**	**1.79**	**1.21**	**2.25**	**1.55**
ClusAug	**1.59**	**1.14**	**1.70**	**1.23**	**1.74**	**1.15**	**1.94**	**1.34**

The data augmentation process (ClusAug) improves the SVM model better than the XGBoost model. XGBoost, an iterative method, is a gradient boosting algorithm in which the individual models are decision trees. Each new decision tree learns the mistakes of the existing ensemble and attempts to correct them. This iterative process provides inherent robustness to gradient boosting models against similar observations of a dataset.

Performance results imply the overall success of workflow. However, the proposed architecture aims for a better prediction for observations within a high tail. Figure *20* and Figure *21* indicate the cluster-based improvements for XGBoost and SVM models, respectively. SVM model has managed a better improvement following the proposed workflow. As shown in the Figures, tail clusters take the lion's share of the improvements. Horizontal lines imply the overall performance of models for the whole dataset. RMSE metric is much sensitive to the skewness of the dataset because of squaring the residuals. MAE, not sensitive to tails, provides a softer comparison for models.

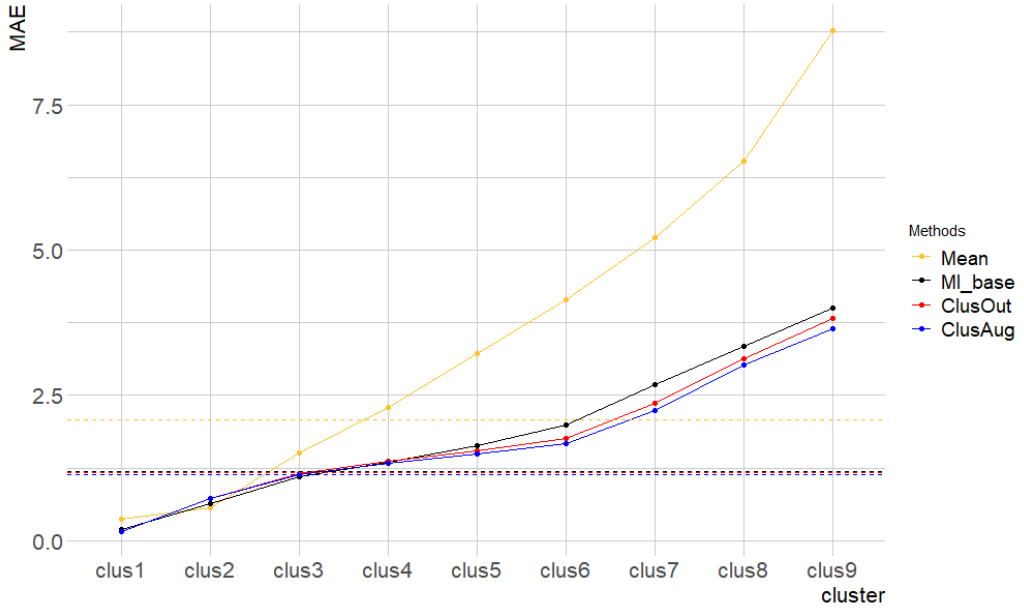


Figure 20: XGBoost model performance per cluster according to methods

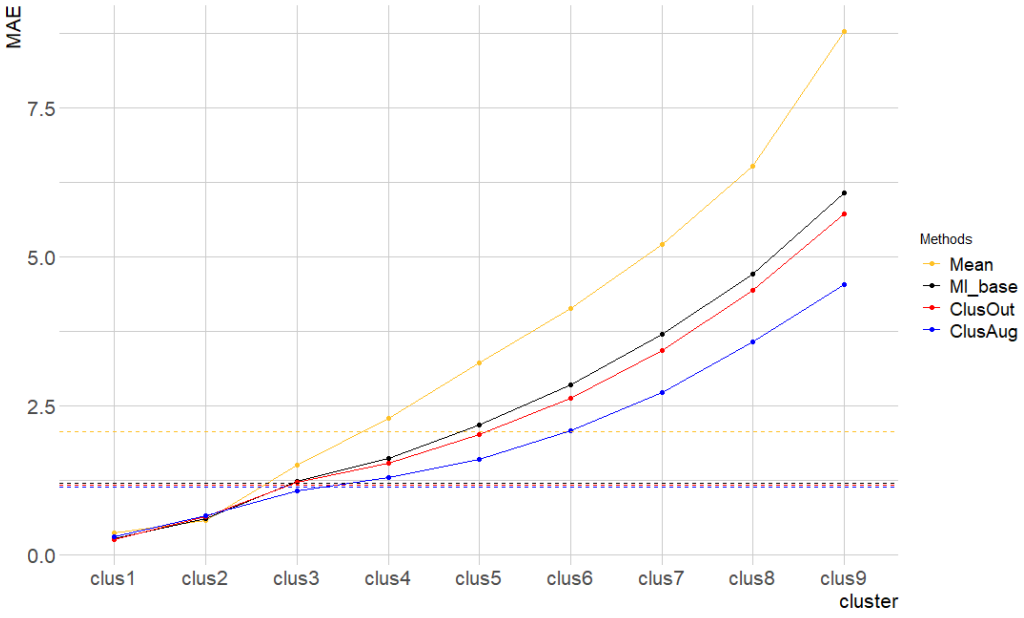


Figure 21: SVM model performance per cluster according to methods

3.5.3 Outlier Detection Mechanism (ClusOut)

This section presents the results of steps in the proposed outlier detection mechanism (ClusOut). The results will be provided for AIC dataset to make it easy to follow. For SBR dataset, the results can be found in APPENDIX

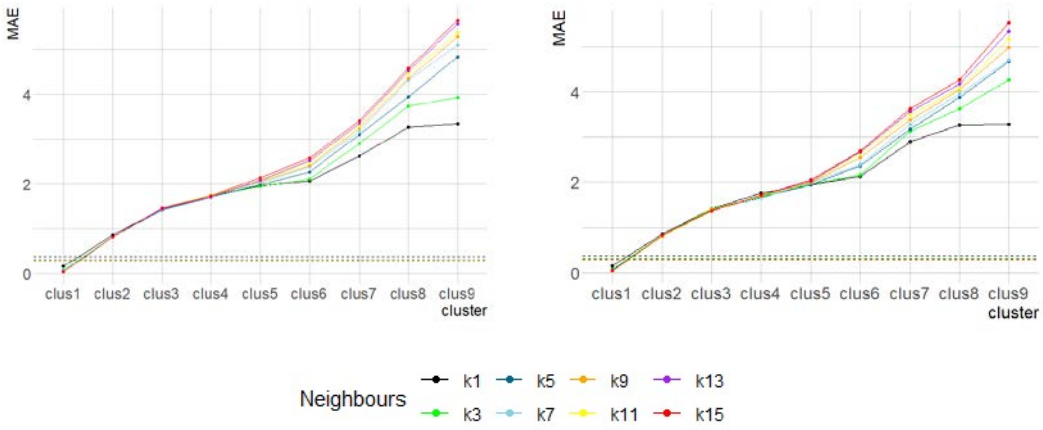
ClusOut starts with the detection of the importance of features via the XGBoost model. This model is run on pre-processed train data of each fold with hyperparameter tuning and five-fold cross-validation. The gain metric is employed to imply the importance of attributes. Besides standing as weights in the KNN model, Table *12* shows the most critical features in the dataset.

Table 12: Importance of attributes

Attribute	Fold1	Fold2
followers_count	0.4151±0.0261	0.4069±0.0126
num_of_alltweets	0.1149±0.0090	0.1236±0.0033
verified_user	0.0936±0.0075	0.0282±0.0051
followees_count	0.0846±0.0059	0.0937±0.0022
timedelta	0.0818±0.0051	0.0974±0.0024
readability	0.0732±0.0104	0.0769±0.0031
polarity	0.0339±0.0030	0.0528±0.0052
num_of_mentions	0.0238±0.0025	0.0207±0.0006
subjectivity	0.0213±0.0019	0.0384±0.0043
num_of_url	0.0198±0.0016	0.0176±0.0004
num_of_hashtag	0.0163±0.0012	0.0160±0.0014
day_of_tweetSunday	0.0058±0.0009	0.0025±0.0003
day_of_tweetTuesday	0.0041±0.0008	0.0061±0.0007
day_of_tweetWednesday	0.0032±0.0006	0.0046±0.0002
day_of_tweetFriday	0.0026±0.0003	0.0021±0.0003
day_of_tweetSaturday	0.0023±0.0002	0.0053±0.0006
day_of_tweetThursday	0.0018±0.0002	0.0037±0.0004
day_of_tweetMonday	0.0009±0.0003	0.0027±0.0004

As expected, followers_count is the essential feature found by the model. The content of a tweet does not significantly affect predictions, or my Natural Language Processing (NLP) models cannot comprehensively analyze the content of a tweet properly. Also, the publishing day of the tweet has been found as the least significant.

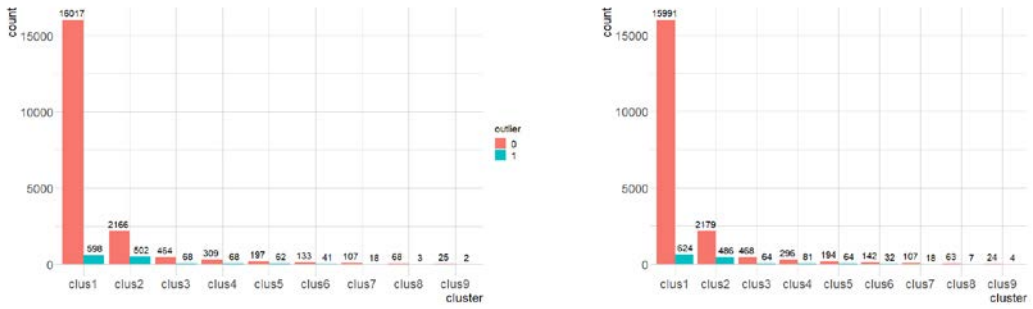
Fifteen neighbors are grouped by odd *k* numbers (1, 3, 5, 7, 9, 11, 13, 15), and the median for each *k* class is the estimation to compute the MAE value within clusters as illustrated in Figure *22*. Smaller k values provide a better fit for tail clusters. Due to the lack of data for tail clusters and high margins between observations in tails, they share similar response values with only a few neighbors.



(a) Fold1 (b) Fold2

Figure 22: MAE within clusters according to different *k* neighbors

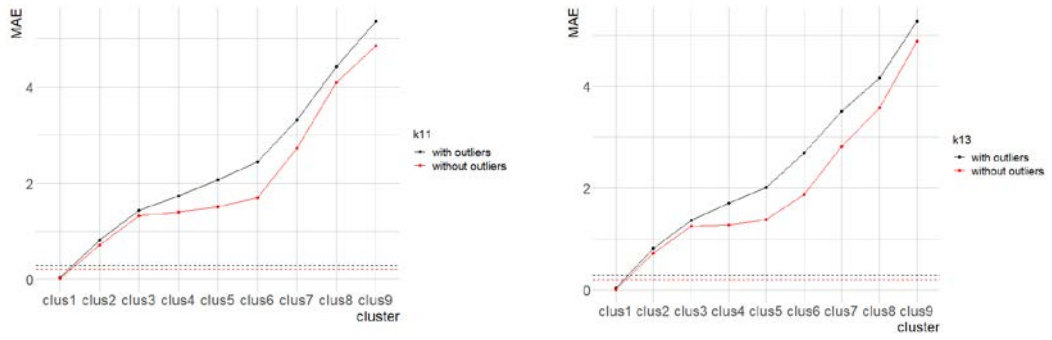
The horizontal dashed lines in Figure *22* show the overall error of observations by median values of k-neighbors. Notwithstanding the closeness of total errors for all observations, k11 and k13 provide the minimum MAE for Fold1 and Fold2, respectively. According to my algorithm (see Section 3.2.3 *Equation (11)*), if the MAE value computed for each observation according to their ideal k neighbors' median value is 1.5x higher than the cluster-based MAE value of k neighbors, these observations are evaluated as an outlier.



(a) Fold1 (b) Fold2

Figure 23: Outliers for per cluster

Model performance slightly increases after outliers are discarded from the training dataset. The models try to decrease the difference between prediction and actual value in each iteration; eliminating outliers helps to model learn patterns better.



(a) Fold1 (b) Fold2

Figure 24: MAE within clusters after outliers are discarded from the training dataset

Each outlier detection method has detected a different number of outliers, shown in Table *13*. Note that these methods have been applied to the training dataset. The models which are shown in Table *11* discarded these outliers before training the ML models. Their test results show the effects of these outliers in terms of performance in Table *11*.

Table 13: Outlier numbers for different methods

Methods	**Number of Outliers**			
	AIC dataset		SBR dataset	
	Fold1	Fold2	Fold1	Fold2
KNNin.k5	673	703	130	123
KNNin.k9	1393	1377	124	124
KNNin.k13	103	288	26	23
KNNAGG.k5-13	1961	1909	475	480
LOF.k5	3823	3845	627	615
LOF.k9	3511	3522	613	574
LOF.k13	3326	3297	604	580
ClusOut	1362	1380	160	88

3.5.4 Data Augmentation Mechanism (ClusAug)

This section provides descriptive information about the proposed data augmentation process (ClusAug). It works on the clean dataset conveyed from ClusOut. Also, it gets the information of neighbors, ideal *k* number, and clusters for observations.

There are 9 clusters in AIC dataset. Observations in clusters that do not have enough samples for the splitting process are selected to be augmented ($cluster\ in\ (4, 5, \dots, 9)$). Not having enough samples for the splitting process is an indicator of under-

representation. The mechanism creates virtual datasets per observations in which selected clusters and their first 50 neighbors. Selected observation and their first k neighbor are labeled in the same class; other observations are in a different class. The ideal neighbor's number is $k11$ and $k13$ for Fold1 and Fold2, respectively. SMOTE algorithm has run in these virtual datasets. For the first half of selected clusters ($cluster4, cluster5, cluster6$), SMOTE generates a new k number of data points; the number of new data points are doubled ($2k$) for the second half of the selected clusters. Note that Table *11* shows the performance of ClusAug when these instances in Table *14* are augmented in the training dataset.

Table 14: Number of augmented instances

Number of Augmented Instances

	AIC dataset		SBR dataset	
	Fold1	Fold2	Fold1	Fold2
ClusAug	13816	16315	2043	2052

CHAPTER 4

CONCLUSION AND DISCUSSION

In this chapter, an interpretable pre-processing workflow, called OutAug, is presented to prepare an ideal dataset for predicting the future reaction number of posts at their publishing time, named the cold-start problem. The workflow is applied to the retweet prediction problem in Twitter, enables fast diffusion of information. A novel architecture is developed, which detects the performance-related outliers and augments the dataset in favor of desired observations. OutAug also eliminates the time-manner uncertainty by creating synthetic observations, having very similar features with their neighbors but having slightly different retweet numbers. The difference is sourced from the wide margin of retweet numbers for observations in tails. It also enables the interpretation of efficient features for prediction. The proposed workflow was compared with the baseline methods, and it achieved a slightly better prediction performance for both datasets.

Removing outliers detected by the ClusOut component from the training dataset has improved the model's learning capacity. Outlier detection may be applied to all datasets, including train-test data; although this provides better performance for the prediction model, it may cause data leakages in my proposed workflow. ClusOut has qualified by precisely detecting similarities and dissimilarities among neighbors. ClusOut can also be used for testing the representativeness of features in modeling. Observing the outlier numbers which ClusOut detects by tendering features provides to analyze the effect of features. Also, if the outlier number is relatively high, it means the lack of related features. Else adding a new feature decreases the number of outliers; it implies the new feature’s importance for the modeling.

All performance-related outliers detected by the proposed ClusOut mechanism were evaluated with qualitative research by identifying basic components, which led to them being an outlier. While being an outlier with overperforming is a destroying factor for model pattern learnings, it is a welcome situation for users of Twitter. Besides the benefits of this study to the academic community, comprehension of the leading factors provides practical implications for sectoral usage. The common point and special features of overperforming tweets ascertained from the visual inspection of tweets are listed below. The listed factors do not guarantee a higher retweet performance; however, they may affect retweet performance under appropriate atmosphere and conditions.

- ✓ Being an eye-witness of a critical situation and put forth it with visual arguments,
- ✓ Being retweeted by a popular account which is mentioned in the tweet,
- ✓ Being the first reporter of something,
- ✓ Fit and proper hashtag and mention selection,
- ✓ Making astonishing claims based upon a famous person or conveying appealing information on behalf of this person,

SMOTE algorithm has been applied many times for classification problems in many different domains. In addition, for the regression problem, there are several attempts to augment the dataset by SMOTE. The proposed component (ClusAug) brings a new insight by creating small virtual datasets and labeling them according to their neighborhood. This method applies SMOTE without overturning the principal logic of SMOTE. As expected, augmenting rare instances will provide a better prediction performance in favor of them. Nevertheless, while increasing the excess data provides a performance improvement for a part of a dataset, it reduces the prediction performance for all data. The natural limit for augmentation starts with constituting new rare instances partly in a dataset. In short, a proper augmentation should be evaluated in both rare instances and the overall dataset.

Even though the proposed workflow is tested for cold-start prediction problems in Twitter, it is applicable for all types of popularity prediction approaches. Independently from derived/selected features for modeling, machine learning model selection, popularity prediction approach, and domain, the ClusOut process can detect the performance-related outliers among similar ones, only considering the ideal t_n, the time for the proper reflection of reaction numbers which content has received may be named as saturation point of content. At that time, the content should have received much of the final reaction number. In an empirical study of the spread of news on Twitter [67], the number of fan votes received by each story is observed as increasing over time. In the first hours following posting, the number of received votes starts dramatically to increase, and a little over 20 hours, they start to follow a horizontal course. Conformably, [40] implies the short lifespan of articles in social media around one day for a separate news agency. Basing on these studies, in my work, the saturation time of content is selected as t_{24}, 24 hours later, the content is posted.

The proposed data augmentation method (ClusAug) is not a specific implementation in the social media domain. It is applicable for all studies from different domains to demand to synthesize very similar new instances and apply SMOTE by preserving the problem as a numeric prediction.

4.1 Limitations and Future Work

There are several limitations of this study. The first limitation is about data scraping process. Both data sets used for analysis are limited because of Twitter API credentials. They constitute approximately 1% sample of all tweets which contain the queried words and are posted in only a two-hour time window. A richer data set with all related tweets without any time restriction could better succeed in the proposed workflow.

Secondly, the computational cost of the proposed workflow is high. Outlier detection employs the standard KNN model, which is an essential component in the workflow. The standard KNN model could be substituted with an advanced neighbor detection method that provides a faster scan among observations, like FAISS[4] (Facebook AI Similarity Search). Moreover, the proposed method could be tested on the same data set to which new state of art features were added to the existing features. The newly added features could be related to the early performance of a tweet. The number of detected outliers is inversely proportional to the representativeness of features, providing a testing platform for feature selection.

Thirdly, tweet content was analyzed with a bag of words approach with neglecting typo errors. Adding more representative features following an advanced text analysis like counting typo errors that are semantically clear for users will provide a better base prediction performance. Likewise, counting external factors in modeling naturally increase base models' prediction performance. However, the proposed workflow aims for an improvement independently from feature and model selection. If more representative features were added to learning, the model with better predictive performance would be further improved.

Lastly, although the lifespan of articles, news on social media is studied, it is challenging to detect the saturation point of tweets which is an ongoing and hot topic. It can vary according to the domain and topic. Twenty-four hours after a tweet is posted, as selected in my work, is a relatively long period and not a precise selection. It is possible to limit this time window or be diversified for different tweets via additional features. This process would help for better preparation of datasets for prediction models.

As future work, following the selection of a state of art methodology in popularity prediction literature, the proposed pre-processing workflow will be employed on the same data set. The same methodology with the same features will be implemented following the pre-processing of the dataset, and prediction results will be compared with the baseline methodology.

[4] https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/, the explanation of FAISS.

REFERENCES

[1] Z. Tufekci, "Big Questions for Social Media Big Data: Representativeness, Validity and Other Methodological Pitfalls," *Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014*, pp. 505–514, Mar. 2014, Accessed: Mar. 04, 2021. [Online]. Available: http://arxiv.org/abs/1403.7400.

[2] M. Lee, H. Kim, and O. Kim, "Why do people retweet a tweet?: Altruistic, egoistic, and reciprocity motivations for retweeting," *Psychologia*, vol. 58, no. 4, 2015, doi: 10.2117/psysoc.2015.189.

[3] I. Arapakis, B. Barla Cambazoglu, and M. Lalmas, "On the feasibility of predicting news popularity at cold start," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Nov. 2014, vol. 8851, pp. 290–299, doi: 10.1007/978-3-319-13734-6_21.

[4] K. Fernandes, P. Vinagre, and P. Cortez, "A proactive intelligent decision support system for predicting the popularity of online news," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9273, pp. 535–546, doi: 10.1007/978-3-319-23485-4_53.

[5] E. Hensinger, I. Flaounas, and N. Cristianini, "Modelling and predicting news popularity," *Pattern Analysis and Applications*, vol. 16, no. 4, pp. 623–635, Nov. 2013, doi: 10.1007/s10044-012-0314-6.

[6] S. van Canneyt, P. Leroux, B. Dhoedt, and T. Demeester, "Modeling and predicting the popularity of online news based on temporal and content-related features," *Multimedia Tools and Applications*, vol. 77, no. 1, pp. 1409–1436, Jan. 2018, doi: 10.1007/s11042-017-4348-z.

[7] Y. Keneshloo, S. Wang, E. H. Han, and N. Ramakrishnan, "Predicting the popularity of news articles," in *16th SIAM International Conference on Data Mining 2016, SDM 2016*, 2016, pp. 441–449, doi: 10.1137/1.9781611974348.50.

[8] T. Uddin, M. J. A. Patwary, T. Ahsan, and M. S. Alam, "Predicting the popularity of online news from content metadata," Feb. 2017, doi: 10.1109/ICISET.2016.7856498.

[9] K. Lerman and T. Hogg, "Using a model of social dynamics to predict popularity of news," in *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, 2010, pp. 621–630, doi: 10.1145/1772690.1772754.

[10] M. Jenders, G. Kasneci, and F. Naumann, “Analyzing and predicting viral tweets,” in *WWW 2013 Companion - Proceedings of the 22nd International Conference on World Wide Web*, 2013, pp. 657–664, doi: 10.1145/2487788.2488017.

[11] K. Wang, M. Bansal, and J. M. Frahm, “Retweet wars: Tweet popularity prediction via dynamic multimodal regression,” in *Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018*, May 2018, vol. 2018-January, pp. 1842–1851, doi: 10.1109/WACV.2018.00204.

[12] H. Y. Chen and C. te Li, “PSEISMIC: A personalized self-exciting point process model for predicting tweet popularity,” in *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, Jul. 2017, vol. 2018-January, pp. 2710–2713, doi: 10.1109/BigData.2017.8258234.

[13] X. Gao, Z. Zheng, Q. Chu, S. Tang, G. Chen, and Q. Deng, “Popularity Prediction for Single Tweet based on Heterogeneous Bass Model,” *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, Nov. 2019, doi: 10.1109/tkde.2019.2952856.

[14] T. Zaman, E. B. Fox, and E. T. Bradlow, “A Bayesian approach for predicting the popularity of tweets,” *Annals of Applied Statistics*, vol. 8, no. 3, pp. 1583–1611, Apr. 2013, doi: 10.1214/14-AOAS741.

[15] J. Cheng, L. A. Adamic, P. A. Dow, J. Kleinberg, and J. Leskovec, “Can cascades be predicted?,” in *WWW 2014 - Proceedings of the 23rd International Conference on World Wide Web*, Apr. 2014, pp. 925–935, doi: 10.1145/2566486.2567997.

[16] T. Paek, T. Paek, M. Gamon, S. Counts, D. M. Chickering, and A. Dhesi, “Predicting the Importance of Newsfeed Posts and Social Network Friends,” Accessed: Mar. 04, 2021. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.187.7204.

[17] A. Zohourian, H. Sajedi, and A. Yavary, “Popularity prediction of images and videos on Instagram,” in *2018 4th International Conference on Web Research, ICWR 2018*, Jun. 2018, pp. 111–117, doi: 10.1109/ICWR.2018.8387246.

[18] H. Kwak, C. Lee, H. Park, and S. Moon, “What is Twitter, a social network or a news media?,” in *Proceedings of the 19th International Conference on World Wide Web, WWW ’10*, 2010, pp. 591–600, doi: 10.1145/1772690.1772751.

[19] Y. Yamaguchi, T. Takahashi, T. Amagasa, and H. Kitagawa, “TURank: Twitter user ranking based on user-tweet graph analysis,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Dec. 2010, vol. 6488 LNCS, pp. 240–253, doi: 10.1007/978-3-642-17616-6_22.

[20] E. Bakshy, W. A. Mason, J. M. Hofman, and D. J. Watts, “Everyone’s an influencer: Quantifying influence on twitter,” in *Proceedings of the 4th ACM International*

Conference on Web Search and Data Mining, WSDM 2011, 2011, pp. 65–74, doi: 10.1145/1935826.1935845.

[21] R. Shreyas, D. M. Akshata, B. S. Mahanand, B. Shagun, and C. M. Abhishek, "Predicting popularity of online articles using Random Forest regression," 2016, doi: 10.1109/CCIP.2016.7802890.

[22] Q. Cao, H. Shen, K. Cen, W. Ouyang, and X. Cheng, "DeepHawkes: Bridging the gap between prediction and understanding of information cascades," in *International Conference on Information and Knowledge Management, Proceedings*, 2017, vol. Part F131841, doi: 10.1145/3132847.3132973.

[23] C. Li, J. Ma, X. Guo, and Q. Mei, "DeepCas: An end-to-end predictor of information cascades," 2017, doi: 10.1145/3038912.3052643.

[24] T. Trzciński, P. Andruszkiewicz, T. Bocheński, and P. Rokita, "Recurrent neural networks for online video popularity prediction," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017, vol. 10352 LNAI, doi: 10.1007/978-3-319-60438-1_15.

[25] R. Crane and D. Sornette, "Robust dynamic classes revealed by measuring the response function of a social system," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 41, 2008, doi: 10.1073/pnas.0803685105.

[26] L. Yu, P. Cui, F. Wang, C. Song, and S. Yang, "Uncovering and predicting the dynamic process of information cascades with survival model," *Knowledge and Information Systems*, vol. 50, no. 2, 2017, doi: 10.1007/s10115-016-0955-7.

[27] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, "SEISMIC: A self-exciting point process model for predicting tweet popularity," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, vol. 2015-August, doi: 10.1145/2783258.2783401.

[28] R. Bandari, S. Asur, and B. A. Huberman, "The Pulse of News in Social Media: Forecasting Popularity," Feb. 2012, Accessed: Mar. 08, 2021. [Online]. Available: http://arxiv.org/abs/1202.0332.

[29] S. do Kim, S. H. Kim, and H. G. Cho, "Predicting the virtual temperature of Web-blog articles as a measurement tool for online popularity," in *Proceedings - 11th IEEE International Conference on Computer and Information Technology, CIT 2011*, 2011, pp. 449–454, doi: 10.1109/CIT.2011.104.

[30] T. Vandal, S. Ganguly, E. Kodra, R. Nemani, J. Dy, and A. R. Ganguly, "Quantifying uncertainty in discrete-continuous and skewed data with Bayesian deep learning," 2018, doi: 10.1145/3219819.3219996.

[31] A. Loquercio, M. Segu, and D. Scaramuzza, "A General Framework for Uncertainty Estimation in Deep Learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, 2020, doi: 10.1109/LRA.2020.2974682.

[32] S. Gao, J. Ma, and Z. Chen, “Modeling and predicting retweeting dynamics on microblogging platforms,” 2015, doi: 10.1145/2684822.2685303.

[33] K. Kuang, M. Jiang, P. Cui, and S. Yang, “Steering social media promotions with effective strategies,” 2017, doi: 10.1109/ICDM.2016.47.

[34] C. Xiao, C. Liu, Y. Ma, Z. Li, and X. Luo, “Time sensitivity-based popularity prediction for online promotion on Twitter,” *Information Sciences*, vol. 525, pp. 82–92, Jul. 2020, doi: 10.1016/j.ins.2020.03.056.

[35] N. v. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, 2002, doi: 10.1613/jair.953.

[36] H. al Majzoub, I. Elgedawy, Ö. Akaydın, and M. Köse Ulukök, “HCAB-SMOTE: A Hybrid Clustered Affinitive Borderline SMOTE Approach for Imbalanced Data Binary Classification,” *Arabian Journal for Science and Engineering*, vol. 45, no. 4, 2020, doi: 10.1007/s13369-019-04336-1.

[37] B. S. Raghuwanshi and S. Shukla, “SMOTE based class-specific extreme learning machine for imbalanced learning,” *Knowledge-Based Systems*, vol. 187, 2020, doi: 10.1016/j.knosys.2019.06.022.

[38] A. S. Hussein, T. Li, C. W. Yohannese, and K. Bashir, “A-SMOTE: A new pre-processing approach for highly imbalanced datasets by improving SMOTE,” *International Journal of Computational Intelligence Systems*, vol. 12, no. 2, 2019, doi: 10.2991/ijcis.d.191114.002.

[39] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, “SMOTE for regression,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, vol. 8154 LNAI, doi: 10.1007/978-3-642-40669-0_33.

[40] D. Bhattacharya and S. Ram, “RT @news: An analysis of News agency ego networks in a microblogging environment,” *ACM Transactions on Management Information Systems*, vol. 6, no. 3, p. 11, Sep. 2015, doi: 10.1145/2811270.

[41] A. Tatar, M. Dias De Amorim, S. Fdida, and P. Antoniadis, “A survey on predicting the popularity of web content,” 2014. [Online]. Available: http://www.jisajournal.com/content/5/1/8.

[42] E. Bakshy, W. A. Mason, J. M. Hofman, and D. J. Watts, “Everyone’s an influencer: Quantifying influence on twitter,” in *Proceedings of the 4th ACM International Conference on Web Search and Data Mining, WSDM 2011*, 2011, pp. 65–74, doi: 10.1145/1935826.1935845.

[43] L. Weng, F. Menczer, and Y.-Y. Ahn, “Predicting Successful Memes using Network and Community Structure,” *Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014*, pp. 535–544, Mar. 2014, Accessed: Jul. 13, 2021. [Online]. Available: https://arxiv.org/abs/1403.6199v2.

[44] G. Szabo and B. A. Huberman, “Predicting the Popularity of Online Content,” *SSRN Electronic Journal*, Dec. 2011, doi: 10.2139/SSRN.1295610.

[45] G. Gürsun, M. Crovella, and I. Matta, “Describing and forecasting video access patterns,” *Proceedings - IEEE INFOCOM*, pp. 16–20, 2011, doi: 10.1109/INFCOM.2011.5934965.

[46] M. Ahmed, S. Spagna, F. Huici, and S. Niccolini, “A peek into the future: Predicting the evolution of popularity in user generated content,” *WSDM 2013 - Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pp. 607–616, 2013, doi: 10.1145/2433396.2433473.

[47] S. D. Roy, T. Mei, W. Zeng, and S. Li, “Towards cross-domain learning for social video popularity prediction,” *IEEE Transactions on Multimedia*, vol. 15, no. 6, pp. 1255–1267, 2013, doi: 10.1109/TMM.2013.2265079.

[48] A. Oghina, M. Breuss, M. Tsagkias, and M. de Rijke, “Predicting IMDB Movie Ratings Using Social Media,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7224 LNCS, pp. 503–507, 2012, doi: 10.1007/978-3-642-28997-2_51.

[49] H. Bunyamin and T. Tunys, “A Comparison of Retweet Prediction Approaches: The Superiority of Random Forest Learning Method,” *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 14, no. 3, pp. 1052–1058, Sep. 2016, doi: 10.12928/TELKOMNIKA.V14I3.3150.

[50] NesiPaolo, PantaleoGianni, PaoliIrene, and ZazaImad, “Assessing the reTweet proneness of tweets,” *Multimedia Tools and Applications*, vol. 77, no. 20, pp. 26371–26396, Oct. 2018, doi: 10.1007/S11042-018-5865-0.

[51] L. Li, R. Situ, J. Gao, Z. Yang, and W. Liu, “A Hybrid Model Combining Convolutional Neural Network with XGBoost for Predicting Social Media Popularity,” *Proceedings of the 25th ACM international conference on Multimedia*, 2017, doi: 10.1145/3123266.

[52] H.-W. Shen, D. Wang, C. Song, and A.-L. Barabási, “Modeling and Predicting Popularity Dynamics via Reinforced Poisson Processes,” *Proceedings of the National Conference on Artificial Intelligence*, vol. 1, pp. 291–297, Jan. 2014, Accessed: Jul. 17, 2021. [Online]. Available: https://arxiv.org/abs/1401.0778v1.

[53] P. Bao, H.-W. Shen, X. Jin, and X.-Q. Cheng, “Modeling and Predicting Popularity Dynamics of Microblogs using Self-Excited Hawkes Processes,” *WWW 2015 Companion - Proceedings of the 24th International Conference on World Wide Web*, pp. 9–10, Mar. 2015, Accessed: Jul. 17, 2021. [Online]. Available: https://arxiv.org/abs/1503.02754v1.

[54] A. G. Hawkes, “Spectra of some self-exciting and mutually exciting point processes,” *Biometrika*, vol. 58, no. 1, pp. 83–90, Apr. 1971, doi: 10.1093/BIOMET/58.1.83.

[55] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, “SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity,” *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 2015-August, pp. 1513–1522, Jun. 2015, doi: 10.1145/2783258.2783401.

[56] D. Z. Abidin, S. Nurmaini, R. F. Malik, Jasmir, E. Rasywir, and Y. Pratama, “A Model of Pre-processing for Social Media Data Extraction,” *Proceedings - 1st International Conference on Informatics, Multimedia, Cyber and Information System, ICIMCIS 2019*, pp. 67–72, Oct. 2019, doi: 10.1109/ICIMCIS48181.2019.8985192.

[57] S. Wankhede, R. Patil, S. Sonawane, and P. A. Save, “Data Pre-processing for Efficient Sentimental Analysis,” *Proceedings of the International Conference on Inventive Communication and Computational Technologies, ICICCT 2018*, pp. 723–726, Sep. 2018, doi: 10.1109/ICICCT.2018.8473277.

[58] M. Morchid, R. Dufour, P. M. Bousquet, G. Linarès, and J. M. Torres-Moreno, “Feature selection using Principal Component Analysis for massive retweet detection,” *Pattern Recognition Letters*, vol. 49, pp. 33–39, Nov. 2014, doi: 10.1016/j.patrec.2014.05.020.

[59] E. Alwagait, B. Shahzad, and S. Alim, “Impact of social media usage on students academic performance in Saudi Arabia,” *Computers in Human Behavior*, vol. 51, pp. 1092–1097, Oct. 2015, doi: 10.1016/J.CHB.2014.09.028.

[60] R. Westerholt, E. Steiger, B. Resch, and A. Zipf, “Abundant Topological Outliers in Social Media Data and Their Effect on Spatial Analysis,” *PLOS ONE*, vol. 11, no. 9, p. e0162360, Sep. 2016, doi: 10.1371/JOURNAL.PONE.0162360.

[61] R. Hassanzadeh and R. Nayak, “A semi-supervised graph-based algorithm for detecting outliers in online-social-networks,” *Proceedings of the ACM Symposium on Applied Computing*, pp. 577–582, 2013, doi: 10.1145/2480362.2480474.

[62] X. Dai and M. Bikdash, “Distance-based outliers method for detecting disease outbreaks using social media,” *Conference Proceedings - IEEE SOUTHEASTCON*, vol. 2016-July, Jul. 2016, doi: 10.1109/SECON.2016.7506752.

[63] S. T. Aroyehun and A. Gelbukh, “Aggression Detection in Social Media: Using Deep Neural Networks, Data Augmentation, and Pseudo Labeling.” pp. 90–97, 2018, Accessed: Jul. 13, 2021. [Online]. Available: https://aclanthology.org/W18-4411.

[64] V. Hautamäki, I. Kärkkäinen, and P. Fränti, “Outlier detection using k-nearest neighbour graph,” in *Proceedings - International Conference on Pattern Recognition*, 2004, vol. 3, pp. 430–433, doi: 10.1109/ICPR.2004.1334558.

[65] F. Angiulli and C. Pizzuti, “Fast outlier detection in high dimensional spaces,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2002, vol. 2431 LNAI, pp. 15–27, doi: 10.1007/3-540-45681-3_2.

[66] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “LOF,” in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data - SIGMOD ’00*, 2000, pp. 93–104, doi: 10.1145/342009.335388.

[67] K. Lerman and R. Ghosh, “Information Contagion: an Empirical Study of the Spread of News on Digg and Twitter Social Networks,” *ICWSM 2010 - Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, pp. 90–97, Mar. 2010, Accessed: Jul. 18, 2021. [Online]. Available: https://arxiv.org/abs/1003.2664v1.

[68] D. Cirqueira, M. Fontes Pinheiro, A. Jacob, F. Lobato, and A. Santana, “A Literature Review in Preprocessing for Sentiment Analysis for Brazilian Portuguese Social Media,” *Proceedings - 2018 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2018*, pp. 746–749, Jan. 2019, doi: 10.1109/WI.2018.00008.

[69] Jenks G F, 1967, “The Data Model Concept in Statistical Mapping" International Yearbook of Cartography, Vol.7, pp.186-190

[70] J. Kincaid, R. Fishburne, R. Rogers, and B. Chissom, “Derivation Of New Readability Formulas (Automated Readability Index, Fog Count And Flesch Reading Ease Formula) For Navy Enlisted Personnel,” *Institute for Simulation and Training*, Jan. 1975, Accessed: Aug. 08, 2021. [Online]. Available: https://stars.library.ucf.edu/istlibrary/56.

APPENDIX

SBR DATASET FIGURES

Table 15: Descriptive Statistics of SBR Dataset

Attribute Name	Min. Value	Max. Value	Median	Mean	SD	Skewness
id	-	-	-	-	-	-
timedelta (unit: day)	0.0	4753.0	2806.0	2453.0	1296.8	-0.4
num_of_hashtag	0.0	14.0	0.0	0.2	0.7	7.3
num_of_mentions	0.0	28.0	1.0	0.9	1.2	4.6
num_of_url	0.0	2.0	0.0	0.3	0.5	1.2
num_tokens_content	0.0	36.0	14.0	15.1	6.7	0.3
polarity	-1.0	1.0	0.0	0.1	0.3	-0.1
subjectivity	0.0	1.0	0.4	0.4	0.3	0.1
readability	-3.5	23.9	12.2	12.6	5.4	0.0
followers_count	0.0	10411095.0	447.0	62974.0	501347.5	13.6
followees_count	0.0	114730.0	575.0	1378.0	3700.2	16.5
num_of_alltweets	1.0	1016594.0	8676.0	39452.0	103672.0	5.3
retweet_count_t1	0.0	913.0	0.0	2.7	24.2	22.8
retweet_count_t3	0.0	961.0	0.0	2.9	25.4	22.8
retweet_count_t5	0.0	961.0	0.0	2.9	25.5	22.7
like_count_t1	0.0	2105.0	1.0	13.6	87.3	15.7
like_count_t3	0.0	2213.0	1.0	14.2	93.4	16.2
like_count_t5	0.0	2231.0	1.0	14.3	93.9	16.2

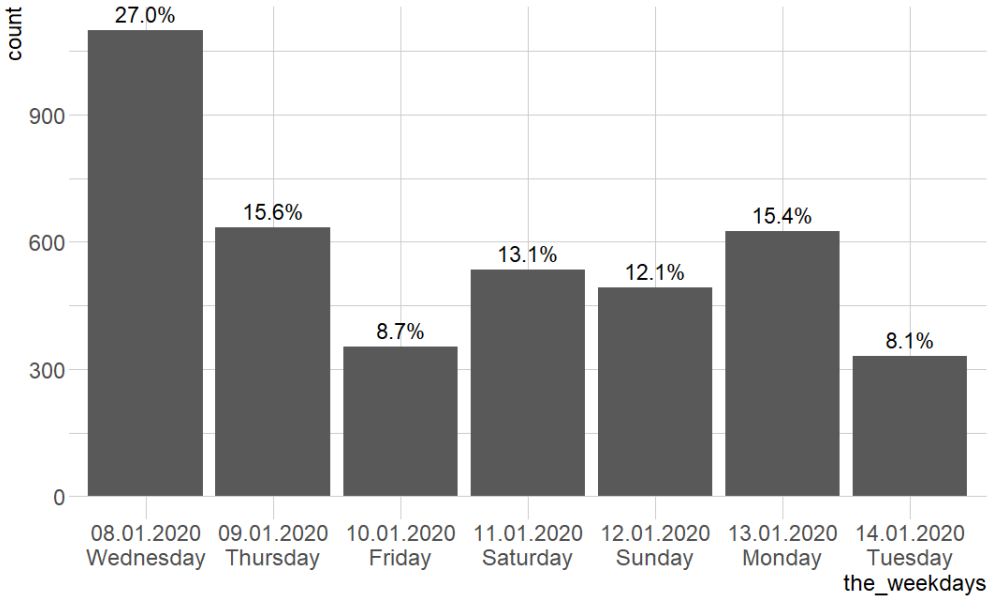


Figure 25: Timeline of tweets for SBR dataset. The histogram shows the number and percentages of collected tweets for each day in the week

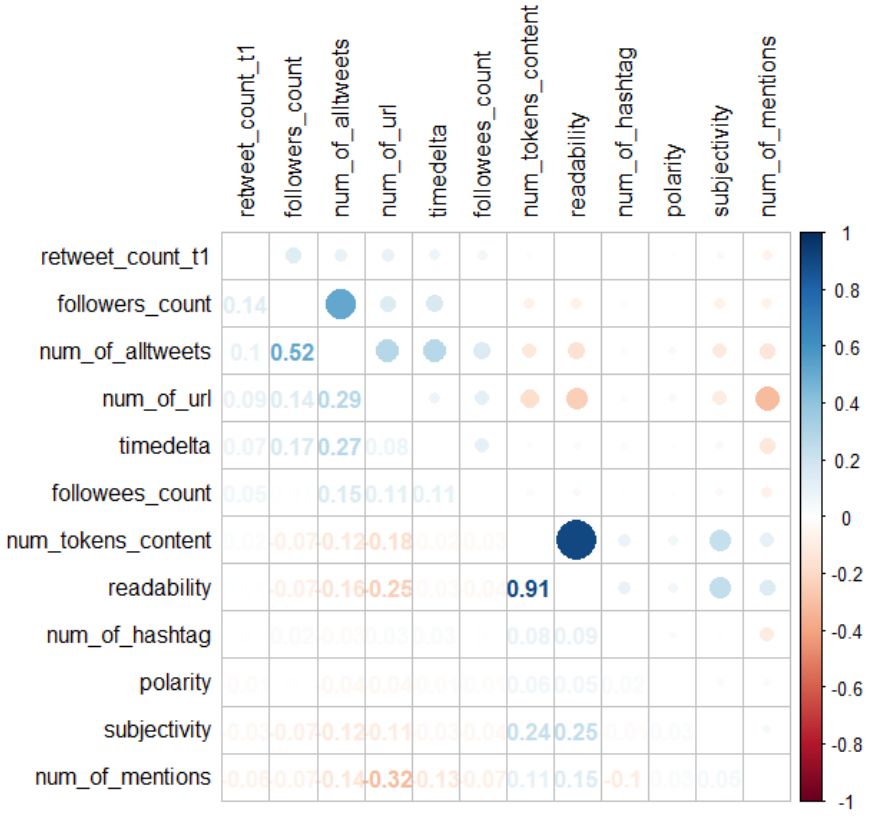


Figure 26: Correlation matrix between the numeric variables and the target feature for SBR dataset

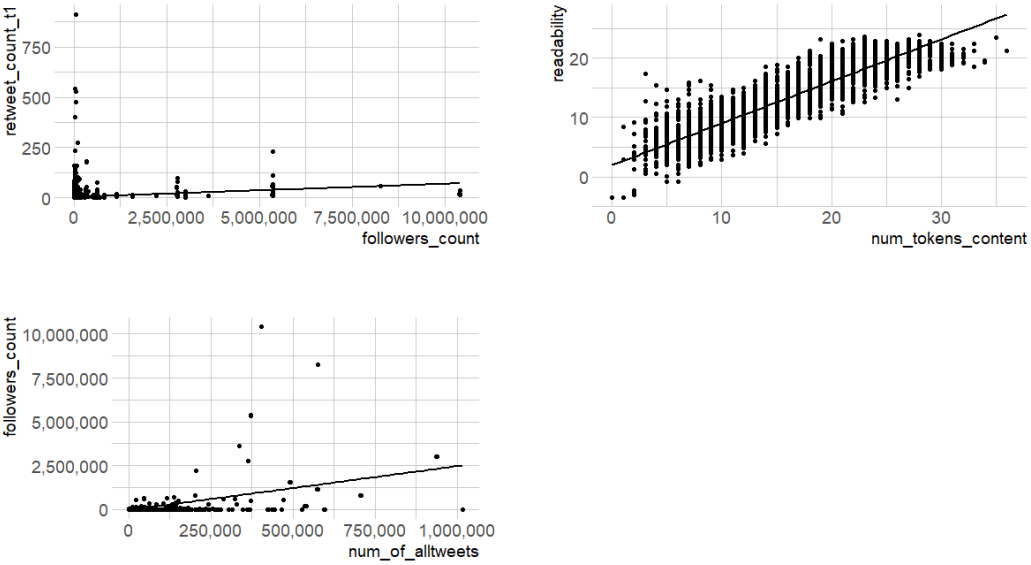


Figure 27: Bilateral plots of correlated features in SBR dataset

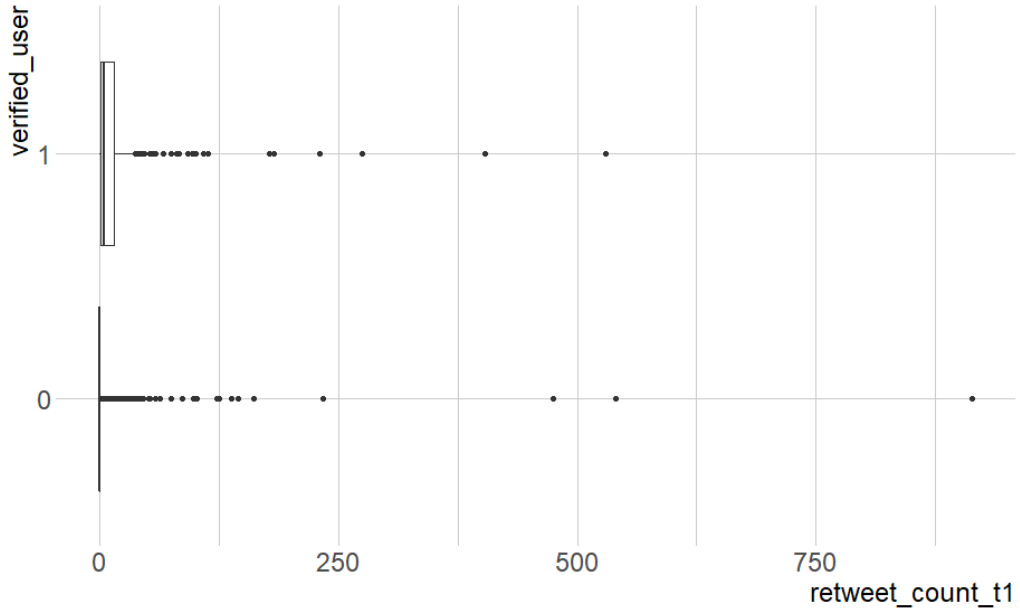


Figure 28: Verified user-retweet number plot in SBR dataset

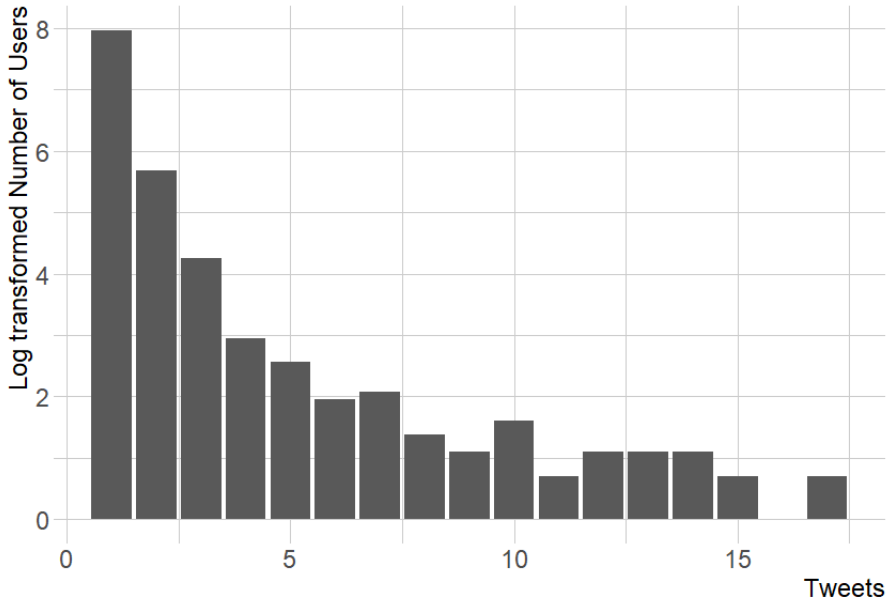
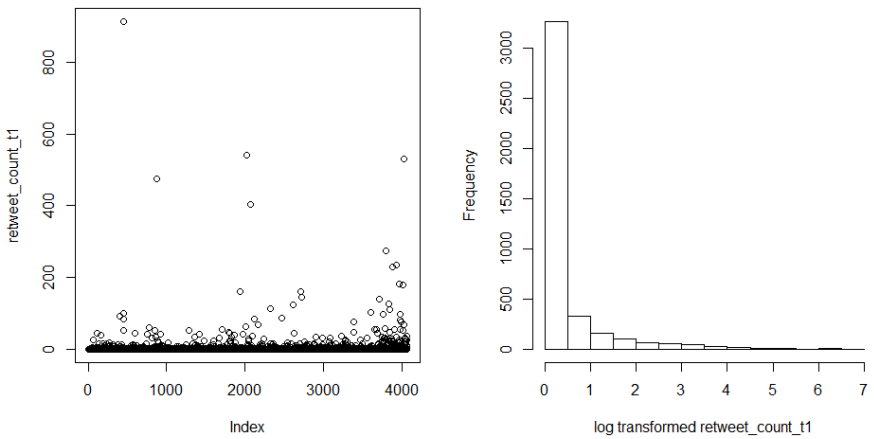


Figure 29: User-Tweet histogram for SBR dataset

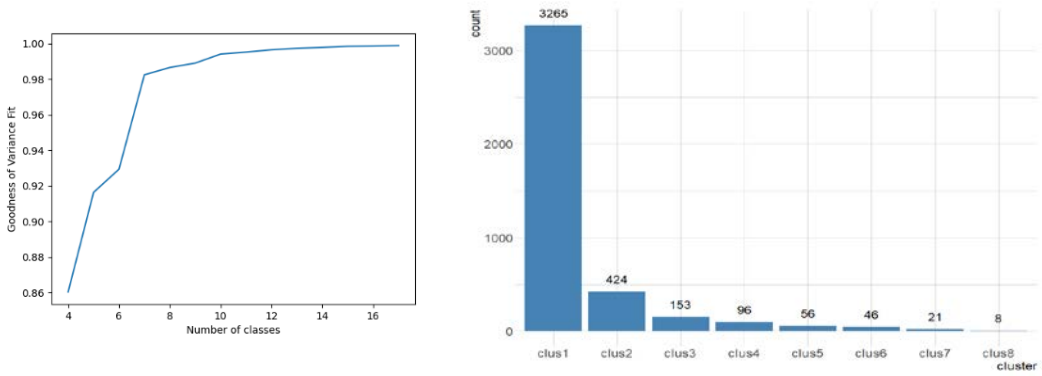
Table 16: Descriptive statistics of retweet numbers in SBR dataset

Min.	**1st Qu.**	**Median**	**Mean**	**3rd Qu.**	**Max.**
0.00	0.00	0.00	2.77	0.00	913.00



(a) Plot (b) Histogram

Figure 30: Fat-tailed distribution of retweet numbers in SBR dataset



(a) (b)

Figure 31: (a) Plotting of goodness of variance fit according to the different number of classes in SBR dataset. (b) Histogram of clusters in the SBR dataset

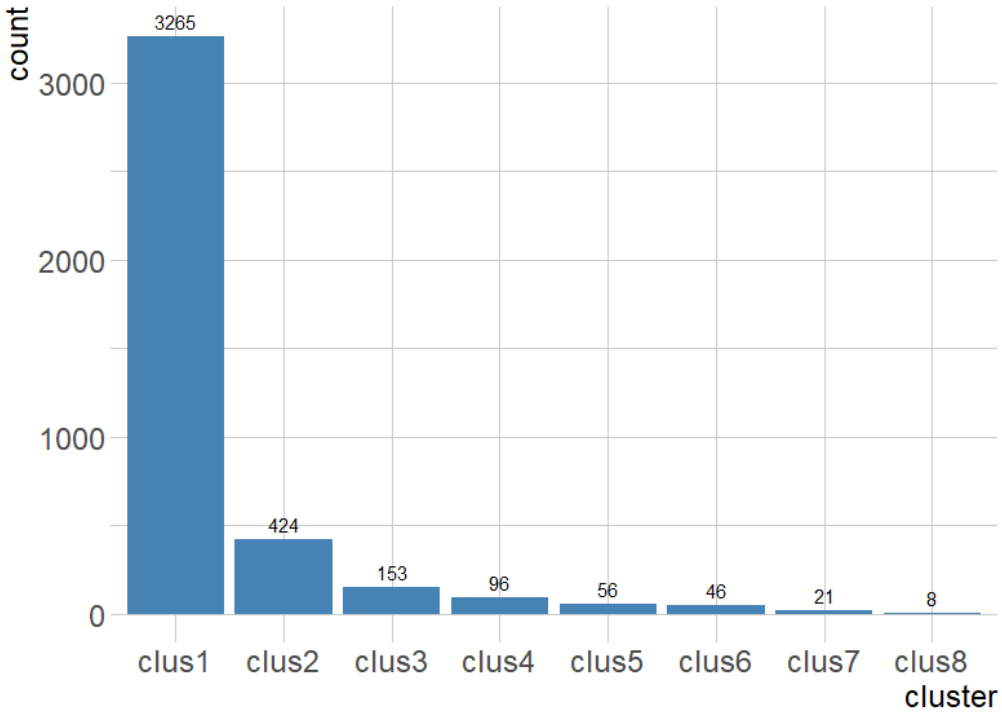
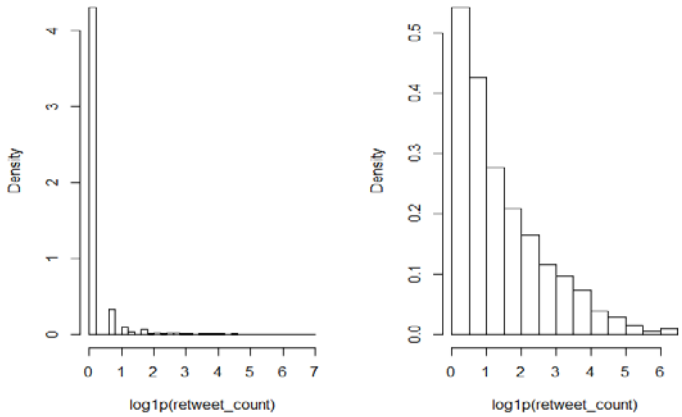


Figure 32: Histogram of clusters in test data for SBR dataset



(a) Train data (b) Test data

Figure 33: Distribution of train and test datasets for SBR dataset

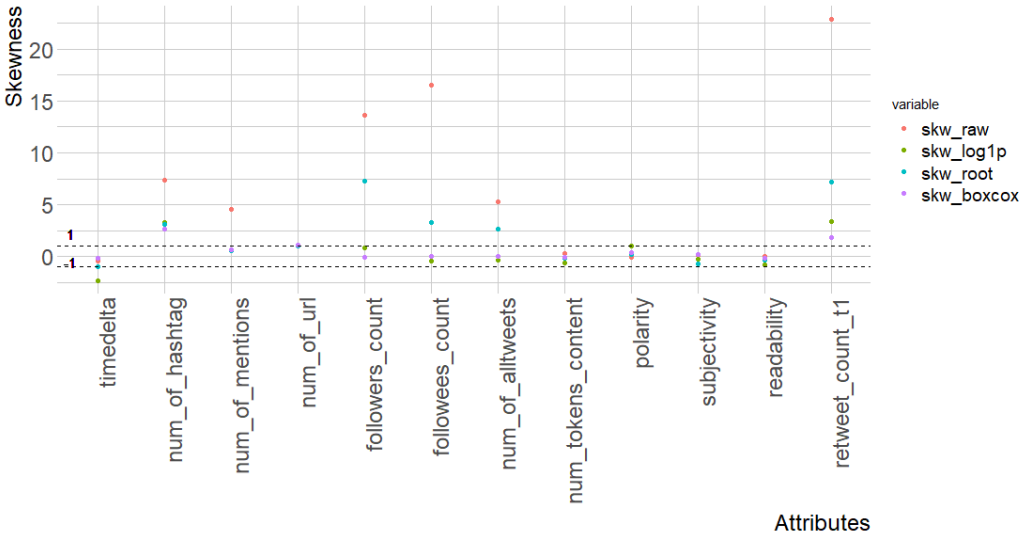


Figure 34: The plot of skewness of attributes with different transformations in SBR dataset

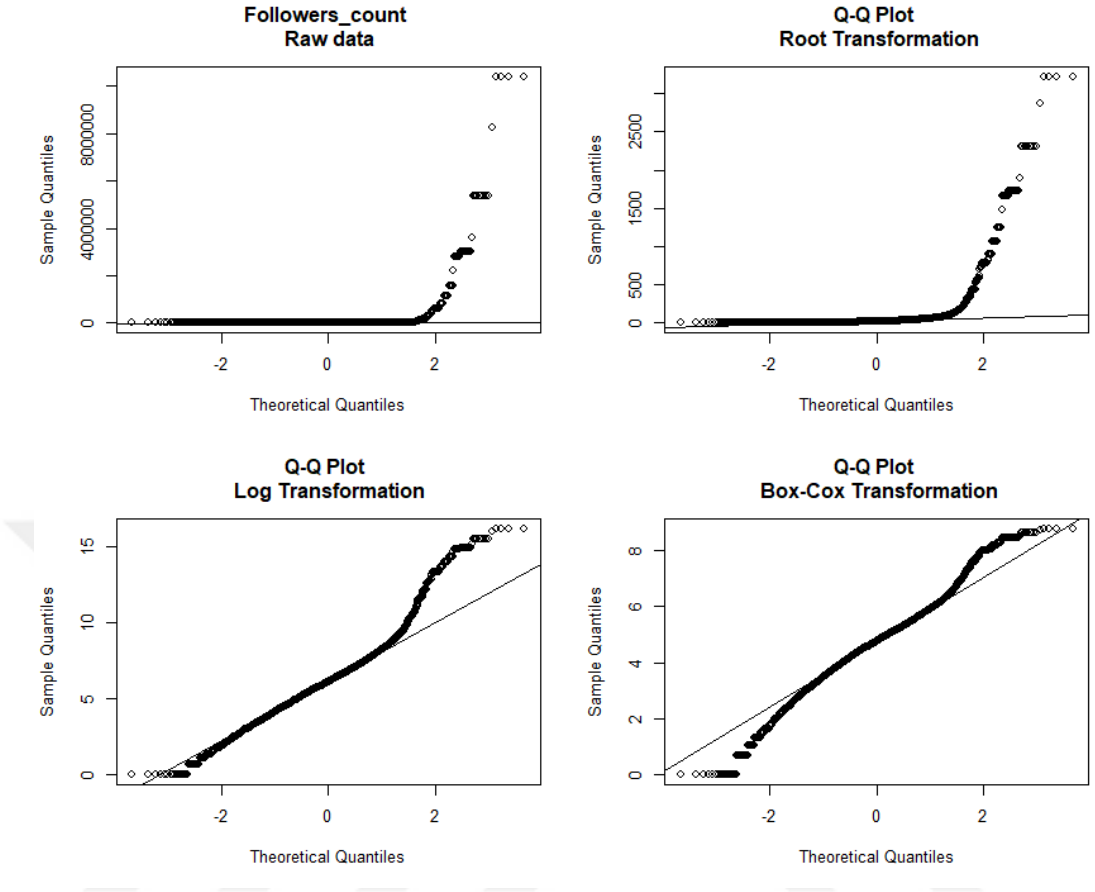


Figure 35: Distribution of followers_count according to different transformations in SBR dataset

Table 17: Skewness and kurtosis values of retweet_count_t1 in SBR dataset

Popularity_indicator	Data	Skewness	Kurtosis
retweet_count_t1	Raw data	22.85	674.66
	Square-Root	7.16	80.32
	Log transformation	3.34	12.66
	Box-cox	1.83	1.82

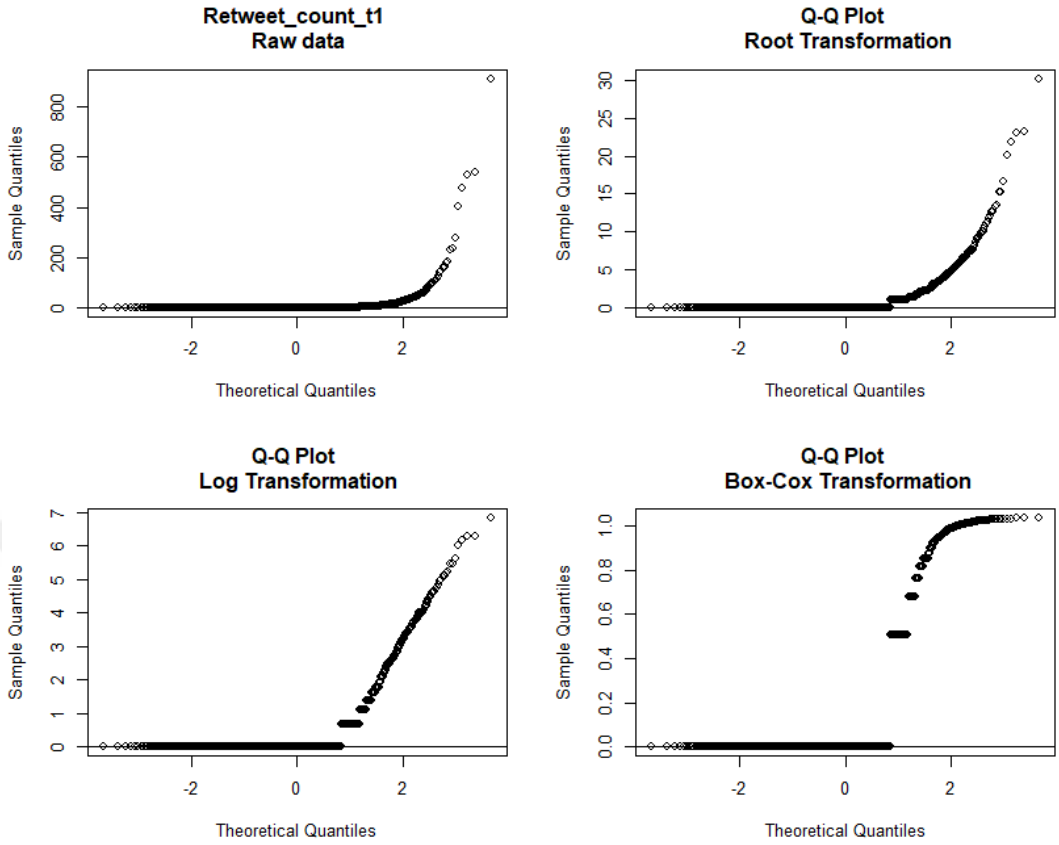


Figure 36: Distribution of retweet_count_t1 according to different transformations in SBR dataset

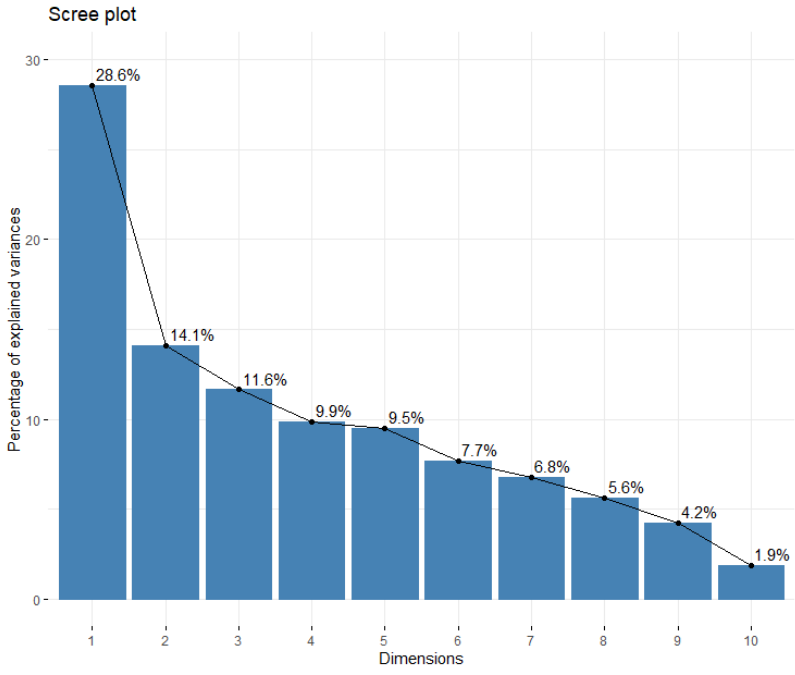


Figure 37: Scree plot of PCA components and their variances in SBR dataset

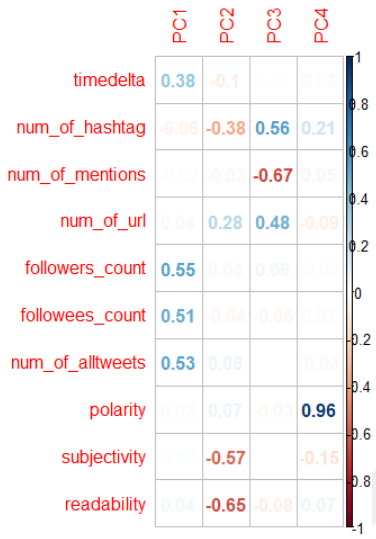


Figure 38: VARIMAX loadings of the first four eigenvectors according to features in SBR dataset

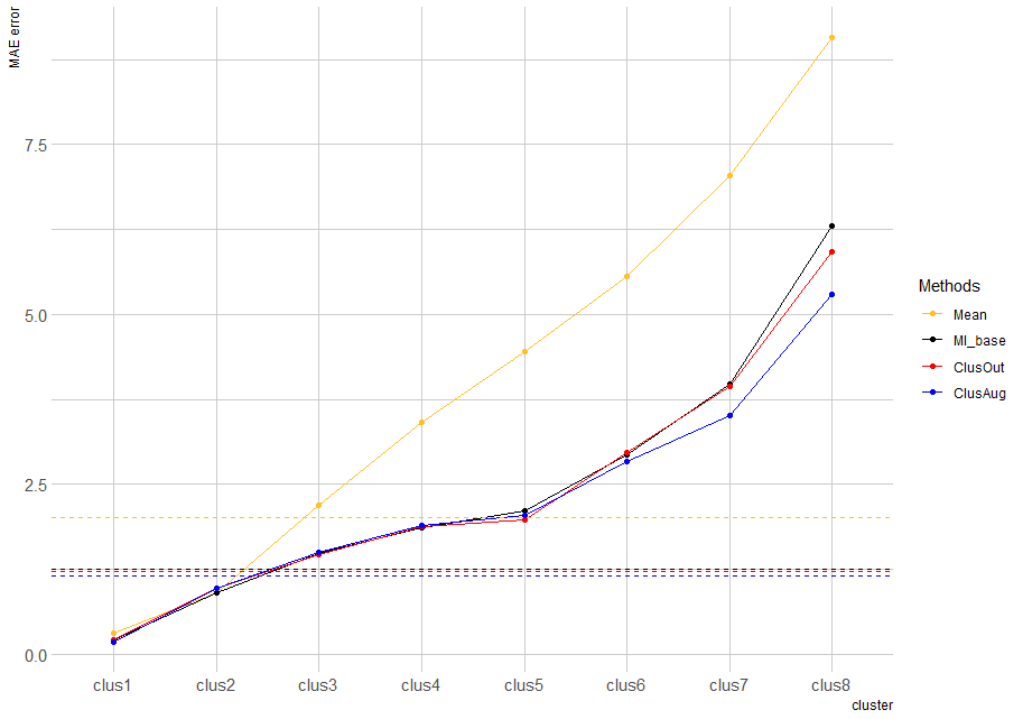


Figure 39: XGBoost model performance per cluster according to methods in SBR dataset

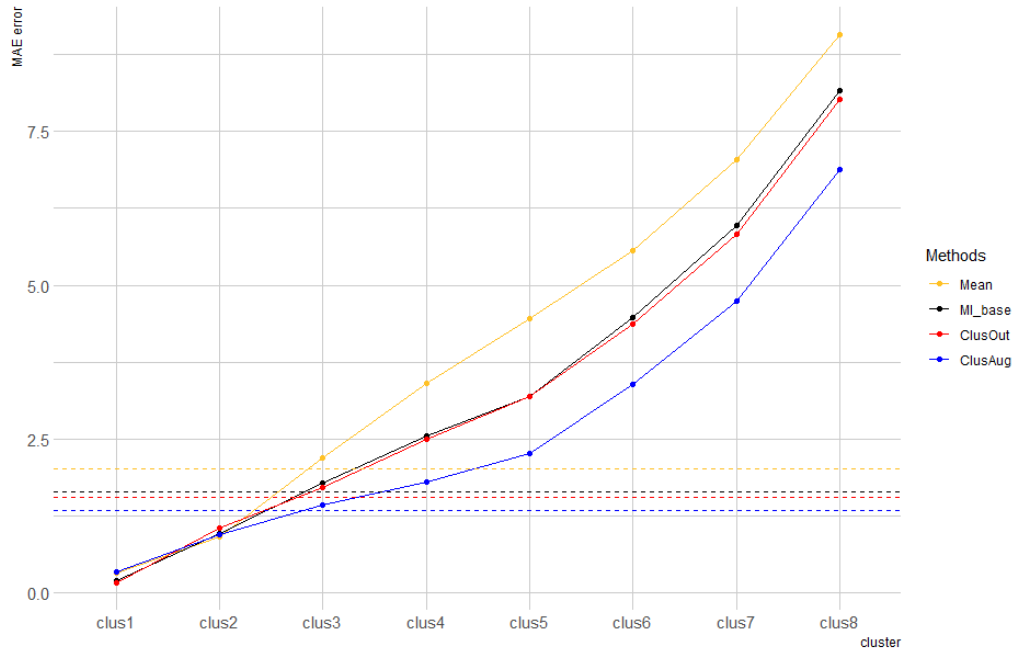
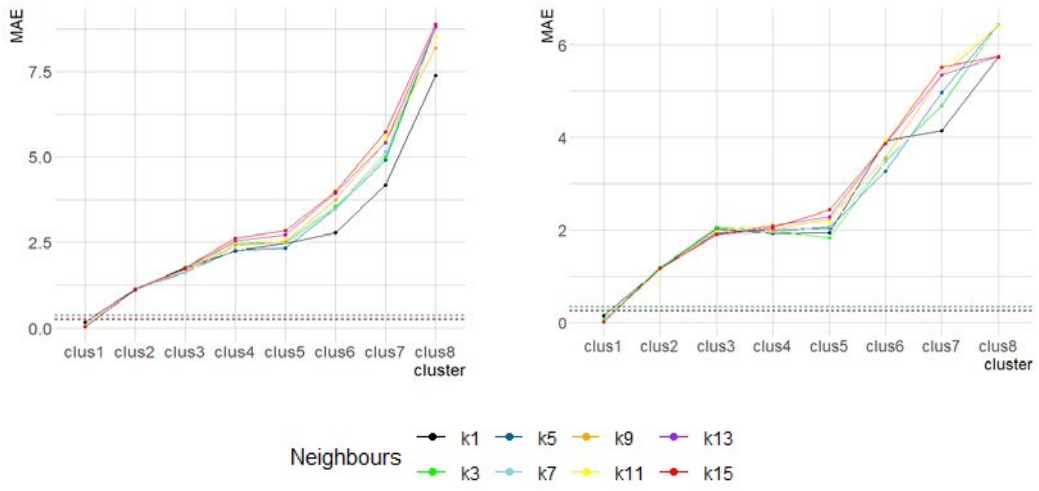


Figure 40: SVM model performance per cluster according to methods in SBR dataset

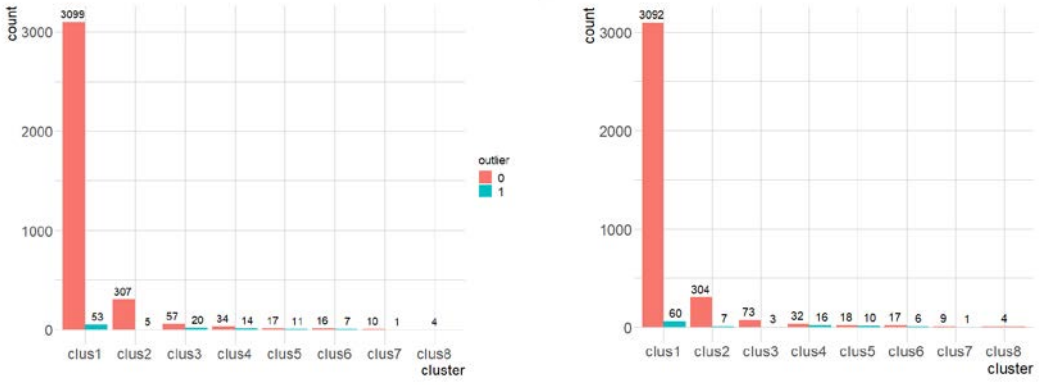
Table 18: Importance of attributes for SBR dataset

Attribute	**Fold1**	**Fold2**
followers_count	0.4329±0.0388	0.4727±0.0112
num_of_alltweets	0.1191±0.0310	0.0643±0.0046
followees_count	0.0887±0.0081	0.0937±0.0022
timedelta	0.0793±0.0068	0.1019±0.0068
verified_user	0.0708±0.0053	0.0844±0.0099
readability	0.0538±0.0082	0.0737±0.0064
polarity	0.0402±0.0023	0.0422±0.0036
subjectivity	0.0355±0.0034	0.0207±0.0006
day_of_tweetSunday	0.0203±0.0050	0.0046±0.0005
num_of_mentions	0.0145±0.0026	0.0321±0.0042
num_of_url	0.0129±0.0031	0.0123±0.0012
day_of_tweetThursday	0.0094±0.0018	0.0057±0.0012
num_of_hashtag	0.0060±0.0013	0.0025±0.0045
day_of_tweetMonday	0.0051±0.0013	0.0072±0.0099
day_of_tweetSaturday	0.0044±0.0013	0.0083±0.0019
day_of_tweetWednesday	0.0035±0.0007	0.0099±0.0011
day_of_tweetTuesday	0.0023±0.0008	0.0053±0.0014
day_of_tweetFriday	0.0004±0.0014	0.0006±0.0003



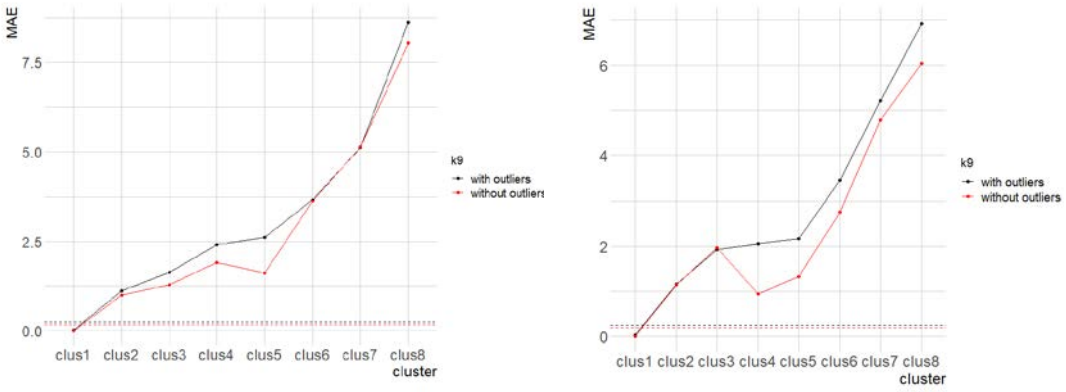
(a) Fold1 (b) Fold2

Figure 41: MAE within clusters according to different k neighbors in SBR dataset



(a) Fold1 (b) Fold2

Figure 42: Outliers for per cluster in SBR dataset



(a) Fold1 (b) Fold2

Figure 43: MAE within clusters after outliers are discarded from the training dataset in SBR dataset