The First International Conference on
Machine Learning and Knowledge Discovery (MLKD 2024)
Amirkabir University of Technology, December 18-19, 2024

Two-Stage Classifier for Detecting Campaign Negativity with Axis Embeddings in Persian Tweets

Fatemeh Rajabi* Ali Mohades†

Abstract

In elections worldwide, candidates often resort to negative campaigning due to pressure and the fear of failure. With the rise of social media platforms like Twitter, political discussions are now more accessible than ever. Given the vast amount of data generated, automated systems for detecting negativity in campaigns are crucial to understanding candidate strategies. In this paper, we propose a hybrid model for detecting negativity in campaigns using a two-stage classifier that leverages the strengths of two machine learning models. We collected Persian tweets from 50 political users, including candidates and government officials, and annotated 5,100 tweets published in the year leading up to Iran's 2021 presidential election. Our model first creates two datasets from the training set for two classifiers by calculating the cosine similarity between tweet embeddings and axis embeddings (the average of positive and negative embeddings). These datasets are then used to train the hybrid model. Our best-performing model (RF-RF) achieved a 79% macro F1 score and 82% weighted F1 score. Applying this model to additional tweets with the help of statistical models, revealed that a candidate's tweet publication timing does not affect its negativity. Still, the presence of political person names and organization names in tweets is closely linked to negativity.

Keywords: Campaign negativity, Two-stage classifier, Persian tweets, Axis embeddings

1 Introduction

In today's information age, vast amounts of textual data are generated daily, making the ability to categorize and interpret this information more critical than ever efficiently. Natural Language Processing (NLP) and Machine Learning (ML) have emerged as powerful tools to automate text classification, offering ways to analyze and understand vast datasets. However, this task presents significant challenges. Text is inherently unstructured, raising issues around feature extraction, dimensionality, and cross-linguistic variations. Furthermore, as data volume increases, scalability and model efficiency challenges become more pronounced. Yet, these challenges also present opportunities for innovative solutions. By leveraging advanced machine learning techniques, researchers can uncover hidden patterns, sentiments, and insights from textual data [1], [2].

Detecting campaign negativity, in particular, demands a nuanced understanding of language. Campaign negativity refers to political candidates attacking their opponents rather than presenting their policies, capabilities, or accomplishments. This strategy often manifests as conceptual or ironic statements [3]. Traditionally, negativity has been measured through manual content analysis by political experts—an approach that is both time-consuming and prone to errors. To address these challenges, researchers have turned to automated tools capable of analyzing large datasets quickly and accurately, providing deeper insights into the tone and content of political discourse on social media. However, these automated methods are not without limitations, especially when dealing with indirect language, such as sarcasm. Thus, a combination of both automated and manual approaches is often necessary to effectively measure campaign negativity and assess its impact on public opinion and election outcomes.

Previous research has examined campaign negativity in various types of elections, such as presidential, senate, and municipal elections, across different countries. These studies have analyzed the timing and causes of negativity and assessed how various factors influence the tone of political campaigns [15], [16]. Typically, this involves a combination of manual content analysis and statistical models to examine the effect of different variables on negativity.

In this study, we propose a machine learning model to detect negativity in political campaigns and apply it to the 2021 Iranian presidential election. Our work addresses several key challenges. First, we collected Persian tweets from political figures and candidates using the Twitter Developer API, ultimately annotating 5,100 tweets into three categories: negativity, personal attacks, and political attacks. The labeling process is detailed in Section 3.1.

Building an effective model for detecting negativity required defining specific features and carefully select-

*Department of Mathematics & Computer Science, Amirkabir University of Technology, fateme.rajabi@aut.ac.ir
†Department of Mathematics & Computer Science, Amirkabir University of Technology, mohades@aut.ac.ir

ing appropriate machine learning methods. Classifying negativity is complex, as it often requires multiple readings by humans to fully interpret. Similarly, for machine learning models, feature extraction is crucial to achieving high accuracy. We employed various techniques, such as preprocessing, feature engineering, and resampling methods, to enhance model performance. Our final model employs a two-stage classifier that uses cosine similarity between tweet embeddings and predefined axis embeddings. This approach significantly improved model performance, particularly in handling the subtle thematic overlap between positive and negative tweets, including sarcastic language.

The challenges encountered during the modeling process included:

- Thematic similarity between positive and negative tweets
- Use of sarcasm in negative tweets
- Limitations of Conceptual embedding models in Persian language

To overcome these issues, we developed an innovative method that separates the dataset into new subsets based on the cosine similarity between tweet embeddings and the average embeddings of positive and negative classes. This allowed us to isolate tweets with positive labels that have similarities with negative tweets, often due to the use of irony or thematic overlap in negative tweets. These subsets were used in the two-stage classification model, which significantly improved performance.

The remainder of this paper is structured as follows: Section 2 provides an overview of related work, and Section 3 details the methodology and data collection process. Section 4 presents the results of our study. In Section 5, we discuss the implications of our findings, while Section 6 concludes the paper, summarizing the key insights and outlining potential directions for future research.

2 Related Works

2.1 Statistical and Analytical Methods

Several approaches involve expert political analysis to examine the content published during elections. These studies often conclude by applying statistical models to evaluate the impact of various factors on campaign negativity. Many articles have used this method to analyze negativity in different elections. Below, we summarize key contributions from 2005 to the present, highlighting different perspectives.

In 2005, Peterson et al. examined the impact of time and political party on campaign negativity by analyzing newspapers covering the 1998 U.S. Senate primary elections. They concluded that campaign negativity is influenced by time, party affiliation, and the number of participants [4]. In 2007, Krebs et al. used newspaper articles and television ads from the 2001 Los Angeles mayoral election to investigate whether candidates' attacks were more focused on issues or individuals. They also examined the extent of attacks on minority versus non-minority candidates, finding that issue-based attacks were more prevalent, and minority candidates engaged in fewer attacks than non-minorities [5].

In 2009, Schweitzer compared negativity patterns between German and U.S. campaigns by analyzing candidate websites from the German national and European parliamentary elections. His study revealed that while overall patterns of negativity were similar across both countries, the topics of attack differed significantly [6]. In 2012, Grossmann explored negative advertising in the 2002 and 2004 U.S. Congressional elections. He concluded that incumbent candidates were less likely to use negative ads compared to their challengers [7].

Hassell et al. (2016) analyzed campaign emails from the 2014 U.S. Congressional elections to determine when candidates chose to adopt a negative tone. Their findings indicated that email negativity did not necessarily make elections more competitive [8]. In a Persian study from 2018, Babaei et al. conducted interviews with 16 Iranian political experts to explore the theoretical foundations of negative election campaigns during the Iranian presidential elections from 2004 to 2016. The authors attributed negativity to factors such as an emotional public atmosphere, a culture of destruction, prioritization of groups over national interests, and weak legal frameworks [9].

Walter et al. (2019) sought to enhance the validity of negativity measurement by analyzing newspapers, voter opinions, and expert assessments from the 2015 U.K. election. They employed a Bayesian statistical model to adjust for bias in voter and expert opinions, offering a more accurate measure of campaign negativity [10]. In 2020, Maier et al. examined the relationship between negativity and media coverage by analyzing tweets and TV ads from 507 candidates in 107 national elections across 89 countries (2016–2019). Their study found that emotionally charged messages, such as those evoking fear or passion, had a greater influence on media coverage than negative campaign content itself [11]. Lastly, Nie (2021) compared the campaign behaviors of populist and non-populist candidates by analyzing articles from 195 candidates in 40 global elections (2016–2017). He discovered that populists were more likely to engage in negativity, personality attacks, and fearmongering than non-populists [12].

2.2 Machine Learning Methods

Machine learning (ML) has recently become a prominent tool for detecting negativity in political campaigns. In 2022, Petkevic et al. developed a multilayer perceptron model to identify negativity, types of attacks (political or personal), and incivility in tweets. The model was trained on 1,186 tweets published in the 90 days before the 2018 U.S. Senate election by 66 candidates. The best model achieved F1 scores of 82% for negativity, 83% for political attacks, 82% for personal attacks, and 85% for incivility. The model was then applied to 16,000 tweets to measure the influence of various factors, such as gender, political affiliation (Republican

or Democrat), and proximity to the election, on campaign negativity [14].

In 2023, Kim focused on the 2020 U.S. presidential election and developed a BERT-based classification model to detect violent tweets. After collecting and filtering tweets with political and violent keywords, he used human labeling for 2,500 tweets and trained various models, with the BERT model achieving the best performance: 71.8% precision, 65.6% recall, and a 68.4% F1 score. Kim applied the model to an additional 5,000 tweets using active learning and further analyzed the impact of gender and political affiliation on the occurrence of violent language. The study concluded that women and Republicans were more frequent targets of violent tweets than men and non-Republicans [13].

3 Methodology

3.1 Data Collection and Annotation

To begin, we collected tweets from 50 political users using the get_all_tweets endpoint via the Twitter Developer Account. Our query was based on the usernames of these 50 users, resulting in a dataset of 42,837 tweets published between January 7, 2011, and June 2, 2021. Among these, 10,292 tweets were published within one year before the election day, of which 1,776 tweets were authored by seven presidential candidates.

From this dataset, we randomly selected and annotated 5,100 tweets, ensuring the selection was proportional to the number of tweets published by each user. Initially, 3,100 tweets were manually tagged. Afterward, basic machine learning models were applied to assess their performance, and ParsBERT was identified as the most effective model. This model was then utilized for an active learning approach. Approximately 2,000 additional tweets were chosen for annotation, specifically those for which the model's classification probability was near the threshold (0.5), indicating difficulty in classification.

For the annotation process, an initial set of 500 tweets was tagged by three experts. Subsequently, the expert whose annotations most closely matched the majority consensus was selected to label the remaining tweets.

Table 1: Number of each label in Dataset (which 1 shows the tweet has campaign negativity and 0 contrariwise)

Class/Label	Presence (1)	Absence (0)
Negativity	1,447	3,653
Political Attack	507	4,593
Personal Attack	894	4,206

For the annotation process, each tweet was evaluated based on its content. If the tweet included a direct or sarcastic attack or exhibited harmful or destructive language, it was labeled as 1, indicating negativity. Otherwise, it was labeled as 0, indicating no negativity. Additionally, for tweets labeled as negative, the specific type of attack—whether aimed at an individual or an organization—was identified. However, due to the limited number of negative tweets in the dataset, we did not develop a model to classify the attack type in this study. Future work could focus on enhancing the dataset with more negative tweets to build models capable of distinguishing between different types of attacks. Table 1 provides a breakdown of the number of tweets in each class.

3.2 Preprocessing and Feature Extraction

In classification tasks in ML, a fixed preprocessing approach does not always yield the best results. Therefore, we applied and tested three different preprocessing methods, ultimately reporting results based on the best-performing approach.

1. **Preprocessing Method 1:** This method involves converting emojis to text, removing emojis, separating hashtags into individual words, eliminating repeated characters within a word, removing words that include numbers, discarding junk characters, removing punctuation, and excluding tweets with fewer than three characters.
2. **Preprocessing Method 2:** Builds on Method 1 by also removing stop words.
3. **Preprocessing Method 3:** Extends Method 2 by further removing links and mentions.

Following preprocessing, we explored four distinct categories of features, based on the nature of the data and the classification problem. We sequentially added these feature sets to traditional classification models and compared the results to determine which features most effectively contributed to detecting negativity. These categories are as follows:

1. **Text Features:** Includes variables such as the number of retweets, likes, mentions, links, hashtags, use of insulting words, names of organizations and political figures, and sentiment analysis.
2. **Metatext Features:** Includes tweet embeddings, unigrams, bigrams, frequent trigrams, and frequently occurring tokens in both class 1 (negative) and class 0 (non-negative), with and without stemming.
3. **User Features:** Considers user-specific attributes like the number of followers, followings, likes, tweets, and the most frequently used tokens in user descriptions.
4. **Time Features:** Includes factors such as the lifespan of a user's account, tweet publication times across four intervals of the day and night, and tweet timing in relation to specific intervals (e.g., 10, 20 days) leading up to the election.

In total, we defined approximately 1,400 features for use with classical models.

3.3 Building New Datasets

Now we want to present the method for building the necessary datasets for the two-stage model. For this purpose, we use two methods: axis embeddings and clustering. Before explaining the mentioned methods, it should be mentioned that according to the results of the basic models that we have

presented in the results section, for building new datasets we used the ALC embedding, which was the most accurate. This method replaces the usual Word2Vec method and is based on GloVe embeddings and a linear transformation. It is also suitable for rare words in the corpus. In the related paper on ALC embedding, the authors claim that the ALC model needs fewer examples for learning than other models. Also, the quality of ALC embeddings has been better than other models in many examples.[17]

3.3.1 Axis Embeddings Trick

Before dealing with the creation of new labels and subsets of the dataset, it is necessary to understand the role of axis embeddings in text classification. Axis embeddings in this paper are the representation of documents in a continuous vector space. These axis embeddings are calculated by averaging the embeddings of tweets in different classes. We define two axis embeddings:

- **Axis-embedding 1 (EMB1):** represents the average embedding of tweets labeled as class 1 (negative), which means the tweet contains campaign negativity.
- **Axis-embedding 0 (EMB0):** represents the average embedding of tweets labeled as class 0 (positive), which means that the tweet has no campaign negativity.

By converting the semantic content of tweets into continuous vectors, we are able to quantify the similarity between tweets and axis representations. In fact, based on this similarity, we form new tags and subcategories.

The innovation presented in this section revolves around identifying tweets in class 0 that are more similar to EMB1. These tweets are often about the topics of negative tweets that have been sarcastically discussed. In our proposed model, we classify them as negative, since Text Features are more important than other features in this problem (according to the feature importance of basic models). In other words, label 2 is introduced for these tweets. Tweets with label 2 have a difference more than the threshold, in their similarity with EMB1 and their similarity with EMB0. For each tweet with label 2, the formula (2) is true. In this condition, CS refers to the cosine similarity based on formula (1) between 2 vectors, X refers to the ALC embedding of tweets in the train set, and t refers to the threshold. [18]

$$CS(A, B) = \frac{A.B}{|A|.|B|} \tag{1}$$

$$CS(X, EMB1) - CS(X, EMB0) > t \tag{2}$$

1. **New Dataset 1:** The first new dataset (train set for first classification in the two-stage model) consists of tweets labeled zero and one, which are considered as follows.
   (a) Positive tweets (labeled positive in the original dataset) that their embeddings are closer to EMB0 than EMB1 according to the cosine similarity. The negativity label for this class is considered 0.
   (b) Positive tweets (labeled positive in the original dataset) that their embeddings are closer to EMB1 than EMB0 according to the cosine similarity. We defined it with label 2 in the previous section. The negativity label for this class is considered 1.
   (c) Negative tweets (labeled negative in the original dataset). It should be noted that all these tweets are closer to EMB1 than EMB0 according to the cosine similarity. The negativity label for this class is considered 1.
2. **New Dataset 2:** The second new dataset (train set for second classification in the two-stage classifier) consists of tweets labeled zero and one, which are considered as follows.
   (a) Positive tweets (labeled positive in the original dataset) that their embeddings are closer to EMB1 than EMB0 according to the cosine similarity. We defined it with label 2 in the previous section. The negativity label for this class is considered 0.
   (b) Negative tweets (labeled negative in the original dataset) that contain negativity. The negativity label for this class is considered 1.

3.3.2 Clustering Trick

Instead of using axis embeddings to construct new datasets for the two-stage model, this section uses different clustering methods to separate tweets. Here we use DBSCAN [19], K-means [20], Agglomerative [21], Birch [22], Gaussian Mixture (GM) [23], and Optics [24] clustering methods. Each of these clustering methods has its strengths and weaknesses, and the choice of which one to use depends on the data's specific characteristics and the clustering task's objectives. It is often a good idea to try several methods and compare their results to find the most suitable clustering approach for a particular dataset.

Here, the difference in building the dataset based on clustering methods instead of the axis embedding trick is separating tweets with label 2. In this trick, the tweets that are placed in the same cluster as the majority of negative tweets are considered label 2, and the rest of the positive tweets are considered label 0. Similarly to the previous section, the new datasets will be made.

3.4 Two-Stage Model

Creating these new datasets serves a dual purpose. First, it addresses the challenge of classifying tweets that exhibit characteristics that vary by topic. Second, due to the combination of some positive tweets with negative tweets in the first classifier as class one, they will be predicted by the second classifier to be identified this time based on their real label. This innovative approach increases the power of the model in detecting the negativity of election campaigns and contributes to a deeper understanding of the complexities of analyzing negativity in the political landscape. We have a hybrid classification model that works in two separate stages. In the first stage, the first classifier model is trained with the entire available training data (the first new dataset) to develop a comprehensive understanding of the patterns in the

text that are relevant to election campaigns. At this stage, the foundation of the classifier's ability to effectively identify and classify tweets that do not contain campaign negativity and are not conceptually similar to negative tweets is laid. In the second stage, the second classification model is trained on the second new dataset, and re-prediction is performed focusing on the tweets predicted as class one (negative) in the first model. Following this process will determine the real tag of positive tweets, which are similar to negative tweets. This two-stage approach not only contributes to a more accurate initial classification but also to increased overall accuracy in detecting campaign negativity. In Figure 1, you can see the final structure of the model. This figure generally includes the dataset preparation (left section) and the two-stage model architecture (right section).

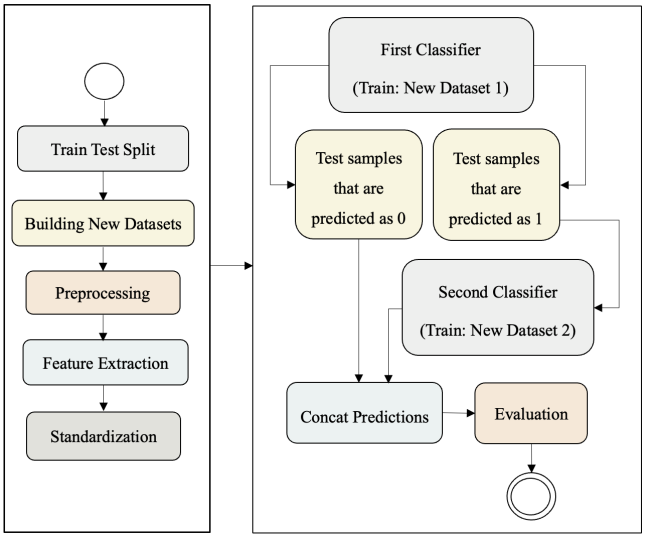

Figure 1: Architecture of Two-Stage Classifier (Proposed Model), left section shows dataset preparation and right section shows the two-stage model architecture

4 Results

4.1 Evaluation Metrics

The process of evaluating machine learning models is essential to understanding their performance, reliability, and suitability for real-world applications. This chapter discusses the various evaluation models, techniques, and metrics essential for comprehensively evaluating the performance of machine learning algorithms. Model evaluation not only helps researchers and practitioners make informed decisions but also plays an important role in advancing new machine-learning methods [25], [26]. In the results section, we examine the F1-score, precision (P), and recall (R) for two classes (positive and negative), F1-macro and F1-weighted, which are listed in formulas (3), (4), (5), (6), and (7), respectively. Because the F1-score is a combination of two criteria: precision and recall, both of which are important in this issue. In formulas (3), (4), and (5) TN refers to the number of true negative labels, TP refers to the number of true positive labels, FN refers to the number of false negative labels, and FP refers to the number of false positive labels (positive and negative are based on case study class). In formulas (4) and (5), N refers to the number of classes (here it is 2).

$$Precision(P) = \frac{TP}{TP + FP} \tag{3}$$

$$Recall(R) = \frac{TP}{TP + FN} \tag{4}$$

$$F1 = \frac{2TP}{2TP + FN + FP} = \frac{2 \times Precison \times Recall}{Precision + Recall} \tag{5}$$

$$F1_{macro} = \frac{1}{N} \sum_{i=1}^{N} F1_i \tag{6}$$

$$F1_{weighted} = \sum_{i=1}^{N} w_i \times F1_i \tag{7}$$

4.2 Basic models

In this section, we first examine the results related to the basic models and then compare the best results with the results related to the two-stage model with axis embedding and clustering tricks. In the basic models, after separating the train and test sets in a ratio of 85 to 15, stratifying on the negativity label and data preparation, we test the models on different preprocessings by sequentially adding the features that we discussed in Section 3.2. Also, due to the inequality of the number of samples in the two positive and negative classes in the dataset, we use Smote and TomekLink techniques. The Smote method is an oversampling method that generates artificial samples for the minority class. It does this by creating synthetic instances that are combinations of existing minority-class instances. It addresses the overfitting problem associated with random oversampling by generating new and diverse data points. The tomeklink method involves separating the samples into pairs (one from the majority class and one from the minority class) that are close to each other but from different classes. Removing the majority of class samples in these pairs can help improve the separation between classes. This method can be used for downsampling to lead to better separation of classes without introducing artificial data. Basic models include classic models and pre-trained Deep Learning models (suitable for the Persian language). Classical models are multilayer perceptron (MLP) [27], eXtreme Gradient Boosting (XGB) [28], Random Forest (RF) [29], Logistic Regression (LR) [30], Naive Bayes (NB) [31], Support Vector Machine (SVM) [32], Gaussian Naive Bayes (GNB) [33], K-Nearest Neighbor (KNN) [34], Ridge [35], Gradient Boosting (GB) [36], and Stochastic Gradient Descent (SGD) [37]. Pretrained Deep Learning models include ParsBERT (DistilBERT-ZWN), ParsBERT (BERT-ZWN) [39], Multilingual DistilBERT [41] and Multilingual BERT (MBERT) [40] which are trained on Persian. For classic models, we have used various embeddings such as FastText, Word2Vec [42], GloVe [43], and ALC. These embeddings are considered metatext features. According to Table 2, the results of the classical RF model with ALC embedding indices have performed better than the rest of the

classic and deep models (the best model of the deep models is MBERT) which can be considered the reason for the strong performance of the ALC model in embedding words with low repetition. Because there are many rare words in the texts in our dataset. Also, the results of model RF (ALC) have been obtained on preprocessing (3) and with all features (text features, metatext features, user features, and time features). Additionally, model MBERT has the best results with preprocessing (3), which are reported in Table 2.

Table 2: Evaluation metrics values for best classic and deep models (class 1 is negative tweets and class 0 is positive tweets)

Model	Class	P	R	F1	F1m	F1w
RF(ALC)	1	72%	50%	60%	72%	78%
	0	81%	92%	86%		
MBERT	1	54%	70%	61%	70%	74%
	0	85%	75%	80%		

Figure 2 shows how tweets are distributed using their ALC embedding, which addresses our paper challenge related to the semantic proximity of some positive tweets to negative tweets. In this figure, by using two-dimensionality reduction methods, PCA, and TSNE [38], it can be seen that some positive tweets are very close to negative tweets. It seems that some positive tweets have fallen on top of negative tweets. Of course, this is only a representation of the reduced dimensions of tweet embeddings. Therefore, in the next section, we want to implement a two-stage method using axis embedding and clustering tricks.

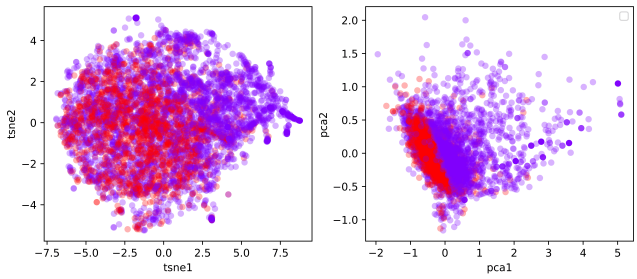

Figure 2: Labels of tweets are shown by TSNE and PCA methods for ALC embedding of tweets (purple color shows positive tweets and red color shows negative tweets)

4.3 Our approach

In this section, we use two clustering methods (Gaussian Mixture and Birch) and three threshold values (0, 0.03, and 0.05) for the axis embedding trick, to create new data sets. The reason for using the two mentioned clustering methods is that they did not separate the tweets that included negativity as much as possible, and their focus was to cluster positive tweets in line with our goal. Based on Figure 3, shows the clustering of each of the methods according to their ALC embeddings. DBSCAN and Optics methods did not perform well. KMeans and Agglomerative methods have done more separation in class 1 (compared to Figure 2), which is not our goal. However, the Gaussian Mixture and Birch methods have focused their separation on positive tweets and put negative tweets in a cluster as much as possible.

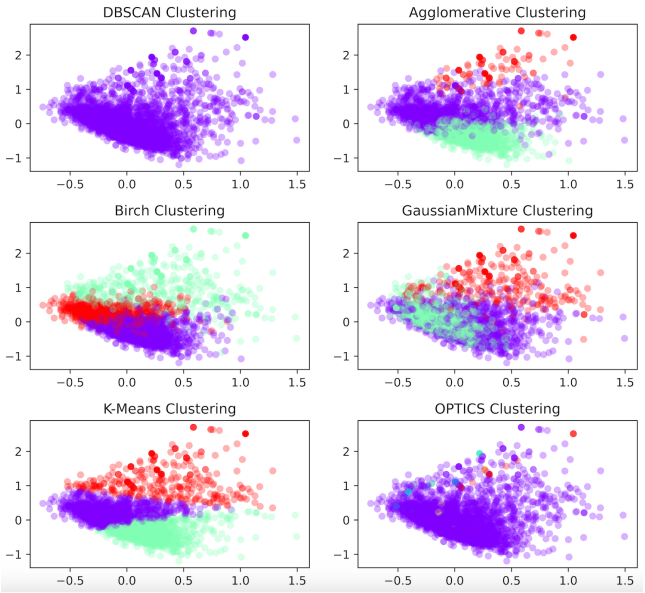

Figure 3: The embedding distribution of tweets in dimensionality reduced by PCA with different clustering methods.)

Table 3 shows the number of labels in the new datasets created with the five mentioned tricks. In this table, 4335 tweets are included in the train collection and 765 tweets are included in the test collection (85 to 15 ratio). Also, in the axis embedding methods, the number of Label 2 has been reduced at higher threshold values. Because according to formula (2), with an increase in the threshold value, less weight is given to positive tweets to be similar to the average of negative tweets. As can be seen in Table 3, in all methods, the number of negative tweets is fixed equal to 1230, which is the total number of negative tweets in the training set.

Table 3: Number of each label in new datasets with 5 methods(axis embedding and clustering)

Method	Train Set			
	label 0	label 1	label 2	sum
Axis_Emb (t=0)	1219	1230	1886	4335
Axis_Emb (t=0.03)	1695	1230	1410	4335
Axis_Emb (t=0.05)	2213	1230	892	4335
Clustering (GM)	1260	1230	1845	4335
Clustering (Birch)	1155	1230	1950	4335

Now, in Table 4, we see the results of the best hybrid models for the proposed two-stage classification method. In this table, all the basic models considered in Section 4.2 in the first and second stages of our approach have been tested, and the best models have been selected in terms of macro F1 score. Finally, with the consensus of the predictions of the first and second classifications, the best model is RF-RF with the axis embedding trick with threshold 0, whose F1-macro is 79% and F1-weighted is 82%. Also, the Birch method has been able to be very close to the best accuracy.

Table 4: Best results of two-stage models in different methods for building new datasets

Method	Model	Class	P	R	F1	F1m	F1w
t=0	RF-RF	1	70.3%	66%	75.2%	78.8%	81.9%
		0	87.3%	89.8%	84.9%		
t=0.03	MLP-GNB	1	57.1%	42.9%	85.5%	60.1%	59.4%
		0	63%	88.2%	48.9%		
t=0.05	SVM-LR	1	72.3%	82.6%	77.1%	32.9%	65%
		0	52.6%	51.9%	51.4%		
GM	MLP-GNB	1	51.8%	50.7%	52.9%	66.5%	72.7%
		0	81.1%	81.8%	80.5%		
Birch	LR-XGB	1	68.4%	72.8%	64.5%	77.2%	80.9%
		0	85.9%	83.7%	88.3%		

5 Discussion

As noted in Section 3.1, a total of 10,292 tweets were published by 50 users in the year leading up to election day. We applied the optimal Random Forest (RF-RF) model to classify 5,192 unlabeled tweets, resulting in 3,258 negative tweets and 7,034 positive tweets. To investigate the factors influencing negativity, we utilized a negative binomial regression model.

One of the independent variables in our analysis is is_candidate, which indicates whether the user is a candidate in the election. As shown in Table 5, this variable is not statistically significant, suggesting no relationship between being a candidate and the occurrence of negative tweets during the 2021 Iranian presidential election. In contrast, the variable person_names_count, which represents the number of political figures mentioned in the tweets, reached statistical significance with a high coefficient value. This indicates a direct relationship between the presence of political figures' names in tweets and increased negativity.

Additionally, we found that variables related to the lifespan of the tweets and the number of swear words present are not statistically significant. However, the user account lifespan variable is statistically significant and indicates an indirect relationship with negativity, albeit with a low coefficient. Notably, the organization_names_count variable is statistically significant, shows a positive relationship with negativity. This suggests that tweets mentioning government organizations and institutions are more likely to be negative. These findings highlight the complexities surrounding negativity in political discourse, indicating that while being a candidate does not significantly affect negativity, references to political figures and organizations play a substantial role in shaping the sentiment expressed in tweets.

Table 5: Drivers of campaign negativity in 2021 presidential election in Iran

variable names	coef	std err	p
tweet_age	-0.0001	1.0e-4	0.273
account_age ***	-0.0002	1.0e-4	0.003
organize_names_count ***	0.1348	4.5e-2	0.003
person_names_count ***	0.2175	6.8e-2	0.001
is_candidate	0.0197	9.7e-2	0.839
swear_words_count	0.1146	1.7e-1	0.5

$* : p \leq 0.1, ** : p \leq 0.05, *** : p \leq 0.01$

6 Conclusion

In this paper, we aimed to develop an automated model for detecting campaign negativity, as outlined in the introduction. A primary motivation for creating this model is to enable quicker identification of negative sentiments in electoral campaigns. This capability can provide timely responses to attacks from opposing parties and assist media outlets in generating more accurate and prompt reports. Additionally, our model can serve as a valuable component in social media analysis dashboards utilized by private companies.

One of the key motivations for this research is the lack of similar work focusing on Persian data related to Iran's elections. We are proud to present an artificial intelligence model specifically designed for this context for the first time.

Looking ahead, future work could explore the integration of Large Language Models (LLMs) to enhance data labeling and improve model accuracy. These models could function independently, allowing for comparative analyses with our current model through prompt optimization and testing on experimental datasets. Furthermore, a potential enhancement to our existing model would be the capability to identify the nature of attacks, distinguishing between political and personal attacks. This would necessitate expanding the dataset to include more negative examples. Another area for improvement is incorporating text recognition from images, as many tweets contain images with accompanying text.

References

[1] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[2] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao. Deep learning–based text classification: a comprehensive review. *ACM Computing Surveys (CSUR)*, 54(3):1–40, 2021.

[3] A. Nai. Going negative, worldwide: Towards a general understanding of determinants and targets of negative campaigning. *Government and Opposition*, 55(3):430–455, 2020.

[4] D. A. M. Peterson and P. A. Djupe. When primary campaigns go negative: The determinants of campaign negativity. *Political Research Quarterly*, 58(1):45–54, 2005.

[5] T. B. Krebs and D. B. Holian. Competitive positioning, deracialization, and attack speech: A study of negative campaigning in the 2001 Los Angeles mayoral election. *American Politics Research*, 35(1):123–149, 2007.

[6] E. J. Schweitzer. Attack politics on the internet: Comparing German and American E-campaigns. *APSA 2009 Toronto Meeting Paper*, 2009.

[7] M. Grossmann. What (or who) makes campaigns negative? *American Review of Politics*, 33:1–22, 2012.

[8] H. J. G. Hassell and K. R. Oeltjenbruns. When to attack: The trajectory of congressional campaign negativity. *American Politics Research*, 44(2):222–246, 2016.

[9] M. Babaei, S. Moradi, and A. A. Ghasemi. Destruction and negative campaigns in Iran's presidential elections; causes and contexts. *American Politics Research*, 14(1):35–62, 2018.

[10] A. S. Walter and C. Van der Eijk. Measures of campaign negativity: Comparing approaches and eliminating partisan bias. *The International Journal of Press/Politics*, 24(3):363–382, 2019.

[11] J. Maier and A. Nai. Roaring candidates in the spotlight: Campaign negativity, emotions, and media coverage in 107 national elections. *The International Journal of Press/Politics*, 25(4):576–606, 2020.

[12] A. Nai. Fear and loathing in populist campaigns? Comparing the communication style of populists and non-populists in elections worldwide. *Journal of Political Marketing*, 20(2):219–250, 2021.

[13] T. Kim. Violent political rhetoric on Twitter. *Political Science Research and Methods*, 11(4):673–695, 2023.

[14] V. Petkevic and A. Nai. Political attacks in 280 characters or less: A new tool for the automated classification of campaign negativity on social media. *American Politics Research*, 50(3):279–302, 2022.

[15] J. Maier and A. Nai. Mapping the drivers of negative campaigning: Insights from a candidate survey. *International Political Science Review*, 44(2):195–211, 2023.

[16] K. Mattes and D. P. Redlawsk. The positive case for negative campaigning. *University of Chicago Press*, 2020.

[17] M. Khodak, N. Saunshi, Y. Liang, T. Ma, B. Stewart, and S. Arora. A la carte embedding: Cheap but effective induction of semantic feature vectors. *arXiv preprint arXiv:1805.05388*, 2018.

[18] K. Zhou, K. Ethayarajh, D. Card, and D. Jurafsky. Problems with cosine as a measure of embedding similarity for high frequency words. *arXiv preprint arXiv:2205.05092*, 2022.

[19] T. Ouyang and X. Shen. Online structural clustering based on DBSCAN extension with granular descriptors. *Information Sciences*, 607:688–704, 2022.

[20] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming. K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Information Sciences*, 2022.

[21] N. Randriamihamison, N. Vialaneix, and P. Neuvial. Applicability and interpretability of Ward's hierarchical agglomerative clustering with or without contiguity constraints. *Journal of Classification*, 38(2):363–389, 2021.

[22] F. Ramadhani, M. Zarlis, and S. Suwilo. Improve BIRCH algorithm for big data clustering. *IOP Conference Series: Materials Science and Engineering*, 725(1):012090, 2020.

[23] S. Adams and P. A. Beling. A survey of feature selection methods for Gaussian mixture models and hidden Markov models. *Artificial Intelligence Review*, 52:1739–1779, 2019.

[24] P. Bhattacharjee and P. Mitra A survey of density based clustering algorithms. *Frontiers of Computer Science*, 15:1–27, 2021.

[25] A. Tharwat Classification assessment methods. *Applied Computing and Informatics*, 17(1):168–192, 2020. Emerald Publishing Limited.

[26] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu, and L. He A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(2):1–41, 2022. ACM New York, NY.

[27] R. Liu, Y. Li, L. Tao, D. Liang, and H.-T. Zheng Are we ready for a new paradigm shift? a survey on visual deep MLP. *Patterns*, 3(7), 2022. Elsevier.

[28] H. Gohiya, H. Lohiya, and K. Patidar A survey of XGBoost system. *International Journal of Advanced Technology and Engineering Research*, 8:25–30, 2018.

[29] P. A. A. Resende and A. C. Drummond A survey of random forest based methods for intrusion detection systems. *ACM Computing Surveys (CSUR)*, 51(3):1–36, 2018. ACM New York, NY, USA.

[30] E. Y. Boateng and D. A. Abaye A review of the logistic regression model with emphasis on medical research. *Journal of Data Analysis and Information Processing*, 7(4):190–207, 2019. Scientific Research Publishing.

[31] I. Wickramasinghe and H. Kalutarage Naive Bayes: applications, variations, and vulnerabilities: a review of the literature with code snippets for implementation. *Soft Computing*, 25(3):2277–2293, 2021. Springer.

[32] M. A. Chandra and S. S. Bedi Survey on SVM and their application in image classification. *International Journal of Information Technology*, 13:1–11, 2021. Springer.

[33] U. W. Wijayanto and R. Sarno An experimental study of supervised sentiment analysis using Gaussian Naïve Bayes. In *2018 International Seminar on Application for Technology of Information and Communication*, 476–481. IEEE, 2018.

[34] M. N. A. H. Sha'Abani, N. Fuad, N. Jamal, and M. F. Ismail kNN and SVM classification for EEG: a review. In *InECCE2019: Proceedings of the 5th International Conference on Electrical, Control & Computer Engineering, Kuantan, Pahang, Malaysia, 29th July 2019*, 555–565. Springer, 2020.

[35] E. Dobriban and S. Wager High-dimensional asymptotics of prediction: Ridge regression and classification. *The Annals of Statistics*, 46(1):247–279, 2018. JSTOR.

[36] R. Sun, G. Wang, W. Zhang, L.-T. Hsu, and W. Y. Ochieng A gradient boosting decision tree based GPS signal reception classification algorithm. *Applied Soft Computing*, 86:105942, 2020. Elsevier.

[37] F. Mignacco, F. Krzakala, P. Urbani, and L. Zdeborová Dynamical mean-field theory for stochastic gradient descent in Gaussian mixture classification. *Advances in Neural Information Processing Systems*, 33:9540–9550, 2020.

[38] J. Pareek and J. Jacob Data compression and visualization using PCA and T-SNE. In *Advances in Information Communication Technology and Computing: Proceedings of AICTC 2019*, 327–337. Springer, 2021.

[39] M. Farahani, M. Gharachorloo, M. Farahani, and M. Manthouri ParsBERT: Transformer-based model for Persian language understanding. *Neural Processing Letters*, 53:3831–3847, 2021. Springer.

[40] T. Pires, E. Schlinger, and D. Garrette How multilingual is multilingual BERT? *arXiv preprint arXiv:1906.01502*, 2019.

[41] V. Sanh, L. Debut, J. Chaumond, and T. Wolf DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[42] S. Thavareesan and S. Mahesan Sentiment lexicon expansion using Word2vec and fastText for sentiment prediction in Tamil texts. In *2020 Moratuwa Engineering Research Conference (MERCon)*, 272–276. IEEE, 2020.

[43] E. M. Dharma, F. Lumban Gaol, H. L. H. S. Warnars, and B. Soewito The accuracy comparison among Word2Vec, GloVe, and FastText towards convolutional neural network (CNN) text classification. *J Theor Appl Inf Technol*, 100(2):31, 2022.