Understanding the Impact of Pre-Processing Text on Rotten Tomato Reviews

Fasih Munir
230057165
MSc Data Science
fasih.munir@city.ac.uk

1 Problem statement and motivation

Online reviews and discussion are especially prevalent for movies. Websites such as Rotten Tomato and IMDB are two popular online sources for people to learn about movies and shows that they are thinking about watching. These websites hold a large amount of reviews that people have written. These reviews tell us how a movie was received and are contextual cues that provide "the other side of the story" of a movie's success as opposed to how much it earned. It thus becomes important to gather, parse and understand this information so that future movies and shows can be guided better.

This, however, is a challenging task to accomplish given the nuances present in human language. The solution to this task is not as simple as hiring people to sit and read every review as the cost is immense and operationally difficult. The challenge thus becomes trying to get machines to understand as much as possible. This becomes difficult as reviews that are left can have varying degrees of sentiment, be filled with sarcasm or humour and, as the human language continues to evolve, filled with generational slang that may not be formalised.

This study will aim to understand how different preprocessing techniques affect the classification of a review as either having a positive sentiment (1) or a negative sentiment (0). There is not straightforward answer given that the way language is expressed can be different. Some reviews might be well crafted with no spelling errors while some may have words or errors that may not fit the available corpus. This makes it tricky, for example, to assert whether stemming or lemmatization is the best way.

The research question for this paper can then be defined as follows: How do different preprocessing techniques impact the ability of machine learning models to classify movie/show reviews into a binary output of positive or negative sentiment.

2 Research hypothesis

The hypothesis is that lemmatization combined with different feature extraction techniques such as Bag of Words (BOW), Term Frequency Inverse Document Frequency (TFIDF) and Word Embeddings (WE) will outperform stemming combined with the same feature extraction techniques. This is because while both bring a word down to a root, lemmatization looks at context and generally returns an actual and accurate (running becomes run) while stemming simply cuts the end few letters off which sometimes results in fake words (beatiful becomes beauti). The performance is checked through the overall accuracy of various trained models which include Support Vector Machines (SVM), Random Forests (RF) and Multilayer Perceptrons (MLP). Additionally, we will compare the accuracies against a pre-trained transformer model, BERT. In the case of this paper, a check point version of BERT known as DistilBERT (Maintainers, 2022) will be used. The comparative difference is in the fact that DistilBERT will be fed unprocessed data.

Running multiple tests on various combinations of feature extraction techniques that process the data and feed it to different models will enable us to find a deeper understanding of the possible answers to our research question. Given that this paper is focused on understanding which preprocessing will work best, a wider approach is used, which is to say that we test many combinations on relatively untuned models to see how varying them will impact accuracy. While model tuning can help improve performance (Arnold et al., 2024) , the goal is less about finding the right tuned model and more about finding the right preprocessing to then tune a model since tuning through gridsearch or random search can be computationally expensive.

3 Related work and background

A broad understanding of the methods and techniques employed in sentiment analysis is described by Wankhade M (2022). The paper informs us about the idea of polarity, which is the classification of text into either positive or negative sentiments, and the various techniques that have been used over the years to identify polarity. The paper also talks about various industries where this work can be applied such as customer reviews which reinforces the aim of this paper to correctly analyze movie review sentiments. Additionally, the authors noted that the ambiguity of words is an obstacle that needs to be overcome.

Another paper by Kenyon-Dean et al. (2018) extends the discussion on the difficulty of sentiment analysis by arguing that many of the datasets being used to train models tend to remove complicated annotated data. Examples of sentiments that had a hard time being categorized by multiple human annotators tend to be dropped from training sets. However, this tends to reduce the amount of data and takes away from model accuracy as many sentiments that are expressed online are complicated.

A final paper on the general space of sentiment analysis was reviewed, which talked about temporal changes in the polarity of lexical features which can lead to degrading performance (Lukes and Søgaard, 2018). It was noted that since datasets being used for training are now a few years old and the our language begins to incorporate new kinds of words, the models will begin to fail if new datasets are unable to be created.

The most relevant study was conducted by Pang et al. (2002), where the authors attempted to classify movie review sentiment using the Naive Bayes (NB), Maximum Entropy (ME) and Support Vector Machine algorithms. Their approach involved the use of an equally balanced dataset of positive and negative reviews (700 each) and they used 3-fold cross validation to inform their accuracy scores. Punctuation was kept in and no stemming was used. They experimented with unigrams, bigrams and Part of Speech (POS) tagging. Their best model was an SVM that achieved an accuracy of 0.829 on data prepared with unigrams. These results outperformed their human baselines which achieved a maximum accuracy of 0.69. Our paper builds on this work by using an SVM model alongside RF and MLP models to add some difference. Additionally our paper extends the preprocessing to trigrams and includes stemming and lemmatization.

Another relevant paper reviewed was authored by Cho and Yoon (2017) where they introduced TF-IDF and incorporated truncated Singular Value Decomposition (SVD) to try and improve their sentiment classification. Their setup involved creating three different datasets which included common, distinct and merged words. This was done by splitting the original dataset based off their review ranking and then using TF-IDF weights to select the top words. They then used truncated SVD to reduce the sparse matrix that gets generated. SVD is computationally efficient as it reduces the large matrix, keeping a good approximation of the data and reducing compute. Their paper was able to achieve an accuracy of approximately 0.61. The SVD technique was incorporated in our paper.

Tweets generally tend to hold the language used currently along with varying degrees of slang, punctuation, errors and emojis. Agarwal et al. (2011) account for special characters and emojis by replacing them with their respective sentiment such as a smiley face emoji being tagged with a positive sentiment. Their paper concluded that POS tags improve classification over a base unigram model. Saif et al. (2014) study the impact of removing stop words within Twitter datasets. Their experiments were run on ME and NB models that used unigram features and their best performance came when stopwords were removed using the TF1 removal method.Yüksel et al. (2019) used transformer encoding to classify sentiment in the Turkish Language. This was interesting as it shows the difficulties of extending techniques to languages that have one to many mapped words. Their study notes the importance of morphological analysis to understand these word mappings. They further note that transformers do not require morphological work and can outperform those that do.

Taboada et al. (2011) introduces an update to lexicon based approaches for sentiment classification. This approach is different from building a classifier in that a score is calculated based of a set of words in the dictionary. While their approach is not used directly in this paper, the sentiment intensity analyzer (nltk libary) was employed during exploratory analysis. The last paper reviewed was authored by Kertkeidkachorn and Shirai (2023) where they introduced a graph neural net to incorporate the relationship between a product and the end user to inform sentiment analysis. Since their model was

not able to outperform BERT models, this paper has stuck to using BERT as a final comparison.

3.1 Accomplishments

- Task 1: Preprocess the data to remove stopwords, punctuation, contractions and convert to lower case - Completed
- Task 2: Tokenise the dataset - Completed
- Task 3: Conduct EDA to understand the dataset and initial sentiment distribution - Completed
- Task 4: Build base models with default parameters for RF, SVM and MLP - Completed
- Task 5: Extend the basic POS tagging in lemmatization to incorporate not just nouns but also verbs and adjectives - Completed
- Task 6: Create different combinations of the dataset using the extended lemmatization, BOW, TF-IDF and different grams - Completed
- Task 7: Create different combinations of the dataset using stemming, BOW, TF-IDF and different grams - Completed
- Task 8: Create different combinations of the dataset using word embeddings (Twitter 25) coupled with stemming and lemmatization - Completed
- Task 9: Experiment with different word embeddings - Failed due to time constraints
- Task 10: Using the models created with default parameters, test the different dataset combinations from tasks 6 and 7 and cross validate the resulting accuracy - Completed
- Task 11: Tune the models on different hyperparameters to improve the accuracy - Failed due to time constraints but one round of tuning was attempted
- Task 12: Use SVD to reduce the feature space of the combinations from tasks 6 - 9 and then experiment on the base models with default parameters - Completed
- Task 13: Implement a BERT model - Failed, while a model was not implemented from scratch, a pre-trained model was pulled from hugging face to compute the accuracy on our dataset
- Task 14: Preform error analysis - Completed

4 Approach and Methodology

A breadth first approach was used where 50 combinations of dataset preprocessing were experimented with. 12 combinations came from stemming and lemmatization coupled with unigram, bigram and trigram versions of BOW and TF-IDF. 2 combinations came from a word embedding approach coupled with stemming and lemmatization. A further 3 sets of 12 combinations came from taking the initial 12 combinations and applying an SVD technique on them to reduce the features down to 100, 1000 and 2000. These combinations not only help capture the naive approach of using every word individually but also capturing the context by pairing surrounding words and understanding their role (through bigrams and trigrams). Before the combinations were created, simple preprocessing was done on the data such as stop word removal to reduce the impact of possible fake reviews or nonsensical reviews. To accomplish this, pre-existing models and architecture was used for example, rather than writing a new stemming algorithm, the Porter Stemmer from the nltk library was used. The gensim library was also used to implement the word embeddings using the Twitter 25 set. These combinations were then experimented with using 3 machine learning models that include SVM, MLP and RF which were implemented using sklearn, pandas and numpy. These models were trained with 5-fold cross validation and maintained the balance of the dataset in each fold. The hugging face transformer library was uses to implement a pretrained version of BERT called DistilBERT (Maintainers, 2022) (without training) to provide a contrast as to what a transformer model can achieve without preprocessing. The classic models used mostly default parameters (found on the sklearn website) and a single iteration of different hyperparameters was implemented. Additionally, matplotlib and seaborn were used for visualization. Using SVM, MLP, RF and DistilBERT gives the widest range of models we can use as we cover ensemble methods, nerual nets, transformers and distance based algorithms.

Many papers before have proven the improvement gain from hyperparameter tuning but, as seen from the literature review, there is no particular consensus on the best way to process data for senti-

ment analysis. 3 papers reviewed that worked with tweets preprocessed the tweets differently. So our focus became to understand how using different techniques would affect results just as how Saif et al. (2014) kept their models simple but changed the way stop words were removed.

The primary issue that came with this work was that of processing time. The numerous combinations and large feature space meant that all 3 classic models were taking hours to train. To combat this, the default parameters were adjusted for RF and MLP. The depth of the RF was limited to 10 trees while for the MLP hidden layer size was reduced to 50 (only 1 layer), early stopping was set to true and the number of iterations without a change in score was set to 5 with a tolerance of 0.01. This greatly improved the training time, likely at the cost of reduced accuracy. A second limitation comes in the form of the word embeddings where to reduce compute a smaller set was used namely the Twitter 25 set as opposed to a Google set. This could have limited the performance of the word embeddings. A third limitation of this approach is that the data used is strictly a movie dataset so the review context is movie only.

5 Dataset

Our dataset can be found on hugging face and is known as the Rotten Tomato dataset (Pang and Lee, 2005). It has 8,530 unique observations divided into a text column that holds the reviews of the movies and a label column that holds a binary sentiment of either positive (1) or negative (0). There are approximately 111,658 words. The dataset can be easily imported directly into a colab notebook using the hugging face libraries. The dataset is small enough to use in full and has an even balance of positive to negative sentiments.

Positive sentiment example: "if you sometimes like to go to the movies to have fun , wasabi is a good place to start . "

Negative sentiment example: "an incredibly irritating comedy about thoroughly vacuous people . . . manages to embody the worst excesses of nouvelle vague without any of its sense of fun or energy ."

While the dataset is split into a train, validation and test set, since we are using a cross validation technique the train and validation sets were merged to form one set.

Figure 1 shows some summary stats of the data.

count	10662
mean	10
std	4
min	1
25%	7
50%	10
75%	14
max	39

Figure 1: Summary Statistics for the Full Data

On average a review is about 10 words (similar to a tweet). The longest review is 39 words while the shortest is 1 word.

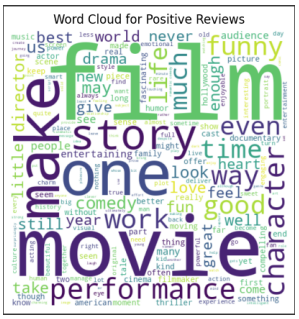

Figure 2: Positive Review Word Cloud

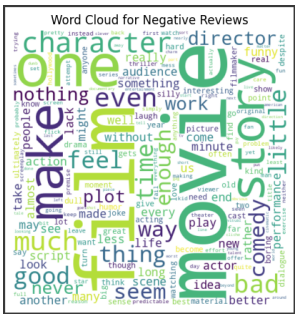

Figure 3: Negative Review Word Cloud

Figures 2 and 3 show the most common words (the larger the word the more common) used for positive and negative reviews. 2 words that standout the most are "character" and "story". Both words are used to describe both sentiments. This points out a slight complexity in our data in that it is likely that the positive and negative sentiments will share many similar words and the contextual cues may hold greater importance towards classification.

This fuzziness is further emphasized by a sentiment score calculated using the sentiment intensity function in nltk. This is based on VADER (Valence Aware Dictionary and sEntiment Reasoner) which is a rule-base lexicon tuned to understand sentiments and is able to account for different punctuation and case. Although our data has been preprocessed (much of it already via hugging face) and

we have lost some of the complexity that VADER would need, we can still understand the sentiment of the words on their own which gives us some surprising results.

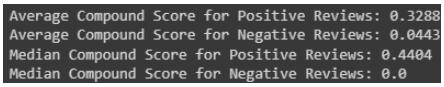

Figure 4: Sentiment Intensity Score on Dataset

Figure 4 shows the intensity score which generally range from -1 to 1 where the sign and size reflect the sentiment and intensity. While the average and median values for positive reviews are low to medium in positive sentiment, the negative reviews are also showing a non-negative sentiment. The average and median being so close together also indicates that there is little room for outliers. This will likely cause a hindrance to our classification results as reviews classified with negative sentiment will likely use words that have positive sentiment.

5.1 Dataset Preprocessing

While sifting through the examples on huggingface for our chosen dataset it seems that the data is already preprocessed in terms of lower casing, we still went ahead and made a simple function that turned the reviews to lower case, removed all punctuation, removed stopwords and fixed contractions. This helped in normalizing the text.

We then made multiple copies of the dataset and applied the following further processing which was then tested on different machine learning models:

1. Lemmatization with BOW (includes unigram, bigram and trigram)
2. Lemmatization with TF-IDF (includes unigram, bigram and trigram)
3. Stemming with BOW (includes unigram, bigram and trigram)
4. Stemming with TF-IDF (includes unigram, bigram and trigram)
5. Word Embeddings through Twitter 25 (followed by stemming and lemmatization)
6. All the above except number 5 were then processed further with SVD to reduce the feature space and tested on 100, 1,000 and 2,000 features.

Given how the research is framed, this is essentially the method and the problem. Since there is no consensus on the best preprocessing steps, the way to find an answer is by exhausting the possible options. Not tuning the hyperparameters of a model but tuning the preprocessing of a dataset. The list above is not indicative of all the possible combinations and was constrained on time and complexity.

Results for 100 and 1,000 features (point 6) can be found in the appendix.

6 Baselines

Three baselines are used to compare the output of results and all results are based on the accuracy metric which is defined as the number of correct predictions across all the predictions made. The first is a random guess which would give us an accuracy of 0.5. This is the same as flipping a coin. The second is used by Pang et al. (2002). The authors created a human baseline (a human classified the sentiments) which ranged between 0.5 to 0.69. The third baseline is a simple preprocessed dataset we created which had only the initial preprocessing done and then BOW was applied directly at a unigram level with no stemming or lemmatization. This achieved a training accuracy of 0.752 (MLP model).

These baselines are useful as they give various check points of what a machine learning model would need to beat to be considered useful. It needs to be better than a random guess, it needs to perform better than a human and if stemming and lemmatization are really that important then those techniques should perform better than a model where they are not used.

7 Results, Error Analysis

Figures 5 and 6 show the training accuracies achieved during experimentation. The highest accuracy we achieved came from the MLP model using stemmed, TF-IDF unigrams (X stem tfidf uni) at 0.765. This is not dramatically higher than the other scores neither is this considered best in class but it does better than the random guess and human base lines and only marginally better than the basic preprocessing baseline. The first thing that stands out is that increasing the number of grams makes the models worse (no better than a random guess) which is surprising given that increasing the grams is supposed to add more context to the training. It seems that the increased context either in full

	Pre-Processing	SVM	RF	MLP	Feature Count
Average Accuracy					
0	X_lemma_bow_uni	0.737703	0.674240	0.750519	16008
0	X_lemma_bow_bi	0.557627	0.551271	0.614527	79014
0	X_lemma_bow_tri	0.509067	0.508545	0.510422	79966
0	X_lemma_tfidf_uni	0.755626	0.665380	0.757085	16008
0	X_lemma_tfidf_bi	0.609838	0.550124	0.606503	79014
0	X_lemma_tfidf_tri	0.506044	0.507711	0.511983	79966
0	X_lemma_we	0.667882	0.614839	0.666632	25
0	X_stem_bow_uni	0.745623	0.683305	0.750623	13645
0	X_stem_bow_bi	0.559607	0.551688	0.606814	78797
0	X_stem_bow_tri	0.510526	0.509066	0.512088	80227
0	X_stem_tfidf_uni	0.759795	0.667466	0.765214	13645
0	X_stem_tfidf_bi	0.611818	0.552626	0.614005	78797
0	X_stem_tfidf_tri	0.506044	0.508649	0.509484	80227
0	X_stem_we	0.627240	0.563881	0.624010	25

Figure 5: All Training Accuracies Without SVD

combined_shrunk_results_df

	Pre-Processing	SVM	RF	MLP	Feature Count
Average Accuracy					
0	X_lemma_bow_uni_shrunk	0.736660	0.540226	0.728219	2000
0	X_lemma_bow_bi_shrunk	0.584619	0.530222	0.591703	2000
0	X_lemma_bow_tri_shrunk	0.511046	0.513025	0.500104	2000
0	X_lemma_tfidf_uni_shrunk	0.749894	0.552000	0.739578	2000
0	X_lemma_tfidf_bi_shrunk	0.603584	0.536054	0.589828	2000
0	X_lemma_tfidf_tri_shrunk	0.523656	0.505835	0.509065	2000
0	X_stem_bow_uni_shrunk	0.742913	0.544079	0.732909	2000
0	X_stem_bow_bi_shrunk	0.580762	0.524803	0.584306	2000
0	X_stem_bow_tri_shrunk	0.509692	0.498228	0.513547	2000
0	X_stem_tfidf_uni_shrunk	0.758648	0.544393	0.738536	2000
0	X_stem_tfidf_bi_shrunk	0.599311	0.534598	0.578574	2000
0	X_stem_tfidf_tri_shrunk	0.511567	0.500832	0.505419	2000

Figure 6: All Training Accuracies With SVD - 2000 Features

or shrunk down is not useful and the simple unigram models perform the best. Next we notice that while results are not better, they are quite similar across the board when comparing it from a with or without SVD perspective. Given the reduction in feature space gives us a comparable output we could say that those models perform better. We can further observe that the word embedding models do not perform well and nearly similar to the human baseline. This is likely due to the limited corpus used. A larger set would have included more words to build from. RF models also do not perform well in this setting likely because these are tree based models and they require more data to split on. The high dimensions, sparse data and likely irrelevant features that rarely occur are probably causing problems for RF. Further, stemming seems to perform marginally better than lemmatization which is again surprising as the latter has robust POS tagging to capture context. TF-IDF versions also perform better than the BOW versions likely because of the normalizing nature of TF-IDF. However, overall the results seem to indicate that the simpler, less computationally expensive techniques are the ones to choose. We further tested 1 iteration of models with different tunings (see appendix) on "X stem tfidf uni shrunk" (scores are comparable and saves compute) but only achieved a training accuracy of 0.747 on SVM.

We chose to take forward the "X stem tfidf uni" combination to calculate test set accuracies using SVM and MLP.

SVM Classifier Report:

	precision	recall	f1-score	support
Negative (0)	0.77	0.80	0.78	533
Positive (1)	0.79	0.76	0.77	533
accuracy			0.78	1066
macro avg	0.78	0.78	0.78	1066
weighted avg	0.78	0.78	0.78	1066

MLP Classifier Report:

	precision	recall	f1-score	support
Negative (0)	0.76	0.83	0.79	533
Positive (1)	0.81	0.74	0.77	533
accuracy			0.78	1066
macro avg	0.78	0.78	0.78	1066
weighted avg	0.78	0.78	0.78	1066

Figure 7: Test Data Classification Report

Figure 7 shows the results of the test set and it is surprising to see that the accuracy actually improves on the test set going up to 0.78 which is a sign of good generalizability. The SVM has more balanced scores while the MLP model looks to prioritize finding the negatives (higher recall). The precision for the negative class being lower in both models compared to the positive class could be due to the overall sentiment intensity of the negative class being non-negative - both models are more sure of the positive class (see data section).

Figure 8 shows the AUC scores of both models and while they are quite similar, they are much higher than the accuracy scores at 0.86. This suggests that there is likely a different threshold for calculating the positive or negative sentiment that can improve the model performances if tweaked and this is likely true given our sentiment intensity scores. Keep in mind, all these scores are without the additional overhead of fine-tuning models and they are already out performing some comparable papers (Cho and Yoon, 2017). We should however note that the test accuracy of the pretrained DistilBERT model was 0.81 which was a plug and play model that required little to no preprocessing. While our scores are not far off , it is likely that with some tuning we can reach the DistilBERT scores,

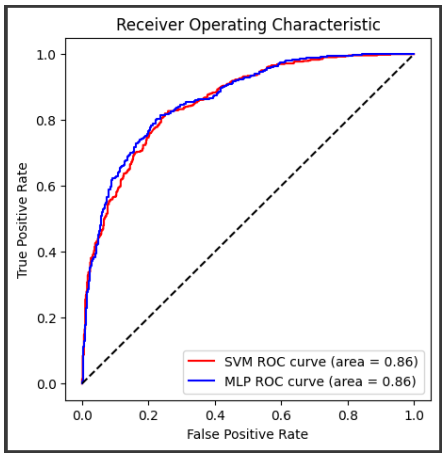

Figure 8: ROC Curve and AUC Scores For SVM and MLP on Test Set

although further tuning DistilBERT will likely lead it to outperform the classic models.

Looking at some direct examples of misclassification can give us a sense of where the models were struggling.

True Label (0), SVM (1), MLP (1):

1. "sandra bullock and hugh grant make a great team , but this predictable romantic comedy should get a pink slip ."

2. "a film without surprise geared toward maximum comfort and familiarity ."

In the above examples, both models incorrectly predicted the sentiment to be positive. Likely because in the first case the nuance in knowing what is meant by a pink slip was not captured. The model must have taken both as separate tokens which, on their own, do not mean much. In the second case there seems to be a presence of more positive sentiment than negative which makes the models less sure.

True Label (1), SVM (0), MLP (0):

1. "what's so striking about jolie's performance is that she never lets her character become a caricature – not even with that radioactive hair"

2. "mostly , [goldbacher] just lets her complicated characters be unruly , confusing and , through it all , human "

In the above examples both models incorrectly predict a negative sentiment. Here a pattern seems to emerge, also present in the pink slip example where the nuance of the words has been missed. The word "never" in the first case is not read with negative connotation while in the second case the complexity of saying that the person lets her characters be human gets lost. Likely "human" is over shadowed by the presence of immediately negative words such as "unruly" and "confusing". While the imbalance of sentiment within a review can likely be tuned away, the approaches presented in this paper will likely not be able to handle nuanced context such as being human or getting a pink slip. These will likely require bigger word embeddings or finely tuned transformer models using large datasets.

8 Lessons Learned and Conclusions

This paper aimed to explore various preprocessing techniques to understand which might be the most effective for sentiment classification for movie reviews. Our initial hypothesis of lemmatization performing better was proven to be incorrect with the models performing better on the simpler stemming technique. Using feature reduction techniques such as SVD while may not result in better accuracy, results in nearly equal accuracy for less compute. Misclassifications tend to occur either when there is imbalanced in-review sentiment or when the meaning of a phrase is not apparent. A larger dictionary of embeddings can be used to capture these cases. Overall, it seems that simpler is better and future iterations of this work can focus on fine-tuning models on simpler preprocessing techniques.

References

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of Twitter data. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, pages 30–38, Portland, Oregon. Association for Computational Linguistics.

Christian Arnold, Luka Biedebach, Andreas Küpfer, and Marcel Neunhoeffer. 2024. The role of hyperparameters in machine learning models and how to tune them. *Political Science Research and Methods*, page 1–8.

Heeryon Cho and Sang Min Yoon. 2017. Improving sentiment classification through distinct word selection. In *2017 10th International Conference on Human System Interactions (HSI)*, pages 202–205.

Kian Kenyon-Dean, Eisha Ahmed, Scott Fujimoto, Jeremy Georges-Filteau, Christopher Glasz, Barleen Kaur, Auguste Lalande, Shruti Bhanderi, Robert

Belfer, Nirmal Kanagasabai, Roman Sarrazingendron, Rohit Verma, and Derek Ruths. 2018. Sentiment analysis: It's complicated! In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1886–1895, New Orleans, Louisiana. Association for Computational Linguistics.

Natthawut Kertkeidkachorn and Kiyoaki Shirai. 2023. Sentiment analysis using the relationship between users and products. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8611–8618, Toronto, Canada. Association for Computational Linguistics.

Jan Lukes and Anders Søgaard. 2018. Sentiment analysis under temporal shift. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 65–71, Brussels, Belgium. Association for Computational Linguistics.

HF Canonical Model Maintainers. 2022. distilbert-base-uncased-finetuned-sst-2-english (revision bfdd146).

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics.

Hassan Saif, Miriam Fernandez, Yulan He, and Harith Alani. 2014. On stopwords, filtering and data sparsity for sentiment analysis of Twitter. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 810–817, Reykjavik, Iceland. European Language Resources Association (ELRA).

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307.

Sekhara A Kulkarni C Wankhade M, Rao. 2022. A survey on sentiment analysis methods, applications, and challenges. pages 5731–5780.

Atıf Emre Yüksel, Yaşar Alim Türkmen, Arzucan Özgür, and Berna Altınel. 2019. Turkish tweet classification with transformer encoder. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1380–1387, Varna, Bulgaria. INCOMA Ltd.