

Sentiment Analysis of Movie Reviews
Comparing Traditional and Transfer Learning Approaches

Ananyaa Tanwar

School of Information Sciences, University of Illinois at Urbana-Champaign

IS 517: Methods of Data Science

Instructor Yaoyao Liu

December 9, 2025

Introduction

Figuring out the emotional tone in written text has become key for organizations looking to make data-driven decisions. Online reviews and discussions often reveal people's feelings about products, services, or experiences. While it's easy for us humans to tell if someone enjoyed or disliked a movie review, computers need a structured approach to interpret the language. That's where sentiment analysis comes in, helping automated systems to classify text as positive or negative, which is super helpful when there's a lot of feedback to process quickly and consistently.

Over the last two decades, natural language processing has really grown. Earlier methods looked at text just as a bunch of words that could be counted and analyzed statistically. One major method, known as Term Frequency–Inverse Document Frequency (TF-IDF), captures the importance of individual words in relation to the whole dataset to represent documents. It's straightforward and easy to understand, and it works well when there's enough domain-specific data available. More recent approaches have switched to distributed word representations, which aim to capture deeper meanings. A popular technique called Word2Vec creates vector representations by examining patterns found in large bodies of text. These vectors put similar words close together in a multi-dimensional space, letting machine learning models take advantage of semantic meaning instead of just focusing on word frequency.

This project takes a look at how these two very different approaches stack up when analyzing the sentiment of movie reviews. Specifically, it investigates if the pretrained Word2Vec embeddings, drawn from a wide news corpus, can do better than TF-IDF features that are created specifically from the movie review dataset. For this comparison, we'll use the IMDB Movie

Reviews dataset, which includes 50,000 reviews evenly divided between positive and negative sentiments. This dataset is a well-known benchmark for research in sentiment classification.

The bigger goal of this work is to figure out how the choice of text representation impacts model performance, accuracy, and behavior. With the increasing use of deep learning and pretrained embeddings in NLP, it's essential to explore if newer methods consistently provide advantages or if traditional models still shine in certain situations. By comparing TF-IDF and Word2Vec under controlled conditions, this project sheds light on the strengths and weaknesses of each approach and helps clarify how to choose the right tools for sentiment analysis.

Research Question & Hypothesis

This project aimed to explore how various text representation techniques affect how well sentiment classification models perform. While a lot of today's natural language processing tools use pretrained embeddings or deep learning, simpler methods like TF-IDF are still popular because they're straightforward, fast, and effective in many areas. In this study, I compared TF-IDF—where the representation comes directly from the movie review dataset—with Word2Vec, a pretrained embedding model that captures semantic information from a larger external corpus.

The main research question driving this project is:

Do pretrained Word2Vec embeddings boost sentiment classification accuracy when compared to TF-IDF features on IMDB movie reviews?

This question arises from claims in existing literature that distributed representations might better capture word meanings, including things like synonyms and contextual relationships. Since Word2Vec embeddings are trained on a vast general-purpose corpus, it's assumed they carry more

nuanced information than just basic word frequency counts. These potential advantages point to the idea that Word2Vec could help a classifier generalize better, even when review wording varies or includes less common vocabulary not prominently featured in the training data.

With this in mind, the original hypothesis was that Word2Vec would do better than TF-IDF. Specifically, it was expected that models using Word2Vec features would show an accuracy increase of around 5 to 10 percent compared to those using TF-IDF. This hypothesis is based on three main premises. First, Word2Vec captures semantic similarities, meaning different words that have similar meanings should lead to more informed decisions by the classifier. Second, Word2Vec is expected to deal with negations or variations in phrasing more effectively because it takes into account word co-occurrence patterns rather than just counting individual words. Third, since Word2Vec is pretrained on a large corpus, it should more easily understand rare or uncommon words compared to TF-IDF, which really relies on word frequency within a specific dataset.

The experiment was set up to test this hypothesis under controlled conditions; since I used the same classifiers for both feature types, any performance differences could be mainly attributed to the representation method. The investigation's results would ultimately reveal whether Word2Vec provides a real advantage in sentiment analysis tasks or if TF-IDF still stands as a strong baseline.

Dataset

For this project, I used the IMDB Movie Reviews dataset, which includes 50,000 reviews from users. It's a well-known benchmark for sentiment analysis research, making it a solid choice

for our work. I got it through TensorFlow Datasets, so it came with a structured format and predefined training and testing splits.

Dataset Description

The dataset is split evenly, with 25,000 reviews for training and 25,000 for testing. Each set has 12,500 positive and 12,500 negative reviews, which is great because it helps keep things balanced and reduces bias. This way, the accuracy and F1-scores can really reflect how well the model is performing.

The reviews themselves vary quite a bit in length and style. Some are short and to the point, while others dive deeper with several paragraphs discussing the plot and personal thoughts. This mix mirrors what you see in real-world user content and adds an interesting challenge for sentiment classification.

Data Source

I got the dataset from TensorFlow Datasets, which helps keep everything formatted consistently and makes it easy to reproduce the results. Since the train-test split is already set up, there's less risk of overlap or data leakage.

Data Structure & Labels

Every review comes with the raw text and a sentiment label. A label of 0 signifies negative sentiment, while 1 stands for positive sentiment. There's no neutral option, so this classification task is all about two sides.

Preprocessing

To get the data ready for modeling, I took several preprocessing steps. I converted all text to lowercase, normalized punctuation, and tokenized the reviews. I kept stopwords because they can carry important sentiment depending on the context.

For the TF-IDF method, I directly used the tokenized text to calculate TF-IDF features. With the Word2Vec approach, I matched each token to its pretrained Google News embedding when it was available. Any words not in that vocabulary were left out. I then averaged the vectors for each review to create a fixed-length representation. These preprocessing steps helped create uniform input representations for the models, which allowed me to fairly compare the TF-IDF features with Word2Vec embeddings.

Methodology

In this project, I compared two different methods for representing text: TF-IDF and Word2Vec, using the same machine learning models to keep everything consistent and fair. The setup was intended to focus on how the method of representation impacted the sentiment classification, rather than the classifier itself.

Overview of the Approach

The analysis unfolded in four key stages: data preprocessing, feature representation, model training, and evaluation. Each review was turned into a numerical format using either TF-IDF or Word2Vec before being classified by two different machine learning algorithms. This framework allowed me to attribute any performance differences mainly to the representation technique I used.

TF-IDF Representation

TF-IDF helped create sparse, high-dimensional feature vectors based on the importance of words in the dataset. Each review was tokenized and changed into a vector that reflected both term frequency and inverse document frequency. This technique highlights those rare yet meaningful words that often convey sentiment. The end result was a TF-IDF matrix with around 10,000 features, showcasing the vocabulary derived from our training set.

Word2Vec Representation

For the transfer learning approach, I tapped into pretrained Word2Vec embeddings from the Google News corpus. Each word in a review was matched with its 300-dimensional vector, wherever possible. Any words that didn't show up in the pretrained vocabulary were left out. To summarize each review, I averaged all available word vectors. This method gives us a broad sense of semantic content, though it doesn't maintain the word order.

Machine Learning Models

I applied two classification algorithms to both types of representation. Logistic Regression was selected because it works well with high-dimensional text data and serves as a reliable baseline. Additionally, I added Random Forest for comparison, as it represents a non-linear, tree-based method. Using both models helped me avoid depending on any single classifier.

Training Procedure

All models were trained on the same set of 25,000 reviews and evaluated on another set of 25,000 reviews. I kept hyperparameter tuning to a minimum, sticking to standard defaults. This

decision was intentional because the focus of the project was not about optimizing each classifier but rather comparing the strengths of TF-IDF and Word2Vec under the same conditions. I also noted training times to gauge computational efficiency.

Evaluation Metrics

To gauge model performance, I looked at accuracy, precision, recall, and F1-score. I also made additional comparisons using confusion matrices, a multi-metric heatmap, and analyzed training times. To check if the differences between TF-IDF and Word2Vec were statistically significant, I used McNemar's test on the predictions from the best-performing models.

Results

I assessed the performance of the models based on accuracy, precision, recall, F1-score, confusion matrices, and training time. I tested four combinations of text representation and classifiers: TF-IDF with Logistic Regression, TF-IDF with Random Forest, Word2Vec with Logistic Regression, and Word2Vec with Random Forest. The results highlighted distinct performance differences between the representation methods, with TF-IDF consistently outperforming Word2Vec across all metrics.

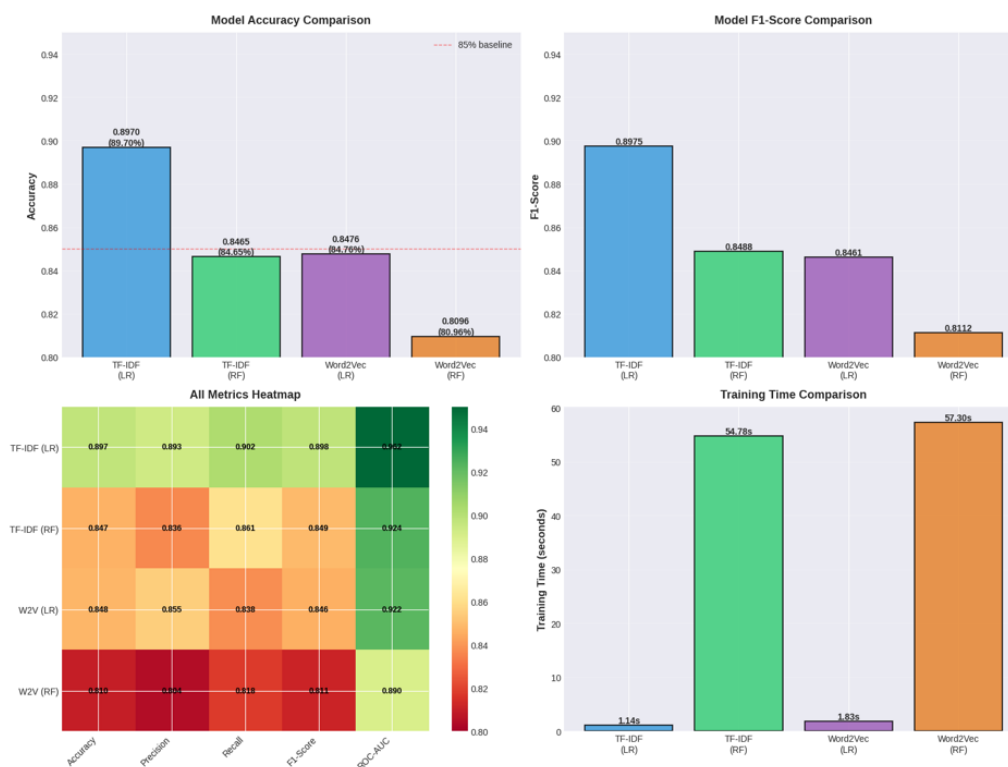
Accuracy and F1-Score

The top-performing model was TF-IDF paired with Logistic Regression, achieving an accuracy of 89.70 percent and an F1-score of 0.8975. This model showed robust performance for both positive and negative classes. The TF-IDF with Random Forest managed an accuracy of 84.65 percent, which was a significant drop compared to the Logistic Regression version.

On the other hand, the Word2Vec models didn't perform as well. Word2Vec with Logistic Regression reached an accuracy of 84.76 percent and an F1-score of 0.8476. The Word2Vec with Random Forest model had the lowest performance, with an accuracy of 80.96 percent and an F1-score of 0.8096. These results suggest that, at least for this dataset, the domain-specific features from TF-IDF offered stronger predictive power than the pretrained embeddings.

Comparative Visualizations

A set of four visual comparisons was used to summarize model behavior:



The first plot showed how accurate each of the four models was by putting them side by side. It clearly separated the TF-IDF models from the Word2Vec ones, with the best combo being TF-IDF paired with Logistic Regression.

In the second plot, I looked at the F1-scores, which mirrored the earlier findings. The TF-IDF Logistic Regression model really stood out with the highest score, while the Word2Vec models were less consistent across different sentiment categories.

Next up was the third visualization, which was a heatmap displaying various metrics for each model. This made it easy to compare accuracy, precision, recall, and F1-score. Again, the TF-IDF Logistic Regression model was the strongest across nearly all the metrics, whereas the Word2Vec Random Forest model didn't perform as well.

The fourth plot focused on training times. The TF-IDF with Logistic Regression trained the quickest, clocking in at about 1.14 seconds. In comparison, both Random Forest models took way longer, with the Word2Vec and Random Forest exceeding 57 seconds. These results clearly show that TF-IDF models not only did better but were also faster to train.

Confusion Matrix Insights

I generated confusion matrices for each model to take a closer look at how they classified. The TF-IDF Logistic Regression model had the highest number of true positives and true negatives, meaning it correctly identified the sentiment of most reviews. On the flip side, the Word2Vec Random Forest model had more false positives and false negatives. This unequal distribution in misclassifications highlights why the embedding-based approaches didn't perform as well.

Overall Takeaway

Looking at all four models, the TF-IDF representations delivered the best results in terms of predictive accuracy and efficiency. The combination of Logistic Regression with TF-IDF consistently ranked at the top for all the evaluation metrics. This suggests that figuring out the importance of specific words directly from the IMDB dataset worked better than relying on pre-trained semantic embeddings from another source.

Statistical Significance Testing

Even though the TF-IDF models showed better accuracy and F1-scores compared to the Word2Vec models, I needed to find out if those differences were real improvements or just random chance. To do this, I used McNemar's test on the predictions from our two top models: TF-IDF with Logistic Regression and Word2Vec with Logistic Regression. I picked these models since they represent the best versions of their respective feature representation methods.

For the test, I set up a contingency table with four categories. These included the number of reviews both models got right, the number they both got wrong, the reviews that only the TF-IDF model classified correctly, and those that only the Word2Vec model got right. Out of 25,000 test reviews, TF-IDF correctly classified 2,128 that Word2Vec misclassified, while Word2Vec managed to correctly classify 893 that TF-IDF got wrong. The rest fell into the categories where both models were correct or incorrect.

The McNemar statistic I calculated from this table came out to 504.06, and I got a p-value lower than 0.0001. This strongly suggests that the performance difference between the two models isn't just a coincidence. It points to the TF-IDF model being significantly more accurate than

Word2Vec. By using a significance test, we further back up the conclusion that TF-IDF was the better choice for this sentiment classification task, reinforcing what we found earlier.

Error Analysis

To get a deeper grasp of how the models performed beyond just overall accuracy, I took a closer look at the errors in predictions from all four models. This analysis was aimed at pinpointing where the misclassifications happened, checking out cases where the models disagreed, and looking into the types of reviews that seemed to trip up both methods. The main goal wasn't just to explain the numerical performance differences but also to shed light on why TF-IDF did better than Word2Vec with this particular dataset.

Patterns of Agreement and Disagreement

I compared the two strongest models, TF-IDF with Logistic Regression and Word2Vec with Logistic Regression. Of the 25,000 test reviews, both models successfully predicted 20,297 reviews. They each made mistakes on 1,682 reviews. The most informative insights came from where they disagreed. TF-IDF accurately classified 2,128 reviews that Word2Vec got wrong, while Word2Vec nailed 893 reviews that TF-IDF misclassified. This trend reinforces the consistent edge that TF-IDF had, aligning with what was suggested by the statistical significance test.

Cases Where TF-IDF Succeeded and Word2Vec Failed

Digging deeper into the reviews that TF-IDF got right but Word2Vec didn't showed some clear patterns. A lot of these reviews included strong sentiment words like 'atrocious,' 'waste of

time,' or 'so fake.' Since these words pop up often in movie reviews, TF-IDF can give them the significance they deserve. On the flip side, Word2Vec averages the meaning of words throughout the entire review, which means that strong sentiment signals can get lost in a sea of neutral words.

The confidence scores also reflected this trend. In many of the disagreements, the TF-IDF model produced moderate to high confidence scores, usually between 0.65 and 0.70, while Word2Vec's scores were often around 0.50, indicating more uncertainty. This suggests that TF-IDF had a clearer sense of the decision-making space it was navigating through.

Cases Where Word2Vec Succeeded and TF-IDF Failed

Even though TF-IDF generally did better than Word2Vec, there were moments when Word2Vec got it right while TF-IDF missed the mark. These instances often involved more subtle expressions of sentiment or reviews that had synonyms or descriptive narratives. Word2Vec sometimes picked up on words like 'flawless,' 'engaging,' or 'uneven' being semantically related to other sentiment markers. In contrast, TF-IDF treats each word separately and can't leverage those connections.

Word2Vec also tended to work a bit better on reviews that didn't lean on specific sentiment words but instead conveyed tone through broader context. This shows how distributed representations can capture meaning beyond individual terms.

Challenging Reviews for Both Models

There were 1,682 reviews that neither model managed to classify accurately. A lot of these featured sarcasm, mixed sentiments, or lengthy narratives that shifted tones. Some reviewers

would praise one part of a movie while critiquing another, making it tough to pin down an overall sentiment. Other reviews included unique writing styles, references to outside media, or personal anecdotes that were only loosely tied to the film. These complexities are typical hurdles in sentiment analysis and underscore the limitations of models that don't take context or discourse structure into account.

Summary of Observations

In the end, the error analysis confirms that TF-IDF worked better for this dataset. It did a good job of picking up on specific sentiment cues and made more reliable predictions. On the other hand, Word2Vec had its advantages in addressing subtle meaning connections, but those weren't enough to overcome the issues caused by averaging word embeddings. Both models had a tough time with reviews that were ambiguous or rich in context, hinting that using more advanced contextual models could lead to better results down the line.

Discussion

The results from this project shed light on how various text representation methods affect the accuracy of sentiment classification. While pretrained embeddings are often thought to be superior to traditional feature engineering, our findings indicate that TF-IDF actually outperformed them for the IMDB movie reviews dataset. This highlights how crucial it is to consider the dataset's characteristics and the method of representation instead of just assuming that newer techniques will always be better than simpler ones.

One key reason TF-IDF did so well is its knack for capturing information specific to the domain. Movie reviews tend to feature strong sentiment words like “boring,” “amazing,”

“terrible,” and “outstanding.” Since TF-IDF gives more weight to words that are particularly informative in its context, it emphasizes the types of signals that are useful for figuring out sentiment. The IMDB dataset is large enough that TF-IDF can pick up solid and reliable patterns without needing extra linguistic information.

On the other hand, Word2Vec embeddings were created using the Google News corpus, which is quite different in nature. The patterns it learned from news articles don’t always mesh well with how people express their opinions about movies. This could mean the embeddings don’t highlight sentiment-related words the same way TF-IDF does. Plus, averaging embeddings for each review takes away details about word order and diminishes the significance of specific sentiment-heavy terms, which likely plays into why the Word2Vec models didn’t perform as well.

The classification results also show some notable differences in how the models read the text. Logistic Regression did particularly well with TF-IDF because those high-dimensional sparse features made it easier to draw clear distinctions. On the flip side, tree-based methods like Random Forest struggled with both representations, probably because they’re not as effective with those high-dimensional sparse vectors and aren’t as well-suited for continuous embedding spaces. This hints that the choice of representation and the classifier can significantly influence outcomes.

Looking at the error analysis brings more clarity to these points. TF-IDF shined in situations where strong sentiment was clearly expressed through key words. Word2Vec only did better when the meaning of a review hinged on more subtle semantic connections or synonyms. Both models had a hard time with reviews that were sarcastic, had mixed feelings, or included long narratives that masked the overall sentiment. These scenarios underscore the limitations of models that don’t take into account the context at the sentence or document level.

Overall, these findings suggest that straightforward, interpretable methods can hold their own for sentiment analysis, especially when working with large, domain-specific datasets. More advanced embeddings might be useful in cross-domain scenarios or with limited training data, but their advantages aren't always guaranteed. This emphasizes the need to assess representation methods within the specific context of the task instead of leaning solely on their theoretical promise or popularity.

Limitations

While the project sheds light on the pros and cons of using TF-IDF and Word2Vec, it's important to point out a few limitations. These limitations come from the dataset, the modeling decisions made, and the overall design of the experiments. Being aware of these factors is key to understanding how widely applicable the results are and can help guide future research.

One limitation relates to the dataset used. The IMDB Movie Reviews dataset only includes binary sentiment labels, which means I could only classify sentiments as either positive or negative. However, real-world sentiment can be a lot more subtle, with plenty of reviews showing mixed feelings or moderate opinions that don't fit neatly into those two categories. Because of this, models that rely on binary data might miss out on those nuances. Plus, since the dataset is limited to movie reviews, it might not hold up when applied to other areas that have different styles or vocabularies, like product reviews, social media, or news comments.

Another limitation comes from how Word2Vec represents features. Averaging embeddings is straightforward and commonly used, but it also means losing important structural details. When we average embedding vectors, we lose information about word order, emphasis, and the syntactic

relationships between words. There are more advanced techniques, like weighted averaging or sequence-based models, that could give better results, but those weren't included in this project.

The models chosen for this study also pose a limitation. I picked Logistic Regression and Random Forest to allow for a fair and interpretable comparison of the different representation methods. However, these models don't really capture the more complex interactions that can occur in text. Using more advanced neural architectures might show different behaviors with Word2Vec embeddings. Also, sticking to default hyperparameters limits the models' performance. This was a deliberate choice to keep things fair among the methods, but it did mean I lost some chances for optimization.

Preprocessing decisions can also play a big role in the results. Deciding whether to keep or remove stopwords can impact how well TF-IDF and Word2Vec work. In this project, I kept stopwords to maintain any sentiment hints, but I didn't explore other preprocessing methods. Likewise, I completely removed words that were not in the vocabulary when using Word2Vec, which can weaken the model, especially when rare or informal words hold sentiment value.

Lastly, I didn't include any contextual or transformer-based representations, which are considered the gold standard in natural language processing right now. Bringing these models into the mix would give a more comprehensive understanding of how traditional techniques stack up against modern deep learning methods.

In summary, while this study clearly shows that TF-IDF outperformed Word2Vec in this specific context, the limitations highlight that the findings should be viewed within the constraints of the dataset and methods I chose.

Conclusion

This project examined how two different methods for representing text, TF-IDF and Word2Vec, performed when it came to figuring out the sentiment of movie reviews. I used the IMDB dataset and created four models by combining each representation method with either Logistic Regression or Random Forest. The outcome revealed that TF-IDF consistently did better than Word2Vec on all the important evaluation metrics, such as accuracy, precision, recall, and F1-score. In fact, TF-IDF paired with Logistic Regression turned out to be the best model overall, achieving strong performance while also being efficient in training.

My analysis indicated that learning features specific to the domain played a crucial role in the success of the TF-IDF models. Because TF-IDF could directly highlight the significance of words tied to sentiment in the movie review dataset, it laid down a solid foundation for linear classification. On the other hand, while Word2Vec can show semantic relationships well, it was hindered by being pretrained and by the averaging method used for creating document representations. These limitations dulled its ability to capture strong sentiment signals in a way that effectively helped with the classification task.

I further backed up these findings with statistical significance tests. McNemar's test showed a noteworthy difference in how the TF-IDF and Word2Vec models disagreed, indicating that TF-IDF was able to classify a significantly higher number of reviews correctly. This confirmed that the performance differences I saw were real improvements thanks to TF-IDF rather than just random chance.

The results also gave me valuable insights into how sentiment data is structured. Reviews that had clear sentiment words were generally easier for TF-IDF models to classify, whereas those that depended on subtle semantic hints were more manageable for Word2Vec. Both methods had a tough time with reviews that included sarcasm, mixed sentiments, or complicated narratives.

In summary, this project shows that straightforward and interpretable methods can still work well for sentiment analysis, especially when there's enough domain-specific data available. Even though pretrained embeddings have their perks in other scenarios, they didn't surpass TF-IDF in this task. Future research could investigate contextual embeddings, multi-class sentiment categories, and cross-domain evaluations to get a deeper insight into how different representation methods impact sentiment classification.

Author Note

The full code, dataset preprocessing steps, and model implementation notebooks are available at:

<https://github.com/Ananyaa-Tanwar/nlp-sentiment-analysis-comparative-study>

Artificial intelligence tools (ChatGPT, Claude) were used to support some portions of the analysis and writing process. This included assistance with resolving coding errors, improving clarity in written sections, and structuring the final report. All analytical decisions, interpretations, and final edits were made by the author.

References

- Sharma, C., Rath, P., Kumar, R., Sharma, S., & Chen, H.-Y. (2025). Mapping the Evolution of Digital Marketing Research Using Natural Language Processing. *Information*, 16(11), 942. <https://doi-org.proxy2.library.illinois.edu/10.3390/info16110942>
- Xiao, L., Li, Q., Ma, Q., Shen, J., Yang, Y., & Li, D. (2024). Text classification algorithm of tourist attractions subcategories with modified TF-IDF and Word2Vec. *PLoS ONE*, 19(10), 1–34. <https://doi-org.proxy2.library.illinois.edu/10.1371/journal.pone.0305095>
- Zhang, L. (2025). Features extraction based on Naive Bayes algorithm and TF-IDF for news classification. *PLoS ONE*, 20(7), 1–17. <https://doi-org.proxy2.library.illinois.edu/10.1371/journal.pone.0327347>