Sentiment Predictions on IMDb Movie Reviews

Chufei Peng, Anna Wang, Sue Tang, Rou Sun, Qinyan Feng

***Abstract*—This project investigates the correlation between textual movie reviews from IMDb and their corresponding binary sentiments (positive or negative). The primary objective is to achieve accurate sentiment prediction based on these textual reviews. The models under consideration encompass Logistic Regression, KNN, LDA, QDA, Decision Tree, and Random Forest. Following comprehensive analysis, our findings highlight that Logistic Regression and LDA exhibit the highest predictive accuracy when tested on the testing dataset.**

***Index Terms*—Natural Language Processing, Data Mining, Binary Classification, Sentiment Analysis, Machine Learning**

I. Introduction

SENTIMENT analysis holds crucial significance in movie reviews as it enables the extraction of underlying emotions and opinions expressed in textual feedback. Sentiment analysis aids in gauging audience reactions and influences decisions in filmmaking, marketing strategies, and audience engagement. IMDb, an extensive online database encompassing information on films, TV series, and video content, serves as a platform that allows audiences to contribute reviews and ratings. These user-generated reviews serve as guiding references for individuals seeking recommendations on movies, shows, and series.

In this project, our focus revolves around analyzing a dataset sourced from IMDb, comprising 50,000 movie reviews[1]. Each review consists of a descriptive paragraph along with a binary sentiment (positive or negative) toward specific movies. Our primary objective involves exploring the correlation between these textual reviews and sentiment. Ultimately, we aim to develop a statistical model that can predict audience sentiment based on the textual content of their reviews with high accuracy.

II. Data Preparation

The IMDb dataset was preprocessed in the following steps.

1) Clean the reviews by removing unnecessary words and adjusting the format of words if necessary.
2) Investigate the descriptive statistics such as the response distribution and word frequency in order to get a better understanding of the dataset.
3) Use words from the pre-defined Lexicon Sentiment Library as predictors for review sentiments.
4) Reduce the number of features and the dimension of our sentiment word lexicon through TF-IDF transformation.
5) Perform a training and testing split on the dataset in a 50:50 ratio.

A. Data Cleaning

We initiated the data cleaning process by removing the unnecessary textual content and lemmatization. We standardized every word into lowercase, removed punctuations and HTML tags, and then tokenized the reviews. Natural Language Toolkit (NLTK)[2] [3]is a powerful Python library that we use to eliminate stopwords, personal pronouns, determiners, coordinating conjunctions, and prepositions. For example, words such as "I", "or", and "which" were removed. For lemmatization, we implemented WordNetLemmatizer, reducing words to their base or root form. Adverbs were transformed into adjectives, and past tense was eliminated. This transformative process mitigates redundancy, exemplified by the conversion of "better" to "good", thereby enhancing the efficiency and interpretability of subsequent analyses.

B. Descriptive Statistics

To better understand our data, we first investigated the descriptive statistics of our IMDb review data.

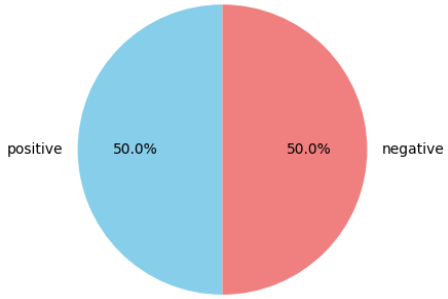

Fig. 1. Class distribution

Through Fig. 1, we can see that our data is perfectly balanced, with exactly half the reviews classified as positive sentiment and the other half classified as negative sentiment. In this case, classifying accuracy will be a fair indicator of the performance of different statistical models, and we will also be able to generalize our results to other balanced movie review data.

In order to get a basic idea of some popular words among these movie reviews, we then constructed word clouds for reviews with a positive sentiment (Fig. 2) and reviews with a negative sentiment (Fig. 3), respectively. Only adjectives were included because they best captured the emotional information associated with the review. We can see that there is a high frequency of words conveying generally positive emotions in the positive sentiment review word cloud. These words include good, great, best, many, and much. There are also more descriptive words such as first, big, beautiful,

wonderful, funny, young, and new. In the negative sentiment review word cloud, there is a high frequency of generally negative words including bad, little, least, old, wrong, awful, and terrible. However, there is also a high frequency of generally positive words in the negative sentiment word cloud. These words coincide with those in the positive sentiment word cloud: good, great, many, much. This may indicate that people might frequently use phrases such as 'not good' and 'not great' to describe their negative sentiments, which can result in a potential problem if we look only at the sentiment words themselves when categorizing without considering the negations that precede those words.

Fig. 2. Positive Adj. Word Cloud

Fig. 3. Negative Adj. Word Cloud

C. Lexicon Sentiment Library

We used a predefined Lexicon-based Sentiment Library as the word dictionary. A lexicon sentiment library is a curated collection of words or phrases associated with sentiment classification (positive or negative) used in natural language processing (NLP) and sentiment classification tasks[4]. We combined all positive and negative words into a large sentiment library of length 6789. Since not all the words included in the sentiment library were useful for our classification objective, the number of words used to predict the sentiment should be reduced. We first eliminated all the words that never appeared in any of the 50000 reviews, which left us 5778 words in the sentiment library.

D. TF-IDF feature selection

To further reduce the dimension of our sentiment features used for classification, we applied a TF-IDF approach. We initialized a TF-IDF vectorizer (TfidfVectorizer()) [5] using our reduced sentiment library (5778 words) as the vocabulary. We then transformed our pre-processed and tokenized review texts into a numeric TF-IDF matrix. The transformed TF-IDF matrix had a shape of 50000 * 5778 where each row corresponded to one piece of movie review and each column corresponded to one unique word in the reduced sentiment library. After getting the TF-IDF matrix, we initialized a feature selector (SelectKBest()) [6] using the chi-squared test as the scoring function. This score measures the dependence between the presence of a word and the class label. A high score indicates that the word is an important feature for predicting the sentiment in this task. We determined the number of features to select to be 300 by trials. In this way, the top 300 features (300 best sentiment words for sentiment prediction) were selected based on their score in the chi-squared test. This TF-IDF feature selection approach left us with 300 sentiment words from the original reduced sentiment library as the best features for sentiment prediction, which implied that the feature dimension is now effectively reduced from 5778 to 300.

E. Training and Testing Split

After preprocessing, we had a 50000 * 300 matrix where each row represented a piece of movie review and each column represented a feature used for sentiment classification. The data is then split into training and testing samples in a 1:1 ratio, where 25000 reviews would be used for model training and 25000 reviews would be used for model testing. In addition, the sentiment was encoded into numeric values, where 1 represented positive and 0 represented negative.

III. Model Implementation

With the processed data, 5 different statistical models were trained and tested to identify the model with the highest prediction accuracy. Our evaluation of the best model contains the following elements:

- Accuracy. It is the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances. Our data is perfectly balanced so accuracy is a strong indicator of the predictive performance of the models. It is our primary indicator for evaluation.
- Precision = TP / (TP + FP). It is the ratio of true positive predictions to the total number of positive predictions made by the model. It measures the accuracy of the positive (and negative) predictions made by the model.
- Recall = TP / (TP + FN). It is the ratio of true positive predictions to the total number of actual positive instances in the dataset. It measures the ability of the model to capture all the relevant instances of a positive class in the dataset.
- F1-score. It is the harmonic mean of precision and recall. It provides a balance between precision and recall. A high F1 score indicates that the model performs well in both correctly identifying positive instances

(precision) and capturing all actual positive instances (recall).
- Area Under the Receiver Operating Characteristic (AUC). It represents the area under the curve when plotting the true positive rate against the false positive rate. The higher AUC value (closer to 1) indicates a better ability of a binary classification model to distinguish between positive and negative instances.

A. Logistic Regression

Logistic regression is a powerful supervised learning model that demonstrates high computational efficiency in binary classification problems, especially with large datasets. It is a parametric and discriminative model. Discriminative classifiers model the posterior $P(Y \mid X)$ directly[7]. Furthermore, we use the Ridge regularization, also known as the L2 regularization, to prevent overfitting by discouraging overly complex models and shrinking less influential coefficients. The significant advantage of logistic regression is its linear feature which is a straightforward and clear interpretation of each variable in the prediction process for large data features.

	Precision	Recall	F1-Score	Support
False	0.8596	0.8165	0.8375	12483
True	0.8257	0.8670	0.8458	12517
Accuracy	-	-	0.8418	25000
Macro Avg	0.8426	0.8417	0.8416	25000
Weighted Avg	0.8426	0.8418	0.8417	25000

TABLE I
LOGISTIC REGRESSION RESULTS

We obtained a relatively high accuracy on our testing dataset for the logistic regression model, which is 0.8418, as shown in Table I. This demonstrates a good ability for this model to make correct predictions. Precision and recall are well-balanced across both classes, which indicates that the model does not overly favor one class over another and avoids biases to some degree. The F1-scores for the two classes are closely aligned with each other. This suggests that precision and recall are balanced with neither being affected by the improvement of the other, so both false positives and false negatives carry significant consequences. The support values indicate an evenly distributed dataset, which helps in creating a model that is not skewed towards a more frequent class. The receiver operating characteristic (ROC) curve further proves the model's credibility, with an area under the ROC curve (AUC) of 0.92 (Fig. 4). This high AUC value indicates excellent discriminatory ability in correctly classifying positive and negative cases.

B. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a discriminative machine learning algorithm for classification and regression. It is a non-parametric model that predicts outcomes by considering the majority class or average of the K-nearest neighbors in the feature space[8]. It has applications in fields like image recognition, recommendation systems, and anomaly detection. It usually outperforms linear regression in complex

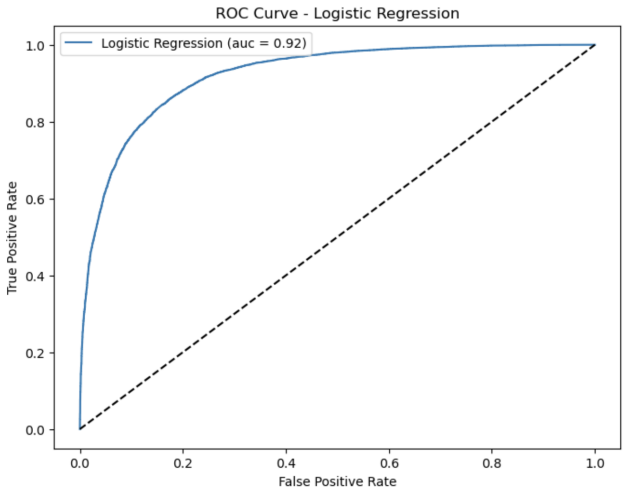

Fig. 4. ROC Curve - Logistic Regression

situations where decision boundaries are not linear. While KNN has no training phase and is easily understood, it has drawbacks like computational inefficiency and sensitivity to noise. In our case, the dimension of our training data is 50000 * 300, which is not very computationally expensive. Therefore, KNN is a relativey suitable model in our case.

Smaller K values result in lower bias but higher variance, whereas larger K values increase bias and reduce variance. Therefore, our team conducted Grid Search Cross-validation to select the best K value to be 25 instead of 5 (the default). The respective accuracy with different K values is shown in Table II.

K Value	Accuracy
1	0.6914
5	0.7452
13	0.7655
17	0.7680
25	0.7722
77	0.7583
93	0.7532

TABLE II
KNN ACCURACY WITH DIFFERENT K

The best result for the KNN model with k = 25 is shown in Table III. The model accuracy is 0.7722. The model achieved an AUC of 0.86 (Fig. 5), signaling that the model effectively distinguishes between the positive and negative classes.

	Precision	Recall	F1-Score	Support
False	0.8470	0.6636	0.7442	12483
True	0.7241	0.8805	0.7947	12517
Accuracy	-	-	0.7722	25000
Macro Avg	0.7856	0.7721	0.7694	25000
Weighted Avg	0.7855	0.7722	0.7695	25000

TABLE III
KNN RESULTS (K=25)

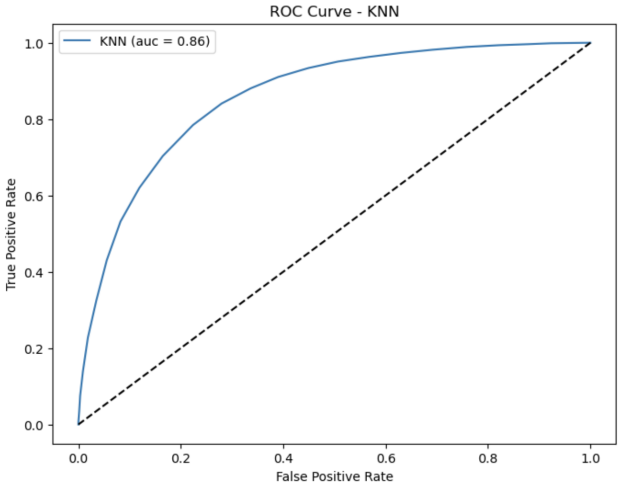

Fig. 5. ROC - KNN (k = 25)

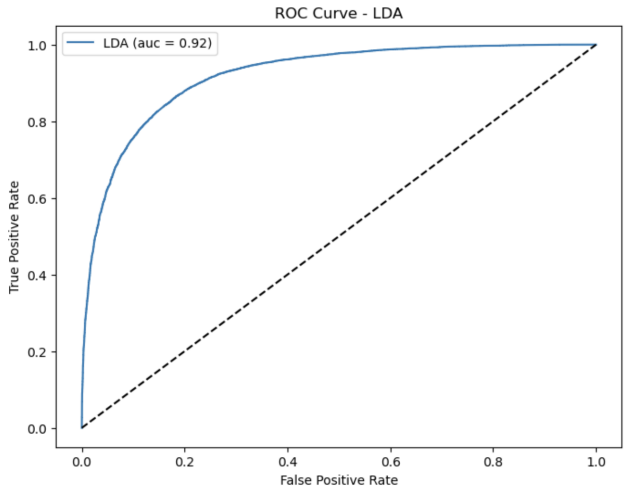

Fig. 6. ROC - LDA

C. Linear Discriminant Analysis and Quadratic Discriminant Analysis

We then transform from discriminative to generative models. Generative models, represented by Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA), analyze the joint distribution of both input features and response variables and learn the underlying probability distribution of the observed data, enabling the generation of new samples resembling the training data[9]. Both LDA and QDA require certain assumptions: the features within each class should follow a multivariate normal distribution, and features should be independent in each class[10]. The difference lies in the covariance matrix. LDA assumes an equal covariance matrix across classes while QDA does not have this limitation. Thus, LDA has a linear decision boundary and QDA has a quadratic decision boundary like ellipses, hyperbolas, and parabolas.

The difference between covariance matrices of the two classes was tested and the result showed a difference of 0.02 in the Frobenius norm. Such a small difference is negligible. Thus, the LDA assumption of equal covariance is satisfied. The LDA model implementation result shows an accuracy of 0.8392 and an AUC of 0.92 (Table IV, Fig. 6). This suggests that the LDA model has a high true positive rate and a low false positive rate across a range of threshold values. The precision, recall, and F1-score exhibit consistency across both classes, reinforcing that the LDA model can make predictions with high accuracy.

	Precision	Recall	F1-Score	Support
False	0.8635	0.8053	0.8334	12483
True	0.8181	0.8731	0.8447	12517
Accuracy	-	-	0.8392	25000
Macro Avg	0.8408	0.8392	0.8390	25000
Weighted Avg	0.8408	0.8392	0.8391	25000

TABLE IV
LDA RESULTS

For the QDA model, the accuracy is 0.7901 and the AUC for the ROC curve is 0.83 (Table V, Fig. 7). These values are lower than those achieved by the LDA model. This discrepancy can be attributed to the fact that QDA, with its assumption of class-specific covariance matrices, introduces extra parameters for estimation and increases the model's flexibility. The increased flexibility raises the risk of overfitting, which explains the relatively low accuracy of the QDA model.

	Precision	Recall	F1-Score	Support
False	0.8393	0.7168	0.7732	12483
True	0.7535	0.8631	0.8046	12517
Accuracy	-	-	0.7901	25000
Macro Avg	0.7964	0.7900	0.7889	25000
Weighted Avg	0.7963	0.7901	0.7889	25000

TABLE V
QDA RESULTS

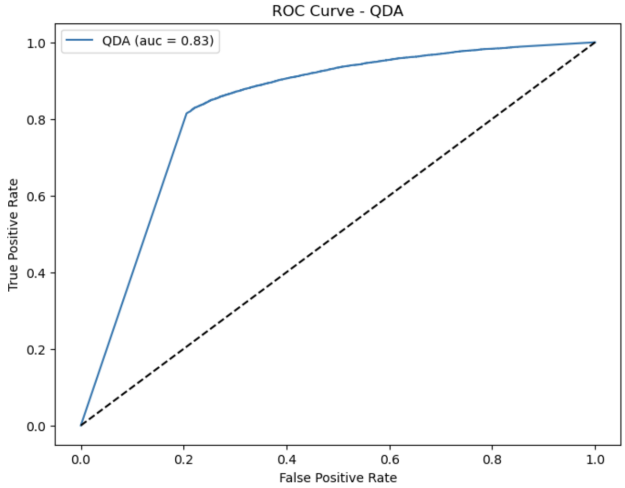

Fig. 7. ROC - QDA

D. Decision Tree and Random Forest

Decision Trees work by recursively splitting the training dataset into subsets based on the decision nodes. It is a discriminative model that focuses on directly learning the decision boundaries that separate different classes directly[11]. Each node in the tree represents a decision based on a feature. Decision Trees can do both classification and regression tasks. Decision Trees are also easy to interpret and can capture complex relationships in the data.

By using Grid Search Cross-validation across 54 different combinations of hyper-parameters, we found that the optimal parameters are criterion= 'entropy', max_depth= None, min_samples_split= 5, and min_samples_leaf= 1. This means that the Decision Tree model excels when utilizing 'entropy' as the criterion for split quality, with no maximum depth, requiring a node to split only if it contains at least 5 samples, and enforcing a minimum of 1 sample per leaf node. As a result of these refined settings, we achieved an accuracy of 0.7322 and an AUC of 0.75, as shown in Table VI and Fig. 8.

	Precision	Recall	F1-Score	Support
False	0.7326	0.7303	0.7314	12483
True	0.7318	0.7341	0.7330	12517
Accuracy	-	-	0.7322	25000
Macro Avg	0.7322	0.7322	0.7322	25000
Weighted Avg	0.7322	0.7322	0.7322	25000

TABLE VI
DECISION TREE RESULTS

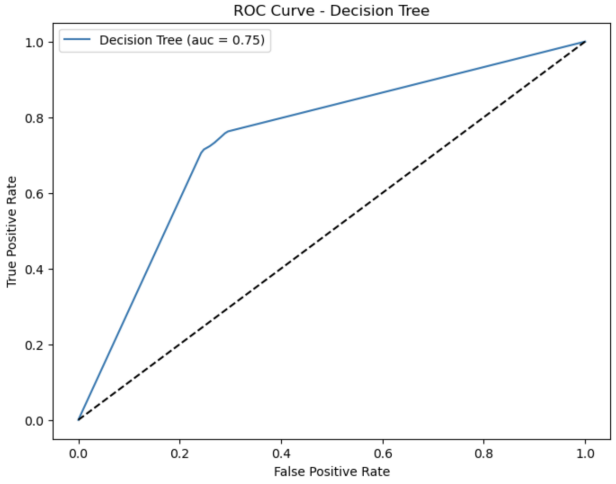

Fig. 8. ROC - Decision Tree

A potential drawback of the Decision Tree model is the overfitting issue because it is sensitive to the small variations in the training dataset, which leads to instability. Thus, we trained an additional model– Random Forest. Random Forest is an ensemble method of multiple Decision Trees and introduces randomness. Each tree in the forest is trained on a random subset of the data and features, introducing diversity. The final prediction is often determined by a majority vote of individual tree predictions. Compared to Decision Trees, the Random Forest model improves generalization and robustness.

By using grid search cross-validation across 54 different combinations of hyper-parameters, we found that the optimal parameters are n_estimators= 200, max_depth= 30, min_samples_split= 10, and min_samples_leaf= 1. This means that the Random Forest model performs best when it consists of 200 decision trees, imposes a maximum depth of 30, requires a node to split only if it contains at least 10 samples, and enforces a minimum of 1 sample per leaf node. With these hyper-parameters, we achieved an accuracy of 0.8052 and an AUC of 0.89. The results are displayed below in Table VII and Fig. 9.

	Precision	Recall	F1-Score	Support
False	0.8378	0.7562	0.7949	12483
True	0.7784	0.8540	0.8145	12517
Accuracy	-	-	0.8052	25000
Macro Avg	0.8081	0.8051	0.8047	25000
Weighted Avg	0.8081	0.8052	0.8047	25000

TABLE VII
RANDOM FOREST RESULTS

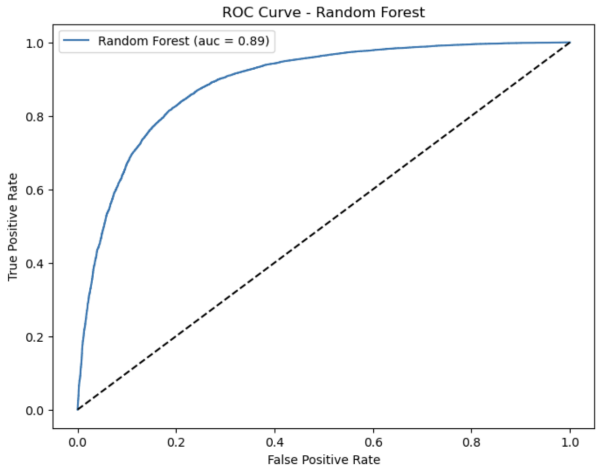

Fig. 9. ROC - Random Forest

IV. DISCUSSION AND CONCLUSION

Our project performed the following steps to build 6 models to predict the sentiments (positive/ negative) of reviews in the IMDb dataset: (1) Through meticulous data preprocessing, irrelevant texts were removed from the original dataset. (2) By feature engineering, we selected the most valuable features and reduced dataset dimensions, which successfully reduced model variances and mitigated the risk of overfitting. (3) To ensure the appropriateness of each model, their assumptions were validated against our dataset. (4) Grid Search Cross-Validation was used to tune the hyperparameters of 3 models (KNN, Decision Tree, and Random Forest), in order to achieve favorable accuracies.

By looking at Table VIII, we can conclude that the models perform well in general with the accuracy higher than 0.7. Specifically, Logistic Regression and LDA model

have impressive accuracy of 0.8418 and 0.8392 respectively. This is probably because our processed data feature a linear trend, which makes these two linear models effective. KNN, QDA, and Random Forest have accuracies of approximately 0.8, which are slightly lower than the accuracies of Logistic Regression and LDA, but are also favorable scores. Decision Tree has the lowest accuracy of 0.7322, likely due to the high noise of the dataset and the assumption that features are independent. This may be improved if the number of features selected in the encoded sentiment dictionary increases, but on the other hand KNN would probably perform worse when the dimension of data is high.

Model	Accuracy
Logistic	0.8418
KNN	0.7722
LDA	0.8392
QDA	0.7901
Decision Tree	0.7322
Random Forest	0.8052

TABLE VIII
MODEL ACCURACY SUMMARY

For future improvement of our project, we shall consider the case of negation. For instance, certain reviews contain phrases like 'not good' or 'not bad.' These expressions are currently processed similarly to 'good' and 'bad' since the sentiment words 'good' and 'bad' are present in both contexts. However, this treatment overlooks the inherent negative sentiment associated with 'not good' in reality, potentially affecting the accuracy of our predictions.

In conclusion, within our IMDb movie review dataset project, Logistic Regression and LDA models predict the sentiment with the highest accuracy on the testing dataset. The further implications of this project extend toward potential applications in enhancing audience engagement and informing marketing strategies within the film industry.

V. ACKNOWLEDGMENTS

We are sincerely grateful to the open-source community and online resources that provided important information for this project. Special acknowledgment is to our dedicated team members, whose tireless efforts and unique expertise have greatly enriched our project. In addition, we would like to extend our special thanks to Professor Shirong Xu for his invaluable guidance. Thank you to all those who have contributed to the success of this project.

APPENDIX A
SOURCE CODE

A full repository of all code used can be found at:
https://github.com/sqtang37/IMDb-Movie-Review-Sentiment-Analysis.

REFERENCES

[1] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. [Online]. Available: http://www.aclweb.org/anthology/P11-1015

[2] T. Hauck, *scikit-learn Cookbook*. Packt Publishing, 2014.

[3] "MS Windows NT nltk," https://www.nltk.org/, accessed: 2023-11-30.

[4] B. Liu, "Opinion lexicon," http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#datasets, 2011, copyright (C) 2011 Bing Liu, Licensed under Creative Commons Attribution 4.0 International (http://creativecommons.org/licenses/by/4.0/).

[5] "MS Windows NT kernel description," https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html, accessed: 2023-12-13.

[6] "MS Windows NT kernel description," https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html, accessed: 2023-12-13.

[7] M. P. LaValley, "Logistic regression," *Circulation*, vol. 117, no. 18, pp. 2395–2399, 2008.

[8] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "Knn model-based approach in classification," in *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings*. Springer, 2003, pp. 986–996.

[9] A. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in neural information processing systems*, vol. 14, 2001.

[10] "MS Windows NT kernel description," https://scikit-learn.org/stable/modules/lda_qda.html, accessed: 2023-12-4.

[11] Y.-Y. Song and L. Ying, "Decision tree methods: applications for classification and prediction," *Shanghai archives of psychiatry*, vol. 27, no. 2, p. 130, 2015.