Sentiment Analysis

Mozhdeh Hajiani
Politecnico di Torino
Student id: s330007
mozhdeh.hajiani@studenti.polito.it

***Abstract*—This study applies machine learning algorithms to classify the sentiment of each text instances. An exploratory supervised data analysis was conducted, where we train models on labeled data. Various feature extractions and classification models were trained and evaluated. The best results were achieved with an Logistic Regression Classification using TF-IDF on a cleaned dataset.**

I. Problem overview

Internet data is increasing rapidly as people share opinions on social media, particularly Twitter. Twitter is a popular site where users interact by creating ”tweets” to express feelings. It is the largest micro-blogging platform, with 330 million active users posting 500 million tweets daily [1]. Analyzing tweets is crucial for businesses to develop marketing strategies. Sentiment analysis classifies emotions in text, helping businesses make informed decisions based on user sentiments.

Sentiment analysis uses a classification approach with a predictive algorithm to identify the sentiment of tweets by learning patterns from training data. The model learns patterns as weight vectors to classify future tweets' sentiment.

This study examines a set of training data that includes tweets in tabular format with six columns: 'ids', 'date', 'flag', 'user', 'text', and 'sentiment'. The 'sentiment' column indicates whether a tweet is positive or negative. The evaluation set serves as test data with five columns: 'ids', 'date', 'flag', 'user', and 'text'. The dataset includes tweets, of which 42.1% are negative and 57.9% are positive. .

In this study, we used various classification algorithms such as random forest, extra tree classifier(ETC), gradient boosting machine(GBM), logistic regression(LR), Naive Bayes(NB), Decision Tree(DT), stochastic gradient descent(SGD), voting classifiers, and Support Vector Machine. Different feature engineering approaches like TF-IDF, TF + TF-IDF, and bag of words were also employed. The best performing classifier was selected for sentiment analysis based on accuracy, precision, recall, and F1-score comparisons.

Classification involves two phases: training the model with data to learn patterns and testing the model with data to evaluate measures.

The Exploratory Data Analysis phase consists of the following operations:

- *text preprocessing:* The first step in sentiment classification is text preprocessing, which will clean up unstructured data. Preprocessing includes tasks like tokenization, removing stop words, converting to lowercase, stemming, and eliminating numbers.
- *feature extraction:* There are different types of text features such as count vectors, bag of words, TF-IDF, word embeddings, NLP based.
- *machine learning algorithms :* Lastly, machine learning algorithms such as support vector machines, decision trees, naïve Bayes, and artificial neural networks are applied for the classification process.

II. Proposed approach

A. Preprocessing

In many cases, the raw data is typically noisy, so it is important to handle it carefully. The data from tweets includes URLs, hashtags, user mentions, brackets, HTML tags, and emojis. The preprocessing phase consists of combinations of these steps:

1. Normalization: Normalization requires various tasks to be carried out simultaneously. These tasks involve converting all text to either uppercase or lowercase, removing punctuation, and changing numbers to their word equivalents. These steps help standardize the preprocessing of each text.

   • First of all, we need to remove the Html tags.

   •Removing punctuations [’”?!,. ():;] and special characters like ‘-‘, ‘:’, ‘,’ from the word.

   • Repeated characters were condensed to 2 characters. Some individuals tend to write tweets such as ”It’s sooo goooood,” utilizing extra characters to stress particular words.

   • Accept only those words which are made up of English letters and are not alpha- numeric.

   • Converting all the word to lowercase.

   • Stopwords only increase the data size, so better to remove them too.
2. URLs: Users frequently include URLs in their tweets, but these URLs do not contribute to sentiment analysis. Therefore, we eliminate them by using regular expressions to identify and remove URLs using the Python library re.
3. Mentions: the user mention other users in their tweets which does not help to decide the sentiment, so we can remove them.
4. Tokenization: In this step, text is divided into tokens. For instance, the sentence ”it is a hard job to do” becomes tokens: {’it’, ‘?’, ’is’, ’hard’, ’job’, ’to’, ’do’}.
5. Eliminating common words like ”are”, ”of”, ”the”, and ”at” is crucial in sentiment analysis as they do not add

value. These words are known as stop words and should be excluded from the analysis.

6. Stemming: The stemming process simplifies words by converting them to their base form, reducing unnecessary calculations. For instance, changing ”fishing, fish, fisher” to ”fish” and ”argue, arguing, argues” to ”argue.”
7. Lemmatization: Lemmatization merges multiple words into one. It analyzes word structure and removes word endings.

After preprocessing the tweets, it is necessary to transform the text into a vector format, which is commonly referred to as word embedding.

1. TF-IDF: The term frequency-inverse document frequency (also called TF-IDF), is a well-recognized method to evaluate the importance of a word in a document. Tf Idf [2] is calculated as the product of tf and idf.
   The Term Frequency (TF) formula (1) is given by:

   $$TF(t,d) = \frac{f_{t,d}}{N_d} \tag{1}$$

   where:
   - $TF(t,d)$ = term frequency of term t in document d
   - $f_{t,d}$ =number of times term t appears in document d
   - N_d = total number of terms in document d

   IDF is used to determine the importance of a term in a document. Term frequency treats all terms equally, while IDF gives more importance to less common words and less importance to common words. Common words like ”is”, ”an”, and ”and” are frequent but not important.
   The Inverse Document Frequency (IDF) formula (2) is given by:

   $$IDF(t) = \log\left(\frac{N}{|\{d : t \in d\}|}\right) \tag{2}$$

   where:
   - $IDF(t)$ = inverse document frequency of term t
   - N = total number of documents in the corpus
   - $|\{d : t \in d\}|$ = number of documents that contain the term t
2. Bag of Words: A bag of words is a way to represent a document, such as a tweet, by counting the occurrence of each word in the document. The BOW approach does, however, have certain limits because tweets may only contain a few characters, which decreases their effectiveness. The accuracy of the BOW-based technique is limited by the lack of adequate word occurrences in textual comments or tweets [3].

B. *Model Selection*

Based on dataset characteristics, we used various machine learning models like regression-based models, probability-based models, and ensemble learning classifiers to improve predictions by reducing variance and bias. We assessed different machine learning classifiers in Python using Scikit learn. The machine-learning models used in the study include:

1. Logistic Regression:
   LR [4] is used for classification problems and is based on a probability model. The logistic function represents binary data. LR uses a sigmoid function to show the relationship between dependent and independent variables. A correlation coefficient is used in LR to measure the connection between the target and independent variables. This coefficient ranges from -1 to 1 and indicates how well expected and actual values align. In LR, the model is linear with Y as the target and X as the independent variable, written as Y = a + bX.
2. Support Vector Machine (SVM) is a classification algorithm used in supervised machine learning for separating positive and negative data points using a hyperplane called the decision boundary. Points on the decision boundary are called support vectors.
   The SVM main purpose is to find a hyperplane in an N-dimensional space to separate positive and negative points. This method works well for regression and classification tasks by creating a hyperplane to distinguish classes. It performs best when the number of dimensions exceeds the number of samples, but struggles with large datasets. Cross-validation is used in SVM to boost computational efficiency.
   We assigned a value of 0.01 to the C term, which is the penalty parameter for the error term in SVM. This parameter affects misclassifications in the objective function. We conducted SVM analysis using Unigram + Bigram features.
3. The Naïve Bayes classifier[5] is a supervised learning method for categorizing text. It is effective and based on probabilities. The algorithm works well even with a large amount of data. It operates using Bayes theorem and probabilities to categorize information. Baye’s theorem is used to predict the class with the highest probability in Naïve Bayes algorithm, which is fast and scalable for Multiclass and Binary Classification. Multinomial Naive Bayes was used with a smoothing parameter of 2 for our analysis.
4. Stochastic Gradient Descent classifier [6] is a machine learning method that finds the best parameter to connect predicted and actual outputs. It is smooth and optimizes the objective function. It learns from large datasets faster than gradient descent and converges quickly by creating batches to determine gradients.
5. Voting Classifiers: A voting classifier is a combination of models that give predictions based on majority votes[7]. It combines the benefits of LR and SGD in the VC(LR+SGD) used in this study.
6. Random Forest (RF) [8] is a method that uses multiple decision trees to avoid overfitting and improve performance. RF trains decision tree classifiers on various

data samples and combines their outputs. This involves creating many decision trees that each make class or regression predictions. RF is popular in machine learning for its simplicity and ability to generate good results without hyperparameter tuning. We decided to parallelize the process to decrease the high time complexity of the model by using multiple CPU cores at the same time to finish the task faster.

7. XGBoost is a type of gradient boosting algorithm that creates a prediction model using weak decision trees. It is important to adjust the number of estimators used in XGBoost to optimize performance. A value of 500 estimators yielded the best results in our study.
8. Decision Tree are a non-parametric supervised learning method used for classification and regression. "GINI evaluates the split at each node and selects the best split. The model had better performance with bow than tfidf. Additionally, using unigrams alone or with bigrams did not lead to significant improvements."

C. Hyperparameters tuning

In our research, GridSearchCV helped us find the best hyperparameters for our sentiment analysis model, enhancing performance and generalization to new data. Cross-validation and exhaustive search led to reliable outcomes in our analysis. By leveraging the power of cross-validation along with exhaustive search, we were able to achieve reliable and reproducible outcomes in our analysis.

III. Results

In this work, The train_test_split method from sklearn.model_selection is used to split the dataset into train and test sets randomly. 80 percent of data is used for training and 20 percent for testing. The dataset for testing is new and not used for training before. Two features TF-IDF (word level) with N-Grams (n=2 selected by cross validation) and bag of word vectorizer are used for feature extraction. Bigrams are used to capture relationships and meanings between words.

Table I displays the results of classification techniques using TF-IDF. Moreover, we investigate the performance of model on the combination of two features, TableII shows the related output.

In certain models(GBM,ETC,XgBoost), the max feature (value=4096) is used to manage the number of features in order to decrease the computational expenses, and improve speed for both training and inference processes.

During training, we examined different models but found them unsuitable. The CNN model was excluded due to high time complexity causing crashes. Therefore, we removed these models from our experiment.

From the results, we observed that the classifiers' accuracy generally improved for SGD from 73.50 to 77.94 when BOW + TFIDF is used. Other classifiers either remained similar or showed slight differences. In many cases, recall improved for class 0 when using BOW + TFIDF, indicating better identification of positive instances of that class. However, precision did not always improve; for some classifiers, it decreased, indicating that the model may have had more false positives. The shifting precision and recall values across different classifiers indicate that they are responding to the imbalanced class situation. Models that increase their recall for class 0 (the minority class) might do so by over-predicting it, thereby hurting precision.

When dealing with an imbalanced dataset, it is crucial to upsample the minority class to achieve balance for better learning outcomes. Addressing class imbalance is essential in classification tasks to prevent biased models that favor the majority class, resulting in poor performance for the minority class. A balanced distribution of classes enables the model to learn effectively from both classes, enhancing overall performance and resilience. In imbalanced datasets, accuracy can be deceiving, so it is important to evaluate other metrics like precision, recall, and F1 score. In our imbalanced dataset, we upsampled the minority class to create a more balanced dataset for effective learning by the model. The results are displayed in Table III.

The result shows that SVM-TFIDF model has the highest accuracy (87.35%) and F1-score (0.88) for class 0, indicating it performs well at classifying both positive and negative cases. In the Logistic Regression models (both BOW and TF-IDF) show strong metrics with high precision and recall across both classes.The SGD-TFIDF model records the lowest accuracy (75.88%) and lowest F1-score (0.75) for class 0, indicating that this model might need further tuning or data handling. Naïve Bayes-BOW shows the highest recall for class 0 (0.91), suggesting it is effective at identifying actual negatives.

we can easily say that tfidf featurization technique in general were recorded higher accuracy than bow. Moreover, we found out that logistic regression gave best predictions of sentiments by giving maximum output for all four comparison parameters namely – accuracy, recall, precision, and f-score and for both feature extraction methods namely – N-Gram and word-level TF-IDF. Thus Logistic regression is the best algorithm for sentiment analysis and both feature extraction techniques are good enough.

IV. Discussion

In this study, we have developed a data science pipeline to predict the sentiment. We ruled out several machine learning model due to their high time complexity like CNN, LSTM.

1. There are other models and techniques that could be explored in future work. These include but are not limited to LSTM, Neural Networks methods

2. There might be room to optimize the models used in the research by adjusting parameters or using different techniques for normalization.

3. comparison of other features like word polarity score features, word embeddings, twitter specific features etc

4. Using Bidirectional Encoder Representation from Transformers model in order to achieve best performancec or stack-

ing best models on top of one another to form an ensemble model which could get us a slightly higher accuracy, but it will have a high time complexity.

References

[1] Omnicoregency.com, “Twitter by the Numbers: Stats, Demographics & Fun Facts,” Feb 10, 2020. Available: https://www.omnicoreagency.com/twitterstatistics/. Accessed: Apr 4, 2020.

[2] P. Bafna, D. Pramod, and A. Vaidya, “Document clustering: TF-IDF approach,” in *IEEE Int. Conf. on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pp. 61–66, 2016.

[3] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas, “Short text classification in Twitter to improve information filtering,” in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Geneva, Switzerland, 19–23 July 2010, ACM, New York, NY, USA, pp. 841–842.

[4] A. Genkin, D.D. Lewis, and D. Madigan, “Large-scale Bayesian logistic regression for text categorization,” *Technometrics*, vol. 49, no. 3, pp. 291–304, 2007. [CrossRef]

[5] A. Perez, P. Larranaga, and I. Inza, “Supervised classification with conditional Gaussian networks: Increasing the structure complexity from naive Bayes,” *Int. J. Approx. Reason.*, vol. 43, pp. 1–25, 2006. [CrossRef]

[6] W.A. Gardner, “Learning characteristics of stochastic-gradient-descent algorithms: A general study, analysis, and critique,” *Signal Process.*, vol. 6, pp. 113–133, 1984. [CrossRef]

[7] J.L. Anderson, “A method for producing and evaluating probabilistic forecasts from ensemble model integrations,” *J. Clim.*, vol. 9, pp. 1518–1530, 1996. [CrossRef]

[8] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, pp. 5–32, 2001. [CrossRef]

TABLE I: Performance Metrics TF-IDF of Classifiers

Classifier	Class	Accuracy	Precision	Recall	F1-score	F1-macro
SVM	0	78.16	0.78	0.68	0.72	0.77
SVM	1	78.16	0.78	0.86	0.82	0.77
SGD	0	73.50	0.83	0.47	0.60	0.70
SGD	1	73.50	0.70	0.93	0.80	0.70
Naïve Bayes	0	76.63	0.85	0.51	0.64	0.75
Naïve Bayes	1	76.63	0.72	0.93	0.81	0.75
Logistic Regression	0	78.73	0.79	0.69	0.73	0.78
Logistic Regression	1	78.73	0.79	0.86	0.82	0.78
GBM (max-feature=4096)	0	75.90	0.80	0.38	0.51	0.72
GBM (max-feature=4096)	1	75.90	0.67	0.93	0.78	0.72
ETC (max-feature=2048)	0	76.47	0.75	0.68	0.71	0.76
ETC (max-feature=2048)	1	76.47	0.78	0.83	0.80	0.76
DT	0	71.63	0.78	0.64	0.66	0.71
DT	1	71.63	0.67	0.77	0.76	0.71
RF	0	76.63	0.74	0.67	0.71	0.76
RF	1	76.90	0.77	0.85	0.81	0.76
Voting Classifier (LR + SGD)	0	78.64	0.80	0.61	0.69	0.78
Voting Classifier (LR + SGD)	1	78.64	0.76	0.89	0.82	0.78
XGBoost (max-feature=2048)	0	77.21	0.76	0.67	0.71	0.76
XGBoost (max-feature=2048)	1	77.21	0.78	0.85	0.81	0.76

TABLE II: Performance Metrics of BOW + TFIDF Classifiers

Classifier	Class	Accuracy	Precision	Recall	F1-score	F1-macro
SVM	0	78.24	0.74	0.75	0.75	0.78
SVM	1	78.24	0.81	0.81	0.81	0.78
SGD	0	77.94	0.79	0.66	0.72	0.77
SGD	1	77.94	0.77	0.87	0.82	0.77
Naïve Bayes	0	76.81	0.72	0.74	0.73	0.77
Naïve Bayes	1	76.81	0.80	0.79	0.80	0.77
Logistic Regression	0	78.06	0.78	0.77	0.77	0.77
Logistic Regression	1	78.06	0.78	0.86	0.82	0.77

TABLE III: Classifier Performance Summary Using Minority Class Upsampling

Classifier	Class	Accuracy	Precision	Recall	F1-score	F1-macro
SVM-BOW	0	86.74	0.85	0.89	0.87	0.87
SVM-BOW	1	86.74	0.88	0.84	0.86	0.87
SGD-BOW	0	80.60	0.81	0.79	0.80	0.81
SGD-BOW	1	80.60	0.80	0.82	0.81	0.81
Naïve Bayes-BOW	0	84.07	0.80	0.91	0.85	0.84
Naïve Bayes-BOW	1	84.07	0.89	0.77	0.83	0.84
Logistic Regression-BOW	0	86.86	0.85	0.90	0.87	0.87
Logistic Regression-BOW	1	86.86	0.89	0.84	0.86	0.87
SVM-TFIDF	0	87.35	0.86	0.89	0.88	0.87
SVM-TFIDF	1	87.35	0.89	0.86	0.87	0.87
SGD-TFIDF	0	75.88	0.77	0.74	0.75	0.76
SGD-TFIDF	1	75.88	0.75	0.78	0.76	0.76
Naïve Bayes-TFIDF	0	83.66	0.80	0.91	0.85	0.84
Naïve Bayes-TFIDF	1	83.66	0.89	0.77	0.82	0.84
Logistic Regression-TFIDF	0	87.19	0.86	0.89	0.87	0.87
Logistic Regression-TFIDF	1	87.19	0.89	0.85	0.87	0.87