Machine learning for sentiment analysis of tweets

Machnine Learnig(DV2599)

1st Jakob Adamsson
dept. of computer science
Blekinge Tekniska Högskola
Karlskrona, Sweden
jaad19@student.bth.se

I. Introduction

In today's digital age, it can be difficult to accurately determine the sentiment of a piece of text, whether it be a social media post, a news article, customer feedback, or a product review. This task is further complicated by the abundance of conflicting information and opinions available. In this paper, the focus is on the application of sentiment analysis to tweets, which are a common platform for expressing opinions and emotions. However, the techniques and models developed in this paper could potentially be applied to other areas as well, such as medical diagnoses or customer feedback analysis. The main focus in this paper and the implementation is on the cleaning of the data. Therefore the question that shall be answered throughout this paper is the following:

What is the impact of using machine learning algorithms for sentiment analysis of tweets on the accuracy and reliability of sentiment predictions compared to traditional methods?

II. Method

A. Datacleaning

When the process of cleaning the data began, the data was first analyzed and important information or unusual features were noted. Then, irrelevant features were removed, in this case, Location and Keyword were removed as they did not add important information about the dataset. The columns that were left after this were Text and Target, where Text contained the tweet itself and Target was set to either 1 or 0 depending on whether the tweet was positive or not. Then the tweets were tokenized, stemmed and stopwords were removed. The steps for fully cleaning the data were:

- Analyze the data to identify and remove any outliers, as well as any useless columns.
- Check for a significant amount of Unicode characters, such as hashtags, square brackets, newlines, weird returns, and digits within words, and replace them with an empty space if necessary.
- Remove stop words, which do not typically carry much weight in natural language processing, using a library designed for this purpose.
- Apply a stemming algorithm to the dataset to replace words that have the same meaning with a more general word. For example, "am," "are," and "is" would be replaced with "be."
- Tokenize the tweets to allow the models to effectively analyze sequences of words and identify patterns.
- Remove words that occur less than 30% of the time in the dataset.

1) Stop words: In natural language processing stop words are words that are pretty frequent in sentences, however they does not necessarily carry a lot of meaning to the overall picture. For instance a sentence like it is cloudy, can simply be replaced by cloudy since it is does not really matter for the machine learning model anyways. With this in mind, the decision was made to remove them from the dataset and it was done by using a Python library called nlkt(*https://www.nltk.org/*, n.d.) which provides tools for working with human language data.(Albon, 2018)

To eliminate stop words from the tweets, a function was created that checks each word in the tweets against a list of English stop words. If a word in the tweet matches a stop word, it is replaced with a blank space. This function was designed to remove all stop words from the tweets. By replacing stop words with blank spaces, the tweets are cleaned and ready to be processed by machine learning algorithms and was in a much better form for tokenization and stemming. (*https://www.nltk.org/*, n.d.)

2) Tokenize: Tokenization in this paper was used in a way where text was divided into smaller parts, these parts are called tokens and are basically a vector with numbers representing each word. Tokens can have various shapes and forms and can for instance be words, punctuation marks, digits and so on. In this paper where tweets were analyzed, tokens might include text such as punctuation marks, commas or periods. The tokenization of the tweets took place after the whole data set were cleaned in order to achieve best performance.
Using tokens in NLP makes it easier to process and analyze text, as it allows the text to be divided into smaller, more manageable pieces. In this particular implementation, tokenization was used in conjunction with the removal of

stop words and other odd characters from the tweets, which made it natural to also implement tokenization. Overall, tokenization helps to improve the efficiency and effectiveness of NLP models by allowing them to focus on specific parts of the text.(Burchfiel, 2022) (Albon, 2018)

3) Stemming: After the tweets were tokenized, the process of stemming was applied. Stemming involves reducing words to their base or root form, for example, converting the words ”tradition” and ”traditional” to the common root ”tradit.” This helps the computer recognize patterns more easily by reducing the number of unique words it encounters. Stemming is useful because it allows the computer to more easily identify the core meaning of a word, even if it appears in different forms. (Albon, 2018)

Since tweets are being analyzed in this implementation, the assumption is that most people expressing themselves online write similarly, so this method was considered appropriate to apply to the data.

B. Support Vector Machine

Support vector machines uses supervised learning and is typically used for classification and regression tasks. In this paper and implementation, it was used for a classification task where the objective was to correctly identify tweets that are positive and negative, basically a sentiment analysis of tweets. (Albon, 2018)

Support vector machine, uses quite a lot of math in order to compute a result and it works by finding a hyperplane in an N-dimensional space where N in this case represents the amount of features in the data set. The goal of finding a hyperplane is to find a hyperplane that maximally separates different classes in the data meaning the margin between the data points divided by the plane shall be the same. In the process of finding the optimal hyperplane, the data points are treated like vectors and this is why it is called support vector machine, the data points stored in the vector closest to the plane are called support vectors. (*Support Vector Machine*, 2022)

The model used for support vector machine comes from a Python module called ”Sci kit learn”. (*1.4. Support Vector Machines*, n.d.). The loss is set to ”hinge”. The hinge loss function is good at finding the largest margin between data points, especially usefull when dealing with support vector machines. (*Hinge loss*, 2022)

C. Gradient Boosting Classifier

Gradient Boosting classifier is an ensemble machine learning algorithm that uses weak learners, usually decision trees in order to grow a strong learner by minimizing the mean squared error as shown (1). A weak learner is a model or algorithm that is just slightly better than random guessing. (*Gradient boosting*, 2022)

Gradient boosting classifier works by training the current weak learner on previous output with the goal to minimize the mean squared error, this is done by using (3). It is a supervised learner. The current weak learner is then added to the ensemble of weak learners and the process is repeated until the desired number of weak learners is reached.
The model used for gradient boosting classifier comes from a Python module called ”Scikit learn”.
(*Sklearn.ensemble.gradientboostingclassifier*, n.d.)

D. Logistic Regression

Logistic regression is another supervised learning algorithm often used for classification tasks like in this paper. It classifies binary data which means that it distinguishes between two classes, in this case the label column of the data set that is either 0 or 1. Because its a binary classifier, it is very well-suited for the data presented in this paper.
The main objective of a logistic regressor is to maintain the functions value between 0 and 1. By doing this, a probability can easily be calculated for each tweet which then makes it easy to classify the tweets as either positive or negative. If the function returns a value greater than 0.5, then the datapoints gets the class of 1, otherwise it is classified as 0(Albon, 2018). With the help of the Sigmoid function this is pissible (7)
The model used for Logistic Regression comes from a Python module called ”Scikit learn” and its fairly straight forward to use. Include the correct module from scikit learn and make a call to create an instance of the class. (*Logistic regression python solvers' definitions*, n.d.)

E. Naive Bayes

Like the other models, Naive Bayes is a classification algorithm that makes use of supervised learning. Based on the mathematical equation Bayes Theorem (**??**). The Bayes theorem is a statistical formula that allows us to determine the likelihood that a data point belongs to a particular class or label based on prior knowledge. By comparing prior probabilities of each potential class for a brand-new given data point, it can predict what class the new datapoint shall have and this Naive Bayes algorithm takes advantage of. In the context of the dataset used in this paper, the prior estimation is what the previous tweet were classified as and the model uses that for next tweets.
In this paper, a specific type of Naive Bayes called Multinomial Naive Bayes. This version of the algorithm is well-suited for dealing with discrete data, where the observations are assumed to be drawn from a multinomial distribution. Multinominal Naive Bayes most common usage is when dealing with text classification, like in this paper. Essentially, Multinomial Naive Bayes is similar to the standard Gaussian version of the algorithm, but it is designed to handle discrete data rather than continuous data. (Albon, 2018)

The model used for Mutlinominal Naived Bayes comes from a Python module called "Scikit learn" (*1.9. naive Bayes*, n.d.) and is simply imported and saved within a variable when creating an instance of the class.

III. Result

The question presented in the introduction will be addressed. As a reminder, the question was:

What is the impact of using machine learning algorithms for sentiment analysis of tweets on the accuracy and reliability of sentiment predictions compared to traditional methods?

Using machine learning algorithms for sentiment analysis of tweets can significantly improve the accuracy and reliability of sentiment predictions compared to traditional methods. As demonstrated by the results of the implementation, models such as Naive Bayes and Logistic Regression perform very well in terms of accuracy, confusion matrix, and F1 score. The impact of using machine learning in this context is significant, and the right model can outperform most humans at tasks like these. The result really shows that investing in these types of implementations can be very profitable for companies, as demonstrated by the results, rather than hiring people to do the work.

IV. Analysis

The models that were developed and trained on the cleaned dataset performed very well overall. Logistic regression, multinomial naive bayes, and support vector machine scored an accuracy above 80%. However, the gradient boosting classifier did not perform as expected and only had an accuracy of 53%. The reason for the poor performance of the gradient boosting classifier may be due to the fact that the hyperparameters were not tuned enough, leading to poor performance. Since logistic regression had an accuracy of 93%, multinomial naive bayes had an accuracy of 94%, and support vector machine had an accuracy of 84%, imbalanced data, poor data partitioning, and insufficient data can be ruled out, as all models were trained on the same training set and tested on the same test set.

Upon examining the confusion matrices of all the models, it appears that most predictions were classified as true positives (positive comments) and true negatives (negative comments). However, the gradient boosting classifier showed a significant number of false positives, meaning that it classified positive tweets as negative. This suggests that the hyperparameters were not sufficiently adjusted, leading to poor performance of the model. It is not fair to judge the performance of a machine learning model solely based on the confusion matrix, as this matrix only provides a general overview of how the model performed overall. In order to fully understand the strengths and weaknesses of the model, it is necessary to consider a variety of other metrics in addition to the confusion matrix.

The gradient boosting classifier did not perform well in terms of F1 score, as indicated by its low score of 0.68. Most other models, with the exception of the gradient boosting classifier and support vector machine, had F1 scores above 86%, indicating high precision and recall. In an effort to improve the performance of these two models, the hyperparameters were adjusted, but this resulted in longer learning times. Despite achieving relatively high accuracy, the gradient boosting classifier still had a poor F1 score, which is a measure of the balance between precision and recall. Using deep learning could potentially address both the long training time and poor F1 scores, but it was not an option for this assignment. The gradient boosting classifier's training time increased significantly after adjusting the hyperparameters, including the number of estimators, learning rate, and number of datapoints in the training set. Despite trying various combinations of these hyperparameters, none of them resulted in a better F1 score. Some combinations were not tested due to time constraints. It is possible that further adjustments to these hyperparameters could have improved the model's F1 score. However, the conclusion drawn is that gradient boosting classifier should not be used with a dataset this big.

The accuracy and F1-score of the support vector machine were good, however an attempt to further improve it was initialized. In order to gain both more accuracy and gain a higher precision and recall(F1-score) the hyper parameters were tweaked. The hyper paramters that were tweaked were: Loss, penalty, alpha and l1_ratio. The loss parameter is measures on how good the model is on separating classes. The penalty parameter was used in order to prevent over fitting. The alpha parameter is mainly used in combination with the penalty parameter, more specific the L1 and L2 penalty. The alpha value regulates the model meaning the model will be more constrained in terms of parameters. The last parameter is l1_ratio and this one is used in combination with Elastic Net penalty and basically determines if L1 or L2 penalty should be used.

The result from doing that was surprisingly good, both the accuracy and F1-score increased significantly. The accuracy went from 84% to 90% and the F1-score went from an array from 85% to 90%. In general these were small improvements however, this clearly shows that if hyper parameters are tweaked correctly, the models performance may increase significantly.

From a company's perspective, implementing a solution using machine learning techniques may require an initial investment, but in the long run, using a machine for tasks like this will likely be more cost-effective compared to using human labor. Additionally, if this technology becomes widely available for use by the public, it could potentially be used to protect vulnerable individuals from scams. For

example, these machine learning models could be trained on a variety of datasets to detect and prevent scams targeting elderly individuals. While the current implementation and dataset may not be perfect, the potential for these models to be applied to other similar tasks is high. In summary, using machine learning for tasks like this has the potential to not only be more cost-effective for companies, but also to provide a valuable service for the public by protecting them from scams.

After analyzing the results of the Kruskal-Wallis and ANOVA oneway tests, it was found that there is a significant difference between the algorithms being compared. The results indicate that at least one of the algorithms is significantly better than the others. This was expected, as the performance of the Logistic Regression and Naive Bayes algorithms was particularly better compared to the other algorithms. However upon examination the results furthermore, the hyperparameters of support vector machine were tweaked and as mentioned before, improved a lot in terms of accuracy and F1-score. So based on these results, it can be concluded that either Naive Bayes, Logistic Regression or support vector machine with some tweaks would be the best choice for tasks that prioritize accuracy, precision, and recall. Overall, the tests appear to be fair and correct in their conclusions therefore gradient boosting classifier and support vector machine can be ruled out with a data set this big and for tasks like these.

V. Conclusion

The conclusion that can be drawn is that using computers that perform operations in the form of machine learning is much more effective than originally assumed. After cleaning the data and implementing the four algorithms, it can be said with a high degree of confidence that a machine can classify tasks like this much better and much faster than a human can in most cases. Some algorithms are better than others, as seen in the results, but if the right algorithm is chosen, such as logistic regression, tasks like this can be classified with a high degree of confidence. The initial idea was that the main focus would be on research about the algorithms and how to implement them. However, during the code writing, about 40% of the time was spent building the models and 60% of the time was spent cleaning the data, learning about tokenization, stemming, and bag of words, just to mention a few of the methods applied to the data. It is reasonable that the cleaning of the data took up most of the time because no algorithm is good at learning from poor data. If a task similar arises in the future, it may be worthwhile to consider using deep learning techniques instead of standard machine learning in order to improve the effectiveness and efficiency of the classification process. Additionally, it may also be beneficial to invest further time and resources into thoroughly cleaning and preparing the data, as the quality of the data has a significant impact on the performance of the model. By using deep learning algorithms and ensuring that the data is as clean and well-structured as possible, it is likely that the classification process will yield more accurate and reliable results.

VI. Contribution

Jakob Adamsson is the author and has written this report as well as the implementation of all code by himself.

VII. Formulas

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{1}$$

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - F(x_i))^2 \tag{2}$$

$$-\frac{\partial L_{mse}}{\partial F(x_i)} = \frac{2}{n}(y_i - F(x_i)) \tag{3}$$

$$P(y_i = 1|x_i) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_i)}} \tag{4}$$

$$P(y_i = 1|x_i) \tag{5}$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{6}$$

$$\frac{1}{1 + e^{-z}} \tag{7}$$

Where:
y_i is the target variable
$\hat{y}_i$ is the predicted value
n is the number of observations
$F(x_i)$ is the current weak learner
x_i is the input variable
L_{mse} is the mean squared error
β_0 is the intercept
β_1 is the slope
$P(A|B)$ should be interpreted as probability of A happening given B occred
$P(A)$ and $P(B)$ should be interpreted as probability of A and probability of B

References

1.4. support vector machines. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/svm.html

1.9. naive bayes. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/naive_bayes/

Albon, C. (2018). *Machine learning with python cookbook: Practical solutions from preprocessing to deep learning*. " O'Reilly Media, Inc.".

Burchfiel, A. (2022, Oct). *What is nlp (natural language processing) tokenization?* Retrieved from https://www.tokenex.com/blog/ab-what-is-nlp-natural-language-processing-tokenization/

Gradient boosting. (2022, Dec). Wikimedia Foundation. Retrieved from https://en.wikipedia.org/wiki/Gradient_boosting

Hinge loss. (2022, Nov). Wikimedia Foundation. Retrieved from https://en.wikipedia.org/wiki/Hinge$_loss$

(n.d.). Retrieved from https://www.nltk.org/

Logistic regression python solvers' definitions. (n.d.). Retrieved from https://stackoverflow.com/questions/38640109/logistic-regression-python-solvers-definition

Sklearn.ensemble.gradientboostingclassifier. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassif

Support vector machine. (2022, Dec). Wikimedia Foundation. Retrieved from https://en.wikipedia.org/wiki/Support_vector_machine