Final Report

CS464 Introduction to Machine Learning

Group 12

Contextual Sarcasm Detector

Abdallah Zaid Alkilani

Muhammad Usman

Asena Rana Yozgatli

Unas Sikandar Butt

INTRODUCTION

The purpose of this project is to detect sarcasm from a given reddit comment using some common machine learning algorithms. Sarcasm is defined on the Dictionary.com website as “the use of irony to mock or convey contempt. This program will be given a dataset of comments from reddit, trained on them, then tested using the same dataset. For this project our team has chosen the Naive Bayes(NB) algorithm and the Support Vector Machine(SVM) algorithm to train our model and predict results. The results of the project were unexpected to a certain extent, as the NB model which was expected to outperform the SVM did not produce reliable results and the SVM produced better and reliable results even though it was not as accurate as expected.

PROBLEM DESCRIPTION

As mentioned earlier the main purpose of this project is to detect sarcasm from a given text comment. It should be mentioned that as the dataset that the model is trained on comprises of reddit comments the model, the accuracy might drop drastically if a random text comment is given which is not part of the reddit comment culture.
The problem of sarcasm detection can appear to be a very complex problem on the surface. However after doing some research online of some papers published on sarcasm detection using machine learning, we found that there are some key features in sarcastic comments that can be identified (explained in detail further). After finding these key features the model could be trained easily to produced some results.

METHODS

Regarding the methodology, first the dataset is explained and features of the dataset are stated. Later on we state the results of our research for the features that can be extracted to analyze a comment for sarcasm.

The Dataset
This dataset contains 1.3 million sarcastic comments from the Internet commentary website Reddit. The dataset was generated by scraping comments from Reddit containing the (sarcasm) tag. This tag is often used by Redditors to indicate that their comment is in jest and not meant to be taken seriously, and is generally a reliable indicator of sarcastic comment content [1].

Features of the Dataset: comment, sarcasm_label, author, subreddit, score, ups, downs, date, created_utc, and parent comment.

Hyperboles
As nouns, the difference between hyperbole and sarcasm is that hyperbole is (countable) extreme exaggeration or overstatement; especially as a literary or rhetorical device while sarcasm is (uncountable) a sharp form of humor, intended to hurt, that is marked by mocking with irony, sometimes conveyed in speech with vocal over-emphasis insincerely saying

something which is the opposite of one's intended meaning, often to emphasize how unbelievable or unlikely it sounds if taken literally, thereby illustrating the obvious nature of one's intended meaning.
However, since hyperboles (countable) can be an indicator of sarcasm (uncountable), it is worth considering hyperboles a feature of sarcastic text. Queues like the number of exclamation points, capitalized words, and use of emoticons could be considered as hyperboles potentially indicating sarcasm.

Exclamation points
Sarcastic text makes use of exclamation to indicate hyperboles and relay sarcasm, and hence can be considered as a feature for sarcasm detection [2]. Usually, one or more exclamation points are used to indicate sarcasm; the more exclamation points used, the more sure the reader can be that the text is sarcastic, again in a hyperbolic way. Exclamation points place emphasis at the end of sentences to provide the reader with hints of the (possibly) sarcastic nature of the text.
For each of the comments in the dataset, the number of exclamation points was counted and stored to be used as a feature. The mutual information scored was calculated, and was found to be **0.0143**

Capitalizations
Sarcastic text makes use of capitalizations to indicate hyperboles and relay sarcasm, and hence can be considered as a feature for sarcasm detection [2]. Usually, one or more words are capitalized to indicate sarcasm; the more words are capitalized, the more sure the reader can be that the text is sarcastic, again in a hyperbolic way. Capitalizations place emphasis at certain parts of sentences to provide the reader with hints of the (possibly) sarcastic nature of the text, usually with increasing number of capitalizations increasing sarcastic interpretability.
For each of the comments in the dataset, the number of capitalized words was counted and stored to be used as a feature. The mutual information scored was calculated, and was found to be **0.0038**

Emoticons
Emoticons (and emojis, but emojis are not used nearly as frequently on Reddit) can be employed as hyperbolic indicators of sarcasm, in a way that emulates facial expressions [2]. Usually, the presence of certain emoticons, most notably the upside-down face “(:” emoticon, can be interpreted and intended to relay sarcasm [3]. The more emoticons used, the more the reader can grasp the sarcastic intentions of the writer, as emoticons relay in some sense facial expressions that would otherwise be difficult to relay in plain text.
For each of the comments in the dataset, the number of emoticons was counted and stored to be used as a feature. The mutual information scored was calculated, and was found to be **0.0038.** The emoticons considered were:

:-) xD :-O =P :| B-) :) :-(:* :-/ -_- B-D (:

:(:-* :/ >:-) (͡° ل͜ ͡°) =)): ;-) =/ >:) ¯_(ツ)_/¯

:3 =(;D :-\ O:-) (´▪ω▪`) =3 D: ;) :

O:) (`▪ω▪´) ☺ Dx :-P =\ 0:-) (▪ω▪) :-D D= :P :I

0:) :D D; XP =| B) =D :O X-P :-| BD

Sentiments

Sarcastic text can be overall more positive or negative than normal text. Or, it can show a huge contrast of sentiments throughout the text. For example, the sentence could start with a positive sentiment and end with a negative sentiment. For example: “I love being poor”. This switch in polarities can be an indicator of sarcasm and can be used as a feature in the model.

Lexical

Text features such as unigrams, bigrams, POS tags are known as the lexical features of the text. According to our research, these features have been used in previous attempts to detect sarcasm in text and yielded pretty good results [4]. So we’ll be using these features in training our model.

Topics

In natural language processing, topic modeling is used for discovering hidden semantic features from documents or text. A text generally consists of several topics, where a topic is a group of co-occurring words. We’ll be using topics as a features to train our model to see which topics are generally more associated with sarcasm.

Models:

I. **SVM**

The SVM model was chosen because our research suggested that it works really well for text data due to the reason that most text classification problems are linearly separable, the generated document vectors are sparse and the text classification problems have a lot of features. All of these qualities make SVM a good model to be used for text classification.

II. **Naive Bayes**

The Naive Bayes model was chosen because research suggested that it was the most efficient model to classify textual comments. The Gaussian version of naive bayes was used because all of our data and features were continuous data eg. polarity, sentiments, capitalizations. Using gaussian naive bayes model from the scikit learn library, we trained the model in hopes to attach weights to each of the features and to use their probabilities to predict the label of a given comment.

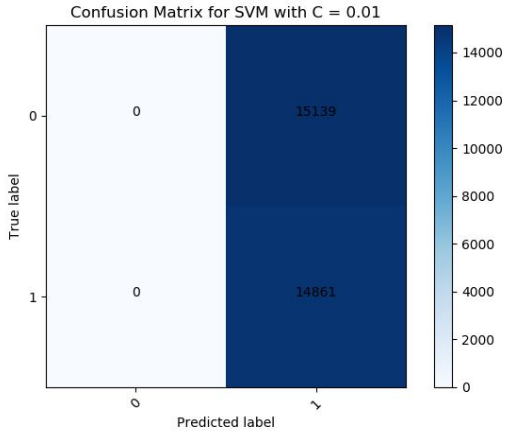
Results

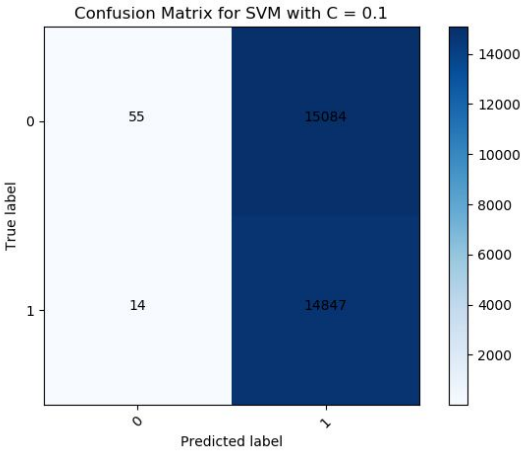
SVM:

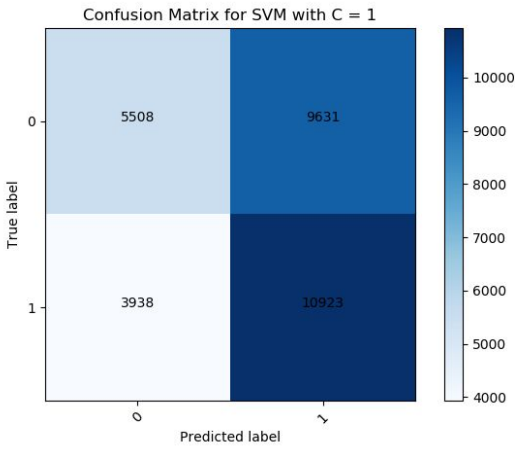
For SVM we used the rbf kernel for training and testing the data. The reasons we choose to use the rbf kernel instead of the linear kernel was that we thought the dataset is not linearly separable and using a non-linear kernel would be a better approach. And our training confirmed our hypothesis. We noticed that training using linear SVM took forever and the reason for that was that the data was not linearly separable and the model couldn't find the correct decision boundaries to separate the sarcastic comments from the non sarcastic ones. Another important factor that affects the accuracy results is how wide the margin in our SVM model is, and that is why we used 5 different models with different C values to see how the accuracy, precision, recall and f1-score changes with different C values.

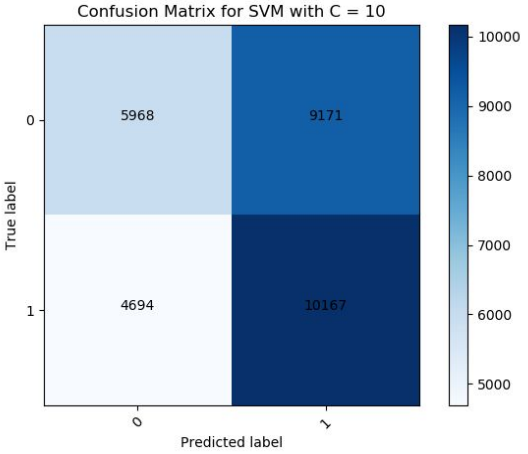
Confusion Matrices:

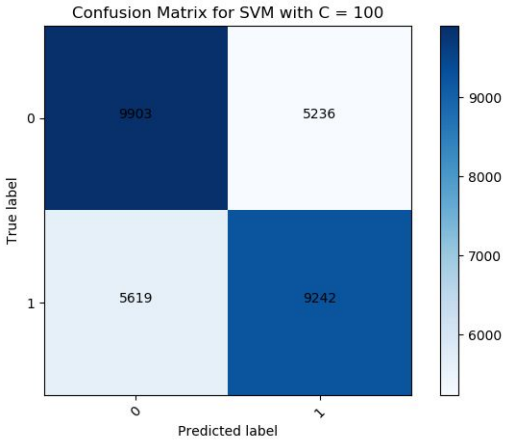
Below are the confusion matrices for C values = 0.01, 0.1, 1, 10, 100



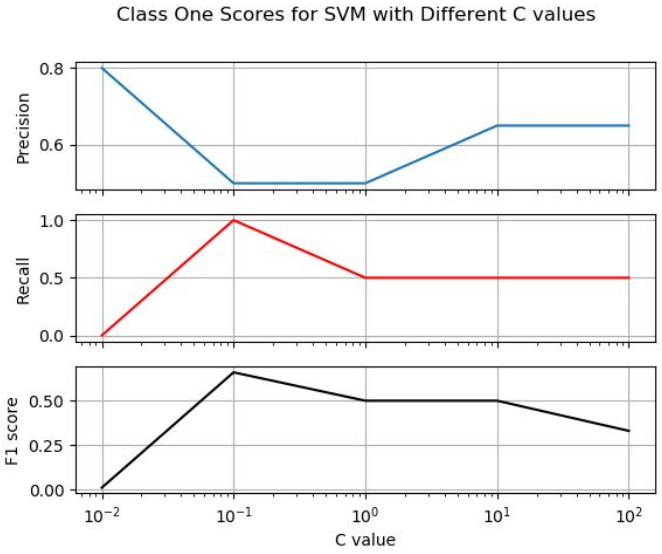


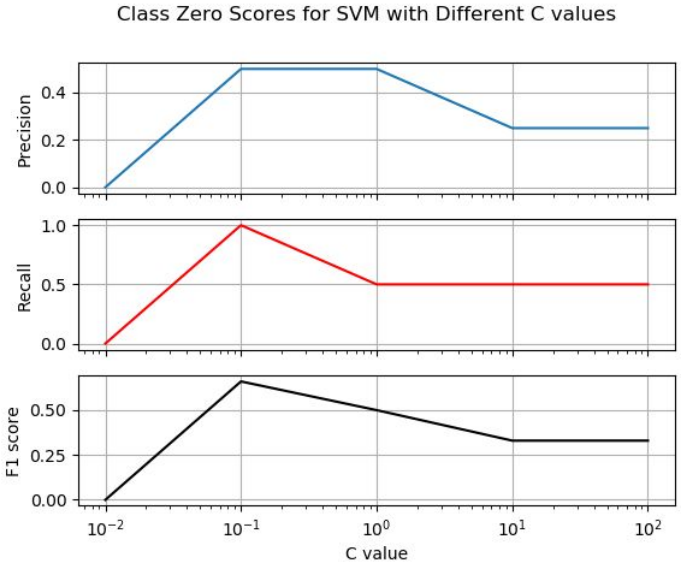






Precision, Recall and F1-scores:





Naive Bayes:

As mentioned earlier the naive bayes model did not work as expected. The Gaussian Naive Bayes, was first tested using the entire data set, even after using cloud computing services the model reported errors in memory. After getting these errors we tried to start fresh and use a short percentage of the data set. Here are some of the results we obtained:

Training set: 5000
Test set: 1000

Confusion Matrix	Predicted Non-Sarcastic	Predicted Sarcastic
True Value Non-Sarcastic	485	0
True Value Sarcastic	0	515

Training set: 7000
Test set: 3000

Confusion Matrix	Predicted Non-Sarcastic	Predicted Sarcastic
True Value Non-Sarcastic	1528	0
True Value Sarcastic	0	1472

Training set: 10000
Test set: 6000

Confusion Matrix	Predicted Non-Sarcastic	Predicted Sarcastic
True Value Non-Sarcastic	0	2495
True Value Sarcastic	0	3505

Discussion:

SVM:

C value in SVM represents how much of an error is allowed in classifying the data points by setting how big of a penalty will be paid for each misclassified point or in other words how big or small the SVM margin should be. If the C value is really small, it means that the margin is going to be really wide and the penalty for misclassification will be really small. Thus a data point actually belonging to class 0 can be predicted as to belong to class 1 and vice versa. And if the C value is really large, the penalty for misclassification is going to be really large as well thus resulting in the margin to be really small. The confusion matrices shown in the results section also show this trend. For C = 0.01, all of the samples are predicted to be positive. And as we increase the C value, the classifier predicts more and more values to lie in the class 0 and less and less in class 1. This happens because, as we increase the C value, the margin becomes smaller and thus then the data points are separated into the 2 class labels much more realistically. In our case, C = 100 gives the best accuracy i.e 63.8% and the best f1-score for both class 0 and class 1 when precision and recall are equal. The reason for that is in the case of C = 100, the number of true negatives and true positives come out to be almost the same. So C = 100 is the best hyper parameter to use in SVM for our dataset.

Naive Bayes

The Naive Bayes results were very hard to interpret. Firstly for a short data set (~5k) the trained model could predict the test data with 100% accuracy. These numbers were very surprising as we could not interpret whether our model was perfect or there was some human error while doing the computations.

Furthermore, as we increased the dataset (~10k) the trained model behaved unexpectedly. Our hopes were that the model would remain perfect as the literature suggests that as we increase the amount of data the resulting model should improve. However, that was not the case; as the training data increased the model started to predict everything as sarcastic. These results suggest that either the approach used for the prediction (features) is wrong as increasing the data is disturbing the results and decreasing accuracy or it might suggest that there are some inconsistencies within our dataset which is causing the model to predict everything as positive.

Either way the Gaussian Naive Bayes approach did not yield conclusive results.
Attempts were made to fit our dataset into a Multinomial dataset however it proved to be impossible.

CONCLUSION

In the conclusion it should be mentioned that this project was good introduction into the vast field of machine learning. We applied multiple feature selection strategies and multiple classifiers to achieve these results.
According to the results it should be safe to suggest that for our dataset the SVM approach was the way to go. As it provided results that were interpretable and understandable and expected. The naive bayes approach failed disappointingly. However the approach should not be criticized based on this project
The SVM approach resulted in a accuracy of ~64% which can be considered a very good number according to the sarcasm detection literature. Our model can be further improved by employing the k-fold cross validation technique.

APPENDIX

Contributions:
Muhammad Usman : Worked with Topic Modeling and SVM
Unas Ali Sikander: Worked with n-grams, POS features and Naive Bayes
Abdullah Zaid Alkilani: Wrote feature extraction code for exclamation points, capitalizations, and emoticons (+ interjections and intensifiers) and code for plots and Mutual Information
Asena Rana Yozgalti: Wrote the code for sentiment analysis and Naive Bayes