**IST 736 Text Mining**

**2020 Summer**

**Final Project – Sarcasm Detection in News Headlines**

Xiwei Shen and Fangzhou Yuan

**1. Abstract**

Sarcasm is widely used in verbal and written languages, while leaving a challenge in automated sentiment classification algorithms as its nature of flexible language patterns. This project aims at building models to detect sarcasm used in news headlines. By doing so, we can classify news headlines into two categories based on if they are sarcastic or not. In addition, such models can be used in sentiment classification tasks. The mechanism of running a sarcasm detection model prior to sentiment classification algorithm gives an advantage of dealing sentiments in sarcastic and flat documents separately.

**2. Introduction**

The inspiration of this project comes from our previous experiments with building sentiment classification algorithms by document sentiments. We found sarcastic documents tend to be more likely mis-classified by models such as Linear SVM and Naïve Bayes, and a large portion of such mistakes are “extreme cases”, e.g. classifying very positive documents as very negative, or vise versa. Therefore, categorizing the corpus into two groups of sarcastic and flat documents, then build and train sentiment classification models on each group should give a more accurate prediction result on document sentiment. We expect to see very different feature weights on these two sentiment classification models.

The dataset we used in this project is from Kaggle. The news headlines are collected from two news media websites, The Onion and HuffPost. The Onion is known for its satire, which uses humor irony, exaggeration, or ridicule to criticize people and events (Media Bias/Fact Check). HuffPost is a popular news and commentary left-biased site. All our sarcastic news headlines are from the Onion and HuffPost counts for all the non-sarcastic data points.

The distribution of the data we used brings a data quality issue can potentially influence our sarcasm detection model. Since we have a unitary data source for each label, besides whether the headlines are sarcastic or not, there can be differences in language pattern, topic coverage, and editors’ wording characteristics, timeliness, etc. Therefore, it is essential to perform descriptive analysis before running the models and feature ranking after the models are built.

To explore the contents of news headlines with each label, we built word-clouds and topic modeling by sarcasm labels. Comparing the two word-clouds generated, we found that there is an overlap in the topics and frequent words used in both sarcastic and flat news headlines. However, not all terms are identical. For instance, “trump”, “Obama”, “Hillary Clinton” are mentioned more frequently in the HuffPost headlines. Via checking the data manually and scraping the date of publication from the webpage links provided, we found the publication distribution of HuffPost is localized at 2015-2016 with a Gaussian-like shape, while the timeline of the data from The Onion is more spread out with the median around 2014.

Since terminologies and names can be unique for different topics, we also did topic modeling to see how well the topics overlap for news with two labels. From the results of our topic modeling, it is we found the same pattern with the word-clouds. Names and other nouns related to the 2016 election were mentioned more frequently in the non-sarcastic news. Besides, it is very challenging to interpret and give an accurate annotation for each topic. What we do capture is that the non-noun features from the sarcastic news headlines are not as formal as the ones found in the flat headlines.

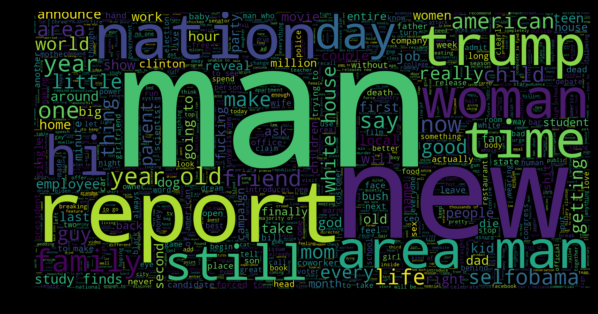
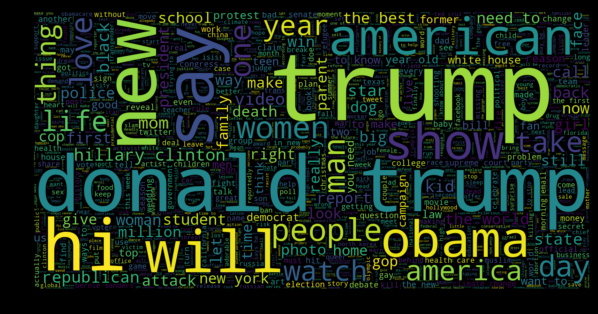


Figure 1. word-cloud for flat and sarcastic news headlines respectively (top is for flat)

By looking at the feature ranking given by the machine learning models, we can interpret and evaluate if the models are catching sarcastic patterns rather than identifying the data sources. The details and the results are given in the method section.

**3. Data Acquisition**

Our dataset contains three attributes. The first one is headline, which is the headline of the news article. The second one is "article\_link", which is the link to the original news article. The last one is "is\_sarcastic", it contains binary values 0 and 1. Value 1 represents the record is sarcastic, otherwise, the value is 0. The dataset contains 28619 records in total. There are 13635 sarcastic records and 14984 non-sarcastic records. The majority vote baseline is about 52%, so overall, we would say this dataset is balanced.

手机屏幕截图

描述已自动生成

Figure 2. Data distribution

**4. Methods**

In this project, we deployed two models, which are linear SVM and Multinomial Naive Bayes on our dataset to perform the classification task. For the evaluation method, we used the n-fold cross-validation method to evaluate our model performance. Since compared with the hold-out test, the cross-validation method is a better estimation of how good the model will be in terms of applying it to the out-of-sample data. We have tried different vectorizer options and different parameter combinations during the preprocessing, in order to find out the optimal model output. The primary evaluation metric that we used for this task is the accuracy score. We will also check other metrics, such as precision and recall, see if there are some interesting patterns appear.

**4.1. Linear SVM**

Support vector machine is a supervised machine learning model with associated learning algorithms that are mainly used in classification analysis. An SVM is a kind of large-margin classifier: it is a vector space based machine learning method where the goal is to find a decision boundary between two classes that is maximally far from any point in the training data (Manning et al.). The decision boundary that the SVM model is trying to find is the one with the maximum margin, which is the maximum distance between data points of different classes. Data points falling on either side of the decision boundary can be attributed to different classes. Margin is only decided by the closest data points on each side of the decision boundary, and such data points usually been called support vectors. The SVM model is very suitable for two-group classification problems due to the model properties, and it works with both linearly separable and linearly non-separable cases. Because of this property, the SVM model has also been commonly used on text classification problems.

**4.1.1 Vectorization and Hyper-parameter Tuning**

For vectorization of the linear SVM model, we did experiments on Boolean, term frequency, and TFIDF vectorizers, trying to find the optimal choice. We have also tuned other parameters, such as document frequency threshold, token patterns, and stop words option, and comparing the outcome every time, see if there are some increment in the model performance. One interesting result is that if we choose to keep stop words, the model accuracy will higher by around 5% than we removed stop words, which is a huge difference. We tried to include bigram and trigram tokens into the vectorizer and check for the model performance, by doing so the model will be able to capture consecutive words in the headlines. We found that adding multi-word tokens is helpful for this classification task, the model accuracy rises when we are adding the bigram tokens. But if we also include trigram tokens in the vectorizer, the model accuracy will drop a little and this feature is consistent on all three vectorizers. Therefore, keep using bigram tokens seems to be a better option.

Then we let the token only identifies alphabetical words during vectorization by setting the token pattern as ‘[a-z]+’. There is a slight improvement on the model performance by adding this constraint. Next, we have tested many combinations on the minimum and maximum document frequency. We finally found that the model has the best performance if we don't set up thresholds on document frequency. By doing this, the vectorizer doesn't ignore any vocabulary, even those words that only appear in very few documents. Among the three different vectorizers, the Boolean vectorizer gets us the lowest accuracy score, the term frequency vectorizer performed slightly better, and the TFIDF vectorizer gets us the highest model accuracy.

For the SVM model, there is a special parameter we need to tune and that is cost C, which is the penalty associated with misclassification. This parameter makes SVM becoming a soft margin classification, which allows a small number of misclassifications. The model with higher costs will have lower bias but the risk of overfitting. The model with lower cost will have higher bias but lower variance. Based on the optimal TFIDF vectorizer we got, we tuned the parameter C from 0.1 to 3 and check for the model performance. We found that the model gets the highest accuracy score when C equals to 2. We printed out the confusion matrix of the optimal linear SVM model and evaluate its performance in detail.

**4.1.2 Model Evaluation**

According to figure 3, we can see that the linear SVM model performed well overall on this dataset. For the non-sarcastic category, there are 3972 examples that were correctly classified as non-sarcastic, and there are 588 examples that were mistakenly classified as sarcastic. For the sarcastic category, there are 3447

|  |  |
| --- | --- |
|  | Linear SVM |
| Unigram Boolean | 0.8324 |
| Bigram Boolean | 0.8614 |
| Trigram Boolean | 0.8610 |
| Unigram TF | 0.8336 |
| Bigram TF | 0.8621 |
| Trigram TF | 0.8617 |
| Unigram TFIDF | 0.8510 |
| Bigram TFIDF | 0.8673 |
| Trigram TFIDF | 0.8657 |

Table 1. Linear SVM CV Accuracy

examples that were correctly classified as sarcastic, and there are 579 examples that were mistakenly classified as non-sarcastic.

手机屏幕截图

描述已自动生成

Figure 3. Linear SVM Confusion Matrix

The classification report can be used to check other model evaluation metrics, such as precision, recall, and F-1 score. Precision can tell us that among all the examples that have been classified as positive, what is the percentage of real positive examples. Recall can tell us what proportion of actual positives was identified correctly. The F1-score is the harmonic mean of precision and recall.

手机屏幕截图

描述已自动生成

Figure 4. Linear SVM Classification Report

According to figure 4, based on the high F-1 score, we can see that the linear SVM model did well in identifying both sarcastic and non-sarcastic examples. The precision and recall are also quite satisfactory. For all non-sarcastic examples, the linear SVM model can correctly identify 87% of them. And within that 87%, there are 87% are actual non-sarcastic examples.

**4.2 Multinomial Naïve Bayes**

The Naive Bayes classifier is a probabilistic machine learning model that’s used for the classification task. It is a simple classifier that classifies based on probabilities of events. The Naive Bayes classifier is based on the Bayes’ Theorem, and it can help us compute the conditional probabilities of occurrence of two events based on the probabilities of occurrence of each individual event. The objective of the Naive Bayes classifier is to use Bayes theorem to calculate the posterior probability for each class, and the class with the highest posterior probability is the outcome of the prediction. The Naive Bayes classifier applied commonly to text classification, and it performs well in many text classification problems. There are two Naive Bayes models that are commonly used for text classification. The first one is the Multinomial model, which uses word frequency and is suitable for classification with discrete features, such as word counts. The second one is the Bernoulli model, which uses word presence/absence and is more designed for binary or Boolean features. In this project, we deployed Multinomial Naïve Bayes model on our dataset.

**4.2.1 Vectorization and Hyper-parameter Tuning**

Since the MNB model counts word frequency, we only used the term frequency vectorizer and the TFIDF vectorizer as the model input. Just like the linear SVM model, we found that the model accuracy will increase by around 5% if we choose to keep the stop words during the vectorization rather than remove them.

Next, we tried to add bigram and trigram tokens in the vectorizer and check for model performance. We got a similar result with the linear SVM model that including multi-word tokens into vectorizer will rise the model accuracy accordingly. Identifying alphabetical words only by setting a constraint on the token pattern during vectorization is also helpful for model performance improvement.

Another consistent result with the linear SVM model is that the MNB model has the best performance if we let the vectorizer to capture all vocabulary in the document by not setting thresholds on maximum and minimum document frequency. Next, we applied the optimal parameter settings on the term frequency vectorizer and the TFIDF vectorizer and check for model performance. We found that the MNB model with the term frequency vectorizer as the input gets us a better classification accuracy score. These are the MNB model outputs by using different vectorizer:

|  |  |
| --- | --- |
|  | MNB |
| Unigram TF | 0.8562 |
| Bigram TF | 0.8636 |
| Trigram TF | 0.8645 |
| Unigram TFIDF | 0.8405 |
| Bigram TFIDF | 0.8522 |
| Trigram TFIDF | 0.8568 |

Table 2. MNB CV Accuracy

**4.2.2 Model Evaluation**

According to figure 5, for the non-sarcastic category, there are 4096 examples that were correctly classified as non-sarcastic, and there are 464 examples that were mistakenly classified as sarcastic. For the sarcastic category, there are 3310 examples that were correctly classified as sarcastic, and there are 716 examples that were mistakenly classified as non-sarcastic. The misclassified examples of the linear SVM seem to be more balanced, while the MNB model did better on classifying non-sarcastic examples. But the MNB model also tends to classify more sarcastic examples as non-sarcastic mistakenly.

手机屏幕截图

描述已自动生成

Figure 5. MNB Confusion Matrix

手机屏幕截图

描述已自动生成

Figure 6. MNB Classification Report

According to figure 6, we can see that the MNB model did a good job of identifying both sarcastic and non-sarcastic examples as well. The F-1 scores of these two models are very close. For all non-sarcastic examples, the MNB model can correctly identify 90% of them. And within that 90%, there are 85% are actual non-sarcastic examples.

**5. Error Analysis**

For error analysis, we checked those sarcastic examples that are mistakenly predicted as non-sarcastic examples by the MNB model and the linear SVM model. The two models actually made similar prediction errors and I found many errors are the same. I noticed that those really short headlines often got mistakenly classified, such as "family business" and "chaps unnecessary". That is probably because those headlines contain very few words, which leads to an ambiguous headline meaning. I also found a common point among some misclassified headlines, that is, these titles contain turning words or implicit turning semantics. These are some examples:

“man who will pay $60,000 in medical bills this year can't afford health insurance right now.”

“Facebook: 'we will make our product worse, you will be upset, and then you will live with it.'”

“U.S. adds 4 million jobs but in St. louis”

These headlines do have sarcastic meanings, but they are mistakenly classified as the non-sarcastic category. The potential reason for this is the model did not pay close enough attention to turning words and implicit turning semantics. In these headlines, the content after the turning words is often the part that contains sarcastic meaning. The two models didn't do a good enough job of identifying this part of the content. Therefore, we believe how to identify and deal with turning words and semantics is a direction of improvement for both models.

**6. Feature Ranking**

After the models are built, we analyzed the feature ranking given by each model. For the MNB model, the words with highest conditional probabilities in the sarcastic category have no apparent connection with sarcasm. Even if the stop words are preserved, the top features do not include many semantic patterns such as negation or degree adverbs that are commonly used in sarcasm. On the contrary, nearly half of the top features from the Linear SVM model are sarcasm related. Words such as “clearly”, “only”, “repeatedly”, “desperate”, etc. can be employed to express exaggeration. Not surprisingly, we saw some curse words in the list as well. In general, both models are not making the classification fully based on sarcasm, but sarcastic features are employed more by the SVM model.

**7. Conclusion and Limitation**

As the confusion matrix and feature ranking stated, while the Linear SVM and Naïve Bayes model having very similar accuracies and F1 scores, the features that the Naïve Bayes model captures are less relevant to sarcasm. Hence, we would pick the Linear SVM model for sarcasm detection. With that being said, the project can be more reliable and by improving the data quality.

There are two options for data with better quality, but each has its own challenges. (1) Collecting both sarcastic and flat news headlines from a single website in a fixed time frame. Then we can ensure both types of news headlines are covering the same topics. In addition, we may have less authors since all the data is from the same site, which can give a more consistent corpus in terms of language patterns and wording choices. However, creditable news sources tend to be very cautious on how sarcasm is used in their news headlines. Therefore, we may have far more flat documents collected comparing to sarcastic ones. Not to mention that by using this method, we need to manually label the data which is extremely time consuming.

The second option is gathering sarcastic and flat news headlines from multiple sources. With more data being fed, the models reply less on topics and names, since the amount of data provides a larger probability of items being covered in both sarcastic and flat news headlines. Nevertheless, data acquisition and labeling are way beyond the scope of this project.

In conclusion, the model performance is influenced by the data quality and our method for sarcasm detection is not the most efficient. There are other models that are more lexical and semantic features oriented which can give a more accurate and efficient prediction (Burfoot and Baldwin, 2009), but this project successfully tested the viability of using distance-based machine learning classification algorithms to detect sarcasm in news headlines.

**8. Reference**

Clint Burfoot and Timothy Baldwin. Automatic Satire Detection: Are You Having a Laugh? 2009. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 161–164, Suntec, Singapore, 4 August 2009. <http://www.aclweb.org/anthology/P09-2041>.

Manning, Christopher, et al. *Introduction to Information Retrieval*. Cambridge University Press, 2008. <https://nlp.stanford.edu/IR-book/html/htmledition/support-vector-machines-and-machine-learning-on-documents-1.html>

Search and Learn the Bias of News Media. (2020, April 09). Retrieved August 09, 2020, from <https://mediabiasfactcheck.com/>