Sarcasm and Irony Detection for German

Student: Lukas Pichler
Graz, University of Technology

Problem and Task

Distinguishing between a sarcastic statement and a non-sarcastic statement can even for humans be a hard challenge. Irony and sarcasm are rhetoric devices present in everyday live. The task was to train a classifier to detect sarcastic and ironic statements in German text. The classifiers I have trained cannot distinguish or differ between sarcasms and irony but they are able to classify between sarcasm and irony, or not in a text-corpus of German tweets. For those who are interested in classifying between sarcasm and irony I prefer to read the paper of Jennifer Ring and Roman Klinger[1] .

Data Set

As a data set I have collected tweets tagged with specific hash tags to create my training set. I have adapted the approach from the seed paper from Jennifer Ring and Roman Klinger[1] and used not only the hashtag #Sarkastisch but a variation of different hashtags with the same meaning. Following hashtags were used to create a training set with tweets.

Hashtags used for streaming Tweets
#Ironie,#Sarkasmus, #Ironisch, #Sarkastisch,#Fail, # Glück, #Glücklich,#Fröhlich,#Freude,#Erfreut,#erbost,#verärgert

Sarcastic Tweet Example
Wow #InfinityWar macht so viel Spaß, ey, kam aus dem lachen gegen Ende gar nicht mehr raus... #Sarkasmus

Non-Sarcastic Tweet Example
Es sind die kleinen Dinge im Leben, die einen glücklich machen… #glücklich

To get this Data I have used the Twitter developer API where it is possible to stream data from twitter.

Bag of Words Approach and Feature Engineering

To get a representation of the data set suitable for our machine learning mechanism the text data must be represented in a different form. For that process, called feature engineering, I have chosen the Bag of Words approach which replace the document by a simple representation consisting of the words that appear in the document. This means that we are creating a feature vector by counting the occurrences of each word in a sentence and create a vocabulary for the whole document.

Uni-Gram Example of Non-Sarcastic Tweet
{die:2, Dinge: 1,Es: 1, einen: 1, glücklich: 1, kleinen: 1, im: 1, machen: 1, sind: 1, Leben: 1}

For each sentence a list as seen above is constructed and put into a vocabulary. This results in a *NxM* matrix where *N* stands for the number of texts in the corpus and *M* for the number of features, where each entry of one row refers to the count of the word appearing in the tweet. To remove common terms which appear very frequently in German sentences I have used a list of **German stop words** to not take this common terms into account. This list of German stops words is provided by the **Natural Language Toolkit** and consists of words like „und“, „oder“, „doch“, „der“,“die“,“das“. A further improvement I have used to get better features was stemming. **Stemming** is the process of reducing inflected words to their word stem. A German stemming corpus is provided by the **scikit learn library.** These two improvements gave me up to 5% better results at the end. In this project the bag of word model is represented as uni-gram representation. The scikit learn library offers also the possibility to realize the bag of word approach. Further it also offers the possibility to further improve the quality of the features by using the **term frequency-inverse document frequency** which is a numerical statistic that is intended to reflect how important a word is to a document in a corpus.

By using these methods the feature generation process is nearly completed, what still had to be done was to create a target vector which labels my output of the bag of word approach accordingly to sarcastic or non sarcastic. For that I have to create a *Nx1* vector which is labeled either 1 or 0 according to if the text comes from the sarcastic data set or non sarcastic data set. This feature engineering process results in a typical **binary classification task** where it is the task to classify if the element belongs to one of the two labels. During **preprocessing** the hashtags, links and usernames were removed from the tweets and further all tweets with less than 3 words were ignored. Additionally in my approach all tweets must pass a sentiment analysis to get into the final data set. This prevents the datasets from tweets with poor German quality.

Sarcastic Dataset	Non-Sarcastic Dataset	Sarcastic Dataset after preprocesssing	Non-Sarcastic Dataset after preprocessing
1553 tweets	1717 tweets	1009 tweets	824 tweets

Training classifiers

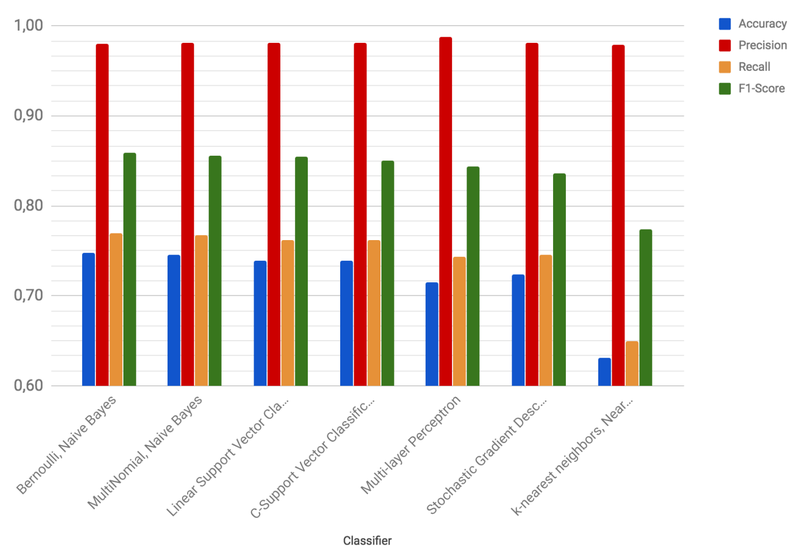
As already mentioned scikit-learn had been used. Its an easy-to-use, general-purpose toolbox for machine learning in Python and it provides various supervised machine learning techniques. When looking at our features we now had a typical binary classification problem to solve. There exist different classification techniques which all differ in the context of usage.

One such technique is **Nearest Neighbors**. Nearest Neighbor uses the notion of similarity to assign class labels. scikit-learn implements two different nearest neighbors classifiers and I have decided to choose the KNeighborsClassifier. Further I have used **Support Vector Machines,** which are a set of supervised learning methods for classification. In detail I have tested the **C-Support Vector Classification** and **Linear Support Vector Classification**. **Naive Bayes methods** are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features. I have chosen the Multinomial Naive Bayes and the Bernoulli Naive Bayes method for testing. Further I have tested the **Stochastic Gradient Descent** a generalized linear model and a **Multi-layer Perceptron classifier** from the family of neural networks. I have used **grid search** for tuning the hyper parameters of my classifier, with that it is possible to recommend or search the hyper-parameter space for the best score. In **k-fold cross-validation**, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. I have used this method for splitting my data set to get the best scores.

Results

Classifier	Accuracy	Precision	Recall	F1-Score
Bernoulli, Naive Bayes	0,75	0,98	0,77	0,86
MultiNomial, Naive Bayes	0,75	0,98	0,77	0,86
Linear Support Vector Classification, Support Vector Machines	0,74	0,98	0,76	0,85
C-Support Vector Classification, Support Vector Machines	0,74	0,98	0,76	0,85

Comparison of the different Classifier



Conclusion

The **best results** were achieved with the **Bernoulli Naive Bayes classifier**. High precision leads to low false positive rate which is very good at all classifiers. The recall is significantly higher than 0.5 for every classifier which is also pretty good. The F1 score (weighted average of Precision and Recall) and accuracy are a good indicator to figure out how well the classifier performs. The F1 score is above 0.8 for most of the classifiers which is good considering the linguistic quality of the tweets I have crawled. The quality of the tweets is also the reason for the moderate accuracies I have got at the end.

Improvements in the approach can be achieved by doing an annotation of the tweets by human experts, because the question remaining here is, can we detect sarcasm now or just classify that a tweet belongs to a certain hashtag. Further the k-fold cross validation could be extended with the leave one out method and in the feature engineering process different n-grams could be used.

Also the approach from Jennifer Ring et al[1] can be further applied to this scenario with German text to also distinguish between sarcasm and irony.

Tools

Twitter Development API, scikit-learn, Natural Language Toolkit, python 3.6

Literatur / Zitat

[1] Jennifer Ring and Roman Klinger, *An Empirical, Quantitative Analysis of the Differences between Sarcasm and Irony,* Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart