Machine Learning & Pattern Recognition

Fall semester 2020

Classifying News Headlines using Naïve Bayes

Professor:	Krzysztof Kryszczuk	Authors:	Chirayil Harry
Date of submission:	18th January 2021		Christopher Keim
			Stefan Schmutz

Introduction

The aim of this project is to create a model which is able to classify headlines (which includes titles and teasers) from two Swiss newspaper organizations, NZZ (Neue Zürcher Zeitung) and 20min. NZZ has a more economical- and 20min a more "mainstream" writing style. Text classification or text categorization is a sub-topic of machine learning and falls within the fields of NLP (Natural Language Processing). It is a supervised learning method in which every new document/headline is classified by assigning one or more class labels from a fixed set of pre-defined classes [1]. Using a Naïve Bayes model for text analysis is very popular in NLP, namely in spam detection [2], [3], document classification [4] or for text categorization [1]. A reason for its popularity is that texts are generally represented using a "bag-of-words" approach, where the order of the words is ignored and each word occurring in the text constitute its features. Since the number of different words can quickly become very large, the feature space increases rapidly. Hence, the algorithms must be able to handle such high-dimensional data in terms of classification performance and computational speed [1].
Therefore, we decided to create a multinomial Naïve Bayes (MNB) model for predicting the newspaper source according the words used in their headlines. We assume that a MNB can distinguish between both sources and can make a clear prediction of the news headlines source. The only concern we had was that headlines from both sources were collected in the same time period. Given that they write about the same news and same words are used, it could have become difficult to make a classification based on those word frequencies. In addition, future news could contain new words which can not be used for classification as they were not present in the training set.

Dataset

In this project we are using data which we have collected on our own. The news headlines (titles and teasers) of the title page were scraped twice a day (6a.m. & 6p.m.) from the online news sites 20min.ch and nzz.ch using a web-scraping program. The content and structure of the raw data is shown in Figure 1.

```
## # A tibble: 65,322 x 4
##   date_time           object order text
##   <dttm>              <chr>  <dbl> <chr>
## 1 2020-01-21 05:00:14 title      1 die konzernlenker sind so pessimistisch wie ~
## 2 2020-01-21 05:00:14 title      2 wef das wohl wichtigste treffen für die schw~
## 3 2020-01-21 05:00:14 title      3 die absperrungen sind aufgestellt die geschä~
## 4 2020-01-21 05:00:14 title      4 präzisionsschützen und ein sprengstoffkomman~
## 5 2020-01-21 05:00:14 title      5 die luanda leaks bieten einblicke in ein kap~
## # ... with 65,317 more rows
```

Figure 1: Raw data of a newspaper source. The data contains a date stamp, an object class (title or teaser), the order in the collection and its containing text (lowercase and without punctuation marks).

Methods

The first step before feeding the model with our data, was the data pre-processing. As the pre-processing is not the main part of the project, only some main concepts are described. Only unique headlines have been considered and the software R has been used. First, the text was tokenized. Tokenization is the process of turning a sentence into a string of characters, referred as a token,

which are the building blocks of natural language [5]. In our case a token is one word (Figure 2). Second, the stop words have been removed, as they usually don't contribute to the classification success. Examples of German stop words are "also" or "und" (a detailed list of used German stop words for this project can be found online [6]). Finally, a document-term matrix has been built. This keeps track of how often a certain word occurs in a headline and converts each text into a numeric vector. This step is called vectorization and is often referred as the "bag-of-words" [4] (Figure 2).

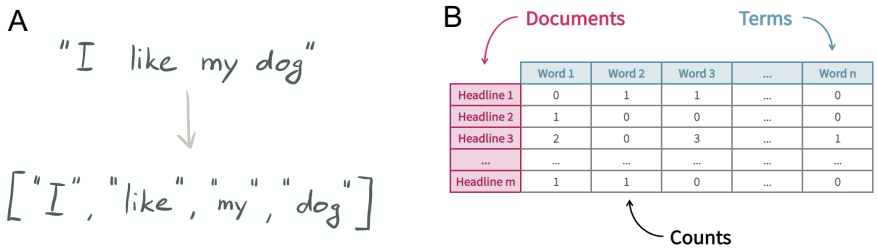

***Figure 2**: **(A)** Shows the process of tokenization, how a sentence is turned into a sequence of tokens (= words); **(B)** Visualization of the Document-Term matrix or also called Bag-of-Words.*

The next step is the data visualisation to get further insights in the data. We have decided to present two different data distributions. A histogram with the headline length distribution shows that the number of unique titles is comparable between the two sources, while there are more teasers available from the 20min title page. It can also be seen that NZZ headlines are typically a bit longer and with a larger spread compared to 20min headlines (Figure 3).

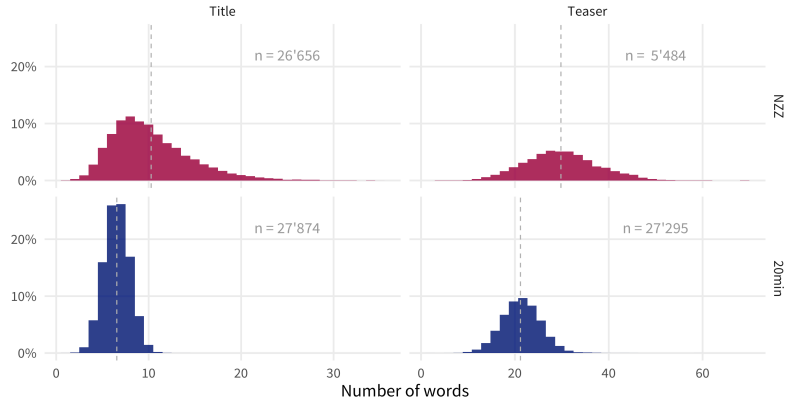

***Figure 3: Headline length distribution of NZZ and 20min.** The number of words (x-axis) is shown for the headlines (titles & teasers) for both sources. The y-axis percentage shows the proportion of the overall word distribution. The amount of each title and teaser is presented in each graph (n).*

The Naïve Bayes classifier is based on the independence assumptions between predictors. This means, the data must be conditionally independent given class, referred as *class-conditionally independent*, and for each class we assume independent features. Therefore, we additionally visualized the proportion differences, where we compared the word frequencies of both sources. The chart shows, that the most frequent words occur at different rates in the headlines of both sources – showing a class-conditional independence. This is necessary that a MNB can classify the headlines based on the words used (Figure 4).

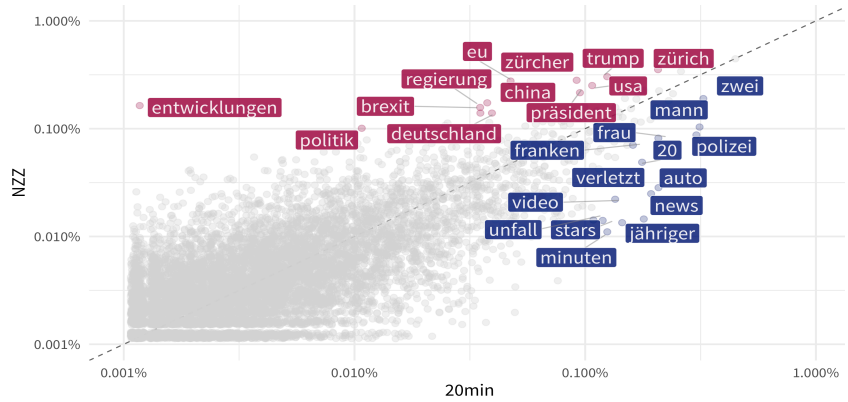

Figure 4: Proportion differences reveal the most frequent words in **NZZ** or **20min** headlines. The most frequent words are highlighted red for NZZ and blue for 20min. Percentages are the word occurrence divided by total number of words.

In our context, the MNB vectorization uses the probability of a particular document (= Headlines) being annotated to a particular category (= Newspaper source), given that the Headline contains certain words. This is equal to the probability of finding those particular words in the source (= Likelihood), multiplied the probability that any headline is annotated to that category (= Priors), divided by the probability of finding those words in any headline (= Evidence). As the evidence is constant for a given data, it can be ignored [2]. The prior probabilities can be calculated using the equation (1):

$$\boldsymbol{Pr(S)} = \frac{\boldsymbol{Total\ Number\ of\ Headlines\ in\ Source}}{\boldsymbol{Total\ Number\ of\ Headlines\ in\ Training\ Dataset}} \quad (1)$$

The class conditional probabilities are calculated using the equation (2). To avoid zero probabilities in models with many features we increased the count by a value of 1. This is also referred to the *Laplace smoothing* [2], [4]. This is done to ensure that each word has a probability of occurrence based on at least a single count even if it does not appear in the training data:

$$\boldsymbol{Pr(H|S)} = \frac{\boldsymbol{Occurance\ of\ Word\ in\ the\ Source + 1}}{\boldsymbol{Total\ Number\ of\ all\ Words\ in\ Source + 1}} \quad (2)$$

According to the Bayes Formula we can now calculate the posterior probability *Pr(Source|Headline)* of each word in the input Headline annotated to each source. Since the numerical values of probabilities of words are very small, multiplying all these probabilities will end up in a so-called underflow which eventually will fail to predict the source. To avoid this underflow the *log* is applied leading to the equation (3) for calculating the overall probability of a headline belonging to a source [4]:

$$\boldsymbol{Pr(S|H)} = \boldsymbol{log(P(H|S))} + \boldsymbol{log(S)} \quad (3)$$

The right source will then be chosen by the highest posterior probability value *Pr(S|H)* as stated in the Bayes Classification rule [4].

The vectorized data was split into 75% Training- and 25% Testing set. To evaluate our model, a 10-fold cross-validation of the training set was performed.

Results

To find the best model parameters the hyperparameters were tested against each other, namely the Number of Headlines, Number of words in vocabulary, both with – and without stop words. They were tested by cross-validation and the mean accuracy was recorded as a measure of quality. The results after cross-validation shows a maximum overall accuracy of 81% for the validation set and that a higher accuracy is reached the more headlines and the more words in the vocabulary are used in the training set (Figure 5).

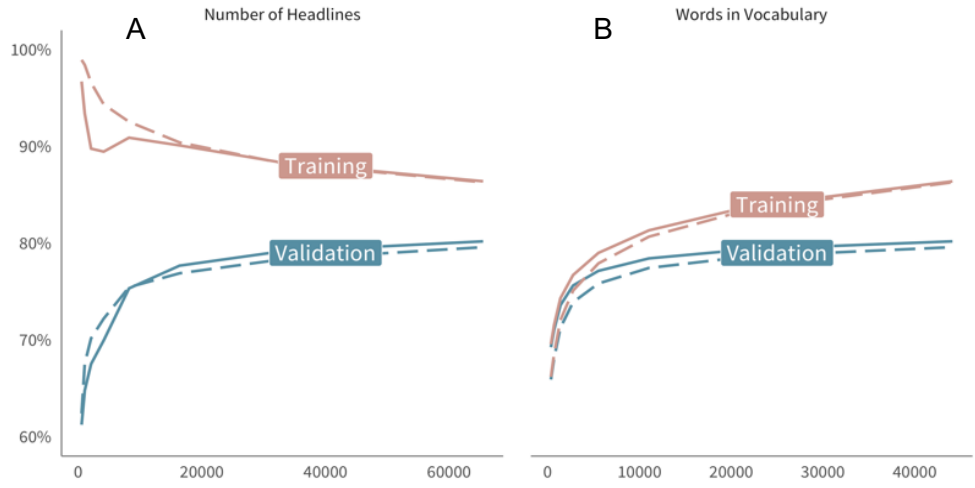

Figure 5: Accuracy after a 10-fold cross-validation. The dashed lines represent the data where the stop words are removed. The solid lines represent the data where the stop words are included. In (A) the number of Headlines were increased while the number of words in the vocabulary was fixed at its maximum. In (B) the number of words in the vocabulary was increased and the number of headlines were fixed at its maximum.

Computing the Confusion matrix gives us a holistic view of how well our classification model is performing and what kinds of errors it is making. From this confusion matrix of the test set (25% of the data) we see that there is no clear bias towards one group (Figure 6). The false-positive and the false-negative having nearly the same value (9.5%, 9.6% respectively).

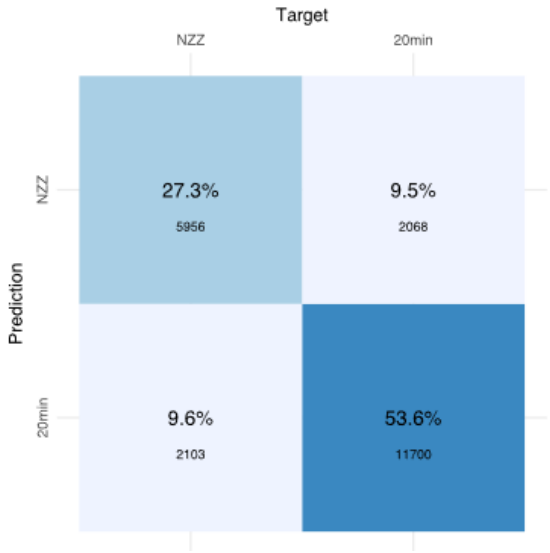

Figure 6: Confusion matrix of the test set shows no clear bias into one direction of the data, as the false-positive (9.5%) and the false-negative (9.6%) values are in the same range.

Discussion

From the results of accuracy after the 10-fold cross-validation (Figure 5) we conclude that it seems to be the best approach to input as much data as it is at hand, if the highest accuracy is requested. It has to be noted though, that for most classification problems maximizing the accuracy is not the main goal. The gains of accuracy is not very high above a certain threshold (around >20'000 headlines and >10'000 words). Therefore, one could be interested in reducing computational resources by reducing the number of input headlines and words.
On the other hand, we are working with time series data when predicting the source of news headlines. Future news might contain new words which cannot be used for classification as they were not present in the training set. In this particular case it could be advantageous to include as much data during possibly a long period of time. As we scraped the data over a period of one year, we had much data available and therefore the classification of the newspaper source was eventually very successful with an overall accuracy of 81% on the never seen test data (25%). Generally, it seems that removing stop words does not improve the test accuracy. In contrary, not removing the stop words resulted in a higher accuracy. This effect may also be explained by the fact that the more data we put in the model, will result in a higher accuracy in turn of using more computational resources. Our model performs well if we compare it to literature where MNB was applied in spam detection. Those models reach accuracies between 80-90% depending on the dataset used [1].

MNB is frequently employed in text classification problems, because it shows good performance, it is computationally very efficient and it is easy to implement, given that it just takes accuracies into account [1], [3]. But the order of words are not considered. Therefore, one could also think of using more sophisticated tools. An improved MNB is presented by Kibriya *et. al.* (2004) by applying a TF-IDF (term frequency–inverse document frequency) transformation to the words count of a document, before applying the learning algorithm. This method lead in an increased accuracy of 1-3% with a final accuracy between 83-93% [1]. Another very popular method in text classification is SVM (Support Vector Machine). In combination with Naïve Bayes it has been shown that Naïve Bayes-SVM even outperforms the TF-IDF method [4]. An innovative classification method was demonstrated by Kadam *et. al.* (2018). They designed an algorithm for spam detection which combines MNB with word embedding and reached 98.3% accuracy. As a result, features closer to each other in the vector space were given more weighting during testing [2]. These methods are just a few suggestions on how our MNB classifier proposed in this project can be optimized to achieve higher accuracies with a reasonable computational complexity.

Bibliography

[1] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, “Multinomial naive bayes for text categorization revisited,” *Lect. Notes Artif. Intell. (Subseries Lect. Notes Comput. Sci.*, vol. 3339, pp. 488–499, 2004, doi: 10.1007/978-3-540-30549-1_43.

[2] S. Kadam, “Bayes Algorithm for Spam Filtering,” *2018 Fourth Int. Conf. Comput. Commun. Control Autom.*, pp. 1–5, 2018.

[3] J. J. Eberhardt and J. Eberhardt, “Scholarly Horizons : University of Minnesota , Morris Bayesian Spam Detection Bayesian Spam Detection,” vol. 2, no. 1, 2015.

[4] H. T. Sueno, B. D. Gerardo, and R. P. Medina, “Multi-class document classification using support vector machine (SVM) based on improved naïve bayes vectorization technique,” *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 3, pp. 3937–3944, 2020, doi: 10.30534/ijatcse/2020/216932020.

[5] A. S. Thanuja Nishadi, “Text Analysis: Naïve Bayes Algorithm using Python JupyterLab,” *Int. J. Sci. Res. Publ.*, vol. 9, no. 11, p. p9515, 2019, doi: 10.29322/ijsrp.9.11.2019.p9515.

[6] Solariz, “German Stopwords,” *github.com*, 2017. https://github.com/solariz/german_stopwords/blob/master/german_stopwords_plain.txt.