Assignment 1: Naïve Bayes Classifier
Bayesian Learning

Antonio Gañán Mora
100438082@alumnos.uc3m.es

Content customization is a topic of key interest for companies. The ability to show information according to the likes of each individual allows companies to create better marketing strategies and to convince users to keep using their applications. For this to work, it is needed to analyze what is the content about, that is, to classify it into some group or groups. Sports is a topic in which there are naturally well-defined classes —each type of sport— to which an individual might be attracted to or not. In this assignment we consider a dataset composed headlines of posts on Reddit and try to classify them into the correct sport each headline belongs to using a Naïve Bayes classifier.

The report is organized as follows. In Sec. 1 we explain how the data was acquired. In Sec. 2 we comment the process of data cleansing and we provide a wordcloud visualization of the cleaned data. In Sec. 3, we train and test the Naïve Bayes classifier and analyze the results. Finally, in Sec. 4 we summarize the results and propose future improvements

1 Data Scraping

Reddit is a social network composed of subforums called subreddits, which are typically denoted by `r/subreddit`. On each subreddit, users are allowed to post content related to the topic of the subreddit itself. We consider the subreddits r/formula1, r/nba, r/nfl, and r/soccer, whose contents are implicit in their names. From each of the four subreddit, we scrape the first 1250 most rated posts of the last year. Then we select the title of the post and the *self-text*, which is a kind of subtitle the poster submits. The self-text is optional and can be substituted by multimedia content, like a video. As this is a text classifier, we do not consider multimedia content. We do not consider either the comments of each post because to see the comments one first to click on the post itself, and, therefore, the user first interacts with the title and self-text (and multimedia content), and later on with the comments.

The Reddit's API is obscure, but, fortunately, writing `.json` right after the URL returns the content of the first 25 posts —a *page*— of the subreddit in `json` format. To obtain the content of the first 1250 (25×50) posts, we iterate the procedure of requesting the web-page content for each page and select the titles and self-texts. The difficulty is to keep track of the last post of the page. To solve this issue, Reddit provides a `json` attribute which, given a page, keeps track of the next page. This attribute must be passed to the next request for the next page. We repeat this process for the four subreddits considered [1].

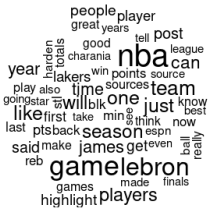
[1] The explanations of the `.json` trick and the Python code for scraping Reddit content are mostly based on the Temple Moore's Reddit scrape function.

2 Data Cleansing and Visualization

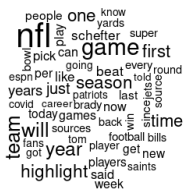
Having scraped the data from Reddit and stored it into files, we now proceed to clean it using R. First, we load the data and merge the four files into a single dataset. The rows of the dataset have two columns: the first one is a tag for the sport and the second one is a line of the input files. Notice that if the self-text is divided into paragraphs, then each paragraph is stored as a line. We do not force to keep the title and its self-text as a single line in order to have strings of approximately the same length. Also, a line can have no text (separating paragraphs with an empty line). Furthermore, lines usually combine uppercase and lowercase, have possession apostrophes ('s), artifacts like `&amp` and `#x200B`, URLs, symbols, stopwords or emoticons. All these problems have been tackled by cleaning the data.

Having cleaned the data, the most frequent words can be seen in Figure 1. Observe that the most common words are those referring to sportsmen or teams and technical word of each sport, like *hamilton* and *race* in F1 or *lebron* and *points* in NBA. Still some stopwords such as *like* or *going* appear as common words. A detailed cleaning of such words could be very time-consuming and we assume that some undesirable words can appear in our corpus. Also, from minimum frequency of the wordclouds, we can observe that lines contain different number of words depending on the subreddit. Indeed, a glimpse to the dataset (or to the subreddits themselves) confirm that lines are shorter in r/formula1 and longer in r/nba. Also, in r/nba there are more self-texts than in r/formula1, and the self-texts of r/nba usually have some statistical analysis of basketball players, thus explaining word like *stl* (steals) or *blk* (blocks).

(a) F1 (min freq = 35).



(b) NBA (min freq = 90).



(c) NFL (min freq = 50).

(d) Soccer (min freq = 60).

Figure 1: Wordclouds for all sports. sportsmen names and technical words are the most common and distinguishable words.

3 Naïve Bayes Classifier

Naïve Bayes classifier is a simple model for predicting which class does a given piece of data belongs to. Let $C = \{C_1, \cdots, C_c\}$ be the classes that the data can be classified into (here, the sports), and let $x_1, \cdots, x_n$ be the features of the problem (here, the occurrences of words of in a line). We want to obtain the posterior probability $P(C_k|x_1, \cdots, x_n)$. By applying Bayes theorem,

$$P(C_k|x_1, \cdots, x_n) = \frac{P(x_1, \cdots, x_n|C_k)P(C_k)}{P(x_1, \cdots, x_n)}.$$

The *naïve* assumption is the conditional independence of x_i with x_j given C_k, that is, $P(x_1, \cdots, x_n|C_k)$ factorizes into $P(x_1, \cdots, x_n|C_k) = \prod_{i=1}^{n} P(x_i|C_k)$. Given this assumption, the classification rule assigns a label $\hat{y}$ such that

$$\hat{y} = \arg \max_{k \in 1, \cdots, C} P(C_k) \prod_{i=1}^{n} P(x_i|C_k).$$

The distribution of the number of appearances of each type of sport follows a multinomial distribution with some weights [2]. Also the word count on each class follows a multinomial distribution. From a frequentist perspective, the weights are parameters estimated from the proportion of occurrences. With this approach, if a given class (sport) and feature value (word) never occur together in the training data, then the posterior is automatically set to zero. To avoid these zero-probability problems, a Bayesian perspective can be applied. Given that the likelihoods follow a multinomial distribution, a Bayesian approach would consider a Dirichlet distribution as prior distribution for the distribution of weights (which are considered as random variables from a Bayesian perspective). It can be shown that a Dirichlet distribution as a prior returns a Dirichlet distribution as posterior when the likelihoods follow a multinomial distribution (it is said that the Dirichlet distribution is a conjugate prior for the multinomial distribution).

The Dirichlet prior distribution has a set of parameters and are fixed in advanced based on the knowledge of the problem. Setting all these parameters equal to one returns a uniform distribution, a particular case of a Dirichlet distribution. This is called Laplacian smoothing, as it causes that the mean of the weight distributions —a point estimate for the weights— to avoid zero-probability problems adding *pseudo-observations*. In general, setting different parameters for the prior distribution results in different posterior Dirichlet distributions [3].

3.1 Results and Analysis

Before training the classifier, we first randomly shuffle our corpus of words (as they are ordered by sport), then consider only words with at least 5 appearances to avoid misspellings or non-cleaned rare words, and, lastly, consider the 75% of data for training. The tests for classification will be done in the other 25% of data.

The `naiveBayes` function of the `e1071` is a simple implementation for computing Naïve Bayes classification with or without smoothing. The smoothing is passed as the

[2] This short discussion is based on this StackExchange question and links therein.

[3] Hyper-parameter tuning should be considered to select the optimal parameters. Unfortunately, trying some values for the parameters already takes a long time, and hyper-parameter tuning is out of the scope of this assignment.

`laplace` argument to the function: `laplace` equals to zero means no smoothing (frequentist); `laplace` greater than zero means smoothing (frequentist), and if it equals to one, then we apply Laplacian smoothing. In Table 1 it can be seen the three confusion tables for three different values of `laplace`, 0, 0.1, and 1 [4]. Notice that Laplacian smoothing (Table Table 1c) notably worsens predictions for NBA and soccer. Also, observe that for Table 1a and Table 1b most of the errors are caused by miss-classifications into NBA. Let us further elaborate on this point.

Table 1
Confusion tables for Naïve Bayes classifier on test partition.

	F1	NBA	NFL	Soccer
F1	310	33	16	17
NBA	25	1232	100	91
NFL	3	68	323	20
Soccer	10	16	15	506

(a) `laplace` = 0.

	F1	NBA	NFL	Soccer
F1	304	30	11	14
NBA	30	1270	73	79
NFL	3	35	350	17
Soccer	11	14	20	524

(b) `laplace` = 0.1.

	F1	NBA	NFL	Soccer
F1	340	540	88	126
NBA	1	680	12	4
NFL	2	113	342	23
Soccer	5	16	12	481

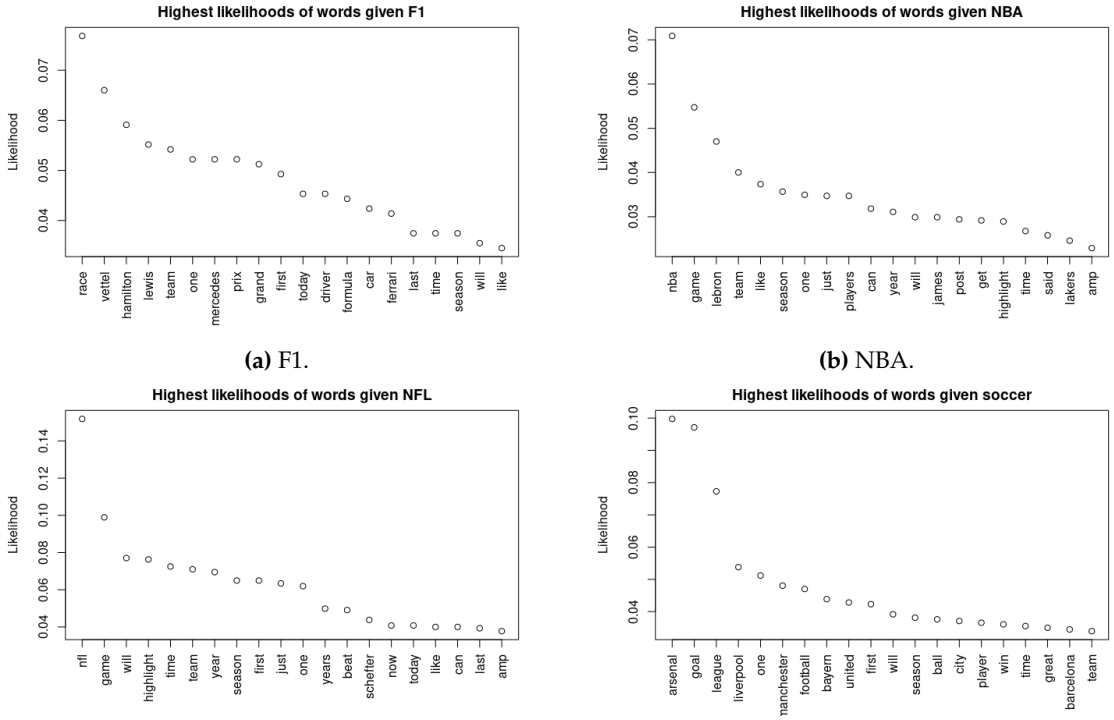
(c) `laplace` = 1.

For simplicity, we will focus on the classifier with no smoothing. Having a large posterior probability depends on having large prior probabilities and likelihoods. Values proportional to prior probabilities can be accessed via `$apriori` on a `naiveBayes` object. It can be seen that NBA (4111) more than doubles soccer values (1914) and almost quadruples NFL and F1 (1315 and 1012, resp.). We can also access the likelihoods for each word via `$table`. For each sport, we select the set of 250 words with highest likelihood among the 3345 total words and compute the intersection with the other sets of each sport. The results are shown in Table 2 and a graphic representation of the 25 words with highests likelihoods are shown in Figure 2. We can observe that in general a notable number of words appear in several sports. Also, we observe that NBA∩NFL and NBA∩Soccer are larger than the rest, which are the precisely the two greatest miss-classifications entries of Table 1a. Thus, we can understand the source of confusion of the classifier.

Table 2
Intersection of the sets of the 250 words with highest likelihoods for each sport.

	F1	NBA	NFL	Soccer
F1	-	83	76	82
NBA	-	-	117	104
NFL	-	-	-	91
Soccer	-	-	-	-

[4] `laplace` equal 4 has also been computed, but the results are quite terrible.



(a) F1.

(b) NBA.

(c) NFL.

(d) Soccer.

Figure 2: Likelihoods of 25 most common words conditioning on each sport. Notice that some words appear in several plots, like *will*.

Finally, we can compute the accuracies of the two best models. We define the global accuracy of the model as the sum of the diagonal of the confusion matrix divided by the sum of all the entries of the matrix. Given the column corresponding to a sport in the confusion matrix, we define the sport accuracy as the quotient of the diagonal entry divided by the sum of elements of the column. The results can be seen in Table 3. From the table we observe that the model with smoothing can predict better in all cases except for F1, where it is a bit worse. We also observe that NFL and Soccer are the two worst classified sports, as rationalized before. NBA accuracies are above 91% in both cases and global accuracies between 85 and 90%.

Table 3
Accuracy of models of Table 1a and Table 1b.

	F1	NBA	NFL	Soccer	Global
`laplace` = 0	0.891	0.913	0.711	0.798	0.851
`laplace` = 0.1	0.874	0.941	0.771	0.826	0.879

4 Conclusions

In general, we observe that the simple Naïve Bayes classifier, with the strong assumption of conditional independence, can achieve reasonable good classifications of posts. We noted that, in our particular case, the classifier gets confused by the large priors probabilities for NBA and the large amount of words with high likelihood both for NBA and other sports. Therefore, in order to improve our current classifier, we should consider the context in which these words appear, because it would allow us to change the likelihoods. By

doing this, we would no longer be under the naive assumption as we would consider correlations between words. Clearly, this model would be more complex in comparison to Naïve Bayes, but we would expect to improve the results.