Practical Machine Learning

Supervised Learning Algorithms
for Text Classification

Ioana Chitic

December 6, 2019

1 Introduction

Given a collection of reviews, can we identify who the author is using supervised learning algorithms for text classification?

I decided to explore the answer to this question by using multinomial naive Bayes and logistic regression. The multinomial naive Bayes algorithm is suitable for classification with discrete features. Logistic regression is a model for classification that outputs the probabilities describing the possible outcomes of a single trial modeled using a logistic function.

I provided a comparison of the two methods, as well as an individual performance analysis.

2 The Dataset

I used a dataset of fine foods reviews from Amazon consisting of approximately 500,000 reviews.

	ProfileName	Text
0	delmartian	I have bought several of the Vitality canned d...
1	dll pa	Product arrived labeled as Jumbo Salted Peanut...
2	Natalia Corres "Natalia Corres"	This is a confection that has been around a fe...
3	Karl	If you are looking for the secret ingredient i...
4	Michael D. Bigham "M. Wassir"	Great taffy at a great price. There was a wid...

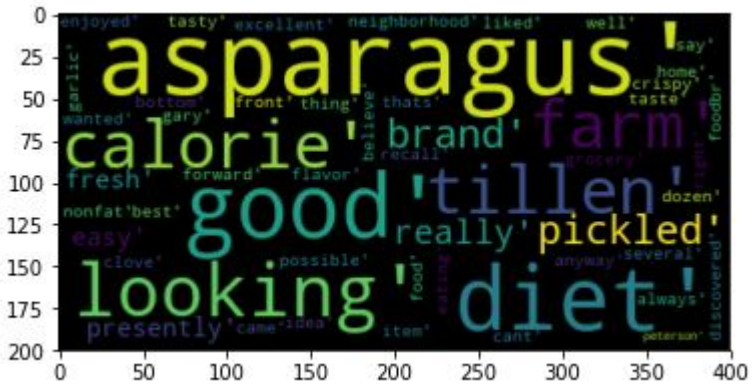
Because the original dataset is so large, I used a subset that contained only reviews made by reviewers who had made more than 200 reviews. I only kept the columns that contained data relevant to the task, namely the profile name of the reviewer, and the text of the review. Afterwards, I processed both the profile names and the review text. The profile names would no longer contain punctuation marks or capital letters, so that they would be uniform. The review text would no longer contain punctuation marks, capital letters, words shorter than length 4, or stopwords. Additionally, all the words were lemmatized. Lemmatization reduces the word-forms to linguistically valid lemmas.

	ProfileName	Text
0	lynrie oh hell no	['strawberry', 'twizzlers', 'guilty', 'pleasur...
1	jen	['okay', 'would']
2	gary peterson	['presently', 'diet', 'fresh', 'easy', 'neighb...
3	sarah	['best', 'investment', 'ever', 'made', 'ginger...
4	c2	['looking', 'le', 'messy', 'version', 'licoric...
5	chris	['work', 'chicken', 'fish', 'beef', 'pork', 'f...

The resulting dataset contains approximately 6000 rows. I examined the data and obtained 23 unique profile names with more than 200 reviews. This means that for this particular dataset, the classification task has 23 possible classes with the following distribution:

```
c f hill cfh                                451
o brown ms o khannahbrown                   421
gary peterson                               389
rebecca of amazon the rebecca review        365
chris                                       363
linda                                       290
john                                        261
mike                                        260
c2                                          256
laura                                       253
sarah                                       244
stephanie                                   240
karen                                       239
lisa                                        237
jessica                                     236
jen                                         232
steve                                       225
david                                       216
anonymous                                   213
mary                                        208
gunner                                      207
amanda                                      202
lynrie oh hell no                           201
```

Here is a wordcloud visualization of one of the reviews:



3 Preprocessing the Dataset

Since I need numeric data for my machine learning algorithm, I have to convert text into numbers. In order to achieve this, I used the ”Bag of Words” method available in Scikit-Learn. The theory behind this method consists of:

- Splitting the reviews into tokens (or separate words)
- Creating a dictionary that contains all the words in our reviews and the number of times they appear in our reviews
- Sorting the word frequency dictionary in descending order
- Creating the ”Bag of Words” model for which we need a matrix in which the columns correspond to the words in our ordered dictionary, and the rows correspond to the reviews
- Every element of the matrix can be either 0, if the word doesn’t appear in the sentence, or 1 if it does

This numeric representation can be used as input for our statistical model. I split the data into training data and test data, using an 80/20 split, and used ”Bag of Words” on each.

4 Multinomial Naive Bayes

The naive Bayes classifiers are a family of probabilistic algorithms that apply Bayes’ theorem with the assumption of conditional independence between features. The multinomial naive Bayes computes class probabilities for a given review. If the set of classes is denoted by C, and N is the size of our vocabulary, then our algorithm assigns a review

r to the class that has the highest probability $Pr(c \mid r)$, which, using Bayes' theorem, is given by:

$$Pr(c \mid r) = \frac{Pr(c)Pr(r \mid c)}{Pr(r)}, c \in C$$

The class prior probability $Pr(c)$ can be estimated by dividing the number of reviews belonging to a class c by the total number of reviews. $Pr(r \mid c)$ is the probability of obtaining a review like r in class c and is calculated as:

$$Pr(r \mid c) = \prod_n Pr(w_n \mid c)^{count_n}$$

where $Pr(w_n \mid c)$ is the probability of a word n belonging to class c, and $count_n$ is the count of word n in our review r. The probability $Pr(w_n \mid c)$ can be calculated as:

$$Pr(w_n \mid c) = \frac{1 + Count_{nc}}{N + \sum_{i=1}^{N} Count_{ic}}$$

where $Count_{nc}$ is the count of word n in all the training documents belonging to class c.

I instantiated a model with multinomial naive Bayes from Scikit-Learn, and trained the model. The training accuracy was:

$$0.8069257096839139$$

and the validation accuracy was:

$$0.604669887278583$$

Using the $\%time$ IPython magic command, I timed the Multinomial Naive Bayes classifier at $22ms$.

5 Logistic Regression

Logistic regression is a linear model for classification. The probabilities describing the possible outcomes of a single trial are modeled using a logistic function. The implementation available in Scikit-Learn can also be used for multinomial logistic regression (also known as softmax regression).

The multinomial logistic classifier uses a generalization of the sigmoid, called the softmax function, to compute the probability that a review r belongs to a class c. The sofrmax function takes a vector $z = [z_1, z_2, ..., z_k]$ of k arbitrary values and maps them to a probability distribution, with each value in the range $(0, 1)$, and all the values summing to 1:

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} 1 \leq i \leq k$$

I instantiated a model with logistic regression from Scikit-Learn, and trained the model. The training accuracy was:

$$0.9985906986108315$$

and the validation accuracy was:

$$0.6626409017713365$$

Using the $\%time$ IPython magic command, I timed logistic regression at $2.95s$.

6 Comparison of the two methods

6.1 Classification Report

I created a classification report for my predictions, also available in Scikit-Learn. The report keeps track of true and false positives, and true and false negatives, and can give a more detailed overview of each particular class. The report's parameters are:

- precision - the ability of a classifier not to label a negative as a positie (what percentage is correctly classified?)
- recall - the ability of a classifier to find all true positives and false negatives
- F_1 score - the weighted harmonic mean of precision and recall, such that $F_1 \in [0.0, 1.0]$
- support - the number of actual occurrences of the class in the specified dataset

	precision	recall	f1-score	support
0	0.79	0.48	0.59	46
1	0.52	0.28	0.36	43
2	0.64	0.95	0.77	82
3	0.54	1.00	0.70	52
4	0.30	0.52	0.38	60
5	0.67	0.32	0.44	37
6	0.43	1.00	0.61	76
7	1.00	0.90	0.95	41
8	0.57	0.44	0.50	45
9	0.53	0.39	0.45	49
10	0.70	0.44	0.54	52
11	0.56	0.27	0.36	37
12	0.54	0.49	0.52	51
13	0.86	0.39	0.53	62
14	0.78	0.35	0.48	40
15	0.80	0.80	0.80	46
16	0.31	0.13	0.18	39
17	0.70	0.44	0.54	64
18	0.82	0.98	0.89	91
19	0.62	0.97	0.76	80
20	0.45	0.26	0.33	50
21	0.89	0.54	0.67	57
22	0.56	0.36	0.43	42
accuracy			0.60	1242
macro avg	0.63	0.55	0.56	1242
weighted avg	0.64	0.60	0.58	1242

(a) Bayes report

	precision	recall	f1-score	support
0	0.62	0.50	0.55	46
1	0.41	0.30	0.35	43
2	1.00	0.98	0.99	82
3	0.96	0.98	0.97	52
4	0.43	0.55	0.49	60
5	0.54	0.38	0.44	37
6	1.00	1.00	1.00	76
7	1.00	0.95	0.97	41
8	0.35	0.42	0.38	45
9	0.49	0.49	0.49	49
10	0.48	0.46	0.47	52
11	0.35	0.35	0.35	37
12	0.61	0.59	0.60	51
13	0.54	0.60	0.56	62
14	0.33	0.50	0.40	40
15	0.97	0.63	0.76	46
16	0.35	0.31	0.33	39
17	0.52	0.53	0.52	64
18	1.00	0.98	0.99	91
19	0.98	0.99	0.98	80
20	0.44	0.54	0.48	50
21	0.79	0.72	0.75	57
22	0.41	0.38	0.40	42
accuracy			0.66	1242
macro avg	0.63	0.61	0.62	1242
weighted avg	0.68	0.66	0.67	1242

(b) Logistic regression report

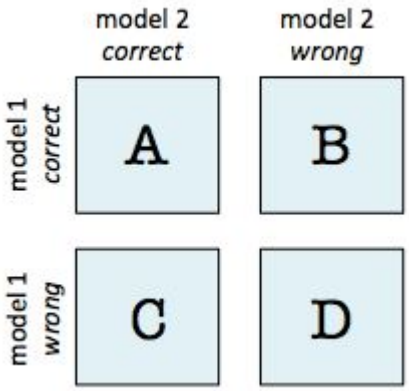
Another important aspect is the timing of the two methods. Naive Bayes is very fast in comparison to logistic regression. However, logistic regression had an overall better training and validation accuracy.

Both methods provided a validation accuracy of approximately 60%. One way to increase this accuracy is to further improve data quality. I have already taken some steps (lowercase words, lemmatization, stopword removal), but more could be done, such as correcting spelling mistakes, which were present in many of the comments, and parts of speech tagging (POS), which retain important parts of speech such as nouns, and look at their overall importance.

Another aspect which can be improved is the validation step. Here, I used a simple 80/20 split between train and test data. Using a cross validation method would likely improve my models' predictions.

6.2 McNemar's Test

Instead of using a confusion matrix, which could have been difficult to read because we have 23 classes, I decided to use a different approach. The McNemar test is a statistical test that can be applied to compare the performance of two machine learning classifiers. It is a version of the 2x2 confusion matrix that compares the predictions of two models to each other.

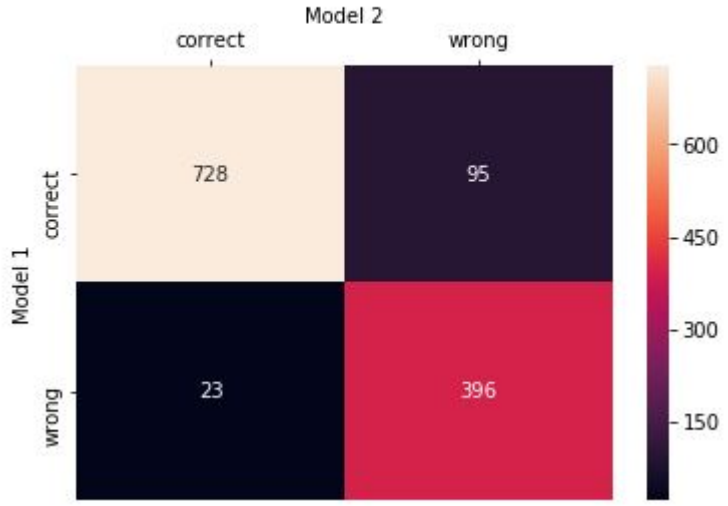


The null hypothesis of the test is that the probabilities for each outcome are the same, or that none of the two models performs better than the other. The McNemar test statistic can pe computed as follows:

$$\chi^2 = \frac{(b-c)^2}{(b+c)}$$

χ^2 has a chi-squared distribution with one degree of freedom. After setting a significance threshold, we can compute the p-value. If the p-value is lower than our chosen significance level α (for instance, $\alpha = 0.001$), then we can reject the test's hypothesis and conclude that the performance of the two models is not equal.

```
chi-squared: 42.720338983050844
p-value: 6.315223437896285e-11
```



Because χ^2 has the value 42.72, it is unlikely to form the distribution implied by the null hypothesis ($p < \alpha$). Therefore, the test provides strong evidence to reject the null hypothesis of equal model performance. Considering the results from the classification report as well, we can conclude that logistic regression is better than multinomial naive Bayes for this task.

7 Conclusion

Multinomial naive Bayes and logistic regression are supervised machine learning algorithms. I used these algorithms for the task of text classification on a dataset containing Amazon reviews of fine food products. I wanted to determine if an unseen review can be correctly attributed to an author. After keeping only a subset of reviewers who had made more than 200 reviews, and applying the usual natural language preprocessing steps of removing punctuation and stopwords and lemmatizing the words, I split the data into train and test using the 80/20 method. I transformed the textual data into numeric values using ”Bag of Words”, keeping train and test data separate. Afterwards, I applied multinomial naive Bayes and logistic regression each to a model, and outputted a classification report for each of the methods applied. I created a McNemar table in order to compare the results of the two classifiers, calculated the χ^2 and p-value, and plotted the table.

Multinomial naive Bayes is a quicker, but less precise classifier than logistic regression for this task. Considering the results of the McNemar test, and keeping in mind that logistic regression takes significantly longer than naive Bayes, we can conclude that logistic regression is the better classifier for this task.