COMP 551 - Mini Project 2

Nicolas Fertout, Luna Dana, Anthony Kumar

February 2022

Abstract

This project involved implementing Naives Bayes and Softmax regression models with a K-fold cross validation to make predictions on data from the 20newsgroup and Sentiment140 datasets, and comparing the performance of both models with each other across both datasets. We found the Softmax model achieved a better performance for 20newsgroups, and Naive Bayes a slightly better average accuracy on the Sentiment140 dataset.

1 Introduction

In this report, we investigate and analyze the performance between the Naive Bayes and Softmax classifiers on the 20newsgroups & Sentiment140 datasets (described in the Datasets section). We use a Naive Bayes and a Softmax regression approach, both of which were implemented in the Python programming language, for classifying new instances of data into one of their respective classes. The Softmax model we used was sklearn's implementation. Our focus was on identifying the most important parameters for the models, as well as the model that yields the highest accuracy for both dataset.

Overall, we found that Softmax gives a better fit (using L2 penalty and liblinear solver) for 20newsgroups and Naives Bayes gives a slightly better fit for Sentiment140. We also discovered that, for Softmax, while the `solver` has only little effect on performance, adding a `penalty` increases the accuracy by 2% to 8% (depending on the penalty and dataset). Moreover, using a list of stop-words to remove some of the most common English words leads to an increase in accuracy of a varying amount for almost all parameters choice. Besides, for the 20newsgroups dataset, we observed that fitting models on the groups rather than subgroups lead to a significant increase in accuracy with both models.

For the Sentiment140 set, a paper published by three Stanford professors and students reported an accuracy of about 81% using Naive Bayes, which is about the same as our results. For 20newsgroup, a study) was carried using tf-idf as text tokenizer and Naive Bayes, and got results of 81.69% accuracy . Those results are larger than ours, but this could be due to the basic vectorization technique we used, and by some different pre-processing methods.

We will start by explaining what our data consists of and how it was cleaned appropriately in order to be used in our models, then we will give some insights on the methods and design choices made during our project, to finish with a presentation of our results and a discussions of our findings.

2 Datasets

2.1 Data Description

Sentiment140

The Sentiment140 set consists of 155 instances of 6 features. The features are the date of the tweet, the id, the query, the user and the text/content of the tweet. The response variable is the sentiment which is a binary variable explaining if the corresponding tweet is either positive=4 or negative=0. It implies that we will fit a binary classification model on this data.

20newsgroups

The 20newsgroups set consists of 20000 instances of 1 feature, a news article. The response variable is a category. There are 20 categories (alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.windows.x, misc.forsale, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey,

sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.m
talk.religion.misc) that can be re-categorize in 6 broader categories using their first prefix (alt, comp, rec, sci, talk, misc). We will fit a multi-class classification model on this data.

2.2 Data Cleaning

In order to analyse the data, we needed the text feature to contain only the most relevant words. For this reason, we removed the words that were contained in a pre-defined list of stopwords. This list is under the data section in the code.zip name stopwords.txt.

Example of stopwords : and, because, don’t, go, if, most, much, my, put, there’s.

2.3 Data Analysis

Key Statistics

Group	Subgroup	#	WC	3 most occuring words (length 4+)
alt	alt.atheism	480	198.544	(’people’, 275), (’believe’, 154), (’atheists’, 131)
comp	comp.graphics	584	180.255	(’image’, 394), (’graphics’, 220), (’file’, 205)
comp	comp.os.ms-windows.misc	591	113.54	(”max¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ax¿’ ax¿’ax¿”, 3289), (’windows’, 464), (’file’, 167)
comp	comp.sys.ibm.pc hardware	590	159.807	(’drive’, 361), (’scsi’, 233), (’disk’, 211)
comp	comp.sys.mac hardware	578	112.6	(’apple’, 189), (’drive’, 130), (’system’, 115)
comp	comp.windows.x	593	255.278	(’file’, 460), (’window’, 457), (’program’, 274)
misc	misc.forsale	585	151.109	(’please’, 154), (’shipping’, 109), (’excellent’, 101)
rec	rec.autos	594	126.034	(’cars’, 123), (’engine’, 82), (’time’, 75)
rec	rec.motorcycles	598	111.771	(’bike’, 227), (’ride’, 90), (’motorcycle’, 84)
rec	rec.sport.baseball	597	141.923	(’team’, 142), (’game’, 114), (’players’, 98)
rec	rec.sport.hockey	600	287.912	(’team’, 337), (’hockey’, 296), (’55.0’, 279)
sci	sci.crypt	595	297.98	(’encryption’, 439), (’government’, 328), (’people’, 317)
sci	sci.electronics	591	141.144	(’power’, 140), (’wire’, 122), (’ground’, 115)
sci	sci.med	594	231.643	(’medical’, 201), (’people’, 181), (’health’, 166),
sci	sci.space	593	232.924	(’space’, 735), (’nasa’, 229), (’launch’, 229)
soc	soc.religion.christian	599	274.773	(’people’, 376), (’jesus’, 314), (’believe’, 269)
talk	talk.politics.guns	546	218.271	(’people’, 321), (’file:’, 258), (’guns’, 158)
talk	talk.politics.mideast	564	363.392	(’people’, 597), (’armenian’, 518), (’turkish’, 434)
talk	talk.politics.misc	465	332.895	(’people’, 409), (’stephanopoulos:’, 341), (’president’, 312)
talk	talk.religion.misc	377	229.565	(’people’, 211), (’jesus’, 181), (’believe’, 105)

Table 1: Analysis of the 20 news group dataset

Sentiment	#	WC	10 most occuring words (length 4+)
Positive	800000	14.781	(’love’, 44579), (’thanks’, 26142), (’time’, 22710), (’hope’, 18287), (’happy’, 18132), (’thank’, 15314), (’night’, 14977), (’nice’, 14010), (’watching’, 13986), (’twitter’, 13634)
Negative	800000	13.984	(’miss’, 30345), (’wish’, 21549), (’feel’, 21487), (’time’, 20954), (’home’, 17634), (’hate’, 16866), (’sorry’, 16190), (’love’, 15562), (”didn’t”, 14605), (’hope’, 14064)

Table 2: Analysis of the Sentiment140 dataset

Concerns

- Resemblance in subgroups : The 20newsgroups analysis shows that some categories have the same most frequent words (soc.religion.christian and alk.religion.misc) which yields higher rates of misclassification.
- Size of the dataset : the high number of rows in the dataset increases the running time which restrict the tuning of many paramaters.

3 Design Choices: Vectorization, Word Filtering, Cross Validation

When computing the Document-term Matrix, we had the choice between several vectorization methods to count word frequency. We chose to use the function `CountVectorizer` from the `sklearn` library to convert a collection of texts into a matrix of token counts. We also ran our models with Bigrams vectorization, which count words by taking all the words and pairs of consecutive words. (Note that we tried using tf-idf as tokenization method, but did not get good enough results, so we decided to stick with the above methods).

Besides, we ran our tests for both models with and without stopwords, to see if that impacted our results.

Finally, in order to tune our parameters for both Naive Bayes and Softmax, we decided to use a k-fold cross validation. This allows us to train and test our data on the train set only, in order to identify the best parameters for each model, before running on the original train and test sets with those parameters.

We used the above methods, as well as several validations for different hyperparameters combinations.

4 Results

4.1 Naive Bayes

When using k-fold cross validation with k = 5, we get the following average accuracies:

20newsgroups

	Normal	bi-gram
Without Stop Words	51.92%	51.48%
With Stop words	72.56%	68.77%

Table 3: Average accuracy for different word filtering and vectorization methods on groups.

	Normal	bi-gram
Without Stop Words	38.74%	38.61%
With Stop words	54.11%	53.84%

Table 4: Average accuracy for different word filtering and vectorization methods subgroups.

Sentiment140

	Normal	bi-gram
Without Stop Words	71.31%	75.48%
With Stop words	77.64%	76.04%

Table 5: Average accuracy for different word filtering and vectorization methods.

4.2 Softmax

When using k-fold cross validation with k = 5, we get the following average accuracies:

20newsgroups

	Normal	bi-gram
Without Stop Words	60.76%	57.55%
With Stop words	63.82%	63.62%

Table 6: Average accuracy for different word filtering and vectorization methods for the groups.

	Normal	bi-gram
Without Stop Words	76.12%	73.48%
With Stop words	76.18%	75.87%

Table 7: Average accuracy for different word filtering and vectorization methods for subgroups.

Using best settings from above (normal tokenization with stop-words), we now keep using k-fold cross validation (on the subgroups only) to identify the best `penalty` and `solver` parameters from the scikit library:

`solver/penalty`	'none'	'l1'	'l2'
'newton-cg'	56.91%	–	63.71%
'lbfgs'	63.44%	–	63.83%
'liblinear'	–	62.52%	**64.61%**

Table 8: Average accuracy on the Softmax classifier for different `penalty` and `solver` values on the 20newsgroups dataset for the subgroups using 5-fold cross validation set.

Sentiment140

	Normal	bi-gram
Without Stop Words	69.64%	52.92%
With Stop words	73.54%	53.33%

Table 9: Average accuracy for different word filtering and vectorization methods

Using best settings from above, we now keep using k-fold cross validation to identify the best `penalty` and `solver` parameters from the scikit library:

`solver/penalty`	'none'	'l1'	'l2'
'newton-cg'	62.31%	–	69.98%
'lbfgs'	63.75%	–	69.52%
'liblinear'	–	67.76%	**70.86%**

Table 10: Average accuracy on the Softmax classifier for different `penalty` and `solver` values on the 20newsgroups dataset for the subgroups using 5-fold cross validation set.

4.3 Final Results

Classifier	20newsgroups	Sentiment140
Naive Bayes	60.4%	**80.2%**
Softmax Regression	**68.1%**	79.8%

Table 11: Comparing the accuracy of the Naive Bayes and Softmax Regression classifiers for both the 20newsgroups and Sentiment140 datasets.

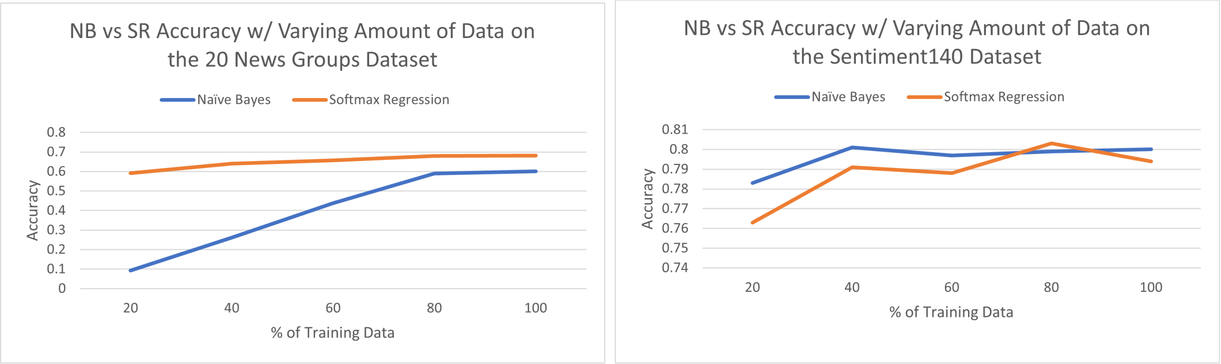

Figure 1: Comparing the accuracy of the Naive Bayes and Softmax Regression classifiers trained on 20%, 40%, 60%, 80% & 100% of the training data.

Futhermore, we can see the impact of removing data from some specific categories below:

Subgroup removed	Subgroup Accuracy	Group Accuracy
talk.religion.misc	69.46%	79.05%
soc.religion.christian	64.95%	76.03%
comp.graphics	64.91%	75.87%

Table 12: Average accuracy per subgroup removed using stop words and the normal vectorizer in decreasing order of accuracy for the Naive Bayes.

5 Discussions and Conclusion

Overall, the Softmax Regression classifier has a better prediction accuracy than the Naive Bayes model on the 20newsgroups dataset by 8%. On the other hand, while performing very similarly, the Naive Bayes classifier has a slightly better prediction accuracy than the Softmax classifier by less than 1%.

In comparing both models' performance on varying the size of the training data for the 20newsgroups dataset, we notice the Naive Bayes model performing significantly better as more training data is given and flat lines at 80%. Overall, for this dataset, the Softmax classifier performs better than the Naive Bayes model for all amounts of training data. On the other hand, for the Sentiment140 dataset, both models get significant performance increases up until the 40% mark and then flatlines. While performing similarly, the Naive Bayes model, overall, outperforms the Softmax classifier.

Besides, we observe that the best settings for both models seem to be using the stop-words list as well as the regular word tokenization. This yields the best performance during validation for the Naive Bayes on the sentiment140 dataset (77.6%), and for the 20newsgroups dataset (54.1% for subgroups, 72.6% for groups), with a total accuracy of 60.4% and 80.2% respectively. Those settings also increased performance for Softmax, in addition to using values of hyperparameters given by the validation step: `penalty:` L2 and `solver:` 'liblinear', yielding a total accuracy of 68.1% and 79.8%.

Overall, the 20newsgroups datasets gave lower accuracies. However, it is worth noting that since the 20newsgroups data contains 20 categories (grouped into 7 major classes), a higher misclassification rate is expected (since there is a probability of 1/20 to get it right at random, against 1/2 for the other dataset). Furthermore, in trying to identify if a specific subgroup was causing several performance issues, we iteratively removed data from each subgroup and observed that removing the data from the group "talk.religion.misc" were causing a substantial increase of 9% with the Naive Bayes model. Removing other groups such as "soc.religion.christian" or "comp.graphics" also led to important increases (of about 4% for subgroups) Besides, we also see that classifying the data within the major groups instead of the subgroups also lead to better results for both models (with an increase of 13%-18% for Naive Bayes and 12%-16% for Softmax on the validation). We can then conclude that a lot of misclassification happens between subgroups within a group.

6 Group Member Contributions

- Everyone: Global planning of the project, discussion of design choices, writing of the report
- Luna Dana: Data cleaning script in Python, data analysis and statistics, running experiments.
- Nicolas Fertout: Implementation of k-fold cross validation, implementation of bigram, running experiments.
- Anthony Kumar: Implementation of Naive Bayes & Softmax (sklearn) classifiers, running experiments.