

Sentiment Analysis

Sentiment analysis of movie reviews using the nltk movie review corpus as training data

Kasper Rahikka, Leif Granfelt | IA - 2 - 013 Text Mining | 15.03.2018

Contents

[The Problem 2](#_Toc508904758)

[Sentiment analysis 2](#_Toc508904759)

[Approach 2](#_Toc508904760)

[Training and Testing Data / Cross Validation 2](#_Toc508904761)

[Different Learning Methods / Classifiers 3](#_Toc508904762)

[Comparing Performance 3](#_Toc508904763)

[Implementation 3](#_Toc508904764)

[Methodology 3](#_Toc508904765)

[Results 3](#_Toc508904766)

# The Problem

By training different classifiers, how accurate is our sentiment analysis when we use NLTK’s movie reviews corpus (Sentiment Polarity Dataset Version 2.0) as a training set and test it against larger datasets of movie reviews, such as Stanford’s Large Movie Review Dataset.

## Sentiment analysis

We computationally identified and categorized opinions expressed in the movie review corpus to determine whether the review was positive or negative towards the movie.

# Approach

## Training and Testing Data / Cross Validation

We trained the classifiers and we ran an initial test against the movie review corpus (95% of dataset as the training set and 5% of the dataset as the test set) to see if there were any classifiers we should purge from the test. In this initial run, we noticed the Support Vector Classifier (SVC) performed poorly in these conditions. The classifier was removed, and we continued with the remaining classifiers.

Once we had removed SVC from the list of classifiers we examined for any bias with the remaining classifiers.

1. Train with both negative and positive data and run a test against only positive data. (Training: 45% positive data, 50% negative data)
2. Train with both negative and positive data and run a test against only negative data. (Training: 50% positive data, 45% negative data)

In this examination all classifiers reported 50% - 80% accuracy in both cases.

## Different Learning Methods / Classifiers

We used seven classifiers:

1. Naïve Bayes
2. Multinomial Naïve Bayes
3. Bernoulli Naïve Bayes
4. Logistic Regression
5. Stochastic Gradient Descent
6. Linear Support Vector Classifier
7. Nu Support Vector Classifier

Lastly, we created our own classifier based on the above classifiers. This classifier takes the mode accuracy of all the classifiers and uses a confidence percentage to report an accuracy based on the previous classifications.

## Comparing Performance

We originally started with eight classifiers and later removed SVC from the list, due to its poor performance. We later added in our own classifier to return to eight classifiers total.

|  |  |  |
| --- | --- | --- |
| **Classifier** | **Accuracy% SLMRD**¹ | **Accuracy% SPD**² |
| Naïve Bayes | 54.604 | 80.000 |
| Multinomial NB | 78.604 | 82.000 |
| Bernoulli NB | 56.088 | 80.000 |
| Logistic Regression | 75.144 | 85.000 |
| Stochastic GD | 74.644 | 87.000 |
| Linear SVC | 73.680 | 84.000 |
| Nu SVC | 76.844 | 85.000 |
| Combined Classifier | 77.120 | 88.000 |

¹SLMRD = Stanford Large Movie Review Dataset

²SPD = Sentiment Polarity Dataset (Using 95% training, 5% testing, randomized training/test data)

Using the combined classifier to classify random parts of the set and get the confidence:

|  |  |  |
| --- | --- | --- |
| Testing set part | Classification | Confidence % |
| testing\_set[0] [0] | pos | 57.142 |
| testing\_set[5] [0] | pos | 85.714 |
| testing\_set[17] [0] | pos | 100 |
| testing\_set[55] [0] | pos | 100 |
| testing\_set[281] [0] | neg | 85.714 |
| testing\_set[1259][0] | pos | 71.428 |

# Implementation

Technologies and libraries used:

1. Anaconda with Python 3
2. Python 3
   1. nltk
      1. corpus.movie\_reviews
      2. classify
   2. pickle
   3. statistics
      1. mode
   4. sklearn
      1. naive\_bayes
      2. linear\_model
      3. svm
      4. metrics
   5. random
   6. os
3. Stanford Large Movie Review Dataset

## Methodology

The Anaconda python file ‘SklearnClassifiers.ipynb’ contains all code in this project.

The classifiers follow the same methodology:

CLASSIFIER\_NAME = SklearnClassifier(NAME\_OF\_CLASSIFIER())

Code for loading an existing training set .pickle save-file for the classifier.

CLASSIFIER\_NAME.train(training\_set)

Call print method to print the accuracy percent for the classifier.

Code for saving the training set to a .pickle save-file.

The classifier data can be saved in files, but we preferred to run the training method each time to make sure the accuracy ratings are reliable. The classifiers take the training data and run it against the test data one after each other, which does clutter the code quite a bit and a for-loop would make the code easier to read.

We ran several tests for each of the classifiers to see their reliability for providing a proper accuracy. Only one classifier failed to do so and was thusly removed from the list of classifiers. The combined classifier was added to provide insight into improvements to the accuracy of the analysis.

The training set was only trained once for each classifier to get a proper insight to their capabilities as classifiers for this problem. Training the classifiers multiple times over would result in a higher accuracy each time, potentially averaging out the classifiers, not providing a proper means to evaluate their differences.

## Results

Out of all classifiers, Bernoulli NB and the original Naïve Bayes classifiers performed the worst, having the lowest accuracy each time. The highest accuracy was received with the combined classifier and the Stochastic classifier.

Each classifier performed well when testing against the SPD testing dataset as the training dataset was from the same dataset.

Bernoulli NB and Naïve Bayes classifiers barely reached 50% in accuracy when tested against the SLMRD dataset. None of the classifiers reached above 80% accuracy against the dataset when trained only once. The Multinomial NB classifier beat the combined classifier in the test against SLMRD, most likely due to the Naïve Bayes and Bernoulli NB classifiers taking down the average with their poor accuracy. If the NB and Bernoulli NB classifiers were removed from the combined classifier, it may well outperform the Multinomial NB classifier in this case.

To improve the accuracy of each classifier, we could use a better training set or tweak the parameters of each classifier and try to adjust them to the problem as best we can.