# Applying Naïve Bayes Classifier in the Natural Language Toolkit to classify recent tweets.

Calvin Canada and Eddy C. Borera

School of Computing and Informatics, Lipscomb University, One University Park Dr., Nashville, TN

## Abstract

Natural Language Processing has been widely used to extract meaningful information from human language that appears in phone dialogues, newspaper stories, emails, product manuals, etc... It has been successfully applied in word editors to check and correct spelling errors, in email applications to detect spams, and in spoken dialogue system to translate spoken words into texts and vice versa. In this research, we combine Natural Language Processing with machine learning techniques to classify tweets as either sad, happy, or neutral. The ability to classify tweets could be considered a useful resource for companies and organizations to track reactions of the mass or specific targeted consumers regarding their products or recently published statements. In our work, we designed feature functions for the Natural Language Toolkit framework, written in Python, and applied them to a Naïve Bayes Classifier to classify recent tweets retrieved from the Twitterarchivist.com website. Randomly selected training data are fed in to the classifier before testing the overall classification model on randomly sampled test tweets. We analyze the classification accuracy results to determine if further modifications are needed in feature functions.

## Introduction

Natural Language Toolkit (NLTK) is a platform that can be used to work with data that deals with managing human language. NLTK contains many classifiers, one of which is the Naive Bayes classifier which is the one that we have implemented. The Naive Bayes classifier is a classifier that was built using the Naïve Bayes algorithm as the cornerstone method of classifying. NLTK has been described as a practical introduction to programming for language processing. Many have used this classifier to perform actions such as sort through Spam, translate languages, and construct sentences in a given language. We combined this wonderful tool with the Python programming language to enable us to interpret and classify human language statements using Twitter data. We start by collecting data which we then divide into a train set and a test set. The train set is used to train the classifier, while the test set is fed to the classifier after this process to determine the accuracy of the our classifier.

## References

[1] http://www-i6.informatik.rwth-aachen.de/~dreuw/latexbeamerposter.php

[2] http://www.ctan.org/tex-archive/macros/latex/contrib/beamer/doc/beameruserguide.pdf

[3] http://www.ctan.org/tex-archive/help/Catalogue/entries/pgf.html

## Introduction

Natural Language Toolkit (NLTK) is a platform that can be used to work with data that deals with managing human language. NLTK contains many classifiers, one of which is the Naive Bayes classifier which is the one that we have implemented. The Naive Bayes classifier is a classifier that was built using the Naïve Bayes algorithm as the cornerstone method of classifying. NLTK has been described as a practical introduction to programming for language processing. Many have used this classifier to perform actions such as sort through Spam, translate languages, and construct sentences in a given language. We combined this wonderful tool with the Python programming language to enable us to interpret and classify human language statements using Twitter data. We start by collecting data which we then divide into a train set and a test set. The train set is used to train the classifier, while the test set is fed to the classifier after this process to determine the accuracy of the our classifier.

## nred

## Naïve Bayes Classifier

### Definition (NBC)

Naïve Bayes Classifier (NBC) is a supervised learning technique that is based on the Bayes' Theorem. A document is classified by taking the most probable class label given a set of feature values (*maximum likelihood*).
Given a set of feature values $(f_1, f_2, \ldots, f_n)$, the likelihood of a document to be in a class label $c$ is computed by the following equation:

$$\begin{aligned} Pr(c \mid f_1, \ldots, f_n) &= \frac{Pr(f_1, \ldots, f_n \mid c) Pr(c)}{Pr(f_1, \ldots, f_n)} \\ &= \frac{Pr(c) \prod_{i=1}^{n} Pr(f_i \mid c)}{Pr(f_1, \ldots, f_n)} \\ &= \frac{Pr(c) \left[ Pr(f_1 \mid c) \times \ldots \times Pr(f_n \mid c) \right]}{Pr(f_1, \ldots, f_n)} \end{aligned}$$

Then, the most likely class is determined by the following equation:

$$\begin{aligned} \operatorname*{argmax}_c Pr(c \mid f_1, \ldots, f_n) &= \operatorname*{argmax}_c \frac{Pr(c) \prod_{i=1}^{n} Pr(f_i \mid c)}{Pr(f_1, \ldots, f_n)} \\ &= \operatorname*{argmax}_c Pr(c) \prod_{i=1}^{n} Pr(f_i \mid c) \end{aligned}$$

## NLTK

Natural Language Toolkit (NLTK) is a platform that can be used to work with data that deals with managing human language. NLTK contains many classifiers, one of which is the Naive Bayes classifier which is the one that we have implemented

## Experiments

## Results

## Conclusion

As you can see it is possible to make your poster very colorfull. But in most cases this will this will overload your poster. If you don't change the color settings you will get the default look, which consits of some shades of the jacobs blue and some decent green highlights. These colors where chosen carefully to keep a consistent look of the poster. The *cpbgposter* style is installed our office computers, so you should be able to compile this example out of the box with pdflatex. If you want to work on your computer make sure that you have a recent TeX distribution (TeXlive 2008, Miktex) and download the beamerthemecpbgposter.sty file from our teamwork page and put it in your local TeX directory.

Hap py
TEX'ing!