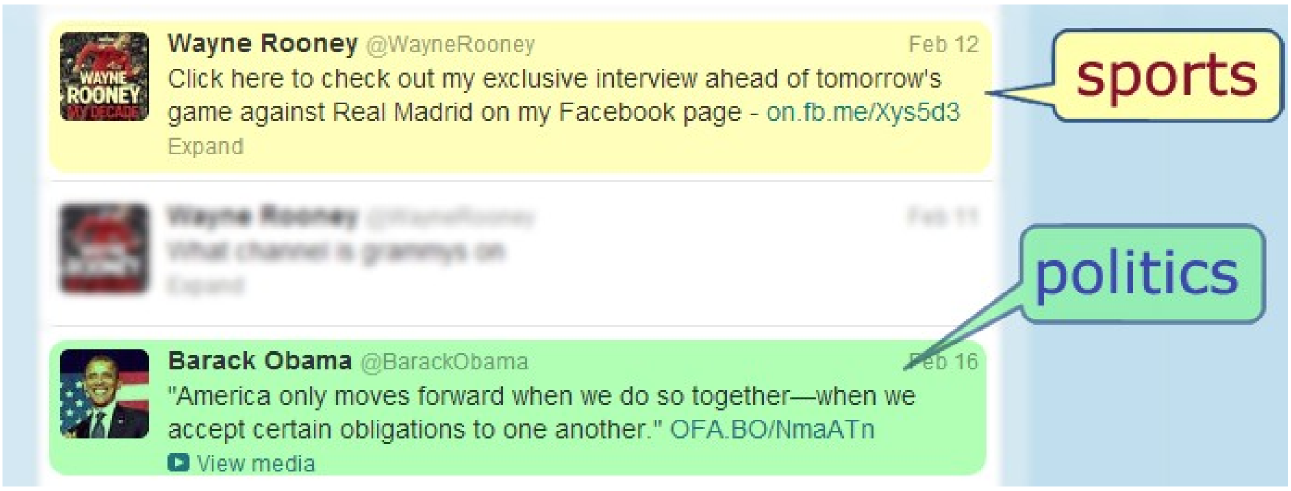
Problem Statement

TwitMiner

The challenge is to predict whether a particular tweet text can be classified to a category of 'Politics' or 'Sports'. This is a simple binary classification supervised learning problem, where you will have a training set consisting of tweettexts and category labels and you have to come up with an algorithm which predicts correct labels for another set of tweettexts.



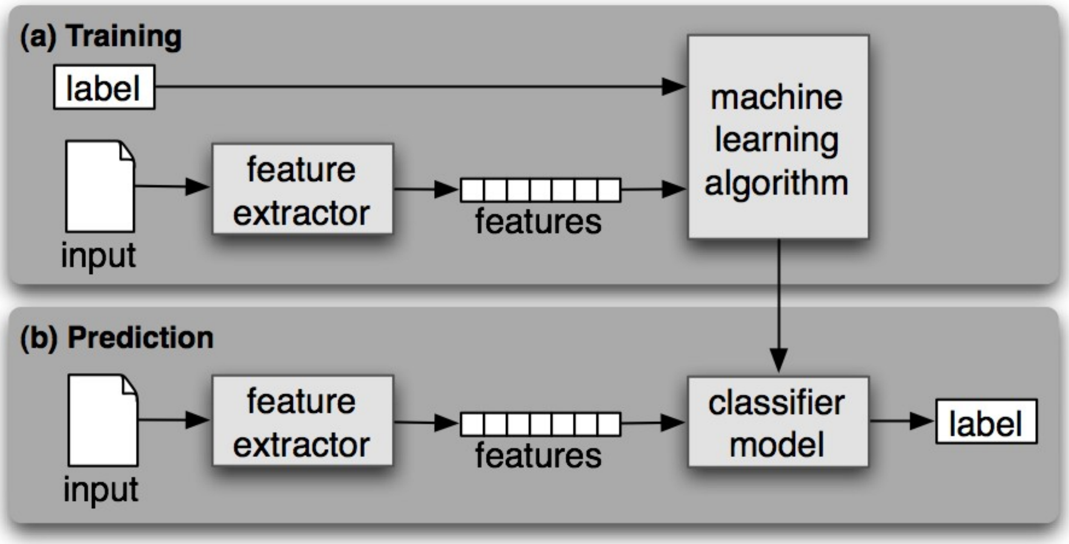
Wayne Rooney @WayneRooney Feb 12
Click here to check out my exclusive interview ahead of tomorrow's game against Real Madrid on my Facebook page - on.fb.me/Xys5d3
Expand

Wayne Rooney @WayneRooney Feb 11
What channel is grammys on
Expand

Barack Obama @BarackObama Feb 16
"America only moves forward when we do so together—when we accept certain obligations to one another." OFA.BO/NmaATn
View media

Background

Machine Learning is one of such tools where people have developed various methods to classify. Classifiers may or may not need data. We propose in this report to use a Supervised Model, wherein the initial phase is Training & then Prediction.



In this problem, we use a ***Naive Bayes Classifier****, a Supervised Model to classify tweets as either 'Politics' or 'Sports'.*

1) Software Used:

Python: *Implemented in Python Programming Language*
scikit-learn *: An Open Source Machine Learning Library for Python Programming Language.*
nltk *: An Open Source Natural Language Processing Library for Python Programming Language.*

2) Features:

The Training set is Enhanced by & the same act as features to the Classifier

- *Replacing URLs by a token*
- *Replacing @user by none*
- *Removed # tags.*
- *Removed all StopWords*

Further a scikit-learn pipeline is used, having:

- *CountVectorizer*
- *TfidfTransformer*
- *MultinomialNB*

3) Classifier:

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions.

In simple terms, a naive Bayes classifier assumes that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable.

Consider the problem of classifying tweets by their content, into Politics and Sports tweets. Imagine that tweets are drawn from a number of classes of tweets which can be modelled as sets of words where the (independent) probability that the i-th word of a given tweet occurs in a tweet from class 'Sports' can be written as:

$$p(w_i|Sports)$$

Similarly, for a tweet from class 'Politics' can be written as:

$$p(w_i|Politics)$$

Here the words are not dependent on the length of the tweet, position within the tweet with relation to other words, or other twee-context.

Then the probability that a given Tweet contains all of the words , given a class Sports or Politics, is:

$$p(Tweet|Sports) = \prod_i^n p(w_i|Sports)$$

and

$$p(Tweet|Politics) = \prod_i^n p(w_i|Politics)$$

So, What is the probability that a given Tweet belongs to a given class Sports?

Now by Conditional Probability definition,

$$p(Tweet|Sports) = \frac{p(Tweet \bigcap Sports)}{p(Sports)}$$

and

$$p(Sports|Tweet) = \frac{p(Sports \bigcap Tweet)}{p(Tweet)}$$

Bayes' theorem manipulates these into a statement of probability in terms of likelihood.

$$p(Sports|Tweet) = \frac{p(Sports)}{p(Tweet)} p(Tweet|Sports)$$

Assume for the moment that there are only two mutually exclusive classes, Sports and Politics, such that every tweet is in either one or the other;

$$p(Tweet|Politics) = \prod_i^n p(w_i|Politics)$$

and

$$p(Tweet|Sports) = \prod_i^n p(w_i|Sports)$$

Using the Bayesian result above, we can write:

$$p(Sports|Tweet) = \frac{p(Sports)}{p(Tweet)} \prod_i^n p(w_i|Sports)$$

$$p(Politics|Tweet) = \frac{p(Politics)}{p(Tweet)} \prod_i^n p(w_i|Politics)$$

Dividing one by the other gives:

$$\frac{p(Sports|Tweet)}{p(Politics|Tweet)} = \frac{p(Sports) \prod_i^n p(w_i|Sports)}{p(Politics) \prod_i^n p(w_i|Politics)}$$

Which can be re-factored as:

$$\frac{p(Sports|Tweet)}{p(Politics|Tweet)} = \frac{p(Sports)}{p(Politics)} \prod_i^n \frac{p(w_i|Sports)}{p(w_i|Politics)}$$

Thus, the probability ratio p(Sports|Tweet)/p(Politics|Tweet) can be expressed in terms of a series of likelihood ratios. The actual probability p(Sports|Tweet) can be easily computed from log (p(Sports|Tweet)/p(Politics|Tweet)) based on the observation that p(Sports|Tweet) + p(Politics|Tweet) = 1.

Taking the logarithm of all these ratios, we have:

$$ln\frac{p(Sports|Tweet)}{p(Politics|Tweet)} = ln\frac{p(Sports)}{p(Politics)} + \sum_{i}^{n} ln\frac{p(w_i|Sports)}{p(w_i|Politics)}$$

Finally, the tweet can be classified as follows. It is Sports if $p(Sports|Tweet) > p(Politics|Tweet)$ *i.e,*

$$ln\frac{p(Sports|Tweet)}{p(Politics|Tweet)} > 0$$

otherwise it is Politics.

3) References:

- *http://en.wikipedia.org/wiki/Naive_Bayes_classifier*
- *http://ravikiranj.net/drupal/201205/code/machine-learning/how-build-twitter-sentiment-analyzer*
- *http://scikit-learn.org/stable/*

ALGORITHM DETAILS:

1) Pre-Prcoessing Step:

The Pre-Processing Step is carried out by enhance_train_valid_test.py

- *Scans through each line of training.txt & processes each tweet for any stopwords,urls,#tags etc & tokenizes them appropriately.*
- *The Enhanced data is written to En_training.txt*

Similar operations are done for validation.txt & test.txt to obtain En_validation.txt & En_test.txt

2) Training Algorithm:

Multinomial Naive Bayes's Classifier of scikit-learn is used to train the data.

Input Format

The preprocessed input format is in the form of a (processedTweet, sentiment) is used used to train the MultinomialNB Classifier.

Tunable Parameters

- ***alpha:*** *Additive (Laplace/Lidstone) smoothing parameter (0 for no smoothing).*
- ***fit_prior:*** *Whether to learn class prior probabilities or not. If false, a uniform prior will be used.*

Output Format

The output format is an array of 0 or 1 for either Politics or Sports.

3) Validation & Parameter Tuning:

The mine_valid_data.py does the Validation of data & produces ValidationOutput.txt in the format as training.txt.

The alpha parameter can be fine tuned to change the prediction accuracy.

4) Testing Algorithm:

Multinomial Naive Bayes's Classifier is used to predict the test data.

Algorithm Accuracy & Scope of Improvement:

The algotithm produced a accuracy of 99.99% on cross validation data(random 10% of train data), whereas 89.94% accuracy on Validation data.

The accuracy could be further improved by fetching data from the urls in the tweets. Also by further enhancing the tweets with more features it is possible to enhance the accuracy.