Text Classification for Sentiment Analysis – Naive Bayes Classifier

Sentiment analysis is becoming a popular area of research and social media analysis, especially around user reviews and tweets. It is a special case of text mining generally focused on identifying opinion polarity, and while it's often not very accurate, it can still be useful. For simplicity (and because the training data is easily accessible) I'll focus on 2 possible sentiment classifications: positive and negative.

NLTK Naive Bayes Classification

NLTK comes with all the pieces you need to get started on sentiment analysis: a movie reviews corpus with reviews categorized into pos and neg categories, and a number of trainable classifiers. We'll start with a simple NaiveBayesClassifier as a baseline, using boolean word feature extraction.

Bag of Words Feature Extraction

All of the NLTK classifiers work with featstructs, which can be simple dictionaries mapping a feature name to a feature value. For text, we'll use a simplified bag of words model where every word is feature name with a value of True. Here's the feature extraction method:

```
def word_feats(words):
        return dict([(word, True) for word in words])
```

Training Set vs Test Set and Accuracy

The movie reviews corpus has 1000 positive files and 1000 negative files. We'll use 3/4 of them as the training set, and the rest as the test set. This gives us 1500 training instances and 500 test instances. The classifier training method expects to be given a list of tokens in the form of [(feats, label)] where feats is a feature dictionary and label is the classification label. In our case, feats will be of the form {word: True} and label will be one of 'pos' or 'neg'. For accuracy evaluation, we can use nltk.classify.util.accuracy with the test set as the gold standard.

Training and Testing the Naive Bayes Classifier

Here's the complete python code for training and testing a Naive Bayes Classifier on the movie review corpus.

```
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk.corpus import movie_reviews

def word_feats(words):
    return dict([(word, True) for word in words])

negids = movie_reviews.fileids('neg')
posids = movie_reviews.fileids('pos')

negfeats = [(word_feats(movie_reviews.words(fileids=[f])), 'neg') for f in negids]
posfeats = [(word_feats(movie_reviews.words(fileids=[f])), 'pos') for f in posids]

negcutoff = len(negfeats)*3/4
poscutoff = len(posfeats)*3/4

trainfeats = negfeats[:negcutoff] + posfeats[:poscutoff]
testfeats = negfeats[negcutoff:] + posfeats[poscutoff:]
print 'train on %d instances, test on %d instances' % (len(trainfeats), len(testfeat
classifier = NaiveBayesClassifier.train(trainfeats)
print 'accuracy:', nltk.classify.util.accuracy(classifier, testfeats)
classifier.show_most_informative_features()
```

```
classifier.show_most_informative_features()
```

And the output is:

```
train on 1500 instances, test on 500 instances
accuracy: 0.728
Most Informative Features
         magnificent = True              pos : neg    =     15.0 : 1.0
         outstanding = True              pos : neg    =     13.6 : 1.0
           insulting = True              neg : pos    =     13.0 : 1.0
          vulnerable = True              pos : neg    =     12.3 : 1.0
           ludicrous = True              neg : pos    =     11.8 : 1.0
              avoids = True              pos : neg    =     11.7 : 1.0
         uninvolving = True              neg : pos    =     11.7 : 1.0
          astounding = True              pos : neg    =     10.3 : 1.0
         fascination = True              pos : neg    =     10.3 : 1.0
             idiotic = True              neg : pos    =      9.8 : 1.0
```

As you can see, the 10 most informative features are, for the most part, highly descriptive adjectives. The only 2 words that seem a bit odd are "vulnerable" and "avoids". Perhaps these words refer to important plot points or character development that signify a good movie. Whatever the case, with simple assumptions and very little code we're able to get almost 73% accuracy. This is somewhat near human accuracy, as apparently people agree on sentiment only around 80% of the time. Future articles in this series will cover precision & recall metrics, alternative classifiers, and techniques for improving accuracy.

Tagged as: bayes, classification, nlp, nltk, python, sentiment, statistics Leave a comment

NLTK-Trainer

Presentations

Sites

Text Processing

Weotta

Recent Posts

Instant PyGame Book Review

Avogadro Corp Book Review / AI Speculation

Monetizing the Text-Processing API with Mashape

Text Classification for Sentiment Analysis – NLTK + Scikit-Learn

NLTK 2 Release Highlights

Post Categories

blog (2)

books (9)

design (6)

erlang (11)

javascript (6)

links (20)

programming (11)

python (57)

talks (2)

Uncategorized (2)

weotta (3)

Tag cloud

architecture authentication bayes bigrams chunking classification database design django doctest email erlang execnet fab feature extraction forms http javascript jquery machinelearning make mercurial mnesia models mongodb nginx nlp nltk nose otp parsing performance precision pycon python recall redis security sentiment statistics tagging templates testing ui unittest

Andres ↗ Jacob Perkins • 5 months ago

I tried with pickle but, once I have reload/unpickle the classifier, it classifies wrong. I used the following code to save and load the files

```
def save_fichB(fich1,data):
fich=fich1+'.dmp'
f = open(fich, 'wb')
pickle.dump(data, f)
f.close()

def load_fichB(fich1):
fich=fich1+'.dmp'
f= open(fich, 'rb')
data = pickle.load(f)
f.close()
return data
```

I'm stuck on it!

^ | v • Reply • Share ›

Jacob Perkins Mod ↗ Andres • 5 months ago

Not sure how to diagnose this. Maybe code has changed? Or the classifier didn't classify correctly in the first place?

^ | v • Reply • Share ›

philgo20 • 3 years ago

Do you have a cue on where to start to classify in multiple categories ?
I mean not positive/negative classification but one or multiple of 280 categories ?

1 ^ | v • Reply • Share ›

Jacob Perkins Mod ↗ philgo20 • 3 years ago

One of the most common ways of doing it is to train 1 binary classifier for each category, with negative training examples coming from all other categories. Then you combine all the binary classifiers into a multi-label classifier. I cover this using the reuters corpus in my book, at the of the Text Classification chapter. There's also a bunch of research papers out there on multi-label classification.

5 ^ | v • Reply • Share ›

Dmitry Chaplinsky • 3 years ago

Funny, that using only half of dataset as training will increase the accuracy to 0.811 and even 1/4th of data producing result better than 3/4.

1 ^ | v • Reply • Share ›

Jacob Perkins Mod ↗ Dmitry Chaplinsky • 3 years ago

Without any pruning, adding more data increases noise, decreasing accuracy. But if you read the last article in the series, http://streamhacker.com/2010/0..., you'll see that with good pruning the results get much better.

1 ^ | v • Reply • Share ›

Selva Saravanakumar • 3 days ago

Hi.. I'm using both nltk NaiveBayesClassifier and SklearnClassifier for

classification of sentences. Is there is a way to find which is the best classification. For eg: If i give "You are looking not so great" , one is classifying it as "Positive" and other as "Negative". I just want to know which is correct, because i will automate for more the 2k data where manual checking is tedious.

Thanks.

• Reply • Share ›

Jacob Perkins Mod → Selva Saravanakumar • 3 days ago

If you can't do manual checking, then you could at least look the class probabilities, and choose the most probable. Or combine the probabilities, and choose based on that. But if you two classifiers disagreeing, you may want to save those disagreements separately from the text where both agree, then use those disagreements, along with manual classifications, to update/fix your training data & classifiers.

• Reply • Share ›

Selva Saravanakumar → Jacob Perkins • 2 days ago

Thanks for the reply.

I was able to find the probability of classification for NaiveBayesClassifier, but not for SklearnClassifier.
So, I decided to stick with NaiveBayesClassifier whose accuracy is more than SklearnClassifier.

• Reply • Share ›

rwanda • 25 days ago

Hi! How can we determine the cases of classification failure? And besides, I would be interested in also applying SVM and Random Forest for text classification, how can this purpose be achieved?

• Reply • Share ›

Naveed • 25 days ago

I have one more question, I have the training data as a whole in one file and I have five different categories that are labelled as positive and negative examples in the training data.

In order to train a classifier on these five categories I need to create a five files (pos and neg) for each category and then train classifier on each categories using pos and neg examples. Is this the right way??

• Reply • Share ›

Jacob Perkins Mod → Naveed • 25 days ago

Yes, if you want to train a classifier for each category, then the simplest thing to do is create separate pos & neg training files for each category.

• Reply • Share ›

Naveed → Jacob Perkins • 24 days ago

Thanks for the reply Jacob. one more thing I want to ask is that while preparing separate pos and neg training files for each category, I will treat all the sentences where category is not mentioned as positive or negative then I should treat those sentences as neutral. As I have positive, negative and neutral labels. Is this right?

• Reply • Share ›

Jacob Perkins Mod ➚ Naveed • 24 days ago

That sounds right

• Reply • Share ›

Anon92115 • a month ago

Hi, thank you very much for this great site you have! I am building a sentiment classifier for product reviews..so,can you please direct me to some resource where I can find the training set for product reviews? Thank you!

• Reply • Share ›

Jacob Perkins Mod ➚ Anon92115 • a month ago

There's a corpus of Amazon reviews online somewhere. I don't remember where, but try searching for "product review corpus" or "review sentiment".

• Reply • Share ›

Anon92115 ➚ Jacob Perkins • a month ago

Thank you!

• Reply • Share ›

venkatesh M • 2 months ago

Mr . Jacob , if you don't mind, can you kindly let me how to get the data set (for eg: some tweets from twitter) so that i can use the data set for doing my project work, (i have just started to working on this) requesting you to reply me...

• Reply • Share ›

Jacob Perkins Mod ➚ venkatesh M • 2 months ago

I recommend the movie_reviews corpus that comes with NLTK. It's very simple to work with, because NLTK already has corpus reader for reading the file contents in various ways. http://nltk.org/data.html

• Reply • Share ›

Rojin • 3 months ago

Hi, for multi class classification I have trained each classifier with logic labels 'yes' or 'no'. Then I ran all classifiers on the same text. But its giving not a better accuracy. I used skleran (scikit) for the classification. Please guide me how I can improve the accuracy.

• Reply • Share ›

Jacob Perkins Mod ➚ Rojin • 3 months ago

There's very little advice I can offer when I know nothing about the data. Try different algorithms. Try filtering the features using information gain: http://streamhacker.com/2010/0.... Try using different features.

• Reply • Share ›

Rojin • 4 months ago

Hello, I am developing a mail classification system using NLP. I have developed a classifier with Naive Bayer's algorithm. The problem I m facing now is classification of a single mail to different categories. Suppose one mail has three category information and Naive Bayer's allow to classify one text to on category. How I can classify a text into multiple categories. In simple way multi label categorization. Please help me

simple way multi-label categorization. Please help me.

• Reply • Share ›

Jacob Perkins Mod ➚ Rojin • 4 months ago

What you need to do is train a classifier for each category. Every classifier should have 2 logical labels: yes or no. Then to classify in multiple categories, you run each classifier over the text and keep the categories where the classifier label is yes. This technique is called "multiple binary classifiers".

• Reply • Share ›

Rojin ➚ Jacob Perkins • 4 months ago

If it is possible please give an example python code for this. So that I can understand this properly. Thanks

• Reply • Share ›

Jacob Perkins Mod ➚ Rojin • 4 months ago

I've implemented this in train_classifier.py from https://github.com/japerk/nltk.... Use the options --multi --binary to train on a corpus with multiple labels.

• Reply • Share ›

Rojin • 6 months ago

hi thanks, but this is not I meant. I have gone through these links. What I need is NLTK example codes with tutorial. for example sentiment analysis with different conditions. Please do reply.

• Reply • Share ›

Jacob Perkins Mod ➚ Rojin • 6 months ago

Every article in my Text Classification for Sentiment Analysis series has example code for sentiment analysis. The NLTK book and my book both have many NLTK example codes for all sorts of different uses. I can't help you more than that without a much more specific question.

• Reply • Share ›

Rojin • 6 months ago

Hello,

I m developing a new application using NLTK. I want to classify mails into different buckets such as query, feedback etc. So I wish to learn more about nltk. How I can develop an application? suggest some tutorials or links.

• Reply • Share ›

Jacob Perkins Mod ➚ Rojin • 6 months ago

I suggest you start with http://nltk.org/book/ and http://docs.python.org/2/tutor...

• Reply • Share ›

tarik setia • 10 months ago

How can i use nltk to calculate a priori probabilities and probability of each word in the feature?

• Reply • Share ›

Jacob Perkins Mod ➚ tarik setia • 10 months ago

The probability module has many useful functions & classes for

calculating probabilities: http://nltk.org/api/nltk.html#...

Reply • Share ›

chaoprokia • 11 months ago

is it possible for me to use to train 3 classes?

negids = movie_reviews.fileids('neg')

posids = movie_reviews.fileids('pos')

neulds = movie_review.fileids('neu'0

Reply • Share ›

Jacob Perkins Mod → chaoprokia • 11 months ago

Sure, NLTK classifiers work with any number of classes, but most classifiers tend to get less accurate as you go beyond 2 classes.

Reply • Share ›

Amar Shanghavi • a year ago

Dear Jacob,

I have tried working with the code you wrote in your book but get stuck on one point which I am not sure why will not execute. When I try to run the negation replacer, I get the following message:

'AntonymReplacer' object has no attribute 'replace_negations'

I am sure I have copied everything exactly as your code.

Thanks

Reply • Share ›

Jacob Perkins Mod → Amar Shanghavi • a year ago

On Page 42, the AntonymReplacer class is defined with 2 methods: replace & replace_negations. Based on the error message, you either did not define the replace_negations method, or defined it incorrectly.

Reply • Share ›

Amar Shanghavi • a year ago

Dear Jacob, thank you for such a great intro to NLTK. I am reading your book closely too to get a better understanding of text analysis. I would like to know if there is already a pre existing corpus for news (tv transcript or print) which has been classified by positive and negative. I would like to do some sentiment analysis of tv news transcripts and wanted to start from an existing database before I create my own classifications (as a first pass).

Reply • Share ›

Jacob Perkins Mod → Amar Shanghavi • a year ago

Hi Amar,

I don't know of any news sentiment corpus, but you might want to look into "corpus bootstrapping", which is a way to create your own custom corpus based on existing corpora and/or models. Here's a presentation I gave on the topic: http://www.slideshare.net/jape...

Reply • Share ›

Sonia Gupta • a year ago

consider i have a sentence that contain multiple sense word and using

sentiwordnet i am getting multiple score of respective word then how can i calcualte positive or negative .can you elobrate ?? if there is any way to do plz. i am not able to use sentiwordnet due to this reason???plz

• Reply • Share ›

Jacob Perkins Mod → Sonia Gupta • a year ago

You need to look into Word Sense Disambiguation: https://en.wikipedia.org/wiki/...

• Reply • Share ›

anonymous • a year ago

can u tell me how do we write a cosine similarity for these reviews when we r creating a dictionary of these features

• Reply • Share ›

Jacob Perkins Mod → anonymous • a year ago

It's not exactly cosine similarity, but I wrote about using information to eliminate low information features at http://streamhacker.com/2010/0....

• Reply • Share ›

anonymous • a year ago

def word_feats(words): return dict([(word, True) for word in words])

negfeats = [(word_feats(movie_reviews.words(fileids=[f])), 'neg') for f in negids]
can anyone explain wat exactly is he doing over here in this piece of code as i analyse a dictionary is created one for negative words and another for positive ,can u high light why r u creating a dictionary of negative and psitive words

• Reply • Share ›

Jacob Perkins Mod → anonymous • a year ago

Quoting from the article, "All of the NLTK classifiers work with featstructs, which can be simple dictionaries mapping afeature name to a feature value. For text, we'll use a simplified bag of words model where every word is feature name with a value of True."

• Reply • Share ›

Bill • 2 years ago

Hey Jacob,

Great write up, but I had a quick question.

What is the best way to display the neutrality and polarity of a test case?

Thanks a lot!

• Reply • Share ›

Jacob Perkins Mod → Bill • 2 years ago

I'd recommend doing prob_classify(), which gives a ProbDist with the confidence/probabilities of each class, then using that to show how confident your system is for each label.

• Reply • Share ›

Joe C • 2 years ago

HI Jacob,

I was wondering if you could clarify some stuff here. If I had a sentence:
pos_sent = 'I love pizza. It is awesome.'

and I tokenize:

tolk_posset = word_tokenize(pos_sent) #Change to tok to rem punc

where in this code do I feed the tokenized sentience so that I can estimate the sentiment of each token and where do I get the result.

I messed around with replacing your "testfeats" with my tolk_posset but I can't seem to get it to work. Any clarification would be awesome.

BTW, your site is the shit.

- Joe

• Reply • Share ›

Jacob Perkins Mod ➔ Joe C • 2 years ago

Thanks Joe,

You should call word_feats(tolk_posset) which will transform a list of words into a dict that looks like {word: True}

Then, you can pass that dict in the classify() method of a trained classifier to get the sentiment.

• Reply • Share ›

Load more comments

Text Classification for Sentiment Analysis – Precision and Recall »

« Linguistic and Natural Language Processing Links

Go to top ↑