**Text Classification: The First Step Toward NLP Mastery**

1. **text classification**

-named entity recognition

-machine translation

-machine question answering

-spam detection

-sentiment analysis

1. Sentiment analysis
2. classification problem:

-Feature: the string representing the input text

-Target: the text’s polarity (0 or 1)

1. numerical features

-one-hot encoding does not work, cause the text aren’t categories.

-follow two basic steps to transform the main feature into numerical feature

(A pre-processing step to make the texts cleaner and easier to process)

(And a vectorization step to transform these texts into numerical vectors.)

1. two basic steps

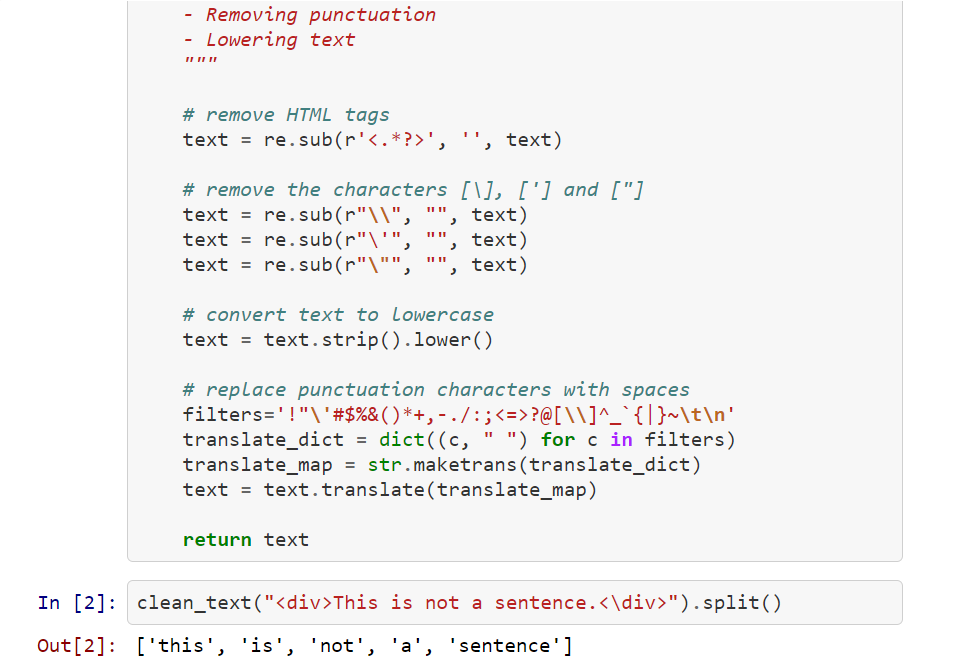
- Pre-Processing

1. representing our text as word sequences

2.regular expressions (regex) a sequence of characters that present a search pattern(e.g. “.”means any character that isn’t the newline character)( <.\*?> can be used to detect and remove HTML tags)

3.re is a python library for regular expressions:





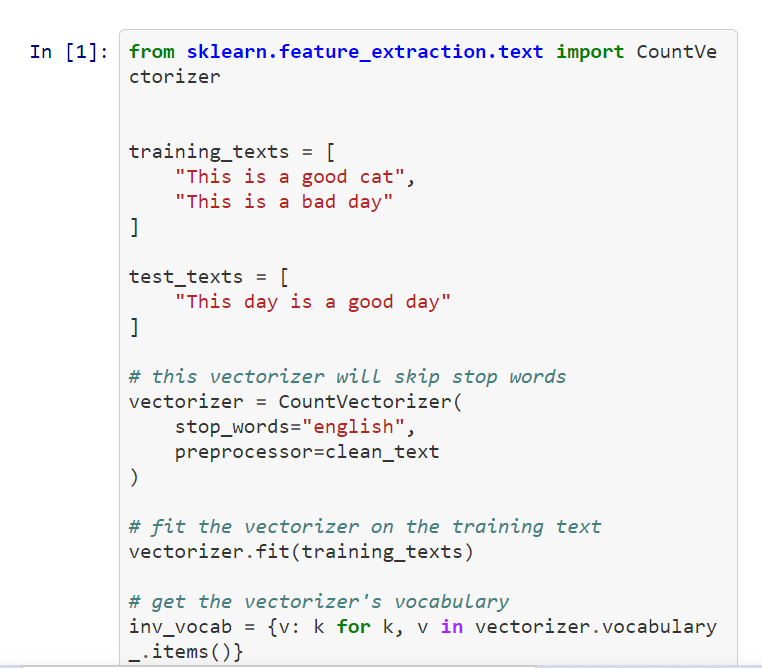
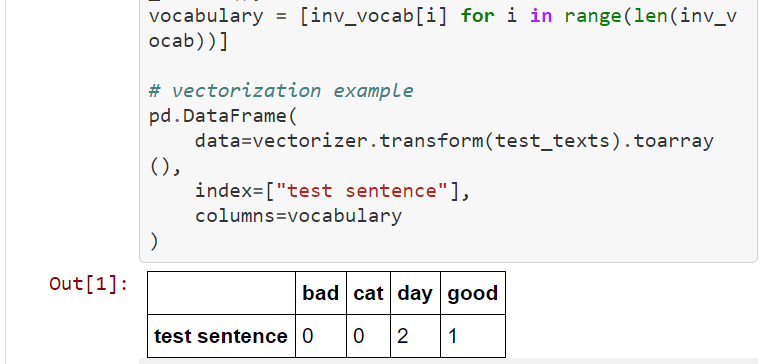
-Vectorization

1. transform word sequences into numerical features(BOW bag of words)the vocabulary generated from the training text, that is all the words that appears in the training text. And then input a new text, output a vector, each element in this vector is the counts of the each word in the vocabulary that in the new text.

2.the more a word appears in a text, the more it is representatives of its meaning.(BOW)

3.detect the word distributions and learn to predict the sentiment of a text based on which words occur and how many times they do(positive sentiment and negative sentiment)

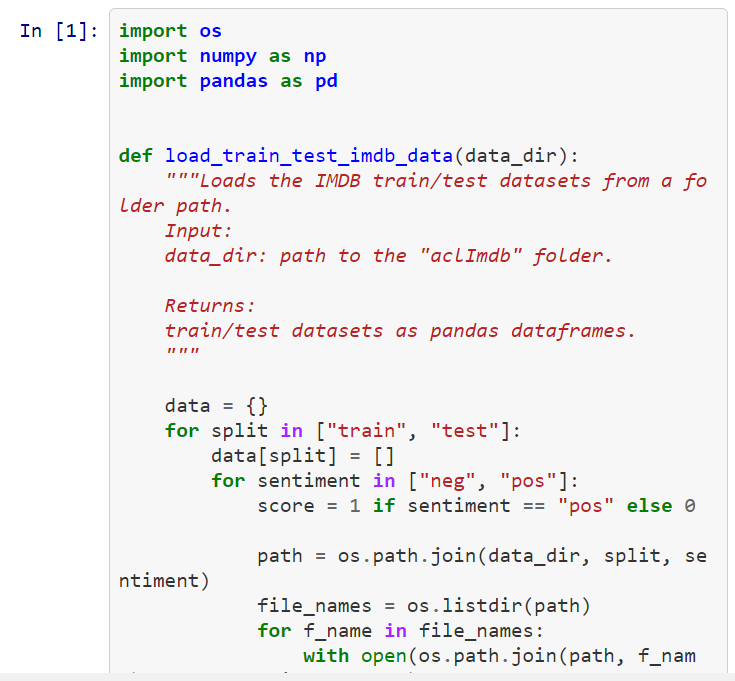
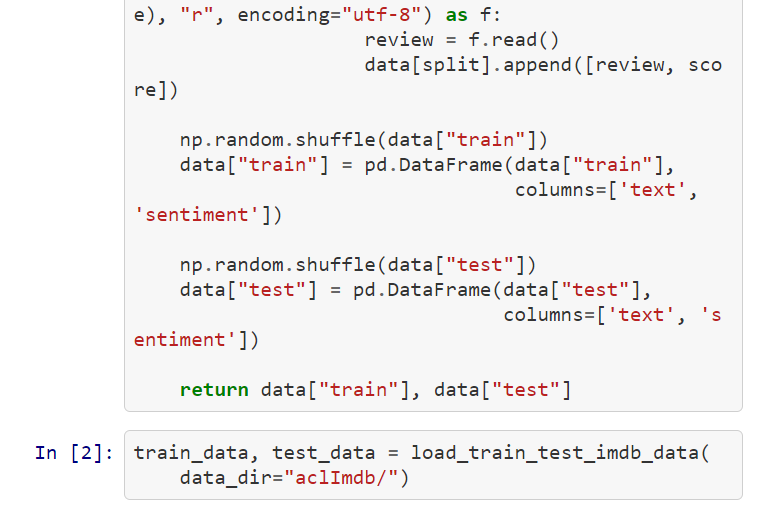
4.use the CountVectorizer from scikit-learn library to performing vectorization:

Use Case : IMDb Movie Reviews

1. Getting the Dataset

A set of 50000 reviews, half are positive and the other half negative, used in sentiment analysis

1. The feature vector result from BOW are usually very large, use simple algorithm that efficient on a large number of features(e.g., Naive Bayes, linear SVM, or logistic regression)
2. SVM classifier:

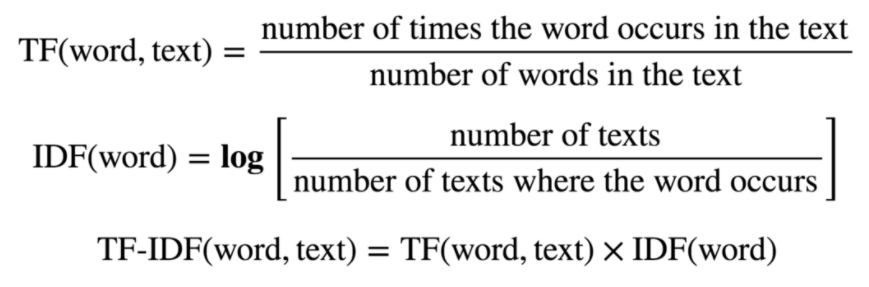


1. Improving the Current Model

1)-Improve the vectorization step. The longer the text, the higher its features will be.

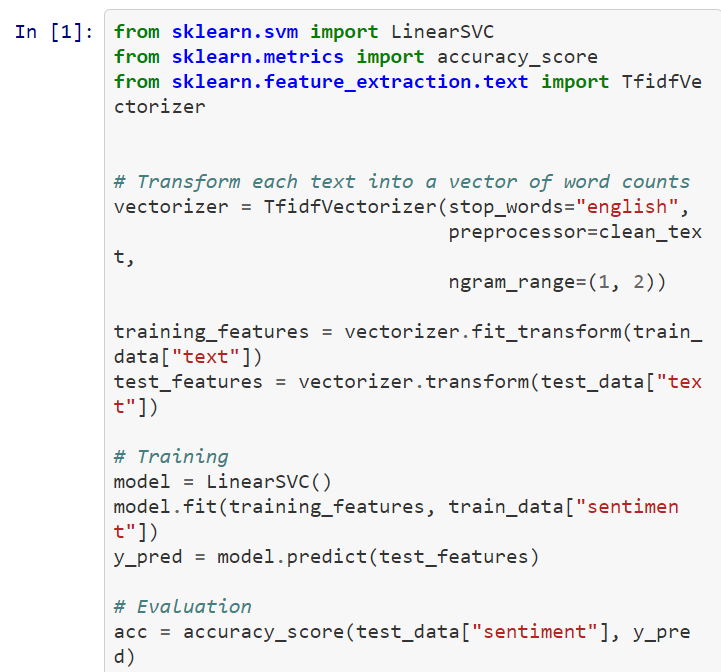
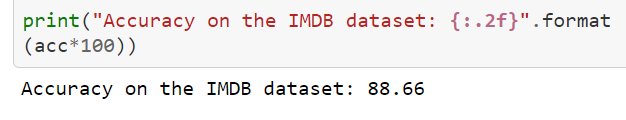
→Term Frequency instead of word counts

→TF-IDF (Inverse Document Frequency)



→ replacing the CountVectorizer with a TfIdfVectorizer

2)improve our model by providing it with more context.( N-grams)

Problem of TF-IDF

1. Don’t account for word position and context
2. Word vectors are usually very high dimensional
3. Not able to capture semantics

BERT

https://zhuanlan.zhihu.com/p/46652512