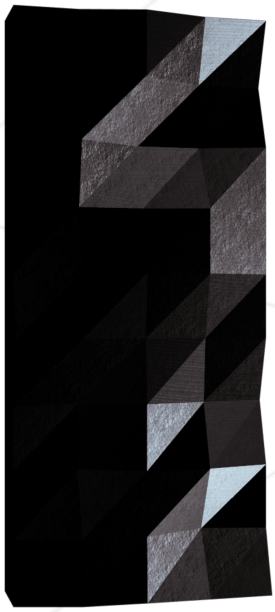

Project 00 – Tweets

Intro to Natural Language Processing. Sentiment Analysis

Summary: This project is an introduction to natural language processing: bag of words, TFIDF, stemming, lemmatization, stop-words, cosine similarity, n-grams, word2vec

Contents

Chapter I

Preamble

Language is important for humans. We use it every day to transfer thoughts from one head to another. It is like the air that we do not even notice. But did you know that it not only reflects the reality but at the same time shapes it? If something does not have a word, it almost literally does not exist for us.

The Amondawa language has no word for "time", or indeed of time periods such as "month" or "year". The people do not refer to their ages, but rather assume different names in different stages of their lives or as they achieve different status within the community source.

The language we use may change how we see different colors. The Himba of northern Namibia – who had never even set foot in a local town – call the sky black and water white, and for them, blue and green share the same word. They have only five words for color, compared to the 11 essential color categories source.

An example from an Aboriginal community in Australia – the Kuuk Thaayorre people. What is cool about them is they don't use words like "left" and "right," and instead, everything is in cardinal directions: north, south, east, and west. You would say something like, "Oh, there's an ant on your southwest leg." Or, "Move your cup to the north-northeast a little bit." In fact, the way that you say "hello" in Kuuk Thaayorre is you say, "Which way are you going?" And the answer should be, "North-northeast in the far distance. How about you?" source.

Chapter II

Introduction

NLP (natural language processing) is a field of knowledge and series of techniques and algorithms that help process text data and extract information from it. You have already worked with structured data – tables full of continuous and categorical features. Text is an example of semi-structured data. You have to do something with it in order to use it, for example, in machine learning tasks.

What machines can understand? That is right – numbers, vectors, and matrices. Imagine that you need to solve a classification task with texts – predict a class for each of them. What can be the features? That is right – words can be the features. This is how the bag of words works.

You can collect a matrix where rows are ids of documents and columns are different words. But what can you put in the cells? The first idea is that you can put 1 if the word exists in the document, and 0 if it does not. The second idea is that you can count the words in the documents and put the numbers as the values in the cells. The third idea is a bit more complicated but can lead to a better result – TFIDF (term frequency – inverse document frequency). It gives greater weights to the words that are specific to the document and lesser weights for the words that are ubiquitous.

But how do you count the words? Will "cat" and "cats" be different words for your task or the same? If you think that they are the same in your task then some preprocessing is required. The first technique is called stemming. It is simple – it just removes some suffixes, prefixes, and other extra stuff from words. In our example, "cats" will become "cat".

But what to do with "better" and "good"? Stemming will not help us with such cases. There is another technique that is called lemmatization. It transforms a word to its base form. In our example, "better" will become "good".

Another problem is that real texts (especially on the Internet) are full of misspellings. Stemming and lemmatization will not help you in these cases. But you can use Levenshtein distance in order to find the words with the minimum number of corrections required to transform one word into another. Some of them will be misspellings.

Actually, we can try yet another approach. What if we will deal not with the words but

with collocations (bigrams, trigrams, n-grams). It will help us to catch “do” and “do not” – they will be different things in our texts. Before it was just “do” and “do”, “not”.

As you see, there are many different ways of how to deal with texts, but you have got the main idea – transform texts into vector formats trying to find (or keep) better features for your task.

Chapter III

Goals

The goal of this project is to give you a first approach to NLP. You will try to preprocess text data and train different classifiers trying to solve a classification task with the best possible score.

Chapter IV

Instructions

- This project will only be evaluated by humans. You are free to organize and name your files as you desire.
- Here and further we use Python 3 as the only correct version of Python.
- The norm is not applied to this project. Nevertheless, you are asked to be clear and structured in the conception of your source code.
- Store the datasets in the subfolder data

Chapter V

Mandatory part

a. Task

In this project, you will work on sentiment analysis of tweets. You will need to predict whether a tweet is positive, negative, or neutral.

- Data preparation. Transform tweets into vectors using different approaches.
- Similarity. Find the top-10 most similar pair of tweets using datasets with different preprocessing.
- Machine learning. Using different machine learning algorithms and different datasets with different preprocessing conduct sentiment analysis.

b. Dataset

- You will work with the dataset of tweets. The tweets are labeled as positive, negative, and neutral. That is it. The source of the dataset.

You can find the dataset in the project page : "p00_tweets.zip"

c. Implementation

- You can work in Jupyter Notebooks. The notebooks should be well-formatted. You need to make a split on the train and test (20
- You can use the NLTK library or any other that you may find useful for you.
- You can enrich the dataset by any other dataset that you may find useful or collect your own.

Data preparation

You need to try different approaches:

	0 or 1, if the word exists	word counts	TFIDF
just tokenization			
stemming			
lemmatization			
stemming +			
misspellings			
lemmatization + misspellings			
any other ideas of preprocessing			

You can perform any cleaning technique before if you find it useful.

You may also try using stop-words to remove the words that create noise and not a signal.

Similarity

Use different datasets that you prepared in the task above and cosine similarity to find the top-10 similar pairs of tweets.

Machine learning

Try different algorithms and different datasets that you prepared before to solve the classification task – sentiment analysis.

d. Submission

You need to achieve accuracy at least equal 0.832 on the test dataset.

Your repository should contain one or several notebooks with your solutions.

Chapter VI

Bonus part

- Try to use word2vec to get a better vectorization of texts.
- Try to achieve a better accuracy on the test dataset – 0.851.
- Try to achieve an even better accuracy on the test dataset – 0.873.

Chapter VII

Submission and peer-correction

Submit your work on your Git repository as usual. Only the work on your repository will be graded.

Here are the points that your peer-corrector will have to check:

- there are all the ways of the preprocessing
- there are the top-10 most similar tweets for any way of preprocessing
- the score achieved on the test dataset