A Project Report on

**Plagiarism detection using python3**

Submitted in Partial Fulfillment of the Requirement for

The Degree of **Bachelors of Engineering in Computer Engineering**

under Pokhara University

Submitted by:

Chhaya Tamang [171343]

Date:6/25/2021

****Department of Computer Science

# 

**NEPAL COLLEGE OF**

**INFORMATION TECHNOLOGY**

Balkumari, Lalitpur, Nepal

Abstract

Plagiarism is the process of copying one’s idea and content without the acknowledgement and consent of the author. It is still a difficult problem hence; many tools have been developed to detect plagiarism. In this project I have created a simple Jupyter notebook that can detect plagiarism using cosine similarity, TF-IDF and jaccard similarity. Cosine similarity measures the similarity between the two vector of an inner product space. It is measured by the cosine angle between the vectors and determines whether two vectors are similar or not. It is used in text and document analysis by converting the text document into vector space. Here the text is converted into vector space. The length of the vector is number of words in vocabulary. The entries in the vector are the number of occurrences of each vocabulary. Similarly, we learn about jaccard similarity.

Keyword: Plagiarism, cosine similarity, software, tools.

Introduction

In academic enterprises like universities, schools, and institutions, plagiarism detection and prevention are one of the most difficult and important challenge. This is because a lot of resources can be found online very easily hence the problem arises. This is solved nowadays with the help of many software tools available in the internet. Many universities also create their own reliable software to detect the plagiarism among the undergraduates.

In information retrieval, using weighted TF-IDF and cosine similarity is a very common technique. It allows the system to quickly retrieve documents similar to a search query. Many plagiarisms software commonly use cosine similarity as a fundamental concept to detect the plagiarism. Cosine similarity measures the similarity between the angles of two vector space. It is measured by cosine of the angle between two vectors and determines whether two are pointing in the same direction or not. Smaller the angle greater the cosine similarity. This means the one of the documents is plagiarized (copied from another). For cosine similarity, the document is converted into vector space where the length of the vector is the number of words in the document. The entries in the vector corresponds to the number of occurrences of each vocabulary word in the document. Then the cosine similarity in this document can be applied.

Project objective

The objective of plagiarism detection software is to find the similarities in the words, text and even document. It helps to find more originality in one’s work and prevents copying from other’s work. The plagiarism detection is very important in todays world as there are many resource easily available in today’s world of internet.

Technical Description

In this project I have created a jupyter notebook using python language.

Jupyter Notebook

The jupyter notebook is an open-source web application which lets us to create and share document that contains live code, equations, visualization and narrative text. Users can create a well explainable and easy to understand code and run it easily on others computer. It is more convenient to share this type of files. IT includes data visualization, machine learning, data cleaning and transformation, numerical simulation, statistical modeling and much more.

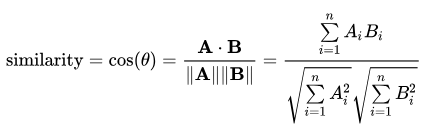
Python

Python is an interpreted, object- oriented, high-level programming language with dynamic semantics. Python is most importantly easy to understand and a simple language. It emphasizes readability and reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

Project Details

In this project, I have used multiple section in jupyter notebook to describe two text similarity metrics. Jaccard similarity and cosine similarity are the most common ones. Hence I have studied these two methods and have come up with this project.

Cosine Similarity

In cosine similarity, the similarity is calculated by measuring the cosine angles between two vectors. This is calculates as:

Here the text is converted into vector. The length of the vector is entries of word in the document. The entry in the document is the number of occurrences of each vocabulary in the text document. The goal is to have a vector space where similar document are close according to a chosen similarity document. This approach takes the name of vector space model, and it’s very convenient because it allows us to use simple linear algebra to compute similarities. We just have to define two things;

* A way of transforming documents into vectors
* A similarity measure for the vectors.

We enter the text in the document into vector form. Here the problem arises when the text document is about a description of a certain item then the word will be repeated multiple times. In such case the same word will increase the vector length unnecessarily. This may result in inaccurate result or biased result. We should use TF-IDF (term frequency – inverse document frequency) weighing method to filter out such bag of words so that the very common set of words (e.g., “the”, “is”, “our”, “of”) doesn’t end up dominating the computation. These word are refered to as stop-words and can be removed in a preprocessing processing step. Tf is good for text similarity in general, but TF-IDF is good for search query relevance. Furthermore, it may be more appropriate to use coordinates of the vectors as trigrams and not individual words.

The idea behind TF-IDF is that we first compute the number of documents in which a word appears in. If a word appears in many documents, it will be less relevant in the computation of the similarity, and vice versa. We call this value the inverse document frequency or IDF, and we can compute it as:

idf(word) = log(\frac{N}{|\{d \in C : word \in d\}|})

In the formula, C is the corpus, N is the total number of documents in it, and the denominator is the number of documents that contain our word.

After this we will the text is processed further:

* The text is converted into lowercase.
* The text is tokenized into sentences and the sentences into words.
* Then the trigrams are generated using n-gram.

The n-gram is a contiguous sequence of n items from a given sample text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The n-gram are collected from text corpus. When n=3 then it is called trigrams. The words are grouped into trigrams for more reasonable similarity detection. It is a little more complex then 1 gram but more accurate, hence trigrams are used. At last, cosine similarity is used to calculate the similarity.

Jaccard Similarity

The preprocessing of the data is almost similar to above. Jaccard Similarity defined as an intersection of two documents divided by the union of those two documents that refer to the number of common words over a total number of words. Here, we will use the set of words to find the intersection and union of the document.

The mathematical representation of the Jaccard Similarity is:

The Jaccard Similarity score is in a range of 0 to 1. If the two documents are identical, Jaccard Similarity is 1. The Jaccard similarity score is 0 if there are no common words between two documents.

Jaccard similarity takes only unique set of words for each sentence while cosine similarity takes total length of the vectors to calculate the similarity. Jaccard similarity is good for cases where duplication does not matter and cosine similarity is good for cases where duplication matters while analyzing text similarity. For two product descriptions, It will be better to use jaccard similarity as repetition of words does not reduce the similarity between the documents.

Conclusion

There are many techniques we can use for plagiarism detection using NLP but, in this project, I have focused on jaccard similarity and cosine similarity method as it is the simplest and commonly used method. In the future I would like to explore more about other topics such as levenshtein distance (edit distance) and many more.