IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

PRODUCT RECOMMENDATION ENGINE Core description about the algorithms and modules used for developing a recommendation engine

[1]Meet G Patel, [2]Rachit M Pandya, [3]Ravi K Virani

[1]B.E. I.T, [2]B.E. I.T, [3]B.E. I.T
[1]Information Technology
[1]Sardar Patel College of Engineering, Bakrol, Anand, India

Abstract: Owing to the frequent elevations in technology, the e-commerce and business websites tend to upgrade and inculcate latest technologies within their system to remain in the market. One of the key things that mostly each website is including is a recommendation system. A Recommendation system is an engine which suggests products to the customers based on their preference and similarity with the product they are currently viewing. This can be done on 2 fronts: 1) Text based product similarity and 2) Image based product similarity. However, the datasets for this system to work are bulky and we have to use complex machine learning modules like Tensorflow and Keras. Apart from that algorithms like Bag-Of-Words, Term Frequency-Inverse Document Frequency (TF-IDF) and Word2Vec can be used for product recommendation.

Keywords: E-commerce, Product Recommendation, Bag-of-Words, Term Frequency, Inverse Document Frequency, Word-2-Vec, Artificial Intelligence, Machine Learning.

I. INTRODUCTION

Recently there has been a spontaneous upsurge in the amount of products available over the internet, there is a heterogeneous range of goods available to buy online especially apparels. As a matter of fact, this has brought some unwanted complications for the customers looking to buy products via e-commerce websites. Under such circumstances, a recommendation system is what is exactly needed for both buyers and vendors as it does not only minimize the user's efforts but also boosts the sales of the sellers which in this case are the e-commerce sites. To implement this concept, we have used python language as it is the best when it comes to compatibility and it can be easily integrated with advanced technologies like artificial intelligence and machine learning. Morever, a major advantage of writing code in python is that it's syntax is relatively very small in size so it reduces our code size. There are a number of libraries that are required to handle bulky datasets such as Matplotlib, Pandas, Numpy, Scipy, Scikit-Learn, Keras and Tensorflow. However, the key for developing such system is the usage and understanding of algorithms like Bag of words, Term frequency, Term frequency-Inverse document frequency and Word 2 Vec. This algorithms provide the text-based product recommendation and suggests the closest products to the users which are computed on the basis of similarity between the products according to their preferences.

II. LITERATURE SURVEY

As Discussed earlier there are several algorithms that are needed to be implemented in order to find similarity between two products, they are:

2.1 Bag of Words (BoW) model

This model is one of among the best and to grasp and implement and it’s one among the foremost used techniques to search out similarities between 2 products. Applications using this algorithm experience nice success in AI issues like language modeling and document classification.
Basically it is used to extract features from text from text to be used in modeling, like machine learning algorithms. A **bag-of-words** is an illustration of text that describes the occurrence of words inside a document. It comprises of 2 things:
1) A vocabulary of better-known words.
2) A measure of the presence of better-known words
This model is merely involved with whether or not already known words occur in the document, not it's location within the document i.e., any data regarding the order of structure of words within the document is discarded, this reason why it’s referred to as "bag" of words. The complexness of this algorithm program depends upon the analyzer’s need as it arises in each deciding a way to style the vocabulary of better-known words and the way to get the presence of those better-known words. The operating of this model consists of following activities:

Step 1: Data Assortment
Step 2: Design the Vocabulary
Step 3: Produce Document Vectors

Managing the Vocabulary:
With relevance to the rise within the size of the vocabulary, the vector representation of documents conjointly will increases. For a massive corpus, the length of the vector perhaps in thousands or millions of positions. Further, every document might contain very few of the better-known words within the vocabulary. This ends up in a vector with countless zero scores, called a sparse vector or sparse representation. They need additional memory and computational resources once modeling and therefore the large variety of dimensions will create the modeling process very challenging for easy algorithms, however there's no ought to decrease the scale of vocabulary in this model. An additional complicated approach is to form a vocabulary of grouped words. This changes not the scope of the vocabulary and permits the bag-of-words to capture a touch additional sense from the document.

Scoring Words:
Once vocabulary has been chosen, the occurrence of words in sample document must be scored. An easy approach is a binary scoring of the presence of absence of words. Some extra straightforward scoring strategies include:
1) Counts: count the quantity of times each word appears
2) Frequencies: measure the frequency that each word appears during a document to the total words in the document.

Word Hashing:
We will use a hash representation of better-known words in our vocabulary. This solves the matter of getting a massive text Document because we can choose the size of hash space. This is referred to as "hash trick" or "feature hashing".

Limitations of BoW:
Though it is very simple to understand and implement, bag of words model will have some shortcomings:
1) Vocabulary: The vocabulary needs to be designed fastidiously, especially in order to manage the size as it directly impacts the scantiness of document.
2) Sparsity : Sparse representations area unit harder to design for both computational and information reasons because it is challenging for the models to generate very little information in such an oversized space.
3) Meaning :Discarding the word not solely ignores the context however conjointly the meaning of the word.

2.2 Term Frequency- Inverse Document Frequency (TF-IDF) Model

This algorithm is a statistical approach that evaluates how relevant a word is to a document in a set of documents. To do this, we need to multiply 2 metrics: number of times a word appears in a document, and the inverse document frequency of the word across a collection of documents. This has several purposes, most significant in analysis of text, conjointly used for scoring words in machine learning algorithms for Natural Language Processing (NLP).

It was invented for document search and information retrieval and works by increasing proportionally to the number of times a word appears in a document, however is offset by the number of documents that contain the word. Therefore, words that are common in each document, like this, they don't mean abundant to it document specially. Interestingly, if a word appears many times in a single document, and is not frequent in other documents, it probably means that it's very significant.

How is TF-IDF calculated:
As mentioned earlier, TF-IDF for a word in document is obtained as a product of 2 completely different metrics.
1) The **Term Frequency** of a word in a document. There are several ways in which to calculate this frequency and the most simple is a raw counting of instances a word appears in a document. There are also several ways of adjusting the frequency by length or by the frequency of the most frequent word in the document.
2) The **Inverse Document Frequency** of the word across a collection of documents i.e., how common or rare a word is in the entire corpus. The closer it is to 0, more common the word is. This can be calculated by dividing the total number of documents by the number of documents containing the word and calculating the algorithm.
3) Therefore, if the word is very common and appears in multiple documents, the number will approach 0, otherwise 1.

Mathematical Form: $tfidf(t, d, D) = tf(t,d)*idf(t, D)$

where $tf(t,d) = \log(1 + freq(t,d))$, $idf(t,d) = \log\left(\frac{N}{count(d \in D : t \in d)}\right)$

Application Areas:
Once we tend to verify the relevance of a word through tf-idf, it can be useful in a number of ways:
1) Retrieving Information: This model can be used to deliver results that are most relevant to what you're seeking for. For example you own a search engine and a user looks for any word, the results will be displayed in order of relevance. Mostly all search engines use TF- IDF in it is algorithm.
2) Extracting Keywords: This is often conjointly helpful to extract keywords from a text, the words with highest words are the most relevant words in the document and so they are the keywords of the text.

2.3 Word 2 Vec model

Word Embedding is one of the most popular approach used to represent a word, which is capable of capturing not only the context of the word but also its semantic and syntactic and semantic similarity and it's relation with other words. **Word2Vec** is the most widely used technique to implement word embedding. It represents word as a vector using shallow two-layer neural networks which are trained to reconstruct linguistic context of the words.

As input, it takes a large corpus of text and produces a vector space, generally having hundreds of dimensions and each unique word being assigned a corresponding vector in the space. Vectors are positioned such that words having common contexts in the corpus are located close to each other. Word2Vec uses two models to produce vector representation:

1) CBOW(common bag of words):
In this method, the context of each word is taken as an input and predicts the word corresponding to the context. For example, consider the text: *have a great day*. The Input is *great* and we are trying to predict a target word *day* using a single context input word *great.* We use the one hot encoding of the input word and determine the error in output compared to one hot encoding. of the target word (*day*). During this process of predicting the target word, we learn the vector representation of the target word.

2) Skip-Gram model:
We can also use a target word (the word whose representation we want to generate) to predict the context, this approach is called skip-gram model. Diagrammatically, it may look like an inverse of bow model and to some extent, it is true. This architecture gives weights to words nearby more heavily than distant words.

CBOW is faster than skip-gram but it does a better job for infrequent words and small amount of data.

Parameterization:
Results of a Word2Vec model training depend on the following parameters:
1) Algorithm used: A word2vec model can be trained either hierarchical softmax or negative sampling. however, according to authors, hierarchical softmax is suitable for infrequent words while negative sampling is better with low dimensional vectors.
2) Sub-sampling: Often little information is provided via high frequency words, so words with frequency above a certain mark may be subsampled to increase training speed.
3) Dimensionality: Higher dimensionality may result in better quality but after a certain point, gain from margins will decrease.
4) Context Window: The number of words before and after the word would be included as context can be determined by the size of the context window.

2.4 Implementation

The models described above are enough to generate an efficient and economical recommendation, however there are certain pre-requisite steps that are required to be performed and output if these steps will be the input of these algorithms like removing null and duplicate values, handling missing data, stemming and eliminating stop-words. To reduce the size of the dataset we have a tendency to use OLAP(online analytical processing) operations like roll-up, drill-down, slice, dice and pivot. For implementation of machine learning approaches, we have a tendency to calculate a mathematical function referred to as Euclidian distance that is employed with the mentioned algorithms to produce the recommendation displayed by using plotting libraries (matplotlib) and plotting functions.

III. CONCLUSIONS

Considering all the points described above we will conclude that, all models explained above play a pivotal role within the operating of a recommendation system. The Bag-of-Words is basic however it is still the foremost wide used technique, whereas TF-IDF generates suggestions based on the relevancy of words within the document. Moreover, Word2Vec permits us to represent words as vectors. However, there are certain aspects of this domain that require a lot of research and want a lot of clearer statistical information processing and improving the accuracy of recommender engines.

REFERENCES

[1] Vanita Jain Achin Jain and New Delhi Nidhi Kapoor Bharati Vidyapeeth College of Engineering. Recommendation System based Sentiment Analysis. Advanced Computational Intelligence : An International Journal (ACII), Vol.3, No.1, January 2016, pages 690–698, 2016.

[2] Vijay Verma Mahak Dhanda. RECOMMENDATION SYSTEM for ACADEMIC LITERATURE WITH INCREMENTAL ANALYSIS. Recommendation system for academic literature with incremental analysis, II-2/W2.

[3] Rong JIn Yin Zhang. Understanding-Bag of words model: A Statistical Framework. pages 1-12, 2010.

[4] Sun, Hao, et al. "Automatic target detection in high-resolution remote sensing images using spatial sparse coding bag-of-words model." IEEE Geoscience and Remote Sensing Letters 9.1 (2011): 109-113.

[5] Huang, Cheng-Hui, Jian Yin, and Fang Hou. "A text similarity measurement combining word semantic information with TF-IDF method." Jisuanji Xuebao(Chinese Journal of Computers) 34.5 (2011): 856-864.

[6] Albitar S, Fournier S, Espinasse B. An effective TF/IDF-based text-to-text semantic similarity measure for text classification. InInternational Conference on Web Information Systems Engineering 2014 Oct 12 (pp. 105-114). Springer, Cham.

[7] Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. "Improved semantic representations from tree-structured long short-term memory networks." arXiv preprint arXiv:1503.00075 (2015).