University of Milan

Milano, Italy

Master's in Data Science and Economics

FINDING SIMILAR ITEMS: STACKSAMPLE ALGORITHMS FOR MASSIVE DATASETS

by

Peesapati Venkata Sai Vamsi

933987

Venkatasaivamsi.peesapati@studenti.unimi.it

September, 2021

Declaration

I declare that this material, which will be now submitted for assessment, is entirely our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of our work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

Contents

Chapter 1

Dataset Description

The aim of the project is to implement a detector of pairs of similar items analyzing the StackSample dataset published on Kaggle and released under the CC-BY-SA 3.0 license. The dataset is the collection of 10% of question and answers from Stack Overflow programming Q&A website. Initially, the dataset contains three tables,

I. Questions contains the title, body, creation date, closed date, score, and owner ID for all Stack Overflow questions whose Id is a multiple of 10.

II. Answers contains the body, creation date, score, and owner ID for each of the answers to these questions.

III. Tags contains the tags on each of these questions
In this assignment, we have used the Spark DataFrame API for loading the data, preprocessing and building the algorithm. As per the requirements of the project, the 'body' of *Questions* is considered.

```
+--------------------+
|                Body|
+--------------------+
|"<p>I've written ...|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
|                null|
+--------------------+
only showing top 20 rows
```

Initially, the data contains 33149724 observations out of which 31745443 are null values. It is vital to remove the null values as they are of no importance. The data after the removal of nulls contain 1404281 observations.

```
+--------------------+
|                Body|
+--------------------+
|"<p>I've written ...|
|"<p>Are there any...|
|<p>Has anyone got...|
|<p>This is someth...|
|<p>I have a littl...|
|<p>I am working o...|
|<p>I've been writ...|
|<p>I wonder how y...|
|<p>I would like t...|
|<p>I'm trying to ...|
| and haven't seen...|
|<p>What's the sim...|
|<p>I need to grab...|
|<p>I'm looking fo...|
|<p>What is the co...|
|<p>I am using CCN...|
|<p>I am looking t...|
|<p>I am using MSB...|
|<p>I'm setting up...|
|<p>I always creat...|
+--------------------+
only showing top 20 rows
```

Chapter 2

Data Preprocessing

The dataset needs to be manipulated or cleaned before it is used as it improvesthe performance. As the process of initial cleaning is completed, that is, removing the null values. Now, consider the data that gives us some information, for example,I remove the regular expressions by lowering the text and removing the expressions like <p>, <code>, <'ve>, <'m> by replacing the expressions with blank space using pyspark.sql.functions library.

```
+--------------------+
|                Body|
+--------------------+
|i written a datab...|
|are there any rea...|
|has anyone got ex...|
|this is something...|
|i have a little g...|
|i am working on a...|
|i been writing a ...|
|i wonder how you ...|
|i would like the ...|
|im trying to main...|
| and havent seen ...|
|whats the simples...|
|i need to grab th...|
|im looking for a ...|
|what is the corre...|
|i am using ccnet ...|
|i am looking to a...|
|i am using msbuil...|
|im setting up a d...|
|i always create a...|
+--------------------+
only showing top 20 rows
```

After the process of regular expression removal, tokenize the data. Tokenization is the process of taking the text and breaking it into individual terms using pyspark.ml.feature and count the number of tokens in each array observations using used defined function 'udf()'.

As the data is tokenzied, we remove the stopwords that gives no information or which are of repeated too often. The process of StopWordRemoval is carried out using pyspark.ml.feature, the stopwordremover taken as input the array of strings and removes

Body	Words	Number_of_tokens
i written a datab...	[i, written, a, d...	18
are there any rea...	[are, there, any,...	13
has anyone got ex...	[has, anyone, got...	9
this is something...	[this, is, someth...	41
i have a little g...	[i, have, a, litt...	15
i am working on a...	[i, am, working, ...	46
i been writing a ...	[i, been, writing...	46
i wonder how you ...	[i, wonder, how, ...	18
i would like the ...	[i, would, like, ...	47
im trying to main...	[im, trying, to, ...	66
and havent seen ...	[, and, havent, s...	21
whats the simples...	[whats, the, simp...	17
i need to grab th...	[i, need, to, gra...	26
im looking for a ...	[im, looking, for...	61
what is the corre...	[what, is, the, c...	26
i am using ccnet ...	[i, am, using, cc...	33
i am looking to a...	[i, am, looking, ...	24
i am using msbuil...	[i, am, using, ms...	17
im setting up a d...	[im, setting, up,...	33
i always create a...	[i, always, creat...	36

only showing top 20 rows

the stopwords. Here, we considered only the default stopwords.

Body	Words	Filtered
i written a datab...	[i, written, a, d...	[written, databas...
are there any rea...	[are, there, any,...	[really, good, tu...
has anyone got ex...	[has, anyone, got...	[anyone, got, exp...
this is something...	[this, is, someth...	[something, pseud...
i have a little g...	[i, have, a, litt...	[little, game, wr...
i am working on a...	[i, am, working, ...	[working, collect...
i been writing a ...	[i, been, writing...	[writing, web, se...
i wonder how you ...	[i, wonder, how, ...	[wonder, guys, ma...
i would like the ...	[i, would, like, ...	[like, version, p...
im trying to main...	[im, trying, to, ...	[im, trying, main...
and havent seen ...	[, and, havent, s...	[, havent, seen, ...
whats the simples...	[whats, the, simp...	[whats, simplest,...
i need to grab th...	[i, need, to, gra...	[need, grab, base...
im looking for a ...	[im, looking, for...	[im, looking, way...
what is the corre...	[what, is, the, c...	[correct, way, ge...
i am using ccnet ...	[i, am, using, cc...	[using, ccnet, sa...
i am looking to a...	[i, am, looking, ...	[looking, allow, ...
i am using msbuil...	[i, am, using, ms...	[using, msbuild, ...
im setting up a d...	[im, setting, up,...	[im, setting, ded...
i always create a...	[i, always, creat...	[always, create, ...

only showing top 20 rows

Chapter 3

Implementation

Secondly, after the data is tokenized, we extract the feature from the raw data. In order to obtain the feature, we implement few algorithms.

Term frequency – Inverse document frequency (TF-IDF) is a feature vectorization method that is used to give the importance of a term to a document in corpus.

Let us consider a term 't', a document 'd' and corpus 'D'. We define the Term Frequency TF(t,d) as the number of times a term 't' appears in a document 'd'. Similarly, Document Frequency can be defined as DF(t,D) - number of documents that has term 't'. Both term frequency and document frequency has equal importance as both gives the measure of importance.

Inverse Document Frequency give the numerical measure of importance of a term or how much information a term provides.

$$IDF(t, D) = log(|D| + 1/DF(t, D) + 1)$$

The Term frequency – Inverse document frequency is the product of both Term Frequency TF(t,d) and Inverse Document Frequency IDF(t,d,D).

$$TFIDF(t, d, D) = TF(t, d).IDF(t, D)$$

The process of Term Frequency and Inverse Document Frequency can carried out separately for the flexibility.

Term Frequency: In this project, HashingTF is used to generate the term frequency vector. HasingTF is a transformer which takes set of wrods as input and converts them

into fixed length feature vector using the hashing trick which maps an index by applying hash function. Currently, MurMur Hash3 algorithm is used to calculate the hash code value for the term. The feature dimension is set as 16, as it is important to take the feature dimension in the power of 2, where the default feature dimension is also the power of 2, i.e, 262114.

Inverse Document Frequency: IDF is an estimator which fits on the dataset and produces an IDFModel. The IFDModel take the feature vectors obtained from HashingTF and scales each feature, it down-weights vectors which appear more often in the corpus and improves performance when using text as features.

HashinhTF requires single pass whereas IDF requires two passes, first to compute the IDF vector and second to scale the features.

Locality Sensitive Hashing (LSH) is a class of hashing techniques used in clustering, approximate nearest neighbor search and outlier detection with large datasets.

The idea of LSH is to use a family of functions to hash data points into buckets, so that the data points which are close to each other are in the same buckets with high probability, while data points that are far away from each other are very likely in different buckets. In Spark, different LSH families are implemented in separate classes (e.g., MinHash).

In LSH, we define a false positive as a pair of distant input features which are hashed into the same bucket, and we define a false negative as a pair of nearby features which are hashed into different buckets.

As finding the similarity using a huge dataset is computationally difficult, we have used MinHash algorithm for Jaccard distance, as it a LSH family.

MinHash for Jaccard Distance is an LSH family for Jaccard distance where input features are sets of natural numbers. Jaccard distance of two sets is defined by the cardinality of their intersection and union:

$$d(A, B) = 1 - (|A \cap B|/|A \cup B) \tag{3.1}$$

MinHash applies a random hash function g to each element in the set and take the minimum of all hashed values:

$$h(A) = min(g(a))$$

The input sets for MinHash are represented as binary vectors, where the vector indices represent the elements and the non-zero values in the vector represent the presence of

that element in the set.

LSH for Minhash Signatures is and approach to hash items many times in such a way that similar items are hashed to the same bucket. We then consider any pair that is hashed to the same bucket as Candidate pair. We then consider candidate pairs for similarity. The pairs which are not similar that hashed to the same bucket are false positives. The pairs that are similar but hashed to different bucket are false negatives.

We divide the minhash signatures into ‘b’ bands consisting of ‘r’ rows. For each band, there is hash function that takes vectors of ‘r’ integers and hashes them to large number of buckets. We use a separate bucket array for each band so that the columns with the same vector in different bands will not hash to the same bucket.

Approximate Nearest Neighbor Search takes a data of feature vectors and a key of single feature vector, and it returns a specified number of rows in the dataset that are closest to the vector. A column is added to the output dataset which shows the true distance between each output row and the searched key.

Chapter 4

Description of Experiment

Initially, after the dataset is loaded, it is cleaned by removing the null values, regular expressions. The cleaned data is tokenized, the tokenizer takes the sets of strings as input and gives an array of string as output, these arrays of strings are the array of tokens or words. Now, as the data is tokenized, the stopwords are removed taking array of strings as input and removes the default stopwords, custom stopwords can also be removed by using the 'stopWords' parameter. As, empty sets cannot be transformed by MinHash, which means any input vector must have at least 1 non-zero entry, it is necessary to filter empty arrays. The tokens are now vectorized using HashingTF. These feature vectors obtained by Term Frequency are used to compute the Inverse Document Frequency which scales each feature. LSH algorithm takes the feature vectors as input

Body	Words	Filtered	RawFeatures	Features
i written a datab...	[i, written, a, d...	[written, databas...	(16,[0,3,4,6,8,10...	(16,[0,3,4,6,8,10...
are there any rea...	[are, there, any,...	[really, good, tu...	(16,[0,1,7,8,10,1...	(16,[0,1,7,8,10,1...
has anyone got ex...	[has, anyone, got...	[anyone, got, exp...	(16,[3,5,6,7,10,1...	(16,[3,5,6,7,10,1...
this is something...	[this, is, someth...	[something, pseud...	(16,[1,2,4,6,8,9,...	(16,[1,2,4,6,8,9,...
i have a little g...	[i, have, a, litt...	[little, game, wr...	(16,[6,7,8,10,11,...	(16,[6,7,8,10,11,...
i am working on a...	[i, am, working, ...	[working, collect...	(16,[1,2,3,4,5,7,...	(16,[1,2,3,4,5,7,...
i been writing a ...	[i, been, writing...	[writing, web, se...	(16,[0,1,2,4,5,6,...	(16,[0,1,2,4,5,6,...
i wonder how you ...	[i, wonder, how, ...	[wonder, guys, ma...	(16,[0,3,4,6,7,9,...	(16,[0,3,4,6,7,9,...
i would like the ...	[i, would, like, ...	[like, version, p...	(16,[0,1,2,5,6,7,...	(16,[0,1,2,5,6,7,...
im trying to main...	[im, trying, to, ...	[im, trying, main...	(16,[1,2,3,4,5,6,...	(16,[1,2,3,4,5,6,...
and havent seen ...	[, and, havent, s...	[, havent, seen, ...	(16,[1,2,3,7,8,10...	(16,[1,2,3,7,8,10...
whats the simples...	[whats, the, simp...	[whats, simplest,...	(16,[3,5,8,9,10,1...	(16,[3,5,8,9,10,1...
i need to grab th...	[i, need, to, gra...	[need, grab, base...	(16,[1,2,4,5,7,9,...	(16,[1,2,4,5,7,9,...
im looking for a ...	[im, looking, for...	[im, looking, way...	(16,[0,1,3,4,6,7,...	(16,[0,1,3,4,6,7,...
what is the corre...	[what, is, the, c...	[correct, way, ge...	(16,[0,2,4,5,8,10...	(16,[0,2,4,5,8,10...
i am using ccnet ...	[i, am, using, cc...	[using, ccnet, sa...	(16,[0,1,4,5,6,7,...	(16,[0,1,4,5,6,7,...
i am looking to a...	[i, am, looking, ...	[looking, allow, ...	(16,[2,3,4,5,8,9,...	(16,[2,3,4,5,8,9,...
i am using msbuil...	[i, am, using, ms...	[using, msbuild, ...	(16,[0,1,6,7,10,1...	(16,[0,1,6,7,10,1...
im setting up a d...	[im, setting, up,...	[im, setting, ded...	(16,[0,2,3,4,5,7,...	(16,[0,2,3,4,5,7,...
i always create a...	[i, always, creat...	[always, create, ...	(16,[0,1,3,9,10,1...	(16,[0,1,3,9,10,1...

only showing top 20 rows

with 10 hash tables and outputs a seq[vector]. I have used the approximate nearest neighbor search to get information about questions that are similar. I have taken key that is used to measure the distance. The approximate nearest neighbor search proved us with a distance column that describes the distance between the dataset and the key that has been specified before and approximately searched for 10 nearest neighbors to

```
+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+
|                Body|               Words|            Filtered|         RawFeatures|            Features|              Hashes|
+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+
|i written a datab...|[i, written, a, d...|[written, databas...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|
|are there any rea...|[are, there, any,...|[really, good, tu...|(16,[0,1,7,8,10,1...|(16,[0,1,7,8,10,1...|[[3.1829568E8], [...|
|has anyone got ex...|[has, anyone, got...|[anyone, got, exp...|(16,[3,5,6,7,10,1...|(16,[3,5,6,7,10,1...|[[3.1829568E8], [...|
|this is something...|[this, is, someth...|[something, pseud...|(16,[1,2,4,6,8,9,...|(16,[1,2,4,6,8,9,...|[[3.1829568E8], [...|
|i have a little g...|[i, have, a, litt...|[little, game, wr...|(16,[6,7,8,10,11,...|(16,[6,7,8,10,11,...|[[3.1829568E8], [...|
|i am working on a...|[i, am, working, ...|[working, collect...|(16,[1,2,3,4,5,7,...|(16,[1,2,3,4,5,7,...|[[3.1829568E8], [...|
|i been writing a ...|[i, been, writing...|[writing, web, se...|(16,[0,1,2,4,5,6,...|(16,[0,1,2,4,5,6,...|[[3.23279535E8], ...|
|i wonder how you ...|[i, wonder, how, ...|[wonder, guys, ma...|(16,[0,3,4,6,7,9,...|(16,[0,3,4,6,7,9,...|[[3.1829568E8], [...|
|i would like the ...|[i, would, like, ...|[like, version, p...|(16,[0,1,2,5,6,7,...|(16,[0,1,2,5,6,7,...|[[3.1829568E8], [...|
|im trying to main...|[im, trying, to, ...|[im, trying, main...|(16,[1,2,3,4,5,6,...|(16,[1,2,3,4,5,6,...|[[3.1829568E8], [...|
| and havent seen ...|[, and, havent, s...|[, havent, seen, ...|(16,[1,2,3,7,8,10...|(16,[1,2,3,7,8,10...|[[3.23279535E8], ...|
|whats the simples...|[whats, the, simp...|[whats, simplest,...|(16,[3,5,8,9,10,1...|(16,[3,5,8,9,10,1...|[[3.1829568E8], [...|
|i need to grab th...|[i, need, to, gra...|[need, grab, base...|(16,[1,2,4,5,7,9,...|(16,[1,2,4,5,7,9,...|[[3.1829568E8], [...|
|im looking for a ...|[im, looking, for...|[im, looking, way...|(16,[0,1,3,4,6,7,...|(16,[0,1,3,4,6,7,...|[[3.1829568E8], [...|
|what is the corre...|[what, is, the, c...|[correct, way, ge...|(16,[0,2,4,5,8,10...|(16,[0,2,4,5,8,10...|[[3.1829568E8], [...|
|i am using ccnet ...|[i, am, using, cc...|[using, ccnet, sa...|(16,[0,1,4,5,6,7,...|(16,[0,1,4,5,6,7,...|[[3.23279535E8], ...|
|i am looking to a...|[i, am, looking, ...|[looking, allow, ...|(16,[2,3,4,5,8,9,...|(16,[2,3,4,5,8,9,...|[[3.1829568E8], [...|
|i am using msbuil...|[i, am, using, ms...|[using, msbuild, ...|(16,[0,1,6,7,10,1...|(16,[0,1,6,7,10,1...|[[3.23279535E8], ...|
|im setting up a d...|[im, setting, up,...|[im, setting, ded...|(16,[0,2,3,4,5,7,...|(16,[0,2,3,4,5,7,...|[[3.23279535E8], ...|
|i always create a...|[i, always, creat...|[always, create, ...|(16,[0,1,3,9,10,1...|(16,[0,1,3,9,10,1...|[[3.1829568E8], [...|
+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+
only showing top 20 rows
```

the key.

```
+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+-------+
|                Body|               Words|            Filtered|         RawFeatures|            Features|              Hashes|distCol|
+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+-------+
|on selection chan...|[on, selection, c...|[selection, chang...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|good day guys can...|[good, day, guys,...|[good, day, guys,...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i want to know wh...|[i, want, to, kno...|[want, know, noti...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i am trying to fi...|[i, am, trying, t...|[trying, find, so...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i am trying to ge...|[i, am, trying, t...|[trying, generate...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i only want scrip...|[i, only, want, s...|[want, scripts, s...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|hi i got a set of...|[hi, i, got, a, s...|[hi, got, set, co...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i am learning up ...|[i, am, learning,...|[learning, method...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i am using code f...|[i, am, using, co...|[using, code, fir...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
|i am trying to us...|[i, am, trying, t...|[trying, use, qui...|(16,[0,3,4,6,8,10...|(16,[0,3,4,6,8,10...|[[3.23279535E8], ...|    0.0|
+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+-------+
```

Chapter 5

Results & Conclusion

As the aim of the project is to implement a detector of pairs of similar questions from StackSample.

To conclude, I have implemented the detector of pair of 10 similar questions corresponding to a key. It has been implemented initially, by cleaning the data, extracting the raw features and scaled features from the cleaned data and passing these features to the learning algorithm. Approximate nearest neighbor search was used to get the group of similar questions.