# Text File Analysis

## *Using* Hadoop/ MapReduce

## *On the* Cloudera HDP data platform

**Cloud Computing Project Final Report**

**David Mannion**

**16365576**

A report submitted in part fulfilment of the Cloud Computing module of
MSc Computer Science.


School of Computer Science and Informatics

University College Dublin

31 December 2020

# Abstract

We are currently living in the data era. Each day massive amounts of information are produced. The purpose of this investigation is to analyse large scale text data. However, Traditional information retrieval techniques have become inadequate due to the size of the datasets available today. Therefore, I will use the Hadoop and MapReduce frameworks on the Cloudera HDP data platform to implement an efficient analysis of text data through K-means clustering.

# Table of Contents

# 1 Introduction

- *Motivation*

Hadoop is one of the most widely implemented big data processing frameworks. I wanted to get hands-on experience using Hadoop and the Cloudera HDP platform as I have a strong interest in Data Science, big data processing and Machine Learning. I have used K means in the past to analyse data, but not using MapReduce so I was interested to see how the two could be combined successfully. I also had the desire of learning python streaming with MapReduce as it I would be able to tie my knowledge of python in with a new data technology. I think MapReduce will undoubtedly be very useful to me as a data scientist in the future due to its big data capabilities, hence I am eager to learn as much about it as possible. As well as this, Natural language processing (NLP) has become a desired skill in data science and this assignment will help me learn more on this interesting topic while also improving my skillset.

- *Objectives*

This project has multiple objectives. The first is to represent the input documents as a table of term vectors using MapReduce and Hadoop. This can be done by implementing a word-count MapReduce program, then modifying the results, so the output is a table of term vectors. The second objective is to reduce the document space of the table of term vectors produced from the first objective. This means reducing the dimensionality of the dataset or reducing the number of columns in the dataset. The third and final objective was to assign a cluster I.D to each of the documents, using K-means clustering. From the cluster I.Ds we will be able to see what documents are similar to each other and what documents are not.

- *Why in your view this is interesting?*

Natural Language processing is a very interesting field of study. It comes at the intersection between Computer Science, Artificial Intelligence and linguistics. The goal of NLP is to process or ‘understand’ natural language. It deals with textual data which is typically unstructured, which can be a challenge to overcome. There are multiple methods used to deal with this, which in themselves are interesting as they try to solve a classic problem using many different approaches, such as Neural networks, or in our case, using a document term matrix and clustering analysis. Natural language processing can be used for something like sentiment analysis which could be useful in predicting things like the stock market or how popular a product is.

- *How Cloud Computing plays a part*

Cloud computing provides a very useful framework for NLP. Since NLP deals with vast amounts of data, even the most efficient algorithms (such as MapReduce) will never successfully run on commodity machines, due to limitations on their processing power, memory and storage. Cloud computing provides a solution to this problem. The processing of data can be done on the cloud instead of on these commodity machines. Cloud computing takes advantage of distributed computing, meaning we can pool resources of multiple machines together, running at the same time (in parallel) to run a task faster and more efficiently. The large data files being processed can also be stored on the cloud, reducing the storage requirements of the local machine. All you need to do is pass the commands to the cloud service you are using, and it will take care of the rest.

The results of the analysis can be accessed by anyone, anywhere if they have an internet connection, which enables better synergies within a business and leads to a more streamlined process.

# 2 Project's Specifications and Requirements

- *Define the project full specifications*

Write mapper and reducer scripts for the word count data and write a script to output this data as a table of term vectors, where each column is a word that occurs in any of the input documents and the rows are the number of occurrences of that word in a given document.

Write scripts for reducing the dimensionality of the table of term vectors. This required writing multiple mappers, reducers, and various other scripts. The goal was to reduce the amount of columns in the dataset, while still maintaining data quality.

Write scripts to implement K means on the data with reduced dimensionality, using the MapReduce approach. This required writing a mapper python script, a reducer python script, a reader python script as well as a shell BASH script.

All these scripts are to be executed on the Cloudera HDP platform.

- *What are the requirements of such application you would like to implement?*

The requirements for the application I would like to implement are

1. Cloudera HDP Platform (which requires a virtual machine such as VirtualBox)
2. Hadoop (which comes with the HDP)
3. Python3 and a python3 editor
4. A computer with at least 8GB of RAM to run the virtual machine

# 3 Methodology & System Architecture

- *Methodology followed to develop and implement the project*

Each map reduce job needs a mapper and a reducer in order for it to run successfully.

I first wrote the mapper and reducers on my machine and tested them locally using stdin and stdout using the python subprocess module.

I uploaded the input files to the HDFS using the shell command -copyFromLocal

I got the original data from various websites such as www.gutenberg.org.

The idea was to use the output from part1 to feed into part 2 and feed the output from part 2 into part 3.

**Word Count (part 1 of the Assignment)**

Mapper function input is the corpus of text documents.

The output of the mapper function is the keys and values. In this case, the keys would be the word, e.g. ‘apple’ or ‘orange’ and the value would be 1.

- the input for the reducer is the list of key pairs above

- The output of a reducer is the sum of the occurrences for a given word for a given document.

1.2 mapper() function:

```
Pairs = [ ]
For word in words:
        Pairs.append(list( word , 1 ))
Reduce() function:
Sort(pairs) by key
Unique_keys = list(dict.fromkeys(keys))
#gets unique keys, ie In a list [a,b,a] ,
returns [a,b]
For key in unique_keys:
        Return sum(word.value) where pairs.key = key
```

Return Table for each word, document, value

## Dimension Reduction

I used Singular Value Decomposition to reduce the dimensionality of the output matrix.

In linear algebra, the singular value decomposition (SVD) is a factorization of a real or complex matrix that generalizes the eigendecomposition of a square normal matrix to any mxn matrix via an extension of the polar decomposition[1].

SVD is the factorization of a matrix(**A**) into three matrices

**A =** $\mathbf{U\Sigma V^*}$

Where V* is the transpose of V

From these matrices we can calculate **T**, which is a reduced dimension representation of **A**

**T =** $\mathbf{U\Sigma}$

**U** is calculated by first calculating $\mathbf{AA^T}$, then by getting the eigenvector of this matrix product.

Sigma is calculated as the square root of the eigen values of $\mathbf{AA^T}$

We take the first k

I used map reduce to calculate $\mathbf{AA^T}$ as well to calculate $\mathbf{U\Sigma}$

## K Means

Definition of the input and the output of the Map and Reduce functions of K-Mean Algorithm

- The input to a mapper function is a partition $D_1$ taken from the original dataset X = $\{x_1, x_2, \dots, x_n\}$
- for each data point given ($\{u_j\}$ , $x_i$ ) , emit ($z_i$ , $x_i$)
- $\{u_j\}$ is a set of centroids, $z_i$ is the centroid id
- The input to the reducer is ($z_i$ , $x_i$)
- The reducer recomputes the centroid position from the $x_i$ that are now assigned to it
- The reducer output is then ($z_i$ , $\{x_i\}$ ) , meaning (centroid id , set of points assigned to that centroid)
- This is then changed to be in the form ($\{u_j\}$ , $x_i$ ) so it can be fed back into a mapper() function, so the k-means algorithm can continue until no new centroids are chosen

1 Wikipedia.org

# 4 Implementation Details

- How did you implement it (language, environment, …)?

I used python3 to write the code for the mappers, reducers and other programs that needed to be run. I used UNIX to send commands to the HDP. I used python streaming in Hadoop to integrate my python commands with Hadoop.

The environment I used was the Cloudera HDP running on a Linux (Red Hat) virtual machine on windows 10.

I used PyCharm to write the python code before running it in Hadoop with the Hadoop streaming functionality.

- How much have you implemented (functions, operations, etc.)?

I have implemented the majority of what was required. Unfortunately, I was not able to represent the word count MapReduce job as a table of term vectors. This proved problematic for the rest of the assignment.

I managed to successfully reduce the dimensionality of the document term matrix

I was able to successfully implement K means in MapReduce on a given dataset

- Evaluation.

While the project is not perfect, it has a lot of the desired functionality.

- Explain how the system should be installed and used?

By running the script in the .BASH files in the various folders

# 5 Conclusion

- State clearly how much have you achieved in developing and implementing the project.

I successfully wrote mapper and reducer scripts in python, which, along with Hadoop streaming, produced the word count for a corpus of word documents.

I also successfully wrote mapper and reducer functions to reduce the dimensionality of the dataset. These were again streamed in

I managed to implement the K-means algorithm to cluster datapoints in a given dataset using the MapReduce framework.

- *Challenges faced*

There were several challenges I had to overcome while completing this project. Setting up the environment – the HDP, took quite some time between downloading it and setting it up correctly.

There were a lot of things that were new to me that required getting familiar with. I had not used UNIX or Hadoop or Java before, or any Linux operating system or a Virtual Machine.

Understanding the HDFS proved a challenge, as well as writing code in python that would be able to interact correctly with Hadoop, MapReduce and the Hadoop streaming functionality.

Getting the virtual machine (VirtualBox) to interact with my local filesystem was a challenge as again I had little to no experience in working with this type of program.

While there was not a lot of java to be written, understanding what a JAR file was and how it interacts with the HDFS and the mappers, reducers and other code was a challenge. Learning the Hadoop filesystem and how this integrated with rest of the Hortonworks sandbox took time to get familiar with.

- *What I learned from this project*

From a data science point of view, I learned a lot about Natural Language processing, dimension reduction and clustering techniques. I learned about several different dimension reduction methodologies, such as Principal Component Analysis, Lemmatizing, stop word removal and of course Singular Value Decomposition when doing the project.

I found out in detail how the K-means algorithm works and its strengths and weaknesses. It is a fast and efficient algorithm that is useful for text data, but it can only model spherical clusters so it not suitable for all clustering jobs.

I learned how to use the Hadoop Distributed file system and how it works, what MapReduce is and how it can be applied to real-word datasets in order to process big data efficiently and accurately.

I learned how to use the Cloudera HDP platform and how to use virtualization to run a different operating system, Linux on my local Windows machine.

I learned some basic UNIX commands, how to use these to implement Hadoop streaming with python.

I learned that Hadoop streaming can be used to run MapReduce jobs on Hadoop using code written in python, even though Hadoop is written in Java.

I learned how to write code to perform a map reduce task, and how fast and effective MapReduce can be at processing large amounts of data.

**Final Comments**

Hadoop and MapReduce are powerful tools in big data processing. They allow large amounts of data to be processed quickly and efficiently. There is no doubt these tools, especially when they are run in the cloud, are the technologies of the future.