Spring 4-29-2020

Detection of Sockpuppet Accounts on Reddit

Swati Adhikari

DETECTION OF SOCKPUPPET ACCOUNTS ON REDDIT

By
Swati Adhikari

A thesis submitted to the faculty of The University of Mississippi in partial fulfilment of the requirements of Sally McDonnell Barksdale Honors College.

Oxford
April 2020

Approved by

Advisor: Dr. Philip Rhodes

Reader: Dr. Yixin Chen

Reader: Dr. Dawn Wilkins

ACKNOWLEDGEMENT

First and foremost, I would like to thank Dr. Philip Rhodes for his advice, guidance, and patience throughout the process of writing this thesis. His motivation and sincerity have deeply inspired me to write this thesis. I would also like to thank my committee members Dr. Dawn Wilkins and Dr. Yixin Chen.

I convey my sincere gratitude to Sally McDonnell Barksdale Honors College for giving me a wonderful opportunity to write this thesis. It has always been a great honor to be a part of Sally McDonnell Barksdale Honors College. I would also like to extend my sincere acknowledgment to the exceptional faculty in the Computer Science department for the past amazing four years here in the University of Mississippi. I would also like to thank my family and friends for always encouraging me with their wise words to strive for success. A very special acknowledgement to my sister, Sweta Adhikari and my friend Bidhan Bashyal for helping me write and present the thesis to my Honors Thesis Committee.

This was a very uncertain and hard semester for all the students due to the COVID-19 pandemic, but all the students and instructors made through the semester with strong will and determination. Cheers to all the May 2020 graduates!

ABSTRACT

Swati Adhikari: Detection of sockpuppet accounts on Reddit

(Under the direction of Philip Rhodes)

The purpose of this thesis is to study and analyze the detection of sockpuppet accounts on Reddit. Sockpuppet accounts are created exclusively to cause mischief or mayhem at a site without the original user being identified. With the rise of sockpuppet accounts, it is very important to identify these accounts and help maintain a healthy online community. The data used for the research was obtained from Reddit which is stored in the dirt cluster. The categorization of sockpuppet accounts and non-sockpuppet accounts was implemented with TF-IDF (term frequency inverse document frequency algorithm), k-Means clustering algorithm and a multilayer perceptron classifier. The goal is to identify sockpuppet accounts from a huge dataset of accounts on Reddit and bring awareness of the level of harm and misuse sockpuppet accounts can create to the online community.

TABLE OF CONTENTS

TABLE OF FIGURES

ABBREVIATIONS

RDD	Resilient Distributed Dataset
UDF	User Defined Functions
JSON	JavaScript Object Notation
API	Application Programming Interface
TF	Term Frequency
IDF	Inverse Document Frequency
ETL	Extract Transform Load
SSE	Sum of Squared Error

CHAPTER 1: INTRODUCTION

Social networking services are one of the most used services today. Almost 4.54 billion people around the world have access to the internet, and one in three people in the world use some sort of social media platform. Among these users, there are many accounts which are fake, and are also referred to as sockpuppet accounts. Sockpuppet is an online fictional identity used for purposes of deception. The use of sockpuppet includes misleading uses of online identities, manipulation of public opinion, or to circumvent restrictions, suspension or an outright ban from a website. Sockpuppets are unwelcomed in many online communities and forums. There are four types of sockpuppets: block evasion, ballot stuffing, strawman sockpuppet and meat puppet. In this thesis, we will be dealing with meat puppets. A meat puppet publishes comments on blogs, wikis and other social networking sites about some phenomenon or product in order to generate public interest and buzz (The Original, 2011). Sockpuppets lead to the misuse or misinterpretation of a data or an identity and also tend to manipulate public opinion.

According to an article by Medium, tens of thousands of bots and hundreds of human-operated, fake accounts acted in a concerted effort to push a pro-Trump agenda across various social media platforms (Morgan, 2017). There were also some suspicions on Trump campaign's ties to Russia and its role in manipulating the media ecosystem. The article says that both automated accounts called "bots" and human-operated fake accounts, called "sockpuppets" attempted to manipulate the conversation in conservative spaces throughout the 2016 election. Sockpuppet language was found to be significantly more aggressive in its support for Trump than language from other users in the same community. These sockpuppet accounts were also more active than regular users, averaging roughly twice as many posts per day (Morgan, 2017).

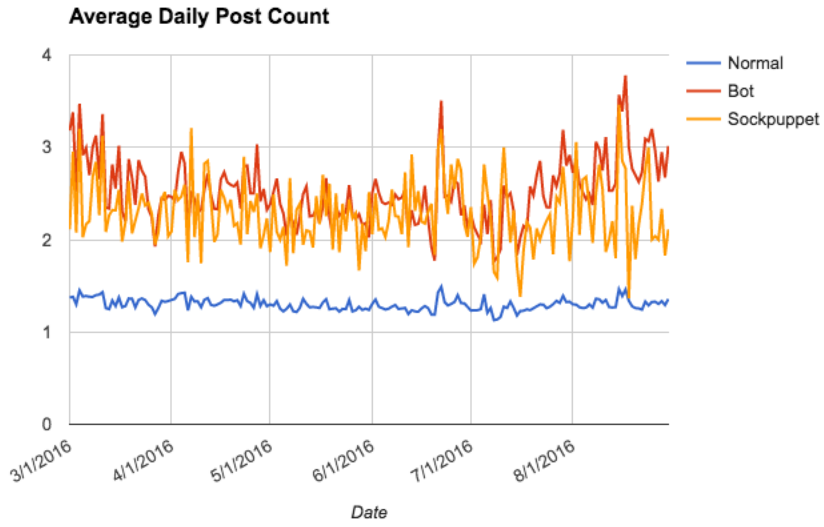

Figure 1: Average Daily Post Count during 2016 election campaign (Morgan, 2017)

The graph in Figure 1 shows that sockpuppet accounts posted way more than average users and less than automated accounts. This example shows how sockpuppet accounts tend to manipulate public opinion.

An article by Vox states that Facebook removed 2.2 billion fake accounts just in the first quarter of 2019 and had 2.38 billion active users remaining. Facebook estimates that 5 percent of its monthly active users are made up and this is a huge issue (Stewart, 2019). This is a great demonstration of the extent to which sockpuppets has affected the online ecosystem. Thus, it is very important to be able to detect sockpuppet accounts from these social media accounts. In this thesis, we investigate the feasibility of a sockpuppet detection tool where we attempt to detect sockpuppet accounts from a dataset of Reddit accounts. We divide these data as our testing and training data and predict sockpuppet accounts using Term Frequency-Inverse Document Frequency algorithm, K-Means algorithm and Multi-layer perceptron classifier.

The second chapter will discuss the fundamental concepts – Parallelization in Spark and the data used in the thesis. The third chapter explains about the algorithms used: term frequency –

inverse document frequency, k-Means clustering algorithm and multilayer perceptron classifier and the evaluation of these algorithms. The fourth chapter discusses the development of sockpuppet detection, and the technologies used in the process. These chapters give a detailed insight into the purpose of this thesis, which is to predict sockpuppet accounts from a list of users on Reddit.

CHAPTER 2: BACKGROUND

2.1 Parallelization in Spark

Parallel programming model is an abstraction of parallel computer architecture, with which it is convenient to express algorithms and their composition in programs. Problems are broken down into instructions and are solved simultaneously as each resource which has been applied to work is working at the same time. In the development of the detection of sockpuppets, we use parallel programming with Spark and thus need to be careful of writing code that can be handled in the parallel mechanism. There are two major advantages of parallel programming over serial programming. In parallel programming, all processes are sped up when using parallel structures, thus increasing both the efficiency and resources used in order to achieve quick results. Another benefit is that parallel computing is cost efficient.

Spark uses Resilient Distributed Dataset (RDD) to perform parallel processing across a cluster or computer processors. It has easy-to-use APIs for operating on large datasets in various languages including Python. PySpark is used for the implementation of the detection of sockpuppet accounts in this thesis. Thus, Spark uses a cluster manager to coordinate work across a cluster of computers, which is a very important concept of parallel processing Big Data. There are two different ways of achieving parallelization in PySpark (Weber, 2019):

1. **Native Spark:** Spark data frames and libraries contribute in parallelizing the code and distributing natively by Spark.
2. **Thread Pools:** The multiprocessing library can be used to run concurrent Python threads and perform operations with Spark data frames as well.

2.2 The Data

All the data used in this thesis is from Reddit. It is stored in the Hadoop File System in the dirt cluster. The data has been fetched from Reddit from the years 2007 to 2018. The data consists of Reddit users, their posts, date of the posts, subreddit and their karma scores. We also have a list of sockpuppet accounts which were web scraped separately from Reddit's 2017 Transparency Report. The data is well-structured and are all stored in Parquet format.

Two years ago, Reddit had been mentioned as one of the platforms used to promote Russian propaganda and its ties to US 2016 election. Then Reddit decided to investigate those accounts promoting Russian propaganda focusing on three broad areas: ads, direct propaganda from Russians, and indirect propaganda promoted by Reddit users. Reddit blocked all ads from Russia and ads from other parts of the world were reviewed by humans. After some investigation, Reddit found and removed a few hundred accounts that were of suspected Russian Internet Research Agency origin. Before entirely removing those accounts, Reddit made those accounts available for other users to see and make the case transparent with its users (Reddit, 2017). These accounts are used in the thesis in identifying sockpuppet accounts.

According to Reddit's 2017 Transparency report, there were 944 accounts that were suspicious and removed from Reddit. Few of these accounts had a visible impact on the site, where 70% of these accounts had zero karma, 1% had negative karma, 22% had 1-999 karma, 6% had 1,000 - 9,999 karma and 1% had a karma score of 10,000+, where karma points are the points scored when one's posts or comments are upvoted. There was a total of 14,000 posts by all of these users (Reddit, 2017). Of the 282 accounts with non-zero karma, more than half (145) were banned prior to the start of the investigation through Reddit's routine Trust & Safety practices. But eventually, at least seven accounts with significant karma scores made past their

defense. So, these sockpuppet accounts will be used to train the rest of the data to recognize sockpuppet accounts.

This data and the list of sockpuppet accounts from the Reddit 2017 Transparency Report was combined and the data was divided as train and test data to be later used while predicting sockpuppet accounts. The sockpuppet accounts were labelled as “1” and non-sockpuppet accounts were labelled as “0”. The basis of the sockpuppet recognition in the pool of accounts was the vocabulary used in the posts and the similarity or differences in the vocabulary of these users. A major issue discovered along the process however was that not all these sockpuppet accounts posted or commented on other posts, so these accounts did not contribute in the training of the neural network on our dataset.

CHAPTER 3: ALGORITHMS USED

3.1. Term Frequency – Inverse Document Frequency

3.1.1 Background

Term Frequency – Inverse Document Frequency is an information retrieval technique that weighs a term's frequency and its inverse document frequency. Each term has its term frequency and inverse document frequency score. The algorithm is used to weigh a keyword in any content and assign the importance to that keyword based on the number of times the word appears in the document. Term frequency is the number of times a term appears in the document. Document Frequency is the number of documents that contain the term. Thus, the actual importance of the term in both one document and number of documents can be obtained by taking the product of term frequency and the inverse of document frequency. Let us denote TF as a function of term (t) and document (d): TF(T, d) and denote DF as a function of term(t) and document corpus (D): DF(t,D). Thus, IDF as a function of term and document is given by:

$$IDF(t, D) = \log\left(\frac{|D|+1}{DF(t, D)+1}\right)$$

And the TF-IDF as a function of term, document and document corpus is given by,

$TFIDF(t, d, D) = TF(t, d) \cdot IDF(t, D)$ (TutorialKart, 2018).

3.1.2 Implementation

For the implementation of TF-IDF, the body of the posts made by reddit users were taken as the dataset. Term Frequency vectors of the words of the posts were generated using HashingTF. HashingTF converts documents to vectors of fixed size. IDF acted as an estimator which was fit on the dataset and produced an IDFModel. The IDFModel took feature vectors and

scaled each column. The model then down-weighted columns which appeared frequently in a document corpus. The implementation is further explained by the figure below.

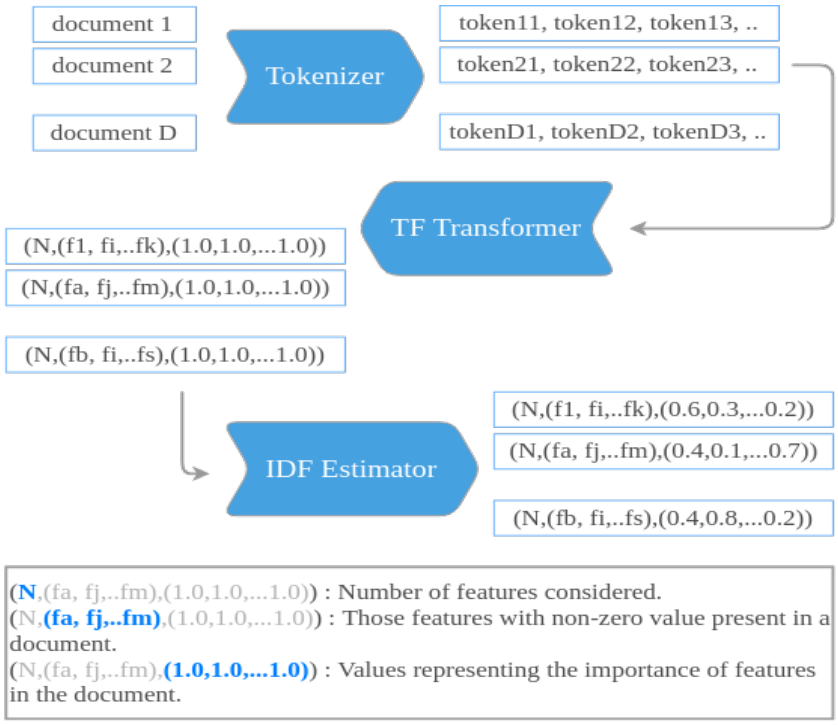

Figure 2: TF-IDF Implementation (TutorialKart, 2018)

3.2. K-Means Clustering Algorithm

3.2.1 Background

K-Means Clustering is one of the most popular unsupervised machine learning algorithms. The K-means algorithm begins with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative calculations to optimize the positions of the centroids. The algorithm halts creating and optimizing clusters when either the centroids have stabilized, and the clustering has been successful or when the defined number of iterations has been achieved. The approach k-means follows to solve the problem is called Expectation-Maximization (Dabbura, 2019). The expectation step is

assigning the data points to the closest cluster and the maximization step is computing the centroid of each cluster. K-means uses distance-based measurements to determine the similarity between data points. This algorithm fits well into the objective of this thesis to find accounts with similarities and group them into clusters.

3.2.2 Implementation

The basis of similarity for the division of clusters is the posts every user has made in Reddit in our dataset. Below are the steps taken for the implementation of k-Means clustering algorithm.

1. Specified the number of clusters *K*. We have the number of clusters k=3.
2. Initialized centroids by first shuffling the dataset and then randomly selecting *K* data points for the centroids without replacement.
3. After the new values of centroid are found, the algorithm performs the same set of steps until the difference between old centroids and the new centroids are negligible.

3.3.3 Evaluation

The most common measure of evaluating k-means clustering is Sum of Squared Error (SSE). For each point in the cluster, the error is the distance to the nearest cluster. To get SSE, the errors are squared and summed up. Then, plotting a graph of SSE for each k-means run against different values of k clusters, gives the optimal number of clusters. This method of finding the optimal number of clusters through SSE is called the Elbow Method (Dabbura, 2019).

Figure 3: Implementation of the Elbow Method for finding the optimal number of clusters

From the implementation of elbow method, the optimal number of clusters for our dataset is 3. These clusters are assumed to be sockpuppet accounts, non-sockpuppet accounts and the accounts that did not post or comment anything but were still used as a part of the sockpuppet list.

3.1 Multi-layer Perceptron Classifier

3.3.1 Background:

Multi-layer Perceptron Classifier is a classifier based on the feedforward artificial neural network with multiple layers of nodes connected to each other in a directed fashion. It uses a supervised learning technique called backpropagation for training the network. In the intermediary layer, the nodes use the sigmoid function to restrict the output between 0 and 1, and the nodes in the output layer use the SoftMax function, a generalized version of the

sigmoid function (Hastie et al., 2009). In this thesis, we use the multilayer perceptron classifier from spark ML.

3.3.2 Implementation

Below are the implementation steps taken into account for multi-layer perceptron classifier.

1. Split the data to train and test data
2. Specified the layers for the neutral network. The input layers of the size of the number of features were specified, and there were two hidden layers and an output layer of size 2 for two outputs “0” and “1”. “1” is the label for sockpuppets and “0” is the label for non-sockpuppet accounts.

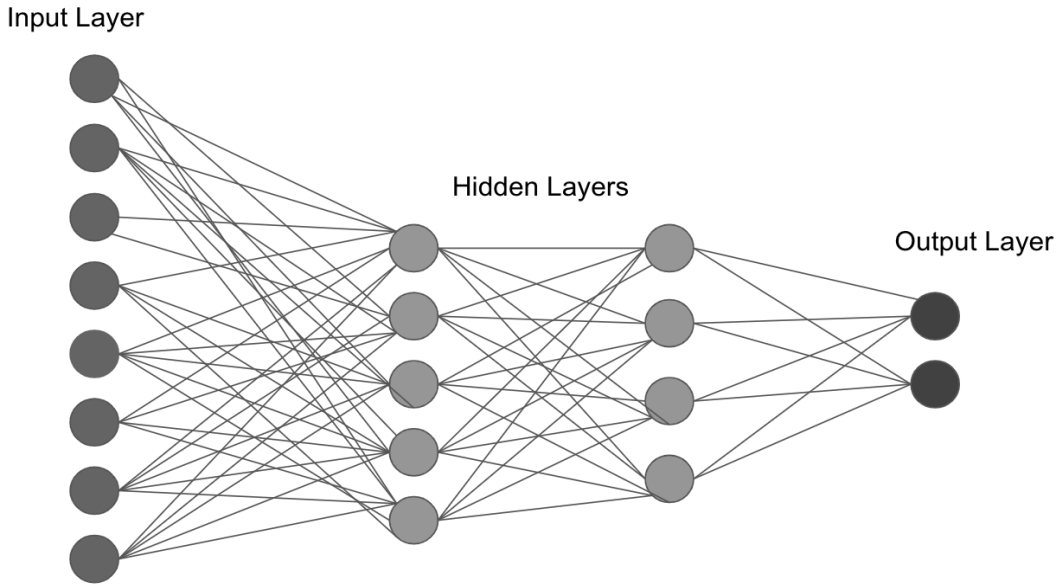

Figure 4: Multilayer perceptron classifier

3. The number of features was input as 8.
4. Trained the training data with the multilayer perceptron classifier.
5. Computed the accuracy on the test set and evaluated the accuracy.

3.3.3 Evaluation

The algorithm was then evaluated on a total number of 2,598,940 accounts where the number of sockpuppet accounts was 38 and non-sockpuppet accounts was 2,598,902. Some of the common machine learning evaluation metrics were used for the evaluation which are explained below.

3.3.3.1 Confusion Matrix

The confusion matrix is a metric that is often used to measure the performance of a classification algorithm. The confusion matrix is of the following form.

	Predicted	**Predicted**
Actual	True Negatives (TN)	False Positives (FP)
Actual	False Negatives (FN)	True Positives (TP)

Table 1: Confusion Matrix

The four cases in the matrix are explained further.

True Positives (TP): The cases for which the classifier predicted sockpuppets and the accounts were actually sockpuppets.

True Negatives (TN): The cases for which the classifier predicted non-sockpuppets and the accounts were actually non-sockpuppets.

False Positives (FP): The cases for which the classifier predicted sockpuppets, but the accounts were actually non-sockpuppet accounts.

False Negatives (FN): The cases for which the classifier predicted non-sockpuppets, but the accounts were actually sockpuppets.

We obtained the following results for confusion matrix on the classifier.

	Negative (predicted)	**Positive (predicted)**
Negative (actual)	1,819,250	0
Positive (actual)	24	0

Table 2: Confusion matrix results

3.3.3.2 Accuracy

Accuracy = (true positives + true negatives) / (true positives + true negatives + false positives + false negatives)

= 1,819,250 / 1,819,274 = 99.99%

A 99.99% accuracy was obtained on the datasets.

3.3.3.3 Precision, recall and f1-score

Precision is the fraction of relevant instances among the retrieved instances. It is given by,

Precision = true positives / true positives + false positives = 0%

Recall is the fraction of the total amount of relevant instances that were actually retrieved.

Recall = true positives / true positives + false negatives = 0 / 0 + 24 = 0%

F1 Score is simple the harmonic mean of precision and recall and given by,

F1 = 2 * ((precision * recall) / (precision + recall)) = 0%

3.3.4 Conclusion

Even though the percent of accuracy is a 99.99% on the dataset with multi-layer perceptron classifier, upon further evaluation, a 0% F1-score was obtained which means the model failed. Some of the possible theories for the obtained results are indifference or very less difference between the vocabulary of posts, fewer number of sockpuppet accounts used for training the dataset and only a fraction of the list of sockpuppet accounts used actually contributed to the dataset as only some of those accounts posted or commented on other posts. These possible theories could have resulted in data imbalance and thus failed to predict sockpuppet accounts with higher accuracy and precision. Thus, the multi-layer perceptron classifier was further evaluated using a much smaller dataset with equal number of sockpuppets and non-sockpuppet accounts.

3.3.5 Re-evaluation of Multi-layer perceptron classifier on a smaller dataset

The algorithm was re-evaluated on a number of 92 accounts with 46 sockpuppet accounts and 46 non-sockpuppets. We obtained the following results for confusion matrix on the classifier with a random split of 70:30 train and test data ratio on the dataset.

	Negative (predicted)	**Positive (predicted)**
Negative (actual)	7 (TN)	12 (FP)
Positive (actual)	3 (FN)	4 (TP)

Table 3: Confusion matrix results

On a dataset of 92 accounts with equal number of sockpuppets and non-sockpuppets, the test set accuracy was 73.08%, which seems more realistic than the 99.99% test set accuracy that was obtained with 38 sockpuppet accounts and 2,598,902 non-sockpuppet accounts. These results will be discussed more in Chapter 5.

CHAPTER 4: DEVELOPMENT

4.1 Introduction:

With rapidly growing sockpuppet accounts, social media sites are overwhelmed with fake ideas and opinions and it has created an unhealthy online community. Reddit is one of such social networking sites which is a network of communities based on people's interests. According to Reddit's 2017 Transparency Report, there were 944 reported fake accounts and there are many added every year. Thus, the motivation for implementing and studying the analysis of sockpuppet and non-sockpuppet accounts raised.

4.2 Objective:

The objective of this thesis is to be able to study and analyze a list of accounts on Reddit and categorize them as a sockpuppet account or a non-sockpuppet account using TF-IDF, K-Means algorithm and Multi-layer perceptron classifier.

4.3 Technologies Used:

4.3.1 PySpark - PySpark is the Python API for Spark. It is a great language for performing exploratory data analysis at scale, building machine learning pipelines, and creating Extract Transfer Load (ETL) for data platforms (Spark Python API Docs). Here are some fundamentals of PySpark used in this thesis:

4.3.1.1. Resilient Distributed Datasets (RDDs):

RDDs are the building blocks of any spark application. RDD stands for:

Resilient: It is fault tolerant and is capable of rebuilding data on failure.

Distributed: Data is distributed among the multiple nodes in a cluster.

Dataset: Collection of partitioned data with values.

RDDs are immutable elements and thus cannot be changed once an RDD is created. Multiple operations can be applied on RDDs to achieve a certain task. There are two ways of applying operations on RDDs:

Transformation: These operations are applied on an RDD to create a new RDD. Filter, groupBy and map are the examples of transformations.

Action: Spark uses a master/slave architecture. It has one central coordinator, the driver, that communicates with many distributed workers, called executors. The driver is the process where the main method runs. First it converts the user program into tasks and after that it schedules the tasks on the executors. Executors are worker nodes' processes in charge of running individual tasks in a given Spark job. They are launched at the beginning of a Spark application and typically run for the entire lifetime of an application. Once they have run the task, they send the results to the driver.

Action are the operations that are applied on RDD, which instructs Spark to perform computation and send the result back to the driver.

The operations count, collect, foreach, etc. can be applied on an RDD as an action.

4.3.1.2 DataFrame - It is the distributed collection of structured or semi-structured data. The data in the dataframe is stored in rows under named columns which is similar to the relational database tables. DataFrames are immutable in nature and supports a wide range of formats like JSON, CSV, TXT, etc. DataFrames can also be loaded from the existing RDDs or by programmatically specifying the schema.

4.3.1.3 User Defined Functions: Spark stores data in dataframe or RDD datasets, so we cannot create our custom function and run that against spark directly. The user has to register the function first which can be achieved by writing user defined functions.

4.3.1.4 Spark Schema: Spark Schema defines the structure of the data. It is the structure of the DataFrame, and Spark SQL provides StructType and StructField classes to programmatically specify the schema. StructType provides ArrayType, StringType, DoubleType, etc. according to the datatype of the values in the DataFrame.

4.3.1.5 MLlib (Machine Learning Library): In PySpark, a Python library called MLlib facilitates Machine Learning which performs data analysis using machine-learning algorithms like classification, clustering, linear regression and a few more.

4.3.2 Hadoop - Apache Hadoop is an open-source framework designed for distributed storage and processing of very large data sets across clusters of computers. Apache Hadoop consists of components including the following specific to this thesis:

4.3.2.1 Hadoop Distributed File System (HDFS): It is the bottom layer component for storage. HDFS breaks up files into chunks and distributes them across the nodes of the cluster.

4.3.2.2. Yarn: It is used for job scheduling and cluster resource management.

4.3.2.3 MapReduce: It is used for parallel processing.

4.3.2.4 Hadoop Libraries: Common libraries needed by the other Hadoop subsystems (HDFS Architecture Guide).

4.3.3 Linux- Linux is a family of open source Unix-like operating systems based on the Linux kernel. It is the leading operating system on servers and other big systems such as clusters.

4.4 Submitting Applications:

There are two ways of submitting the PySpark applications:

1. spark-submit: The spark-submit script in Spark's bin directory is used to launch applications on a cluster. The spark-submit script can use all of Spark's supported cluster managers through a uniform interface so the application does not have to be configured separately for each cluster.

2. pyspark-shell: Spark shell is an interactive shell through which we can access Spark's API. Spark provides the shell in Python and it can be started in command line with the command 'pyspark'.

4.5 Conclusion:

For the development of the algorithms to detect sockpuppet accounts from a pool of users in Reddit, the major technologies used were pySpark, Hadoop and Linux. The fundamentals of pySpark used in this thesis were RDDs, DataFrames, User defined functions, spark schema and machine learning libraries in spark. Parallel programming was a main factor in the development as these algorithms were implemented with parallel solutions in mind.

CHAPTER 5: FUTURE WORK

Currently, because of the high volume of the data and the time required to run all the data in the server, one file is being read through at a time but eventually, the goal is to be able to run all the files at once and detect sockpuppet accounts from a pool of users in Reddit. The test set accuracy was calculated to be 99.99% from the multi-layer perceptron classifier which seems great, but further evaluating the results, a 0 F1-score was obtained. I believe this result was obtained because there are very few numbers of sockpuppets in comparison with the number of non-sockpuppets due to which the dataset for training does not have enough sockpuppets. Less data for training the dataset could lead to lack of generalization, data imbalance and difficulty in optimization (Maheswari, 2019). But re-evaluating the dataset with multi-layer perceptron classifier on a small dataset with equal number of sockpuppet accounts and non-sockpuppet accounts gave a test set accuracy of 73.08%. The next step is to balance the data with both sockpuppet and non-sockpuppet accounts with over-sampling or under-sampling under further study of the data and hopefully obtain better results in the detection of sockpuppet accounts on Reddit.

REFERENCES

Morgan, J. (2017, April 1). Sockpuppets, Secessionists, and Breitbart. Retrieved from https://medium.com/data-for-democracy/sockpuppets-secessionists-and-breitbart-7171b1134cd5

Stewart, E. (2019, May 23). Facebook has taken down billions of fake accounts, but the problem is still getting worse. Retrieved from https://www.vox.com/recode/2019/5/23/18637596/facebook-fake-accounts-transparency-mark-zuckerberg-report

Reddit's 2017 transparency report and suspect account findings. (n.d.). Retrieved from https://www.reddit.com/r/announcements/comments/8bb85p/reddits_2017_transparency_report_and_suspect/

Sockpuppet (Internet). (2020, March 2). Retrieved from https://en.wikipedia.org/wiki/Sockpuppet_(Internet)

Weber, B. (2019, January 21). Methods for Parallelization in Spark. Retrieved from https://towardsdatascience.com/3-methods-for-parallelization-in-spark-6a1a4333b473

Welcome to Spark Python API Docs! ¶. (n.d.). Retrieved from https://spark.apache.org/docs/latest/api/python/index.html

HDFS Architecture Guide. (n.d.). Retrieved from https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

“Spark MLlib TF-IDF - Example.” *TutorialKart*, www.tutorialkart.com/apache-spark/spark-mllib-tf-idf/.

Dabbura, I. (2019, September 3). K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. Retrieved from https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

Maheswari, J. P. (2019, April 23). Breaking the curse of small datasets in Machine Learning: Part 1. Retrieved from https://towardsdatascience.com/breaking-the-curse-of-small-datasets-in-machine-learning-part-1-36f28b0c044d

The Original (July 16, 2011).The Computer Language Company. Retrieved from https://web.archive.org/web/20110716025003/http://www.techweb.com/encyclopedia/defineterm.jhtml?term=meat+puppet

Hastie, Trevor. Tibshirani, Robert. Friedman, Jerome (June 2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York, NY.