Implementation of Various Classifiers using Mahout through Hadoop

Annetoinette Figueroa, Cassandra Ho, Aaron Kuo, Sahan Paliskara, Eric Yuan

Abstract

There is a multitude of methods used to classify data into subcategories. However, due to the numerous algorithms for classifying, it is difficult to know which method is preferred. The algorithms involved in this experiment were based on Mahout, a tool for Hadoop. Three of the more common classification algorithms are Complementary Naive Bayes, Naive Bayes, and Stochastic Gradient Descent. Each of the classification algorithms were executed on varying amounts of data to find which method was ideal in terms of reliability and accuracy when classifying information. Classification is used frequently by large companies, such as Facebook and Amazon, to provide services including friend recommendations and item recommendations, respectively. This is accomplished through the pattern recognition and processing of classification algorithms similar to the ones that were tested in this experiment. By having more accurate and reliable classification algorithms, companies and people in general will be able to benefit from more streamlined predictions of what is wanted or needed. With the growing internet presence of people around the world, more data is available to be analyzed as the basis of recommendations, so this area of research is increasingly important.

Nomenclature

HDFS	Stands for Hadoop Distributed File System. It's a system tool that is incorporated into Hadoop and is designed to run efficiently on low-specification systems, with the use of data-nodes and name-nodes.
GPU	Graphics processing unit; processes images for display.
RAM	Random access memory; memory within the computer used for temporary data storage.
SGD	Stands for Stochastic Gradient Descent. It is one of many classification algorithms used in the process of machine learning.
C Naive Bayes	Stands for Complementary Naive Bayes. Another classification algorithm also based on Bayes' Theorem. However, it is a little different from Naive Bayes, which is another classification algorithm.

Introduction

Hadoop is a framework that is designed to run off of multiple nodes while handling faults. Hadoop contains numerous tools including Spark, MapReduce, and Mahout. Spark allows Hadoop to run up to one hundred times faster, though more memory is required. Another tool, MapReduce, allows Hadoop to process data more easily through parallelization, which is the act of running data through multiple nodes then compiling the processed data. Mahout is a tool within Hadoop that supports machine

learning, which is the ability of computers to detect patterns based on previous data. Mahout is also able to classify, provide recommendations, and cluster data.

This experiment focused solely on three main Mahout classification algorithms: CNaive Bayes, Naive Bayes, and Stochastic Gradient Descent. The algorithms were compared across different amounts of data for their accuracy and reliability. By testing this, the ideal classification algorithm could be determined.

Literature Review

Hadoop:

Hadoop is a framework that supports data processing through multiple computers. While Hadoop is traditionally used with cloud computing, it can also be used on individual computers and servers. Hadoop can run on one to thousands of computers, each providing storage. Also, Hadoop can find and deal with failures by directing work to other computers as necessary when failures occur. Hadoop's architecture consists of Hadoop Common, Hadoop HDFS, Hadoop MapReduce, and Hadoop YARN. [1][2]

Hadoop Distributed File System (HDFS):

The Hadoop Distributed File System (HDFS) is a system designed to run off a multitude of cheap technology. HDFS has many applications with big data (often gigabytes to terabytes) and has a large amount of fault tolerance. The architecture consists of a single server, called NameNode, which manages the access files and other devices. The system provides files that store data in sequences of blocks known as DataNodes. Except for the final DataNode, all DataNodes are the same size. To prevent faults, the blocks are copied by NameNode. Due to the intelligent nature of HDFS, it is able to recognize errors and recover quickly with the use of the earlier DataNode copies. Apart from fault tolerance, HDFS also features streaming access to data. HDFS applications are created to be written and closed without needing change in the future, allowing simplification of data coherency - the consistency of data across the multiple nodes. Additionally, HDFS allows applications to move to the location of data, which is useful in the case of big data where it's more efficient to move the applications themselves rather than all of the data. As such, HDFS is also designed to be easily portable. A possible error in HDFS is a loss of connection between the DataNode and NameNode, where the data from the DataNode can't be transferred to HDFS. [3]-[5]

JobTracker and TaskTracker:

The JobTracker is used to submit jobs. Clients submit jobs to the JobTracker, which communicates with the NameNode to determine the location of certain data sets. Upon finding the location, the JobTracker locates TaskTracker nodes with available slots at or near the requested data. Then, the JobTracker submits work for the TaskTracker nodes to carry out and monitors the TaskTracker nodes; if the TaskTracker nodes don't submit periodic signals quickly enough, they are labeled as failed nodes, and the work is given to other TaskTracker nodes. TaskTracker nodes will also notify JobTracker if a task has failed, so the JobTracker can decide future steps: JobTracker chooses to either submit the job to other TaskTracker nodes elsewhere or drop the task altogether. When the work is completed, the JobTracker will update its status, which client applications can parse info from to monitor the

performance and location of data. The JobTracker is a weak point of the Hadoop MapReduce service, since if JobTracker crashes, all running jobs instantly stop. [6][7]

TaskTracker is run through the DataNodes. They oversee the Mapper and Reducer tasks run through the DataNodes. TaskTracker nodes also give Mapper and Reducer tasks to the JobTracker to run. They constantly talk to the JobTracker to oversee progress and are given jobs from the JobTracker. TaskTrackers have slots; the number of slots is proportional to the jobs the TaskTracker can do. JobTrackers look for empty slots to give tasks. The TaskTracker nodes creates a JVM process to finish the work; the JVM process also prevents TaskTracker failure. [6][7]

Hadoop MapReduce:

Hadoop MapReduce is a software framework designed to make processing multi-terabyte sized data sets much easier by running them in-parallel on large clusters, which can consist of thousands of nodes of commodity hardware. The framework sorts outputs of maps, which are then outputted to reduced tasks. The framework, which consists of a single master, JobTracker, and one slave, TaskTracker, per node, takes care of scheduling and monitoring. [8]-[12]

MapReduce requires applications to specify the input and output locations and supply map and reduce functions via implementations of appropriate interfaces and abstract classes. This along with other parameters creates the job configuration that Hadoop job client submits to JobTracker. The Hadoop framework is implemented in Java as it is used in this experiment, but MapReduce can be used with other languages. MapReduce operates on key value pairs <key, value> similar to a hash table. These pairs are interpreted as maps where they are sorted and combined. Then they are reduced into the output files. The following line is how the input of a job is viewed:

(input) <k1, v1> -> **map** -> <k2, v2> -> **combine** -> <k2, v2> -> **reduce** -> <k3, v3> (output) [8]-[12]

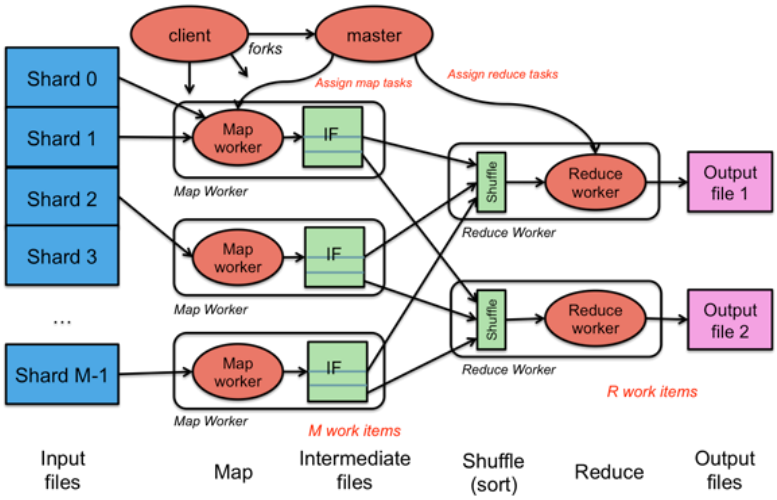

Fig. 1. A visualization of MapReduce. [13]

MapReduce is beneficial due to its scalability, speed, and simplicity; however MapReduce lacks in its ability to multitask and security. MapReduce possesses the ability to have a large amount of nodes, making it extremely cost-effective and easy to use on large amounts of data. It is also extremely flexible as it is capable of accessing many different types of data, and is relatively time efficient because it allows for terabytes of data to be processed in mere minutes. Next, its simplicity and community are advantageous as there is a wide range of support to develop through MapReduce provided by the Apache community. Unfortunately, MapReduce does not deal with isolated jobs running simultaneously very well. The large overhead of MapReduce makes it inefficient in running relatively small jobs. It is also less work to just use a stand-alone system instead of breaking up the task using MapReduce, so on small data sets it becomes unneeded. It also has security issues. Hadoop inherently does not have encryption at storage and network levels, so one's data is not as protected as it could be; Hadoop is connected to multiple machines, which making data less secure while MapReduce is active. [8]-[12]

Mahout:

Mahout is a type of machine learning, developed in 2008. Designed to be an open source so people could create accessible machine-learning algorithms, Mahout has been used in many Map-Reduce algorithms for clustering, classifying data, along with initializing and creating matrix and vector libraries. Mahout has been split into three main subcategories: classification, recommendation, and clustering. [14]

I. Classification

Classification is part of predictive analytics, which tries to create technology that can make human-like decisions, and involves a choice from potential answers. Classification can be used for predicting and detecting, by looking at patterns of behavior and identifying whether or not an action fits within the behavior. It also includes sorting of any type of data such as emails, messages, music, and movies into intelligent subcategories. Email providers such as Google and Yahoo use machine learning to track and analyze users' habitual responses in order to sort spam from worthwhile emails. Classification can also be applied to credit card fraud detection by predicting the card-owner's usual spending habits and locations and determining whether or not a transaction is fraudulent. [15][16]

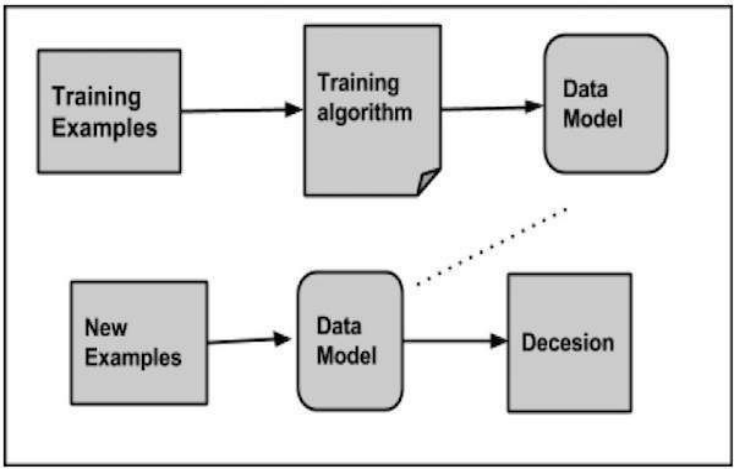

Fig 2. A visualization of Mahout classification. [16]

Classification involves creating data and models and predicting where new data will be assigned. Models are updated through training: it looks at examples and attempts to predict the original decisions.

The model is tested by giving 90% of the data to the model to try to predict the remaining 10% of the data; this repeats in order to continue updating the model. The model attempts to classify based on the features, which are discovered through measuring and calculating. Clues can be given to the model to help the model classify the examples, which can be continuous, categorical, word-like, or text-like. Continuous examples have numerical values; categorical refers to predetermined values; word-like values have an infinite possibility of predetermined values; and text-like values refer to sequences. [15]

Naive Bayes classifiers apply Baye's Theorem of probability. In general, they are highly scalable in general and are used to construct classifiers. Classifiers are models that create class labels for features and instances of data, based on the assumption that features of objects are independent from each other. [17]

Another type of classification learning algorithm is the stochastic gradient descent (SGD) algorithm, which is an iterative approach that changes a model slightly with each training example to be more correct for one example. After a modest number of examples, SGD is capable of accurately classifying new data, provided that some special tricks are used to modify how much the model should be changed each time. SGD algorithms are difficult to effectively parallelize, but with the fast runtime of SGD algorithms, parallelizing is often unnecessary. Other than its speed, SGD advantages in its scalability, as only simple operations are performed between examples for constant amounts of memory. This means that twice the amount of data takes twice the processing time. [15]

Within Mahout, there is supervised and unsupervised learning. Classification is part of supervised learning, where preferred answer is provided within the data, while unsupervised learning searches for an answer. Supervised learned is given to the machine in order to test the system. Within a classification project, there are three steps: training, testing, and using the model. While using the model, more training can undergo. Training involves describing categories, collecting data, and defining the variables for the features. Accuracy is tested prior to using the model, and new data must be tested to prevent a decrease in quality. The cycle then repeats of training, testing, and using. [15]

II. Recommendations

Recommendation in Mahout provides users with suggested items in lists. These recommendation lists are created with recommender engine algorithms, which are split into two main types: user-based and item-based. User-based recommendations are created from a series of attributes provided by a user of a specific program or shopping website. Based on the user's feedback, such as rating, purchase, and other factors, on certain items, Mahout is able to compare that data with the records and habits of other users of the same program and provide recommendations. For example, if Person A rated Movie A five stars, and Person B rated Movie A and Movie B five stars, then a Mahout recommendation process may suggest Movie B to Person A. Meanwhile, item-based recommendation is based on individual elements. For example, on Amazon's shopping website, they collect data based on user purchases where if many people buy Item A and B together, then if a person buys only Item A, Amazon will recommend Item B for purchase to the user. Then there are content based recommendation. In Mahout, there are data preferences, where the program gathers data by asking people to rate items. Then, based on similar ratings, the program can also recommend other items. The program simulates how someone will like something based on test data, a part of the actual data set. Preferences can be scored by looking at the difference in rating between the actual rating and estimated rating. One method of rating is known as the root-mean- square, where the program takes the differences, squares them, adds them, divides by the total number of items, and then takes the square root. Another method of rating is known as the average

difference, where the average is taken of the differences. There is also precision and recall, where "precision is the proportion of top recommendations that are good recommendations, and recall is the proportion of good recommendations that appear in top recommendations. However, both precision and recall are based on how someone rates the item. [15][18][19]

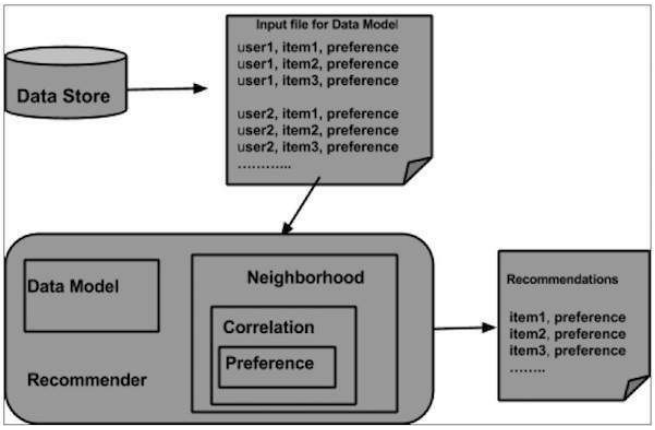

Fig. 3. A visualization of Mahout Recommendations. [19]

In Mahout, preferences are inputted into arrays based on the user in the general data model. There are also filedata models, similar to the general datamodel, except the data compresses. Filedata models allow for updating information, where files and preferences can be changed. When data is too large, it can be entered into a database. There are also Boolean preferences, which don't have preference numbers. Sometimes data is disregarded, especially when a person dislikes one type of item but likes another type, except both are part of a larger group of item. Mahout remembers a liking for the larger type of item and forgets the ratings of the original two items. However, boolean and non-boolean data can't be both included within a data set. [15][18][20]

In Mahout, good recommendations involve numerous portions, including data sets, user similarity results, and recommender programming. Similarity in users (user-based recommendations) involve looking at how often the other user chooses a number, known as the Pearson Correlation. There can be a positive and opposing relationship. There is another method, where the distances between users are measured. Each user is represented with a point. Points closer in distance are more similar users, in their likes and dislikes. The Spearman Correlation re-rates preferences, giving the least liked item a rating of one, the next-least liked item a two, and so on. Then, it follows the Pearson correlation. There is also the Tanimoto Coefficient, which looks at two user's preferences and looks at the overlap and the log-likelihood test, which tries to guess preferences if one isn't given. However, item-based recommendations involve a data set and a comparison of items. It also runs more quickly than user-based recommendations. There are slope-one recommendations, where preferences are inferred based on the difference in rating between two items. It uses massive amounts of data, however. There are Singular Value Decomposition-based recommenders, where the program tries to lessen time by looking at specific features, Linear Interpolation-based recommenders, Cluster-based recommendations, and Model-based recommendations, which looks at preferences. Then, the program predicts how well the user will like a new item. [15][18][20][21]

III. Clustering

Clustering involves grouping items based on similarities, such as the words in the items' name or word counts for similar words. Unfortunately, clustering based on the name limits similarity based on plot, and clustering based on similar words is time-consuming. In clustering, each word is given a weight; less weight is given to commonly seen words, while more weight is given to less frequently seen words. Then, the sum of the weights could measure the similarity between two similarly lengthed books; however, this method depends on the length of the text or item. One weighing method is known as Term Frequency-Inverse Document Frequency. Clustering is commonly used in research data processing to find a specific group of data or means and in identifying, sorting, and classifying data. In the real world, clustering is used to derive plant and animal taxonomies and categorize the DNA in plants and animals based on their structures. [15][22]

There are several types of clustering: k-means, fuzzy k-means, canopy, etc. In Mahout, data is inputted as vectors, which are saved for the clustering formula. Through k-means, the vectors are converted into points. The points undergo Euclidean Distance measuring to judge the similarity. Mahout clusters points into multiple sets, but with more data the clustering becomes more difficult. Another method of distance measuring is Manhattan Distance Measure, which takes the absolute difference of each of the coordinates. There is also the Cosine Distance Measure, the Tanimoto Distance Measure, and the Weighted Distance Measure. [15]

K-means clustering limits the number of clusters and looks at the distances between the clusters; the process repeatedly finds points and then estimates the center of the points. Canopy clustering guesses the number of clusters as well as their position, allowing it to be faster than k-means clustering. However, the algorithm for canopy clustering needs to know the size of a cluster. It is extremely fast, but it isn't always accurate. Problems that occur during clustering include: exclusive clustering, overlapping clustering, hierarchical clustering, and probabilistic clustering. K-means tries to delegate each point to a specific cluster, but in fuzzy k-means points can belong to multiple clusters, resulting in overlapping clustering. Model-based clustering tries to overcome problems with the other methods through Dirichlet Clustering. [15]

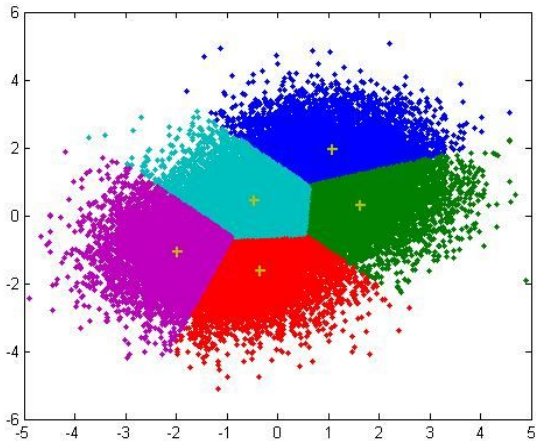

Fig. 4. A visualization of K-Means Clustering. [23]

Spark:

Hadoop, being an open-source software, facilitated the creation of various tools, including Pig, Hive, Spark, Cassandra, and Mahout, to perform varying tasks with modified versions of Hadoop. [5] One of those tools, Apache Spark, was first developed in 2009 by Matei Zaharia in the UC Berkeley AMPLab for fast computation with cluster computing technology. The purpose behind its development was to speed up Hadoop computation processes, but Spark itself is not dependent on Hadoop. Instead, Spark has its own cluster management system, and uses Hadoop for storage and processing. [24]

Apache Spark is mainly built for fast speed, good analysis, and to be easy to use. Unlike Hadoop, Spark has a large unified framework that can deal with a very diverse field of data and data gathering techniques. For example, Spark can get text data, graph data, and numerical data. The source of the data can also be diverse in Spark in that it can come many sources, including real time data streaming. This allows processes to run about 100 times faster on Hadoop clusters with Spark than they would normally. [24] However, Spark requires much more memory than Hadoop, so Spark is usually used for low latency, streaming, and parallel computing applications while keeping Hadoop for large data set storage. [25]

Spark's architecture consists of three main parts: Data Storage, Management Framework, and the Application Program Interface (API). Data storage for Spark is accomplished with HDFS and works with any Hadoop compatible data sources, and for management framework, Spark can be a stand-alone server or on distributed computing frameworks line Mesos or YARN. The API allows developers to create Spark based applications. Spark contains built-in APIs for Java, Python, or Scala and comes with up to 80 high-level operators for interactive querying. [24][26]

Other the API, Spark contains libraries that add more applications of Spark on Big Data analysis and Machine Learning. Spark Streaming supports the processing of real-time streaming data by using the DStream which is essentially several Reliable Distributed Datasets (RDDs) to process the streaming data. Spark SQL allows structured data processing, letting users Extract, Load, and Transfer (ETL) data into different formats for ad-hoc querying, the obtaining of information when needed. Next, Spark GraphX is a Spark API for graphs and parallel computation of graphs. GraphX supports graph computation by exposing fundamental operators, and it also includes graph algorithms and builders to make graphing analysis easier. Machine learning is possible in Spark with Spark MLib, which consists of common learning algorithms such as classification, regression, clustering, and more, along with the optimized primitives enabling those algorithms. [27]

Apart from those libraries, Spark contains BlinkDB, an engine for running interactive SQL tasks for big data that trades query accuracy for response time, Tachyon, a distributed file system centered on memory, and integration adaptors that allow Spark use with other products such as Cassandra and R (the integration adaptors for those are Spark Cassandra Connector and Spark R, respectively). [27]

Project Description

Using the 20 NewsGroup dataset (20 groups consisting of an approximate total of 20000 news documents), three classification algorithms (CNaive Bayes, Naive Bayes, and Stochastic Gradient Descent) were tested for accuracy and reliability. The datasets were evaluated based on the confusion matrices, created by the algorithms' models. The testing data was partitioned to evaluate how different amounts of data affected the accuracy and reliability. The different levels of data used were 40%, 60%, 80%, and 100%. This would provide the data needed to determine the most effective algorithm. Our final evaluation was based on both the reliability and accuracy of each algorithm.

Method

The three classification algorithms used for the experiment were CNaive Bayes, Naive Bayes, and Stochastic Gradient Descent. The first two, CNaive and Naive Bayes, are based off of Bayes' Theorem (Equation 1). Bayes Theorem describes the probability of an event based on the condition that features are independent of each other.[16]

Bayes' Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

The third method is based off of the Stochastic Gradient Descent (SGD) algorithm, where a model is created and edited with every example, resulting in a model more targeted for a specific example. [14]

Process

1. First we created a virtual machine created through Oracle VM VirtualBox 5.1 running Ubuntu 14.04.4 with the specifications of 4 cores of an Intel i7 4771,4 gigabytes of RAM, and a 128 megabyte virtual GPU.
2. We then downloaded and configured Apache Hadoop 2.7.2. on the virtual machine.
3. We created a single namenode in the virtual machine.
4. We installed the Apache Mahout 0.12.2 library through Apache Maven 3.3.9.
5. Then, we downloaded the 20 NewsGroup data set.
6. We then tested the native CNaive Bayes, Naive Bayes, and SGD algorithms using the classifiers provided by Apache during installation, while using 100% of the data as input through the following procedure.
    a. The program first partitioned 80% of the dataset that was inputed data as training data and 20% as testing data.
    b. Then the chosen algorithm created a model based on the the training data.
    c. Then the model was checked for both accuracy and reliability using the testing data.
    d. The accuracy and reliability were then recorded.
7. Afterwards, we repeated steps 6. However, this time we changed the amount of data input data to ≅80%(64% of the entire dataset), than ≅60%(48% of the entire dataset), and finally ≅40%(32 of the entire dataset) by deleting 200 random articles from each news source each time.

Data Results and Discussion

Data Results

Our data seemed suggest that on average, the naive bayes method of classification was the most accurate classification method that we tested. It also had the highest reliability percentage compared to the other methods. As shown in table 1 below, naive bayes consistently scored higher on the accuracy test when compared to the other classification algorithms. This was true across all the 4 levels of the independent variable, which was the amount of training data that was given to the classification algorithm. This seems to suggest that naive bayes is more accurate and precise than the other methods. However, it is important to consider that while it was consistently more accurate than the CNaive Bayes method, this was only by an average of about 1.4381% in accuracy. This combined with a difference of 1.4374% in reliability shows that the performance is almost the same. However, both methods exceeded the performance of the SGD on both tests. CNaive Bayes scored an average of 13.9185% better than the SGD on the performance test, while the naive bayes scored an average of 15.35657% better on the performance test when compared to the SGD method. This is also shown in the reliability data, as CNaive Bayes and naive bayes scored 13.83833% and 15.27575%, respectively, above the SGD average accuracy.

All together, this data seemed to support that naive bayes is the best classification algorithm that we tested. This is followed shortly by CNaive Bayes, which had results that were almost as good as the results of naive bayes. However, SGD seemed to be the worse classification algorithm, having results that were significantly lower than the other two algorithms.

Table 1: This table shows the accuracy of different classification algorithms on the same set of data.

Types of Classification Algorithms	≅40% of the Data Set	≅60% of the Data Set	≅80% of the Data Set	100% of the Data Set	Average
Cnaive Bayes	88.5037%	88.7782%	89.7178%	88.7672%	88.94173%
Naive Bayes	90.3461%	90.1783%	90.4365%	90.5583%	90.3798%
SGD	74.376%	74.376%	74.376%	76.9649%	75.02323%

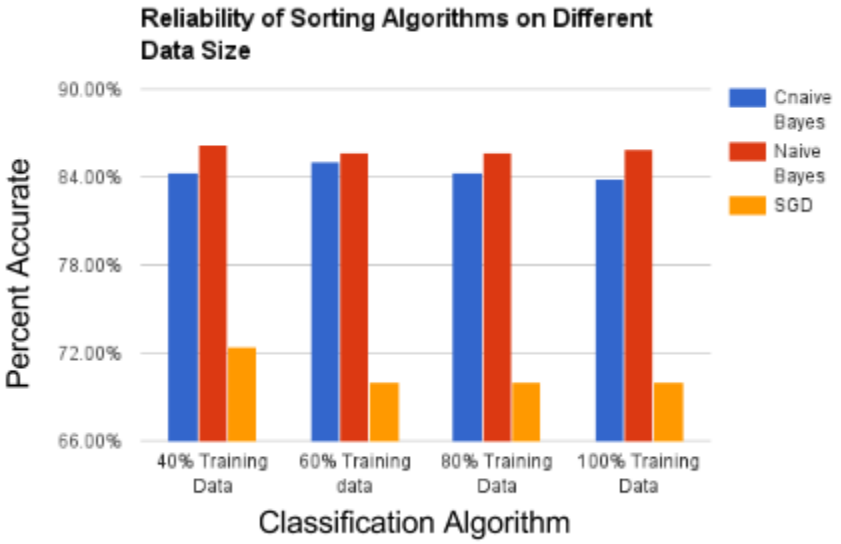

Fig. 5: This chart provides a visual representation of the accuracy of different classification algorithms.

Table 2: This table shows the reliability of different classification algorithms on the same set of data.

Types of Classification Algorithms	≅40% of the Data Set	≅60% of the Data Set	≅80% of the Data Set	100% of the Data Set	Average
CNaive Bayes	84.3435%	85.0623%	84.3639%	83.9352%	84.42623%
Naive bayes	86.1963%	85.7048%	85.6617%	85.8918%	85.86365%
SGD	72.3813%	69.9901%	69.9901%	69.9901%	70.5879%

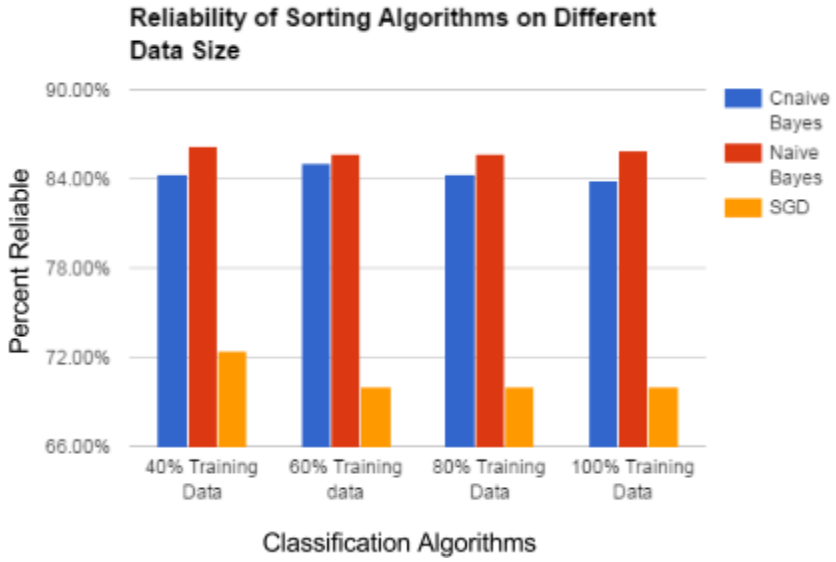

Fig. 6: This chart provides a visual representation of the reliability of different classification algorithms.

Conclusion

The experiment compared the accuracy and reliability of different algorithms with varying amounts of data for classification within Mahout. The Naive Bayes method was the most accurate with an average of 90.38% as well as the most reliable with an average of 85.86%, above the CNaive Bayes method and SGD method. Overall, there could have been errors with how the amount of data changed and within the algorithms in how accurate each classification was. It is possible that the Naive Bayes method was the best in this experiment due to the specifications of the data, particularly its size. Similarly, more trials could have been conducted to obtain more data to see how accurate the results were. In the future different sorting algorithms could be tested to see how they compare to the ones tested in this experiment. Future research could also allow us to process more data to find algorithms that exceed the accuracy and reliability of the ones used in this experiment.

Acknowledgements

The research team would like to thank Haysam Selim Abdelhamid and Sai Phani Krishna Parsa for providing us with the tools and knowledge necessary needed to carry out this research project. They were an immense help and were essential in the learning and research process for this paper. We would also like to thank the U.S. Department of Defense for funding this summer program.

References

[1] "Welcome to Apache™ Hadoop®!," (2016, Jun. 21). Apache. [Online]. Available: hadoop.apache.org

[2] "What Is Hadoop?," (n.d.). SAS. [Online]. Available: www.sas.com/en_my/insights/big-data/hadoop.html. Accessed Jun. 30, 2016.

[3] "HDFS Architecture Guide," (n.d.). Apache. [Online]. Available : hadoop.apache.org/docs/r1.2.1/hdfs_design.html. Accessed Jun. 30, 2016.

[4] "What Is the Hadoop Distributed File System (HDFS)?," (n.d.). IBM. [Online]. Available: www-01.ibm.com/software/data/infosphere/hadoop/hdfs/. Accessed Jun. 30, 2016.

[5] A. Kala Karun and K. Chitharanjan, "A review on hadoop — HDFS infrastructure extensions," *Information & Communication Technologies (ICT), 2013 IEEE Conference on*, JeJu Island, 2013, pp. 132-137.

[6] "JobTracker and TaskTracker - Hadoop In Real World," (2015, Jul. 14). Hadoop In Real World. [Online]. Available: www.hadoopinrealworld.com/jobtracker-and-tasktracker

[7] "TaskTracker," (n.d.). Apache. [Online]. Available: wiki.apache.org/hadoop/TaskTracker. Accessed Jun. 30, 2016.

[8] "MapReduce Tutorial," (n.d.). Apache. [Online]. Available: hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html. Accessed Jun. 30, 2016.

[9] "Hadoop MapReduce," (n.d.). TutorialsPoint. [Online]. Available: www.tutorialspoint.com/hadoop/hadoop_mapreduce.htm. Accessed Jun. 30, 2016.

[10] "Advantages of Hadoop MapReduce Programming," (n.d.). TutorialsPoint. [Online]. Available:

www.tutorialspoint.com/articles/advantages-of-hadoop-mapreduce-programming. Accessed Jun. 30, 2016.

[11] "What Are the Disadvantages of Mapreduce?," (n.d.). StackOverflow. [Online]. Available: stackoverflow.com/questions/18585839/what-are-the-disadvantages-of-mapreduce. Accessed Jun. 30, 2016.

[12] "What Are the Limitations of Hadoop?," (n.d.). Quora. [Online]. Available: www.quora.com/What-are-the-limitations-of-Hadoop. Accessed Jun. 30, 2016.

[13] P. Krzyzanowski. (2011, Nov.) "MapReduce: A framework for large-scale parallel processing," Rutgers. [Online]. Available: www.cs.rutgers.edu/~pxk/417/notes/content/mapreduce.html

[14] "Introducing Apache Mahout," (n.d.). IBM. [Online]. Available: www.ibm.com/developerworks/library/j-mahout/index.html. Accessed Jun. 30, 2016.

[15] Owen, Sean, Robin Anil, Ted Dunning, and Ellen Friedman. *Mahout in Action*. Shelter Island, NY: Manning Publications, 2012.

[16] "Mahout - Classification," (n.d.). TutorialsPoint. [Online]. Available: www.tutorialspoint.com/mahout/mahout_classification.htm. Accessed Jul. 7, 2016.

[17] "Naive Bayes classifier," (n.d.). Wikipedia. [Online]. Available: en.wikipedia.org/wiki/Naive_Bayes_classifier. Accessed Jul. 11, 2016.

[18] V. Eluri, M. Ramesh, A. Al-Jabri, M. Jane, "A Comparative Study of Various Clustering Techniques on Big Data Sets using Apache Mahout," *3rd Ann. Int. Conf. on Big Data and Smart City*, Vol. 3, 2016

[19] "Mahout - Recommendation," (n.d.). TutorialsPoint. [Online]. Available: www.tutorialspoint.com/mahout/mahout_recommendation.htm. Accessed Jul. 7, 2016.

[20] R. Esteves, R. Pais, C. Rong, "K-means clustering in the cloud - a Mahout test," *Workshops of Int. Conf. on Adv. Inform. Networking and Appl,* 2011

[21] L. Ma, E. Haihong, K. Xu, "The Design and Implementation of Distributed Mobile Points of Interest(POI) Based on Mahout," *6th Int. Conf. on Pervasive Computing and Appl,* 2011

[22] "Mahout - Clustering," (n.d.). TutorialsPoint. [Online]. Available: www.tutorialspoint.com/mahout/mahout_clustering.htm. Accessed Jul. 7, 2016.

[23] Y. Cao. (2008, Mar. 27). "Efficient K-Means Clustering Using JIT," MathWorks. [Online]. Available: www.mathworks.com/matlabcentral/fileexchange/19344-efficient-k-means-clustering-using-jit

[24] "Apache Spark Core Programming," (n.d.). TutorialsPoint. [Online]. Available: www.tutorialspoint.com/apache_spark/apache_spark_core_programming.htm. Accessed Jun. 23, 2016.

[25] A. C. Oliver. (2015, Jun. 5). "Straight talk on Apache Spark," JavaWorld. [Online]. Available: www.javaworld.com/article/2360185/big-data/straight-talk-on-apache-spark-and-why-you-should-care.html

[26] "Big Data Processing with Apache Spark - Part 1," (2015, Jan. 30). InfoQ. [Online]. Available: www.infoq.com/articles/apache-spark-introduction

[27] "Spark SQL, DataFrames and Datasets Guide," (n.d.). Apache. [Online]. Available: spark.apache.org/docs/latest/sql-programming-guide.html. Accessed Jul. 7, 2016.