ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

Support Vector Machines in Apache Spark

Ann Paul.
Department of Information Technology, Rajagiri School of Engineering & Technology, Kerala

Manuscript Info

Key words:
Support Vector Machine,
Text Categorization,
classifier,
Spark

Abstract

Complex and huge quantities of data are produced every second. These data need to be categorized. Hence, text categorization is the method of categorizing text documents into one or more predefined categories or classes. A number of methods are proposed for text categorization. The most popular method is the Support Vector Machine (SVM) because it is more efficient when compared to other proposed methods for text categorization. The objective of the paper is to implement SVM using Apache Spark and to predict the accuracy and confusion matrix of the classifier. Apache Spark is an open source framework that helps in computation of large datasets in an efficient way much faster than Hadoop. Spark is a computational tool that processes the data.

Introduction:-

BigData normally computes datasets that cannot be processed by normal means and are very much complex in nature. These data sets can be stored, analysed and processed to obtain more accurate results for various applications and other decision making process. There is an increase in the amount of data being generated each day from different sources. Today, most of the data are in digital format and they are exchanged through Internet at a rapid rate. The large volumes of data include photos from different users in social media websites, blogs, comments, meteorological data, remote sensing, location data streams of mobile subscribers and devices, and audio and video recordings, log files, database, cyber interactions between different users. According to researches, data are generated faster than ever in a figure that increases at a rapid rate. It is said that 90% of data in the world has been created in last 2 years. It is a mind boggling figure, but the sad part is instead of having more information, people feel less conversant. The speed at which the data is being generated is expected to increase more rapidly due to wide range of applications and uses of different kinds of data all over the world. All of the above reasons led to the evolution of Big Data as a new research area and led to the evolution of various methods of computation.

Data pertains to ranges varying from terabytes to yottabytes. It helps in processing wide range of data. It experiences challenges related to volume, velocity and variety. Big Data is comprised of 3Vs (i.e. Volume, Velocity and Variety). Volume means large amount of data, Velocity pertains to data arriving at high speed, Variety means data originating from heterogeneous resources. To this two more terms are added namely, Veracity and Value. Veracity refers to the messiness or trustworthiness of the data whereas value stands for having well and good access to big data but unless it‘s value is turned to useless. Success of Big Data depends on its analysis. Big Data analysis is an ongoing process in today‘s world. For this, it requires a set of activities instead of an isolated activity. Therefore, for finding the data and determining the new insights and then integrating and presenting those new insights properly to conferring unique goals requires a unified set of solutions.

Big Data is implemented through different kinds of framework that helps in processing the large amount of data that are huge enough to be handled by normal methods. These frameworks include Apache Hadoop, Spark, Drill and so on which supports databases such as HBase, SQL and so on. Big Data is a wide topic that can be researched and thought about for producing efficient results.

In Big Data definition, Big means a dataset which makes data to grow faster such that it becomes difficult to manage it by using existing data management concepts and tools. The various data processing methods in general include optimization, statistic methods, data mining, machine learning, signal processing, and visualization and text categorization. This paper makes use of the text categorization method to bring out the differences in performance factors of Hadoop and Spark. Categorization of text is of proves to be efficient in information retrieval and natural language processing systems. Today's Business is unexpectedly affected by this growth of Data.

Apache Spark:-

Apache spark is a framework for performing general data analytics on distributed computing cluster like Hadoop. It produces in memory computations for increased speed and data processing over MapReduce. It runs on top of existing Hadoop cluster and access HDFS. Spark uses more RAM instead of network and disk I/O it's relatively fast as compared to Hadoop. Spark uses data in memory and Hadoop uses data in disk. Despite the fact that it might not be possible for all the future allocations or existing applications to completely abandon Hadoop MapReduce, but there is a scope for most of the future applications to make use of a general purpose execution engine such as Hadoop Spark that comes with many more innovative features, to accomplish much more than that is possible with MapReduce Hadoop.

A number of frameworks and tools are used for the processing of large quantities of data such as Hadoop, Storm, Spark, Drill, Tez with an access to database such as HBase, MongoDB, MySQL, HiveQL and so on. These frameworks are efficient in processing the data in their own way. The data can be structured, semi-structured or unstructured. The tools are designed in such a way as to match with the scalability and granularity of the large quantities of data. There are a lot of Hadoop based frameworks that can used: hive, pig for data pre-processing and aggregation and mahout offers a range of large scale machine learning algorithms which were implemented in the underlying map-reduce paradigm, Giraph offers implementation in a wide range of graph algorithms. This paper offers study on Spark.

Text Categorization has spread its applications in a variety of areas such as Spam filtering, Transliteration, Auto-Content management, on-line new classification, etc. There are varieties of Text Categorization algorithms such as Support Vector Machine, Naive Bayes, Rule based Systems, K-Nearest Neighbour, etc. SVM is considered to perform better than most of algorithms (Saurabh et al., 2013) in Text Categorization problem as it can handle large feature set. Thus to test our experimental results we have used SVM as underlying algorithm using different sets of training documents.

Apache spark is a framework for performing general data analytics on distributed computing cluster like Hadoop. It produces in memory computations for increased speed and data processing over MapReduce. It runs on top of existing Hadoop cluster and access HDFS. Spark uses more RAM instead of network and disk I/O its relatively fast as compared to Hadoop. Spark uses data in memory and Hadoop uses data in disk. Despite the fact that it might not be possible for all the future allocations or existing applications to completely abandon Hadoop MapReduce, but there is a scope for most of the future applications to make use of a general purpose execution engine such as Hadoop Spark that comes with many more innovative features, to accomplish much more than that is possible with MapReduce Hadoop.

Spark is an Apache project advertised as "lightning fast cluster computing". It has a thriving open-source community and is the most active Apache project at the moment. Spark provides a faster and more general data processing platform. Spark lets you run programs up to 100x faster in memory, or 10x faster on disk, than Hadoop. Last year, Spark took over Hadoop by completing the 100 TB Daytona GraySort contest 3x faster on one tenth the number of machines and it also became the fastest open source engine for sorting a petabyte. Spark allows programmers to develop complex, multi-step data pipelines using directed acyclic graph (DAG) pattern. It also supports in-memory data sharing across DAGs, so that different jobs can work with the same data.

Spark takes MapReduce to the next level with less expensive shuffles in the data processing. With capabilities like in-memory data storage and near real-time processing, the performance can be several times faster than other big data technologies. Spark also supports lazy evaluation of big data queries, which helps with optimization of the steps in data processing workflows. It provides a higher level API to improve developer productivity and a consistent architect model for big data solutions. Spark holds intermediate results in memory rather than writing them to disk

which is very useful especially when you need to work on the same dataset multiple times. It is designed to be an execution engine that works both in-memory and on-disk. Spark operators perform external operations when data does not fit in memory. Spark can be used for processing datasets that larger than the aggregate memory in a cluster. Spark will attempt to store as much as data in memory and then will spill to disk. It can store part of a data set in memory and the remaining data on the disk.

Spark Architecture:-

Spark Architecture is mainly comprised of three components namely:

- Data Storage: Since Spark is built over HDFS, it makes use of this file system for the purpose of storing data. The other compatible data sources are HBase, Cassandra, etc.
- API: This is actually used by the developers to create standard Spark based applications using a standard API interface. Spark provides API for Scala, Java, and Python programming languages.
- Management Framework: Spark can be deployed as a Stand-alone server or it can be on a distributed computing framework like Mesos orYARN.

Resilient Distributed Datasets:-

Resilient Distributed Dataset (RDD) is the core concept in Spark framework. Spark stores data in RDD on di_erent partitions. They help with rearranging the computations and optimizing the data processing. They are also fault tolerance because an RDD has an idea on how to recreate and recompute the datasets. RDDs are immutable. The RDDs can be modified with a transformation but the transformation does return a new RDD whereas the original RDD remains the same.RDD supports two types of operations:

- Transformation: The transformation does take an RDD and it returns a new RDD whenever it is called. It does neither return a single value nor does it not get evaluated whenever a function call is made.
- Action: Unlike transformation, the action operation does evaluate as well as return a new value. Whenever the function is called on a RDD object, all the queries are computed and the time of call and the resultant value is returned.

Support Vector Machine:-

SVM is supervised machine learning mechanism. Improving classifier effectiveness has been an area of intensive machine-learning research over the last two decades, and this work has led to a new generation of state-of-the-art classifiers, such as support vector machines, boosted decision trees, regularized logistic regression, neural networks, and random forests. An SVM is a kind of large- margin classifier: it is a vector space based machine learning method where the goal is to find a decision boundary between two classes that is maximally far from any point in the training data (possibly discounting some points as outliers or noise). Support vector machines are based on the Structural Risk Minimization principle from computational learning theory. The idea of structural risk minimization is to find a hypothesis h for which we can guarantee the lowest true error. The true error of h is the probability that h will make an error on an unseen and randomly selected test example. An upper bound can be used to connect the true error of a hypothesis h with the error of h on the training set and the complexity of H (measured by VC-Dimension), the hypothesis space containing h. Support vector machines find the hypothesis h which (approximately) minimizes this bound on the true error by effectively and efficiently controlling the VC-Dimension of H.

SVMs are very universal learners. In their basic form, SVMs learn linear threshold function. Nevertheless, by a simple plug-in" of an appropriate kernel function, they can be used to learn polynomial classifiers, radial basic function (RBF) networks, and three-layer sigmoid neural nets. One remarkable property of SVMs is that their ability to learn can be independent of the dimensionality of the feature space. SVM‘s measure the complexity of hypotheses based on the margin with which they separate the data, not the number of features. This means that we can generalize even in the presence of very many features, if our data is separable with a wide margin using functions from the hypothesis space.

Support vector machines are supervised learning mechanisms that make use of computational models for the purpose of classification of huge datasets. Supervised Classifiers are a group of statistical machine learning techniques that attempt to attach a "class", or "label", to a particular set of features, based on prior known labels attached to other similar sets of features. This is clearly quite an abstract definition, so it may help to have an example. Consider a set of text documents. Each document has an associated set of words, which we will call

"features". Each of these documents might be associated with a class label that describes what the article is about. The support vector machine has gained more popularity than Naive Bayes because of their accuracy and it requires on small training samples.

The SVM has been successfully used in many applications, such as pattern recognition, multiple regression, nonlinear model fitting, and fault diagnosis. The idea of SVM classification is to transform the input data to a higher dimensional feature space, and find an optimal hyperplane that maximizes the margin between the classes. The group of examples that lie closest to the separating hyperplane is referred to as support vectors.

The classification of data into their respective categories is a tedious task as the size of dataset increases rapidly. While testing a model, suppose a set of data points exist in either two or more classes, the testing phase must be capable of classifying the new data into these classes respectively. In the case of support vector machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and the objective is to know whether these points can be separated with a (p-1)-dimensional hyperplane. This is called as a linear classifier. There are many hyperplane that might classify the data within the p dimensional space. The objective of SVM is to choose the best hyperplane. This is decided by the one that represents the largest separation, or margin, between the two classes. The hyperplane is chosen such that the distance from it to the nearest data point on each side is maximized. When such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier; or equivalently, the perceptron of optimal. The points that pass through these hyperplane is called as support vectors.

Proposed Method:-

The implementation of support vector machines was done using python programming language as it was simpler and easy to develop. The Apache Spark consists of machine learning library known as MLlib. MLlib is Apache Spark's scalable machine learning library. Spark excels at iterative computation, enabling MLlib to run fast. At the same time, algorithmic performance are also considered: MLlib contains high-quality algorithms that leverage iteration, and can yield better results than the one-pass approximations sometimes used on MapReduce. This library was used to develop the code.

- **Dataset:-**

The dataset can be obtained automatically from many many web-based articles and effectively extract their text-based data from the HTML with the help of Python libraries. This paper has made use of a famous dataset Reuters21578. It is one of the most widely used testing datasets for text classification. This dataset consists of a corpus that are predefined and tagged with selection of topics and the geographic location. The dataset is then downloaded and extracted.

- **Parsing:-**

The news articles in each folder were in .sgm format i.e. SGML files. The Standard Generalized Markup Language (SGML) is for defining generalized markup languages for documents. HTML was theoretically an example of an SGML-based language until HTML 5, which admits that browsers cannot parse it as SGML (for compatibility reasons) and codifies exactly what they must do instead. The latest version of Python does not support the sgmlllib. Hence, these articles had to be parsed before any training was done on them. For the purpose of parsing, HTMLParser was used. The first goal is to actually create the SGML Parser that will achieve this. To do this, subclass Python's HTMLParser class was used to handle the specific tags in the Reuters dataset. Upon subclassing HTMLParser, three functions are handled, which tell the parser what to do at the beginning of SGML tags, what to do at the closing of SGML tags and how to handle the data in between. It also handles the internal state of the class and to parse the actual data in a chunked fashion, so as not to use up too much memory.

After the completion of parsing, a list of predictor-response pairs was created. This is a list of two-tuples that contain the most appropriate class label and the raw document text, as two separate components. This result in a multiple topic label associated with a document rather than a single document label. This label does not give importance to geographic locations. If there are no associated topics, then the articles are eliminated from the corpus.

- **Vectorization:-**

At this stage we have a large collection of two-tuples, each containing a class label and raw body text from the articles. The tuples undergo the process of vectorization, which converts these large set tuples into a numerical representation for further processing. Vectorisation allows widely-varying lengths of raw text to be converted into a numerical format that can be processed by the classifier.

This is achieved by creating tokens from a string. A token is an individual word (or group of words) extracted from a document, using whitespace or punctuation as separators, numbers from within the string as additional "words". Once this list of tokens has been created, they can be assigned an integer identifier, which allows them to be listed. Once the list of tokens has been generated, the number of tokens within a document is counted. Finally, these tokens are normalised to de-emphasise tokens that appear frequently within a document (such as "a", "the"). The Term-Frequency Inverse Document-Frequency (TF-IDF) value for a token increases proportionally to the frequency of the word in the document but is normalised by the frequency of the word in the corpus. This essentially reduces importance for words that appear a lot generally, as opposed to appearing a lot within a particular document. In other words, the TF-IDF method removes the stop words from the document in much efficient manner when compared to the rest of the methods.

- **Training the Support Vector Machine:-**

The output of vectorization results into a sparse matrix which is a normalised matrix (XX) of document-token occurrences and a vector of class labels (yy). In order to train the Support Vector Machine it is necessary to provide it with both a set of features (the XX matrix) and a set of "supervised" training labels, in this case the yy classes. However, we also need a means of evaluating the trained performance of the classifier subsequent to its training phase. One approach is to simply try classifying some of the documents from the corpus used to train it on. Such an evaluation procedure is known as in sample testing. However, this is not a particularly effective mechanism for assessing the performance of the system. An alternative approach is to partition the training set into two distinct subsets, one of which is used for training and the other for testing. This is known as the training-test split. Such a partition allows us to train the classifier solely on the first partition and then classify its performance on the second partition. This gives us a much better insight into how it will perform in true "out-of-sample" data going forward. The only challenge faced here was to decide what percentage of the document must be retained for training and testing. the more that is retained for training, the "better" the classifier will be because it will have seen more data. However, more training data means less testing data and as such a poorer estimate of its true classification capability. The next step is to actually create the Support Vector Machine and train it. In this instance we are going to use the SVC (Support Vector Classifier) class from scikit-learn. We give it the parameters C=1000000.0, Υ = 0.0and choose a radial kernel.

Conclusion:-

After training the SVM model, the testing phase is to be done. This resulted in calculating the hit ratio and the confusion matrix. The former is simply the ratio of correct assignments to total assignments and is usually quoted as a percentage. The confusion matrix goes into more detail and provides output on true-positives, true-negatives, false-positives and false-negatives. In a binary classification system, with a "true" or "false" class labelling, these characterise the rate at which the classifier correctly classifies something as true or false when it is, respectively, true or false, and also incorrectly classifies something as true or false when it is, respectively, false or true. The SVM model resulted in 0.836 and a large confusion matrix was obtained.

References

1. **Saurabh Khatri, Emmanuel M. (2013)** Review on Classification Algorithms in Email Domain.“ in International Journal of Applied Research and Studies ISSN 2278 – 9480.
2. **Lei Gu, Huan Li, (2013)** Memory or Time: Performance Evaluation for Iterative Operation on Hadoop and Spark, IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing.
3. **A.Basu, C. Watters, and M. Shepherd** Support Vector Machines for Text Categorization. Faculty of Computer Science Dalhousie University Halifax, Nova Scotia, Canada B3H 1W5.
4. **Setu Madhavi Namburu, Haiying Tu, Jianhui Luo and Krishna R. Pattipati** Experiments on Supervised Learning Algorithms for Text Categorization, Dept. of ECE, University of Connecticut.