Hadoop Spark Framework for Machine Learning Using Python

J V N Lakshmi

Abstract— Information is mounting exponentially and hungry for knowledge. As data is increasing, world is moving in hunting knowledge with the help of analytics in this age of Big Data. The flooding of data emerging from diverse domains is labelled for automated learning methods for data analysis is meant as machine learning. Spark is a framework build upon Hadoop for implementing the Linear Regression and Naïve Bayes statistical method used for predictive analysis. It calculates the Random mean square error and time required for this to be processed. Both the techniques are compared for time efficiency. The results for calculating the RMSE are evalated to minimize the complexity. These methods are implemented in this paper using python programming tool for analysing the datasets.

Index Terms — Big Data, Data Analytics, Linear Regression, Machine Learning, Naïve Bayes, Spark, RMSE.

1 Introduction

Computations and Analyzing the Big Data is a novel tendency in feature abstractions, acquiring knowledge from Data, rapidity in processing information, and future prediction.

Big Data is dynamically evolving with variant features of velocity (analysis time has drastically decreased subsequently), volume (corpus size raise from Big Data to Bigger Data) and Vectors (consonance to dissonance). Organizations are ascending in analyzing to make sense of the reams of data that are getting accumulated.

Deploying analytics is forthcoming challenge as end users gather information. An open source framework for processing contaminated analytics on Big Data is Spark. This unified framework gives us a wide-range of practices on diverse text data, graph data and structured either static or real time streaming as well.

Spark uses MLlib for developing Machine Learning algorithms. These algorithms uses less memory, less processing time and are largely hand tuned on specialized architecture to parallelize large cluster of machines for data analytics.

In this article an attempt is made to implement this machine learning techniques on spark framework to represent the distribution of data on different machines and recording the time for analysis.

The paper is organized as follows. Section 2 briefs Spark framework and Hadoop. Section 3 presents the Linear Regression and Naïve Bayes methods using spark framework.

The computation of Random mean squared error using anaconda tool is discussed in section 4. The results and evaluation of the study conducted is discussed in section 5. The paper is concluded in Section 6.

2 Spark Hadoop Map Reduce

Spark is implemented on top of HDFS infrastructure to provide augmented and supplementary functionality. Spark uses improvised map reduce framework with facilitating in memory data storage, enhanced shuffle phase, rapid performance and real time processing. This operates on huge datasets than the average capacity of a cluster.

2.1 Spark Architecture

Spark Architecture Model has three components Data Storage, API and Resource Management.

Data Storage: In spark is structured using HDFS attuned on data sources such as Hbase, Cassandra etc,.

API: Spark afford wide range of Application interfaces such as Scala, java and Python for development

Resource Management: Spark can be deployed in two ways one as a standalone server and other as a distributed computing network like yarn.

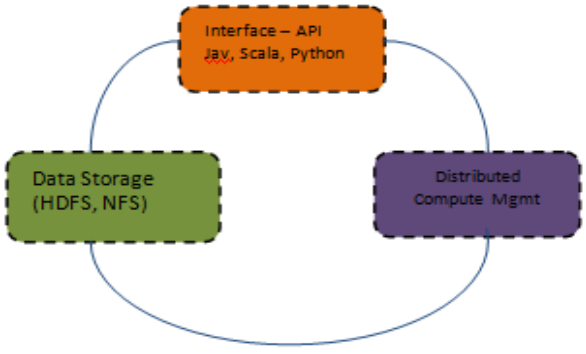

Figure 1: Spark Architecture

3 Linear Regression and Naïve Bayes

Machine Learning algorithms are elegantly expressed with a novel framework such as Spark for substantial improvement in performance on analytics. Iterative methods include statistical techniques such as Linear Regression and Naïve Bayes are evaluated for optimization procedures.

3.1 Linear Regression

A straight line is assumed between the input variables (x) and the output variables (y) showing the relationship between the values. Statistics on the training data is required to estimate the coefficients.

These estimates are used in model for prediction for further data processing. The line of simple linear regression model is y= a1+a2*x where a1 and a2 are the coefficients of the linear equation.

Estimating the coefficients is given as follows:

a1 = (Sum((x(i)-mean(x))*(y(i)-mean(y)))/(sum(x(i)-mean(x))2)

$$a0 = mean(y) - a1 * mean(x)$$

To process linear regression using python Spark framework is used the following algorithm illustrates the procedure.

To process linear regression using python Spark framework is used the following algorithm illustrates the procedure.

Algorithm for Linear Regression with pyspark framework

Step 1: Read the data in spark data frame and use time method to invoke time.

Step 2: Splitting the data into train data and test data using randomSplit function from spark.

Step 3: Vector Assembler converts the data in terms of vectors.

Step 4: Transform function modifies the vectors into necessary data frames.

Step 5: Mapping the labelcol and featurecol using LinearRegression method from MLlib.

Step 6: Pipeline consists of stages each acts as an estimator or a transformer when fit() is initiated.

Step 7: Regression Evaluator uses to evaluate the prediction on the featured data.

Step 8: RMSE is calculated to find the mean square error.

Step 9: Total evaluated time to process the data set is computed.

3.2 Naïve Bayes

Naïve Bayes are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. The spark.ml implementation currently supports both multinomial naive Bayes and Bernoulli naive Bayes. The following is the algorithm.

Algorithm to implement Naïve bayes using pyspark

Step1: importing the naïve bayes method from ml.classification

Step2: Load training data in spark.read.format ("filename").load("file")\

Step 3: Split the data into train and test using randomSplit()

Step 4: Create the trainer and set its parameters

Step 5: train the model to fit the dataset

Step 6: transform the model on the test data

Step 7: predicting the data from the model

Step 8: Calculate the accuracy on the test data using the MSE

4 Computation Of Root Mean Square Error Using Anaconda

The training data are used to make predictions on the test data to estimate the root mean squared error of the predictions using following procedure.

Step 1: Calculate the mean value of a list of numbers.

Step 2: Calculate the Standard Deviations and Covariance of the test data

Step 3: Calculate the correlation coefficient from the data

Step 4: evaluate the coefficients from the covariance and correlation

Step 5: Predict the values with the respective coefficients

Step 6: Computation of RMSE

5 Results and Evaluation

To better validate this framework real time datasets temperature of India corresponding to each month and year is been used.

Spark tool is used as a framework. In this paper we build Linear Regression and Naïve Bayes based model with the reduced training data, and evaluate it on the test data after applying the same feature selection. This model predicts temperature of year impression created with the test data.

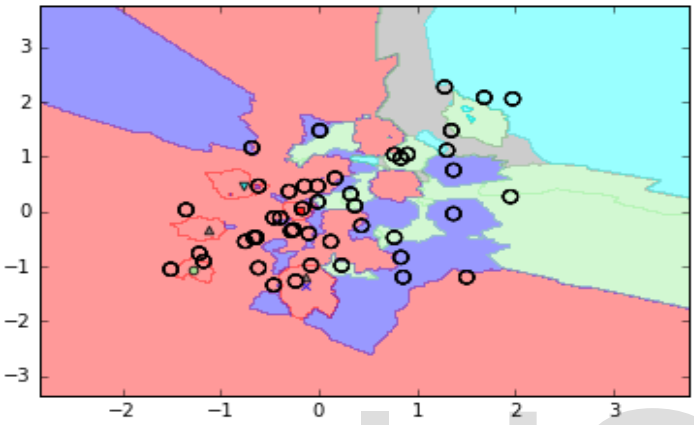

Figure 2: Naïve Bayes Classification

The results are illustrated in the Figures 2 is the Naïve Bayes Classification and Figure 3 illustrates the Linear Regression Model.

The prediction is made on the model created on the training dataset. If the model learning is effective, we expect the impression with high prediction to be actually resulting. Spark uses effective pipeline across various stages that gets updated to handle real time data efficiently.

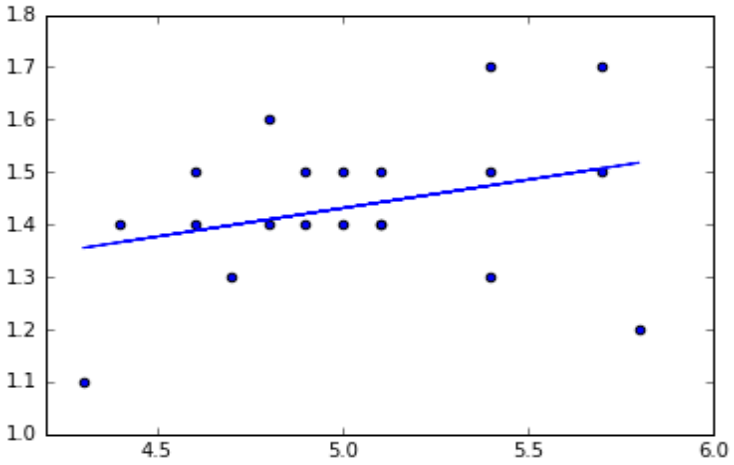

Figure 3: Linear Regression Model

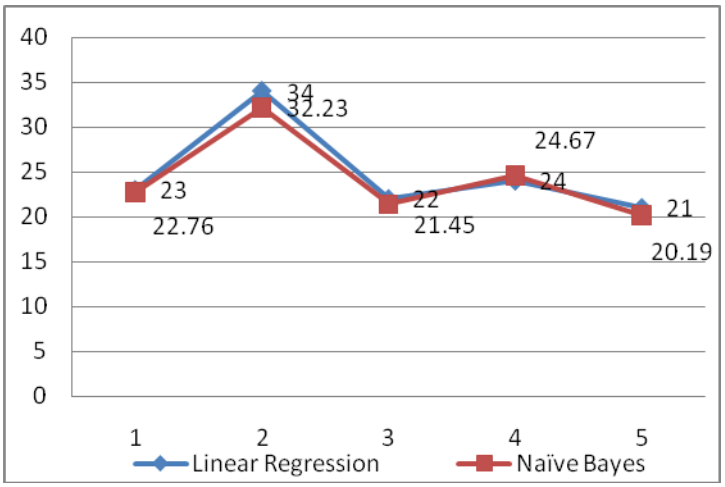

Figure 4: Model Predictions of the dataset

Figure 4 represents the results where the linear regression model prediction is 67.78% of the annual temperature. Naïve Bayes classification shows the 85.62% random mean square error.

6 Conclusion

In this paper a model is presented using pyspark that enables the development of large-scale machine learning algorithms. The languages in which machine learning algorithms are expressed are compared with highly optimized execution plans over existing techniques. Systematic empirical results in this paper have shown the benefit of a number of optimization strategies such as blocking, local aggregation, and the applicability scaleup on diverse set of machine learning algorithms. Development of additional constructs to support machine learning meta-tasks such as model selection, and enabling a large class of algorithms to be probed at an unprecedented data scale.

References

[1]. Lakshmi JVN, Ananthi S. (2015) "A Theoretical Model for Big data Analytics using Machine Learning Algorithms", at ICACCI Delhi.

[2]. Yong Chul Kwon, Bill Howe. (2014) "A Study of skew in MapReduce Application", in International Conference at USA.

[3]. Dr. Ananthi Sheshasayee, J V N Lakshmi. (2014) "A study on hadoop architecture for big Data analytics", in Delhi Conference ICETSCET.

[4]. Juigui Li, Yue Ye, Xuelian Lin. (2013) "Improving the Shuffle of Hadoop MapReduce", in IEEE ICCCTS at Beijing, China.

[5]. Yanfei Guo, Jia Rao, Xiaobo Zhou. (2013) "IShuffle-Improving Hadoop Performance with Shuffle-on-Write", in USENIX ICAC at USA.