Weather Prediction Model using Random Forest Algorithm and Apache Spark

Thin Thin Swe[1], Phyu Phyu[1], Sandar Pa Pa Thein[2]

[1]Lecturer, Faculty of Information Science, [2]Lecturer, Faculty of Computing,
[1,2]University of Computer Studies, Pathein, Myanmar

ABSTRACT
One of the greatest challenge that meteorological department faces are to predict weather accurately. These predictions are important because they influence daily life and also affect the economy of a state or even a nation. Weather predictions are also necessary since they form the first level of preparation against the natural disasters which may make difference between life and death. They also help to reduce the loss of resources and minimizing the mitigation steps that are expected to be taken after a natural disaster occurs. This research work focuses on analyzing algorithm on big data that are suitable for weather prediction and highlights the performance analysis with Random Forest algorithms in the spark framework.

KEYWORDS: *Weather forecasting, Apache Spark, Random Forest algorithms (RF); Big Data Analysis.*

How to cite this paper: Thin Thin Swe | Phyu Phyu | Sandar Pa Pa Thein "Weather Prediction Model using Random Forest Algorithm and Apache Spark" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-3 | Issue-6, October 2019, pp.549-552, URL: https://www.ijtsrd.com/papers/ijtsrd29133.pdf

IJTSRD29133

I. INTRODUCTION

Weather forecasting had always been one of the major technologically and scientifically challenging issues around the world. This is mainly due to two factors: Firstly, it is consumed for several human activities and secondly, because of opportunism, which is created by numerous technological advances that are directly associated to the concrete research field, such as the evolution of computation and improvement in the measurement systems. Hence, making an exact pre- diction contributes to one of the major challenges that meteorologists are facing around the world. From ancient times, the weather prediction had been one of the most interesting and fascinating study domains. Scientists have been working to forecast the meteorological features by utilizing a number of approaches, some of these approaches being better than the others in terms of accuracy. Weather forecasting encompasses predicting in what way current state of atmosphere will get altered. Existing weather situations are attained by ground observations, such as the observations from aircrafts, ships, satellites, and radars. The information is directed to the meteorological centers, which collect, analyze, and project the data into a variety of graphs and charts. The computers imprint lines on graphs with the help of meteorologists, who look for correcting any errors, if present. These computers not only make graphs but also predict how the graphs may look sometime in the near future. This estimation of weather by computers is acknowledged as numerical weather prediction[1]. Hence, for predicting weather by numerical means, meteorologists went on developing some atmospheric models, which approximate atmosphere by consuming mathematical equations to portray how atmosphere and rain will have transformations over time. These equations are automated into the computer, and the data for the current atmospheric conditions are provided into the computer. Computers solve these equations to conclude how different atmospheric variables may change over upcoming years. The resultant is known as prognostic chart, which is a forecast chart drawn by the computer.

II. PREDICTING WEATHER

Fig. 1 shows that initially the weather data source is collected from weather sensors and power stations. These weather data can be collected in the different data sources like kafka, flume etc. In the proposed system the data set is loaded into the spark API and using random forest algorithm to regress and classify the weather data.

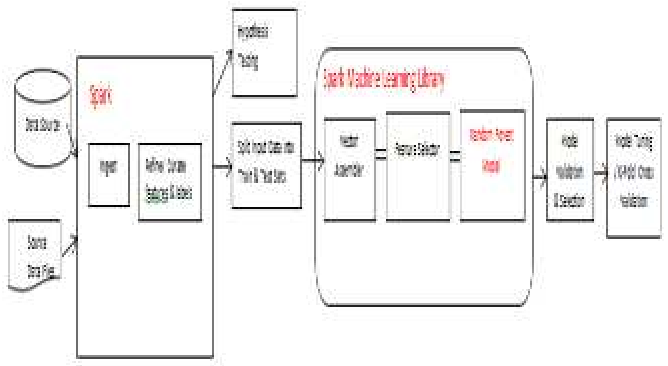
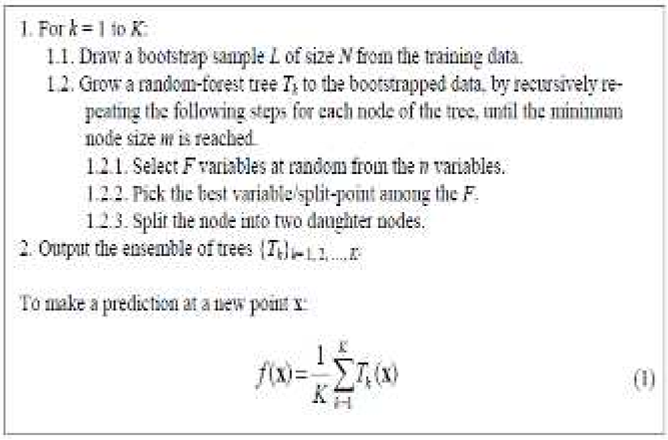


Figure 1: Design of the system

A. RANDOM FORESTS MODEL

Random Forests (RF) is the most popular methods in data mining. The method is widely used in different time series forecasting fields, such as biostatistics, climate monitoring, planning in energy industry and weather forecasting. Random forest (RF) is an ensemble learning algorithm that can handle both high- dimension classification as well as regression. RF is a tree- based ensemble method where all trees depend on a collection of random variables. That is, the forest is grown from many regression trees put together, forming an ensemble [4]. After individual trees in ensemble are fitted using bootstrap samples, the final decision is obtained by aggregating over the ensemble, i.e. by averaging the output for regression or by voting for classification. This procedure called bagging improves the stability and accuracy of the model, reduces variance and helps to avoid overfitting. The bias of the bagged trees is the same as that of the individual trees, but the variance is decreased by reducing the correlation between trees (this is discussed in [10]). Random forests correct for decision trees' habit of overfitting to their training set and produce a limiting value of the generalization error [6].

The RF generalization error is estimated by an out-of-bag (OOB) error, i.e. the error for training points which are not contained in the bootstrap training sets (about one-third of the points are left out in each bootstrap training set). An OOB error estimate is almost identical to that obtained by *N*-fold cross-validation. The large advantage of RFs is that they can be fitted in one sequence, with cross-validation being performed along the way. The training can be terminated when the OOB error stabilizes [7]. The algorithm of RF for regression is shown in Figure-2[5].



```
1. For $k = 1$ to $K$:
    1.1. Draw a bootstrap sample $L$ of size $N$ from the training data.
    1.2. Grow a random-forest tree $T_k$ to the bootstrapped data, by recursively re-
         peating the following steps for each node of the tree, until the minimum
         node size $m$ is reached.
        1.2.1. Select $F$ variables at random from the $n$ variables.
        1.2.2. Pick the best variable/split-point among the $F$.
        1.2.3. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_k\}_{k=1,2,\ldots,K}$.

To make a prediction at a new point x:
```

$$f(x)=\frac{1}{K}\sum_{i=1}^{K}T_k(x) \quad (1)$$

Figure2. Algorithm of RF for regression [8]

Where K represents the number of trees in the forest and F represents the number of input variables randomly chosen at each split respectively. The number of trees can be determined experimentally. And, we can add the successive trees during the training procedure until the OOB error stabilizes. The RF procedure is not overly sensitive to the value of F. The inventors of the algorithm recommend F = n/3 for the regression RFs. Another parameter is the minimum node size m. The smaller the minimum node size, the deeper the trees. In many publications m = 5 is recommended. And this is the default value in many programs which implement RFs. RFs show small sensitivity to this parameter.

Using RFs we can determine the prediction strength or importance of variables which is useful for ranking the variables and their selection, to interpret data and to understand underlying phenomena. The variable importance can be estimated in RF as the increase in prediction error if the values of that variable are randomly permuted across the OOB samples. The increase in error as a result of this permuting is averaged over all trees, and divided by the standard deviation over the entire ensemble. The more the increase of OOB error is, the more important is the variable.

The original training dataset is formalized as S = {(xi,yj), i=1,2,…..,N; j=1,2,….,M} where x is a sample and y is a feature variable of S. Namely, the original training dataset contains N samples, and there are M feature variables in each sample. The main process of the construction of the RF algorithm is presented in Fig. 2.

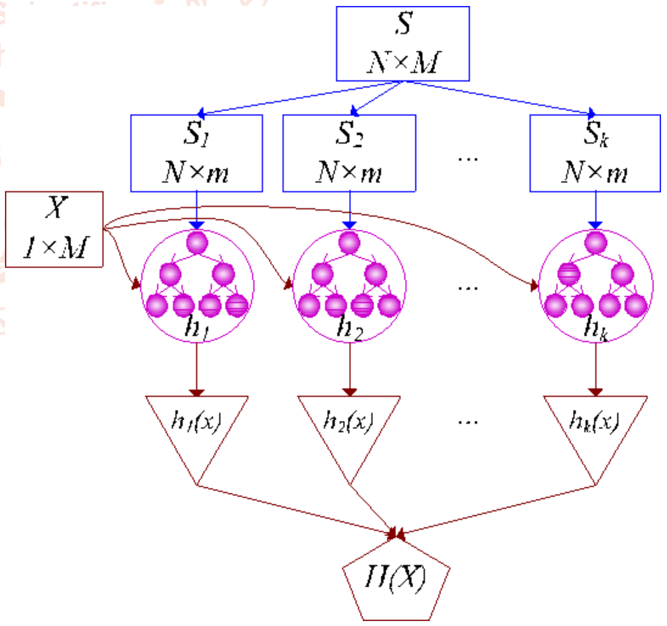


Fig.2. Process of the construction of the RF Algorithm

The steps of the construction of the random forest algorithm are as follows.

Step1: Sampling k training subsets.

In this step, k training subsets are sampled from the original training dataset S in a bootstrap sampling man-ner. Namely, N records are selected from S by a random sampling and replacement method in each sampling time. After the current step, k training subsets are constructed as a collection of training subsets $S_{T\,rain}$:

$S_{Train} = \{S_1; S_2, \ldots\ldots, S_k\}$.

At the same time, the records that are not to be selected in each sampling period are composed as an Out-Of-Bag (OOB) dataset. In this way, k OOB sets are constructed as a collection of S_{OOB}:

$S_{OOB} = \{OOB_1; OOB_2, \ldots, OOB_k\}$,

where $k << N$, $S_i \cap OOB_i = \phi$ and $S_i\ OOB_i = S$. To obtain the classification accuracy of each tree model, these OOB sets are used as testing sets after the training process.

Step2: Constructing each decision tree model.
In an RF model, each meta decision tree is created by CART algorithm from each training subset S_i. In the growth process of each tree, m feature variables of dataset S_i are randomly selected from M variables. In each tree node's splitting process, the gain ratio of each feature variable is calculated, and the best one is chosen as the splitting node. This splitting process is repeated until a leaf node is generated. Finally, k decision trees are trained from k training subsets in the same way.

Step3: Collecting k trees into an RF model.
The k trained trees are collected into an RF model, which is defined in Eq. (1):

$$H(X, Kj) = \sum_{i=1}^{k} h_i(x, Kj), (j=1,2,\ldots,m) \qquad (1)$$

where $h_i(x;j)$ is a meta decision tree classifier, X are the input feature vectors of the training dataset, and j is an independent and identically distributed random vector that determines the growth process of the tree.

To dig why we select random forest algorithm, the following presents some benefits:

- Random forest algorithm can be used for both classifications and regression task.
- It provides higher accuracy.
- Random forest classifier will handle the missing values and maintain the accuracy of a large proportion of data.
- If there are more trees, it won't allow overfitting trees in the model.
- It has the power to handle a large data set with higher dimensionality [3].

B. APACHE SPARK

Apache Spark is an all-purpose data processing and machine learning tool can be used for a variety of operations. Data scientist, application developer can integrate Apache Spark into their application to query, analyze, transform as scale. It is 100 times faster than Hadoop MapReduce. It can handle petabytes of data at once, distribute over a cluster of thousands of cooperating virtual or physical servers. Apache Spark has been developed in Scala and it support Python, R, Java and off course Scala Apache spark is fast and general purpose engine for large scale data processing [9-10]. Architecture of spark has spark core at it bottom and on top of which Spark SQL, MLlib, Spark streaming and GraphX libraries are provided for data processing[2].

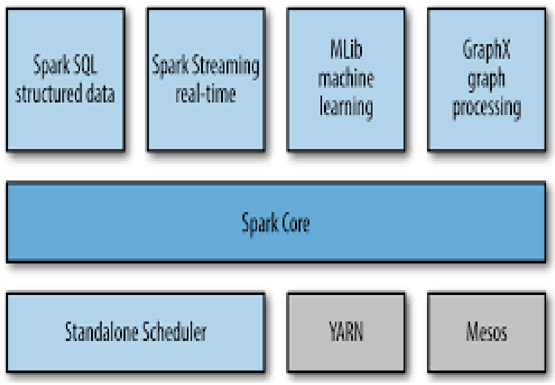


Fig.3. Architecture of spark

Apache Spark is very good for in memory computing. Spark has its own cluster management but it can work with Hadoop also. There are three core building blocks of Spark programming. Resilient Distributed Datasets (RDD), Transformations and Action. RDD is an immutable data structure on which various transformations can be applied. After transformation any action on RDD can lead to complete lineage execution of transformation before result is produced.

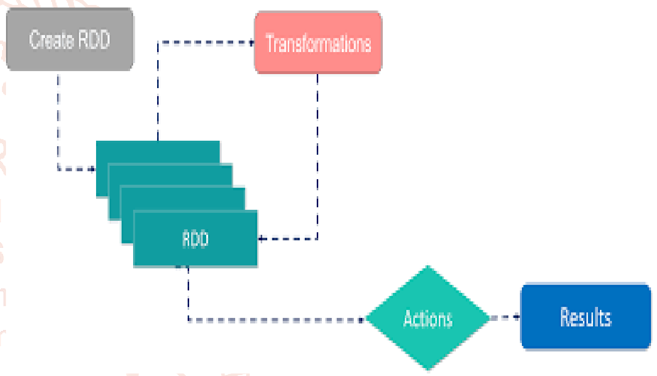


Fig.4. Working with RDD in Spark

III. CONCLUSIONS

In this paper, a random forest algorithm has been proposed for big data. The accuracy of the RF algorithm is optimized through dimension-reduction and the weighted vote approach. Then, combining data-parallel from different data station and task-parallel optimization is performed and implemented on Apache Spark. Taking advantage of the data-parallel optimization, the training dataset is reused and the volume of data is reduced significantly. Benefitting from the task-parallel optimization, the data transmission cost is effectively reduced and the performance of the algorithm is obviously improved. Experimental results indicate the superiority and notable strengths of RF over the other algorithms in terms of classification accuracy, performance, and scalability. For future work, we will focus on the incremental parallel random forest algorithm for data streams in cloud environment, and improve the data allocation and task scheduling mechanism for the algorithm on a distributed and parallel environment.

References

[1] Guidelines on Climate Metadata and homogenization World Climate data and monitoring program, Geneva.

[2] https://spark.org

[3] https://www.newgenapps.com/blog/random-forest-analysis-in-ml-and-when-to-use-it

[4] K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, "Big data analytics framework for peer-to-peer botnet detection using random forests," Information Sciences, vol. 278, pp. 488–497, September 2014.

[5] Apache, "Spark," Website, June 2016, http: //spark-project.org. [9] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, October 2001.

[6] G. Wu and P. H. Huang, "A vectorization-optimization method-based type-2 fuzzy neural network for noisy data classification," Fuzzy Systems, IEEE Transactions on, vol. 21, no. 1, pp. 1–15, February 2013.

[7] H. Abdulsalam, D. B. Skillicorn,and P. Martin, "Classification using streaming random forests," Knowledge and Data Engineering, IEEE Transactions on, vol. 23, no. 1, pp. 22–36, January 2011.

[8] C. Lindner, P. A. Bromiley, M. C. Ionita, and T. F. Cootes, "Robust and accurate shape model matching using random forest regression-voting," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 25, no. 3, pp. 1–14, December 2014. [4] What is Twitter and How does it work? http://www.lifewire.com/ what-is – twitter.

[9] S. Tyree, K. Q. Weinberger, and K. Agrawal, "Parallel boosted regression trees for web search ranking," in International Conference on World Wide Web, March 2011, pp.387–396.

[10] D. Warneke and O. Kao, "Exploiting dynamic resource allocation for efficient parallel data processing in the cloud," Parallel and Distributed Systems, IEEE Transactions on, vol. 22, no. 6, pp. 985–997, June 2011.