Prediction Breast Cancer as Benign or Malignant in Apache Spark Framework

1 Wafaa S. Albaldawi, 2 Rafah M. Almuttairi

Rafah@uobabylon.edu.iq ; stud.wafaa.shakir@uobabylon.edu.iq

1Ministry of Higher Education and Scientific Research, Department of Computer Science, College of Science, University of Kerbala, Karbala, Iraq
1,2Ministry of Higher Education and Scientific Research, Department of Software, College of Information Technology, University of Babylon, Hillah, Iraq

Abstract

There are number of diseases that increase the number of deaths over the world. Breast cancer can be considered as the most common of them. Therefore, there is a need to use classification and others data mining methods to study the health datasets in order to diagnosis and make decisions. In this paper, Support Vector Classifier model, Logistic Regression algorithm, and Random Forest algorithm are conducted on the public available Wisconsin Breast Cancer dataset. The experiment is executed in a Scala environment. Moreover, in single and multi-nodes spark cluster. The results show the high accuracy in Support Vector Classifier model and the low error rate in less time consumed when compared with other studies. The authentication in spark are applied in the application by using shared secret method.

Keywords: Classification, Prediction, Apache Spark, Single Node, Multi-nodes Cluster.
1. Introduction

Discovering the breast cancer in an early stage will reduce the mortality rate of middle aged women. Classification systems and accurate predictions help the relative and health care providers to make the most valuable decisions in patients' treatments. Therefore, they can play the effective role in disease care by applying them in medical dataset to produce the high performance in outcome and improve the value and quality of health care in order to save people's lives. Apache Spark is fast as it uses memory worked in preparing. It continues running on distributed environment and chops down the data in bunches to give a high profitability rate.

The paper is divided into ten sections which related to this cancer type and the prediction system. Structure of this the second section allows reader to have knowledge about the breast cancer and the related work. The third section illustrates the description of the dataset. Section Four, Apache Spark and the usage of Scala with Spark is discussed. The followed section, Support Vector Classifier, Logistic Regression algorithm, and Random Forest Model in cloud environment (Apache Spark) are applied in single node and in multi-node cluster. Optimize Apache Spark Jobs, Authentication in Spark and Visualization as the six, seven and eight sections respectively. The next section, prediction results is shown and discussed. Conclusion of the paper in the followed section. Acknowledgment section and the references are listed in the end of the paper.

2. Related Work

Cancer cases number increases and medical management develops along with it. Breast cancer can be considered as the most spreading one. Breast cancer detection can be done using many methods. Machine Learning application to support cancer detection have the ability to provide good diagnostic result.

According to 1 least square support vector machine classifier algorithm is used to conduct the breast cancer diagnosis. The performance of the algorithm is evaluated by using various methods as classification accuracy, k-fold-validation, Sensitivity and precision analysis, and confusion matrix. The accuracy of the model classification obtained is 98.53%.

Akay2 has used support vector machines combined with feature selection for breast cancer diagnosis and obtained 99.51% classification accuracy. Moreover, sensitivity, specificity, positive and negative predictive values, receiver operating characteristic (ROC) curves and confusion matrix have been used to evaluate the robustness of the method.

An efficient hybrid data mining approach to diagnose breast cancer has been proposed. The algorithm is based on the fundamental concept of the discrete particle swarm optimization. The obtained results as following: classification accuracy is 98.71%, sensitivity is 100% and specificity is 98.21%.3

Applying artificial metaplasticity neural network for pattern classification is inspired by the biological metaplasticity property of neurons and Shannon's information theory. The Wisconsin Breast Cancer Database (WBCD) has been used to test the algorithm by using classification precision, sensitivity, and confusion matrix and specificity analysis. The accuracy of classification obtained is 99.26%.4

Nahato et al5 build a classifier to predict diseases by learning from the clinical dataset attributes. Rough set in-dicernibility relation method with back propagation neural network are used after handling of missing values. The classification was used with hepatitis, Wisconsin breast cancer, and Statlog heart disease datasets collected from the University of California at Irvine (UCI). The precision reached for hepatitis, heart disease, and breast cancer, and heart disease is 97.3 percent, 90.4 percent, and 98.6 percent respectively.
Author in 6 presents the evolutionary artificial neural network approach based on the Pareto-differential evolution algorithm augmented with local search for the prediction of breast cancer.

Madhu and Vijendra7 in their work presented the breast cancer prediction system by using support vector machine, linear regression and k nearest neighbor. The experiment was conducted using Wisconsin breast cancer dataset. Using classification precision, evaluation of the proposed method is obtained. The achieved classification accuracy is 99.28%.

Ablen presents a comparison of six machine learning algorithms on the Wisconsin Diagnostic Breast Cancer dataset: GRU-SVM8, Multilayer Perceptron, Linear Regression, Soft max Regression, Support Vector Machine, and Nearest Neighbor Search by testing their classification precision, sensitivity and specificity. Results show that Multilayer Perceptron algorithm has the highest accuracy among the implemented algorithms with a test accuracy of 99.04%.9

Tianming10 represents the hybrid of genetic algorithm and support vector machine model to diagnose breast cancer. The dataset has been used in paper is the Diagnosis Breast Cancer Wisconsin which contains 30 features for classification. By applying the model the classification accuracy achieved is 98.07%. Genetic Algorithm used to select out features that can train a support vector machine classifier making a small error rate while diagnosing breast cancer.

Mohammed, Akash, Shahriar, Ahmed and Mayaz used six classification techniques in the form of Decision Tree, K-Nearest Neighbors, Linear Discriminant Analysis, Logistic Regression, Naïve Bayes and Support Vector Machine on the Wisconsin in Breast Cancer (original) data sets, before and after applying Principal Component Analysis. Classification accuracy, precision, recall, specificity and F1 Score are measured. The obtained results have shown that Logistic Regression and Support Vector Analysis have recal score=1.000 with PCA performs well with Breast Cancer Prediction.11

Asri et al12 have measured the performance comparison between different machine learning algorithms by conducted on the Wisconsin Breast Cancer datasets the Decision Tree, Support Vector Machine, Naïve Bayes and k Nearest Neighbors. The evaluation of their analysis is the correctness in the classification of data regarding the efficiency and effectiveness of each algorithm in terms of precision, accuracy, sensitivity and specificity. Support vector machine gives the highest accuracy which is 97.13% with lowest error rate.

Aaron and Taghi13 present a cloud-based approach for learning from electronic health record data and show its usefulness for prediction melanoma risk. For data preprocessing and marking, they used a hybrid distributed processing with Apache Spark, and non-distributed processing with scikit-learn for machine-learning model testing. In addition, they demonstrated the performance enhancement effects of sampling the training data set. The predictive success was obtained by a gradient-boosted classifier with cross-validation as 0.688 of Specificity, 0.799 for AUC, and 0.753 for Sensitivity.

Zhang et al14 propose Meteor, a novel architecture that can significantly improve the performance of short-run applications. They extend Spark to three modes of operation: one-container, one-thread, and distributed. The single-thread mode performs all tasks on one thread only; the multi-threaded one-container; the last mode assigns all tasks over the cluster. The mechanism for submitting applications is also built that uses a fine-grained Spark performance model to decide which mode is the most effective to invoke upon submission of a new application. The results show that experiments with Amazon EC2, one-thread mode are the best option when the size of the input is small. In other instances it is easier to have the distributed mode. Overall Meteor is about two times faster than Spark for short applications.
3. Data set

In this paper, the data set is utilized for training the svc classifier, is Breast Cancer Wisconsin (Diagnostic) Data Set. The dataset is derived from Kaggle, developed by William H. Wolberg, a doctor at Madison, Wisconsin, USA University of Wisconsin Hospital.

For the calculation of the characteristics a digitized representation of a fine needle aspiration (FNA) of a breast mass. This describes the characteristics of the nuclei of cells in the image. Dr. Wolberg used fluid samples from patients with firm breast masses and an interactive computer program named Xcyt, capable of evaluating cytological characteristics based on digital scanning. The program uses a curve-fitting algorithm to calculate ten features from each of the sample cells than it calculates the mean value, the extreme value, and the standard error of each feature for the image, returning a real-valued vector of 30.

There are 569 instances composed of thirty two attributes. No missing value in the dataset. In addition, the balance of dataset is acceptable, in which 212 instances are malignant and the other 357 instances are benign breast cancers. For the 32 attributes in the data set, the first one is the ID number of the instance, which is no help for training the Support Vector Classifier. The second attribute is the label which has two classes benign and malignant breast cancer. The remaining 30 attributes are computed from a digital image of a Fine-needle aspiration of a breast mass which are the used features to train the Support Vector Classifier. ¹⁵

In this paper, it can be decided to use Support Vector Classifier model, Logistic Regression algorithm, and Random Forest Model.

4. Usage of Scala with Spark

Scala is an open source language, developed by Professor Martin Odersky, founder of Type safe, that promotes and supports Scala's programming language commercially. Scala programming language provides the best path in terms of data size and program complexity to build scalable Big Data applications. Scala provides remarkable support for functional programming with support for unchangeable data structures, for-comprehensions, and unchangeable named values.

Performance and pace play a vital role, no matter how fast the processor increases. Scala is quick and efficient, making it an ideal language option for computer-intensive algorithms. When using Scala for Spark programming, the compute cycle and memory performance are also well balanced.

According to a recent Databricks survey, Scala programming language is also used by 71 per cent of Spark users-making it the de facto programming language for Spark.

All these numerical reports show that Apache Spark's Scala software becomes the choice to make data analytics faster.

Sitting on Ubuntu laptop running, data can be processed on parallel nodes in a Spark Cluster. The syntax is powerful. Data Cleaning and Migration operations can be carried out in real time throughout the cluster as Map-Reduce tasks.

Spark transformations are also many times faster, because they occur in memory on parallel nodes.

5. Implementation

5.1 Dataset Preprocessing

Since the first attribute in the data set is the ID number of an instance, which is no help for
training the Support Vector Classifier. Therefore, it is save to drop the first column of the data set. The second column of the data set is the label whose value is ‘M’ or ‘B’ which indicate the instance is a malignant or a benign breast cancer. For the convenience of training the Support Vector Classifier, ‘M’ is converted into 1 and ‘B’ is converted into 0. The remaining 30 attributes are useful features for training the classifier.

5.2 Implement Support Vector Classifier model, Logistic Regression algorithm, and Random Forest Model in Scala in local, standalone, and yarn modes.

Table 5-1: Types of nodes in the distributed system

| Node            | Description                                                      |
|-----------------|------------------------------------------------------------------|
| A server        | A server                                                        |
| Worker Node     | Available to run Spark jobs                                      |
| Master Node     | Coordinates the Worker nodes                                     |
| Executor        | A sort of virtual machine inside a node. One Node can have multiple Executors |
| Driver Node     | Initiates the Spark session                                      |
| Driver (Executor)| The Driver Node will also show up in the Executor list           |

If we use the client mode with the cluster in the same local network, and send it from the laptop. Also, if the cluster is far away, we would either submit it locally in cluster mode, or rsync the jar to the remote cluster and send it there, in client or cluster mode, depending on how heavy the driver program is on resources.

YARN manages resource management, scheduling, and protection when we run spark applications thereon. The program can be executed in any mode, whether cluster or client mode. Spark executor also runs as a YARN container while we run spark on YARN.

There is one case where Map Reduce schedules a container for each function and begins a JVM for each. Hosts lots of functions within the same container fire. So it makes faster startup time for several mission orders of magnitude.

The steps of our application:

1. Implement PCA:
2. Analyzing the data: check for missing values: no missing values found.
3. Convert diagnostics to binary: benign=0, malignant=1
4. Exploratory data analysis for the entire dataset.
5. Advanced level exploratory data analysis for Benign Tumor.
6. Advanced level exploratory data analysis for Malignant Stage Cancer.
7. Identify the features and labels.
8. Include PCA

Despite the fact that PCA was able to distinguish the groups extremely well, it can be noticed afterwards that a few critical highlights—which resulted in diminished precision—were too reduced by the PCA technique. That's how the standard scaler will be used on the dataset.
val standardscaler = new StandardScaler()
   .setInputCol("Rfeatures")
   .setOutputCol("scaledFeatures")
   .setWithStd(true)
   .setWithMean(false)

Build the Support Vector Machine, Logistic Regression, and Random Forest Models to predict the tumor type in Scala with Apache Spark 2.4.3 version.

1. Split the training data into training and validation data frame.
2. Create the Random Forest Classifier, Logistic Regression and SVC.
3. Setup the binary classifier evaluator.
4. Fit the model on Random Forest
5. Random Forest - Training set
6. Print the Random Forest Model Parameters
7. Check the accuracy on validation set
8. Accuracy on Test Set
9. Support Vector Classification Model
10. Fit the model on SVC
11. SVC Model - Training set
12. Print the SVC Model Parameters
13. Check the accuracy on validation set
14. Accuracy on Test Set
15. Logistic Regression Model
16. Fit the model on logistic regression model
17. logistic regression Model - Training set
18. Print the Logistic regression model Parameters
19. Check the accuracy on validation set
20. Accuracy on Test Set
21. Summarize the results for these 3 models, in terms of accuracy on the test set
22. Compute Other Classification Metrics for Support Vector Classifier Model
23. True positive is the frequency with which the model correctly predicted a tumor is malignant.
24. False positive is the frequency with which the model predicted a tumor was malignant when it was benign.
25. False negative reflects how the model correctly predicted a tumor was benign.
26. False Positive shows how often a tumor was predicted to be benign when it was actually malignant.
27. SVC model - Training Set
28. Get the area under the curve
29. SVC Model - validation Set
30. ROC for Validation Set - Logistic Regression
31. SVC Model - Test Set
32. ROC for Validation Set - SVC Classification Model

6. Optimize Apache Spark Jobs
1. Monitoring Cluster Metrics
   YARN Resource Manager UI provides a view of cluster resources including the number of executors, CPUs and memory per executor.
   Spark Web UI, providing a view of all planned tasks and spark configurations.
2. Manual Configuration
By getting these few key parameters right, we would mostly be able to maximize cluster utilization:

- `spark.executor.memory`
- `spark.executor.cores`
- `spark.executor.instances`
- `spark.yarn.executor.memoryOverhead`
- `spark.default.parallelism`

Spark.executor.cores informs Spark how many simultaneous tasks each executor will run, and that usually means the number of CPUs to be allocated to each executor.

A better approach would be to have many smaller executors launched within a node. Be mindful that sometimes having several small executors also has some disadvantages because if broadcast variables are used, it will end up having overheads.

spark.executor.memory determines the amount of memory per executor is to be used. Due to the fact that Spark processes data in memory, the more memory an executor has, the less often data can spill into a disk. But we don't want to use 100 percent memory on an executor node because it requires space to run OS and Hadoop daemons. It must also remember `spark.yarn.executor.memoryOverhead`.

The last step is to decide `spark.default.parallelism` which controls the number of data partitions to be created after some operations for example `aggregateBy` to reduce of shuffle data size. This affects the number of concurrent activities to be run.

7. Authentication in Spark

Spark's protection is OFF by default. Spark supports multiple types of deployments. Security concerns come in many different forms. Spark does not necessarily guard against everything. Currently Spark supports RPC channel authentication using a shared secret. Authentication may be switched on by setting `spark.authenticate`. The exact deployment-specific method used to produce and distribute the shared secret is.

Spark can automatically manage producing and transmitting the mutual secret for the Spark on YARN and local deployments. Each application uses a shared secret unique to it. In YARN's case, this function is based on allowing YARN RPC encryption to protect the distribution of secrets. For other resource managers it is important to configure `spark.authenticate.secret` on each of the nodes. All daemons and applications must share this secret.

8. Visualization

The components of the visualization are:

1. Timeline of events at Spark
2. Direct acyclic graph (DAG)
3. Summary Metrics for Completed Tasks

1. Timeline View of Spark Events
In Fig. 8-1, Fig. 8-2 and Fig. 8-3, the stage has many partitions spread out across 4 machines (slave8 and master just are shown).

A single task is represented by one bar. From this view of the timeline, we can gather several insights into the stage. Firstly, the partitions are distributed fairly across the nodes. Also, a lot of the task execution time consists of raw computation rather than network or I/O overheads, which is not shocking because shuffled data is low. Third, if we allocate more cores to the executors, the level of parallelism can be increased. At present each executor seems to be able to perform no more than four tasks at one time.

2. Execution direct acyclic graph (DAG).

The second diagram shows the DAG for each job execution. A job is connected to a chain of RDD dependencies arranged within a DAG. DAG below is in one job in our program, for one stage (stage 10).
The result is something that resembles a SQL query design mapped to DAG that underlies it. The total time for all the tasks in the stage (10) is different when we select the local, cluster and client yarn mode.

Fig. 8-4 Execution direct acyclic graph (DAG) of local, cluster yarn and client yarn modes respectively.

3. Summary Metrics for Completed Tasks

Fig. 8-5 Stage (10) consists 8 completed tasks when we are using Local mode.

Fig. 8-6 Stage (10) consists 12 completed tasks when we are using Client yarn mode.

Fig. 8-7 Stage (10) consists 12 completed tasks when we are using Cluster yarn mode.

In Fig. 8-5, Fig. 8-6 and Fig. 8-7, the Summary Metrics shows different metrics at percentiles 0 (Min), 25th, 50th (Median), 75th, and 100th (Max) (among the activities, for example, in
stage (stage 10)). Selecting hidden checkboxes under "Show Additional Metrics" earlier on the page can reveal more metrics.

We would see all the statistics clustered closely around the 50th percentile value in a perfect world, where our calculation was completely symmetric across tasks. There would be slight variance; there would be small distance between the values of 0 and 100 percent. Nonetheless, things do not always work that way in the real world. However, we can see how far away they are, and get an understanding of why.

The 25 percent-75 percent range is not too large but some Max metrics are significantly higher than the corresponding 75 percent. This indicates a range of "straggler" tasks that take too much time to calculate (or cause excess GC) and/or over partitions with larger distorted data volumes.

On the other hand, the distribution is rational, except we have a bunch of Min values at or near zero. That means we have empty (or nearly empty) partitions and/or tasks that do not compute anything (our compute logic may not be the same for all kinds of records, so we might have some partitions that do not work at all).

9. Results:

Load the CSV dataset from local position in the PC to hdfs. The class attribute is in factor format so before use it. It has to be converted to numeric value 0 and 1 instead of B and M.

Table 9-1 The results for the train prediction, Accuracy on Validation and Test Set for Random Forest, Support Vector, Logistic Regression Models

| Model                          | Random Forest Classification Model | Support Vector Classification Model | Logistic Regression Model |
|--------------------------------|-----------------------------------|------------------------------------|--------------------------|
| train_prediction               | 0.9924242424242424               | 0.9706060606060606               | 0.9706060606060606       |
| Accuracy on validation set     | 0.9426705370101597              | 0.952104992743106               | 0.9366231253023706      |
| Accuracy on Test Set           | 0.9629515282494597              | 0.9936708860759494               | 0.9746835443037974      |

Table 9-2 Total Records, True Positives + False Negatives, True Negatives + False Positives, True Positive, True Negative, False Negative, False Positive, Precision, Recall for SVC Model.

| SVC Model                      | Training Set | Validation Set | Test Set |
|--------------------------------|--------------|----------------|---------|
| Total Records                  | 357          | 92             | 120     |
| True Positives + False Negatives | 348          | 88             | 119     |
| True Negatives + False Positives | 9            | 4              | 1       |
| True Positive          | 126 | 36  | 41  |
|-----------------------|-----|-----|-----|
| True Negative         | 222 | 52  | 78  |
| False Negative        | 6   | 3   | 0   |
| False Positive        | 3   | 1   | 1   |
| Precision             | 0.9545454545454546 | 0.9230769230769231 | 1.0 |
| Recall                | 0.3620689655172414 | 0.4090909090909091 | 0.3445378151260504 |

Area under the curve - SVC Classification Model = 0.9706060606060606
ROC for Validation Set - SVC Classification Model = 0.9936708860759494
ROC for Validation Set - Logistic Regression = 0.9521044992743106

10. Conclusion:
Compared with other machine learning methods, Support Vector Classifier is the best model which is used to predict whether the breast cancer is Benign or Malignant (class of Breast Cancer Wisconsin (Diagnostic) Data Set from Kaggle) based on accuracy measure: ROCAUC = 99.36%. The model has good predictability and interpretability, this is valuable for both medical researchers and patients.

Spark has two different components. The driver and the workers are there. The driver operates remotely on a data node in yarn-cluster mode, and the worker run on separate data nodes. The driver is on the computer that began the job in yarn-client mode and the workers are on the data nodes. The driver and workers are on the computer that began the job in local mode. The data from the nodes of the worker will be pulled into the driver when programmer run collect(). Basically, it is where the final bit of processing takes place.
When the programmer at home, yarn-cluster mode can be considered as great option. However, yarn-client mode is better when we run code from the data center.

11. Acknowledgment
An appreciation to the open source community (GitHub, Stack Overflow) for the virtually infinite source of information and knowledge.

12. References
1. Polat K, Güneş S. Breast cancer diagnosis using least square support vector machine. Digital Signal Processing. 2007;17(4):694-701.
2. Akay M. Support vector machines combined with feature selection for breast cancer diagnosis. Expert Systems with Applications. 2009;36(2):3240-3247.
3. Yeh W-C, Chang W-W, Chung YY. A new hybrid approach for mining breast cancer pattern using discrete particle swarm optimization and statistical method. Expert Systems with Applications. 2009;36(4):8204–11
4. Marcano-Cedeño A, Quintanilla-Domínguez J, Andina D. WBCD breast cancer database classification applying artificial metaplasticity neural network. Expert Systems with Applications. 2011;38(8):9573-9579.

5. Nahato K, Harichandran K, Arputharaj K. Knowledge Mining from Clinical Datasets Using Rough Sets and Backpropagation Neural Network. Computational and Mathematical Methods in Medicine. 2015;2015:1-13.

6. Abbass H. An evolutionary artificial neural networks approach for breast cancer diagnosis. Artificial Intelligence in Medicine. 2002;25(3):265-281.

7. Kumari M, Singh V. Breast Cancer Prediction system. Procedia Computer Science. 2018;132:371-376.

8. Agarap A. A Neural Network Architecture Combining Gated Recurrent Unit (GRU) and Support Vector Machine (SVM) for Intrusion Detection in Network Traffic Data. Proceedings of the 2018 10th International Conference on Machine Learning and Computing - ICMLC 2018. 2018;

9. Agarap A. On breast cancer detection. Proceedings of the 2nd International Conference on Machine Learning and Soft Computing - ICMLSC ’18. 2018;

10. Zhao, T. Breast Cancer Diagnosis via the hybrid of genetic algorithm and support vector machine. 2018;1–6.

11. Mohammed T, Akash, Shahriar F, Ahmed, Mayaz. Machine learning as an indicator for breast cancer prediction [Internet]. Machine learning as an indicator for breast cancer prediction. BRAC University; 2018 [cited 2020Jun17]. Available from: http://hdl.handle.net/10361/11431

12. Asri H, Mousannif H, Moatassime H, Noel T. Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis. Procedia Computer Science. 2016;83:1064-1069.

13. Richter A, Khoshgoftaar T. Efficient learning from big data for cancer risk modeling: A case study with melanoma. 2020.

14. Zhang H, Huang H, Wang L. Meteor: Optimizing spark-on-yarn for short applications. Future Generation Computer Systems. 2019;101:262-271.

15. Wolberg, W, Nick Street, W. and Mangasarian , O. Breast Cancer Wisconsin (Diagnostic) Data Set. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic), 1995.

16. Spark Security, Available at:https://spark.apache.org/docs/2.4.3/security.html#standalone-mode-only__(Accessed: 18 February 2020).