Anlu Chen (ac4218), Guanqun Wang (gw2353)

Prof. Jelenkovic

Statistics Learning in Bio & Info

May 9 2018

Breast Cancer Prediction Based on Statistical Learning Techniques

Data classification is one of the main hot study fields in statistical learning and has been extended to a wide rage of real-world applications. Among these, statistical learning based disease diagnosis is a representative practice. In the aim of developing an automated diagnostic system to provide assistance for professional doctors, we focus our attention on the diagnostic prediction of the breast cancer, which is the most common cancer among women, excluding the skin cancer. To achieve a more precise cancer prediction and to understand the cancerous breast cells in depth, multiple statistical techniques and machine learning methods are implemented.

In general, we reviewed one paper(Karabatak and Ince) for predicting breast cancer and reproduced their results. Furthermore, we provided evaluations of some other machine learning algorithms applied on the dataset. The prediction model in the paper consists of two modules: feature selection and classification. We implemented the paper's method and also implemented some other techniques. In general, for the feature selection part, Association rules (AR) and Principle Component Analysis (PCA) are both applied to retrieve the top significant factors, i.e. transformed or non-transformed tissue and cell expressions in this project. In terms of the classification task, Neural Network, Support Vector Machine (SVM) and XGBoost are utilized for breast cancer prediction based on the selected features obtained from the former process.

1. Dataset

In this project, Breast Cancer Wisconsin (Original) Data Set[1] and Breast Cancer Wisconsin (Diagnostic) Data Set[2] are both selected for study. The paper (Karabatak and Ince), 'An expert system

[1] https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original)

[2] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

for detection of breast cancer based on association rules and neural network' that we choose for re-producing uses the original breast cancer dataset. As there are only 9 cell features for breast cancer prediction (benign or malignant), we additionally pick the diagnostic breast cancer dataset with 30 cell features, in order to better explore the different feature dimension reduction techniques.

1.1 Breast Cancer Wisconsin (Original) Data Set

The original breast cancer dataset was created by Dr. WIlliam H. Wolberg at the University of Wisconsin Hospitals. It contains 699 instances with 9 features of integer values ranging from 1 to 10. Values of attributes represent the degree of abnormality in breast cells. The larger value an attribute holds, the more abnormal it stands. The statistical details are shown in Fig.1. From the figure, we can see that this dataset holds 65.5% malignant records and 34.5% benign records, which indicates that it is a relative balanced dataset.

1.2 Breast Cancer Wisconsin (Diagnostic) Data Set

The diagnostic breast cancer dataset was created by Dr. William H. Wolberg, W. Nick Street and Olvi L. Mangasarian at University of Wisconsin. It contains 569 instances with 30 features of real numbers. For the 'diagnosis' attribute, 'B' means the class of 'Benign' and 'M' means the class of 'malignant'. It is also a balanced dataset as the ratio of two classes is approximately 17:10. The statistical details are shown in Fig.2.

```
       Id               Cl.thickness   Cell.size     Cell.shape  Marg.adhesion  Epith.c.size
 Length:699         1       :145   1       :384   1       :353   1       :407   2       :386
 Class :character   5       :130   10      : 67   2       : 59   2       : 58   3       : 72
 Mode  :character   3       :108   3       : 52   10      : 58   3       : 58   4       : 48
                    4       : 80   2       : 45   3       : 56   10      : 55   1       : 47
                    10      : 69   4       : 40   4       : 44   4       : 33   6       : 41
                    2       : 50   5       : 30   5       : 34   8       : 25   5       : 39
                    (Other):117   (Other): 81   (Other): 95   (Other): 63   (Other): 66
  Bare.nuclei   Bl.cromatin  Normal.nucleoli    Mitoses          Class
 1       :402   2       :166   1       :443     1       :579   benign   :458
 10      :132   3       :165   10      : 61     2       : 35   malignant:241
 2       : 30   1       :152   3       : 44     3       : 33
 5       : 30   7       : 73   2       : 36     10      : 14
 3       : 28   4       : 40   8       : 24     4       : 12
 (Other): 61    5       : 34   6       : 22     7       :  9
 NA's    : 16   (Other): 69   (Other): 69     (Other): 17
```

Figure 1: Statistical details of original breast cancer dataset

```
diagnosis  radius_mean      texture_mean   perimeter_mean     area_mean      smoothness_mean
B:357     Min.   : 6.981   Min.   : 9.71   Min.   : 43.79   Min.   : 143.5   Min.   :0.05263
M:212     1st Qu.:11.700   1st Qu.:16.17   1st Qu.: 75.17   1st Qu.: 420.3   1st Qu.:0.08637
          Median :13.370   Median :18.84   Median : 86.24   Median : 551.1   Median :0.09587
          Mean   :14.127   Mean   :19.29   Mean   : 91.97   Mean   : 654.9   Mean   :0.09636
          3rd Qu.:15.780   3rd Qu.:21.80   3rd Qu.:104.10   3rd Qu.: 782.7   3rd Qu.:0.10530
          Max.   :28.110   Max.   :39.28   Max.   :188.50   Max.   :2501.0   Max.   :0.16340
compactness_mean  concavity_mean    concave_points_mean symmetry_mean    fractal_dimension_mean
Min.   :0.01938   Min.   :0.00000   Min.   :0.00000     Min.   :0.1060   Min.   :0.04996
1st Qu.:0.06492   1st Qu.:0.02956   1st Qu.:0.02031     1st Qu.:0.1619   1st Qu.:0.05770
Median :0.09263   Median :0.06154   Median :0.03350     Median :0.1792   Median :0.06154
Mean   :0.10434   Mean   :0.08880   Mean   :0.04892     Mean   :0.1812   Mean   :0.06280
3rd Qu.:0.13040   3rd Qu.:0.13070   3rd Qu.:0.07400     3rd Qu.:0.1957   3rd Qu.:0.06612
Max.   :0.34540   Max.   :0.42680   Max.   :0.20120     Max.   :0.3040   Max.   :0.09744
  radius_se        texture_se      perimeter_se       area_se        smoothness_se      compactness_se
Min.   :0.1115   Min.   :0.3602   Min.   : 0.757   Min.   :  6.802   Min.   :0.001713   Min.   :0.002252
1st Qu.:0.2324   1st Qu.:0.8339   1st Qu.: 1.606   1st Qu.: 17.850   1st Qu.:0.005169   1st Qu.:0.013080
Median :0.3242   Median :1.1080   Median : 2.287   Median : 24.530   Median :0.006380   Median :0.020450
Mean   :0.4052   Mean   :1.2169   Mean   : 2.866   Mean   : 40.337   Mean   :0.007041   Mean   :0.025478
3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190   3rd Qu.:0.008146   3rd Qu.:0.032450
Max.   :2.8730   Max.   :4.8850   Max.   :21.980   Max.   :542.200   Max.   :0.031130   Max.   :0.135400
 concavity_se     concave_points_se   symmetry_se       fractal_dimension_se  radius_worst
Min.   :0.00000   Min.   :0.000000   Min.   :0.007882   Min.   :0.0008948    Min.   : 7.93
1st Qu.:0.01509   1st Qu.:0.007638   1st Qu.:0.015160   1st Qu.:0.0022480    1st Qu.:13.01
Median :0.02589   Median :0.010930   Median :0.018730   Median :0.0031870    Median :14.97
Mean   :0.03189   Mean   :0.011796   Mean   :0.020542   Mean   :0.0037949    Mean   :16.27
3rd Qu.:0.04205   3rd Qu.:0.014710   3rd Qu.:0.023480   3rd Qu.:0.0045580    3rd Qu.:18.79
Max.   :0.39600   Max.   :0.052790   Max.   :0.078950   Max.   :0.0298400    Max.   :36.04
texture_worst   perimeter_worst    area_worst     smoothness_worst  compactness_worst concavity_worst
Min.   :12.02   Min.   : 50.41   Min.   : 185.2   Min.   :0.07117   Min.   :0.02729   Min.   :0.0000
1st Qu.:21.08   1st Qu.: 84.11   1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720   1st Qu.:0.1145
Median :25.41   Median : 97.66   Median : 686.5   Median :0.13130   Median :0.21190   Median :0.2267
Mean   :25.68   Mean   :107.26   Mean   : 880.6   Mean   :0.13237   Mean   :0.25427   Mean   :0.2722
3rd Qu.:29.72   3rd Qu.:125.40   3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910   3rd Qu.:0.3829
Max.   :49.54   Max.   :251.20   Max.   :4254.0   Max.   :0.22260   Max.   :1.05800   Max.   :1.2520
concave_points_worst symmetry_worst   fractal_dimension_worst
Min.   :0.00000      Min.   :0.1565   Min.   :0.05504
1st Qu.:0.06493      1st Qu.:0.2504   1st Qu.:0.07146
Median :0.09993      Median :0.2822   Median :0.08004
Mean   :0.11461      Mean   :0.2901   Mean   :0.08395
```

Figure 2: Statistical details of diagnostic breast cancer dataset

2. Original Paper Review

In the paper (Karabatak and Ince), the authors present an intelligent diagnosis system for breast cancer detection based on association rules (AR) and neural network (NN). In their study, dimension reduction of breast cancer database is performed using AR and intelligent classification is achieved using NN. The whole architecture of their detection system is presented in Fig.3.

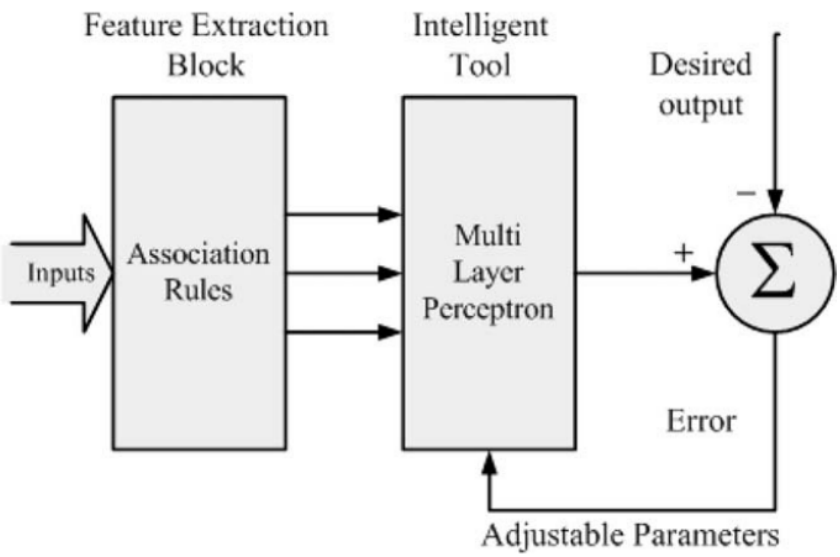

Figure 3: The model architecture in paper

2.1 Association Rules

Association Rule (AR) is a popular machine learning technique for data mining, with aim of retrieving the relations among variables in datasets. The core algorithm of the association rule, the Apriori Algorithm(Agarwal, Srikant, et al.), was proposed by Agrawal et al. in 1994 and is generally used to find rules with high support and confidence. Association rule is widely applied in market basket analysis to explore whether certain itemsets would be bought together. And in this paper, it is further extended to be applied to the feature extraction in classification issues for breast cancer prediction.

There are two types of AR mentioned and examined in their study. The idea of AR1 is to find a rule of $X \Rightarrow Y$ with the highest support value and confidence value. With this rule, itemsets in Y could be regarded as being dependent on itemsets in X and thus would not be included in the further neural network procedure. In terms of AR2, largest itemsets are collected for both classes respectively.

2.2 Neural Network

In paper, they implement a simple neural network with three layers: input layer, output layer and one hidden layer with 11 neurons. The activation functions for the first two layers are sigmoid, and linear activation function is applied on the output layer.

Table 2
MLP architecture and training parameters

Architecture	
The number of layers	3
The number of neuron on the layers	Input: 4, 8, 9 Hidden: 11 Output: 1
The initial weights and biases	Random
Activation functions	Tangent-sigmoid Tangent-sigmoid Linear
Training parameters	
Learning rule	Levenberg–Marquardt Back-propagation
Sum-squared error	0.01

Figure 4: Neural network settings in the original paper

2.3 Evaluation and Analysis

The proposed method is then evaluated by the 3-fold cross validation error rate. Through the comparative experiment among methods of NN, AR1+NN, AR2+NN, they concludes that AR is a feasible way to reduce feature dimension and the feature extraction process indeed could provide great benefits for pattern recognition and classification tasks. By reducing feature vector to a lower dimension while extracting the most useful information, it is promising to expect a better classification performance. Their detailed experimental result is shown in the following figure:

Table 3
Performance comparison for breast cancer detection using NN, AR1 + NN and AR2 + NN

The classifier	The epochs	Correct classified	Miss classified	Correct classification rate (%)
NN (9, 11, 1)	61	216	11	95.2
AR1 + NN (8, 11, 1)	44	221	6	97.4
AR2 + NN (4, 11, 1)	33[a]	217	10	95.6

[a] Goal is 0.01.

Figure 5: Results in the original paper

3. Work Reproduce

To reproduce the method proposed in this paper, our work includes following two parts: data preprocessing and implementation of association rules and neural network.

3.1 Data Preprocessing

First we analyzed the original dataset and found a missing value problem. There are 699 records in the dataset, however there are attributes missing for some specific records. So we deleted these observation with missing attribute and got 683 observations finally. Then normalization is applied to the dataset.

In the following experiments, we will use these complete observations to train and evaluate most models except XGBoost (Since it can handle missing value by itself, we will discuss it in the following section).

As with original paper, we use 3-fold cross validation to build and test our models. To keep the balance of dataset, we extract and split the positive and negative records from the dataset. Then split them evenly in three parts, and combine each positive and each negative chunk into one group (now we have three groups). To implement the final cross validation method we then make three different combinations of these three groups of data (two groups for training and one group for testing, so now we have three cross validation sets). Now these three sets have the same

distribution for training and test data.

3.2 Algorithm implementation

3.2.1 AR rules

To implement AR rules, we used *apriori* function in R. The results of AR rules are as the following:

```
    lhs                    rhs             support confidence     lift count
[1] {Cl.thickness=1,
     Cell.shape=1,
     Marg.adhesion=1,
     Normal.nucleoli=1} => {Cell.size=1} 0.1516452          1 1.820312   106
```

Figure 6: AR1 to eliminate one feature

```
                          items   support count
37                {Mitoses=1} 0.5560166   134
36            {Bare.nuclei=10} 0.5352697   129
                                        items   support count
181    {Cell.size=1,Normal.nucleoli=1,Mitoses=1} 0.7620087   349
```

Figure 7: AR2 to choose 4 features

Both results are consistent to the original paper.

3.2.2 Neural network

We implemented the neural network by using *neuralnet* function. As with the original paper, we set the number of neurons in the hidden layer to 11, then set *linear.output* to F in order to solve classification problem. The following graph is the architecture of the neural network. As default in R, the activation function for each hidden layer is sigmoid function. Each edge between neurons represents the weight that shall be used to calculate the weighted input of the next layer. The blue neuron represents the bias term for each layer.

When we use the *neuralnet* package, we find that the update of weights are based on each training data point rather than each batch. This can slow down the speed of convergence and make

the performance worse because the direction of updating vector is not stable for a single data point but stable for a batch of data points.

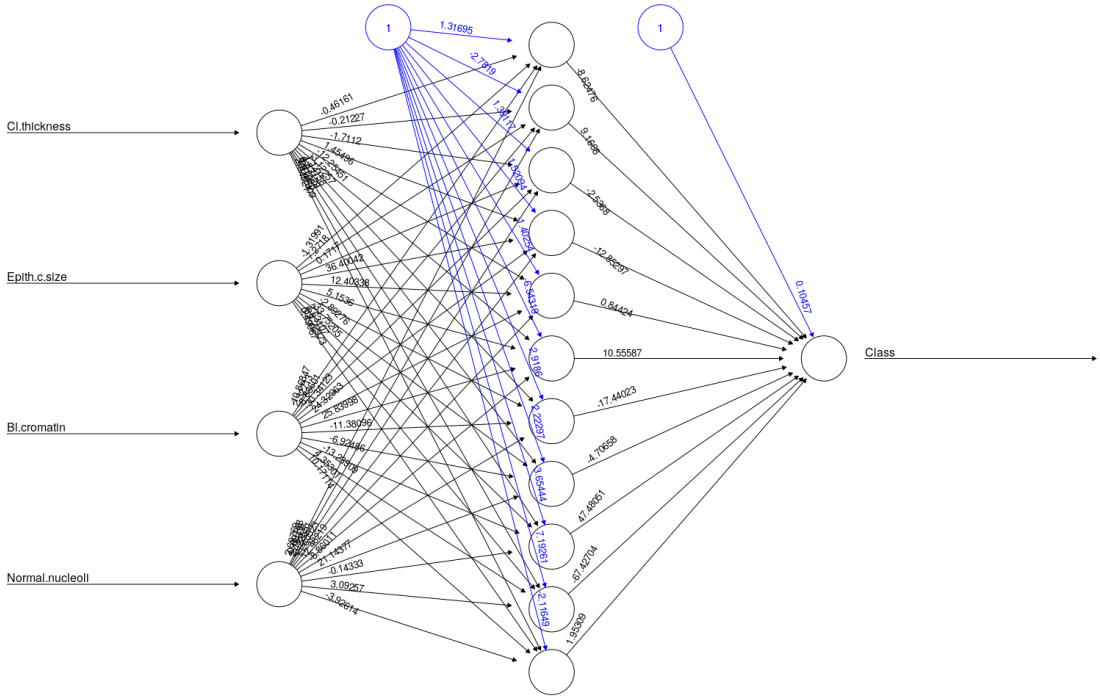

Figure 8: The architecture of our neural network

3.3 Experiments - Reproducing Result vs Original Result

The following table are the comparison of reproducing results and the original results. AR1 rule + NN shows the best performance in both results, and our results are closed to paper's results.

Table 1: Results of reproducing work and original paper

	AR1+NN	AR2+NN	NN
Reproducing Work	**0.968**	0.962	0.953
Original Paper	**0.974**	0.956	0.952

4. New Approach

We also did many other experiments to evaluate different models' performances on these two datasets

4.1 PCA

Both AR rules and PCA are used to reduce the dimension of dataset. In general, there are three reasons to do dimension reduction:

1. Reduce the computation of algorithm
2. Eliminate some noise in original datasets
3. Make the data more explainable

The principle and details about PCA will not be discussed in this paper because it has been discussed during the class hours. We focus on the comparison of PCA and AR rules. The following table shows that the AR1 + NN method performs better than our PCA + NN method. Some reasons behind this may be:

1. AR rules perform better on data with less attributes
2. PCA is less explainable than AR rules

There is a tip when we did PCA experiment. When we did the cross validation, we **cannot** apply PCA on the whole dataset, because the test data will be used. So we need to apply PCA separately on training and test data.

Table 2: Results of original paper and PCA + NN

	AR1+NN	PCA+NN
Original Dataset	0.974	0.969

4.2 SVM

SVM(Boser, Guyon, and Vapnik) is a general model for classification. The core idea of the two-class support vector machine is to find the best hyperplane that could perfectly or softly split data of two classes. To evaluate the performance of SVM for both original dataset and diagnosis dataset, we also invoked SVM method. We did two experiments for this purpose. One is AR1 rule + SVM and the other is SVM for all features. The following table shows the performances of two models. One can find that if we use AR1 to eliminate one feature (Cell.size) from the original features, the accuracy is better.

Table 3: Results of SVM

	AR1+SVM	SVM
Original Dataset	0.966	0.965
Diagnosis Dataset	0.973	0.968

4.3 XGBoost

XGBoost is a main method that we focus on.

4.3.1 Overview

XGBoost(Chen and Guestrin) is a general Tree Boosting algorithm based on Gradient Boosting Decision Tree (GBDT). Because of the powerful learning ability and fast training speed, XGBoost is the most popular machine learning algorithm in Kaggle competition. Basically, this algorithm dominates the field of standard tabular dataset as opposed to high-correlated data, such as images and videos. Moreover, in KDD Cup 2015[3] All of the top 10 teams use XGBoost as their algorithm.

4.3.2 Improvements over GBDT

The first improvement of XGBoost is that it uses Taylor series to expand the loss function. By using the first two order of the loss function as the residual, it generalizes the format of error as opposite to only residual squared error.

[3] http://kddcup2015.com/submissionrank.html

The second improvement of XGBoost is that it adds a regularization term of the loss function which reduces the complexity of the model. Basically, there are two components in the regularization term: one is the number of tree leafs, the other is the sum of squared scores on each leaf. From the point of Bias-variance trade-off, this regularization term reduces the variance of the model and fixs the overfitting problem.

The third improvement of XGBoost is that it supports column sampling which means it does not use all features of the data during training process(the same as Random Forest). In this way, it reduces not only the overfitting but also the computing time. However, the traditional GBDT algorithm does not use column sampling.

The fourth improvement of XGBoost is the treatment of missing value. XGBoost will consider the missing values as sparse matrix, and it does not calculate the missing value when splitting the node. The data point with missing value will be split to the left or right node based on the cost of splitting. If there are no missing values in the training dataset, then the test data point with missing values will be split to the right node according to the XGBoost paper. However, there is a potential risk here because we assert that the distribution of the training dataset is the same as the test dataset.

4.4 XGBoost in R

XGBoost has been implemented in R language.[4] Users can custom their own loss function as they need. There are some typical parameter settings of XGBoost in R:

1. *max.depth*. The depth of each tree. Usually set 2 in simple data.

2. *nthread*. The number of cpu threads that are used in computation.

3. *nround*. The number of passes when training data. Usually 2 and the second round will further reduce the difference between prediction and ground truth.

4. *verbose*. Equals 1 if one wants to see the training process. It will display the training error for each round.

[4] http://xgboost.readthedocs.io/en/latest/Rpackage/xgboostPresentation.html

4.4.1 Feature selection of XGBoost

XGBoost package also provides a function of estimating feature importance during the training process. In general, importance of features provides a metric to evaluate how much each feature is used when building the ensembles of decision trees. To be more specific, we can use gini-index to calculate the importance of each feature in a single decision tree. The amount that each feature split improves the gini-index is weighted by the number of data points in that node. Finally, the feature importances are averaged over the number of all decision trees in the model.

4.5 XGBoost experiments

To evaluate the performance of XGBoost with different settings of parameters, we did the evaluation of models with different tree depths. We set the depth to 1, 2 and 3 in two datasets. The result is as the following table: To visualize the architecture of different XGBoost trees, we

Table 4: Accuracy of classification for trees with different depths

	Depth = 1	Depth = 2	Depth = 3
Original Dataset	93.13%	97.00%	99.57%
Diagnosis Dataset	94.72%	98.94%	97.89%

also plot them in the small dataset. The followings are three trees with different depths and their corresponding importance measurements.

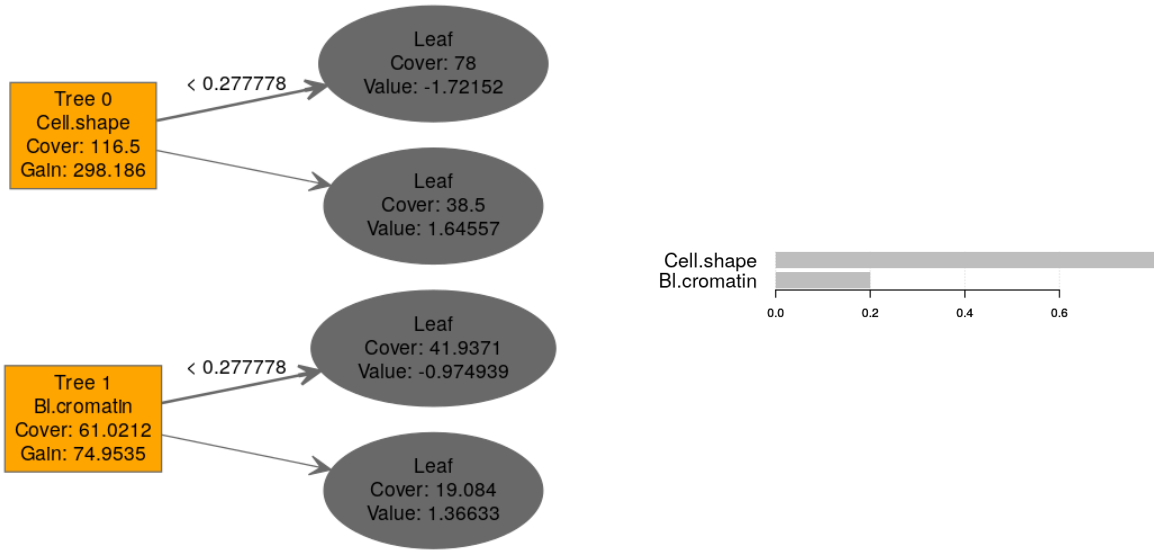

Figure 9: Architecture for 1-depth XGBoost tree

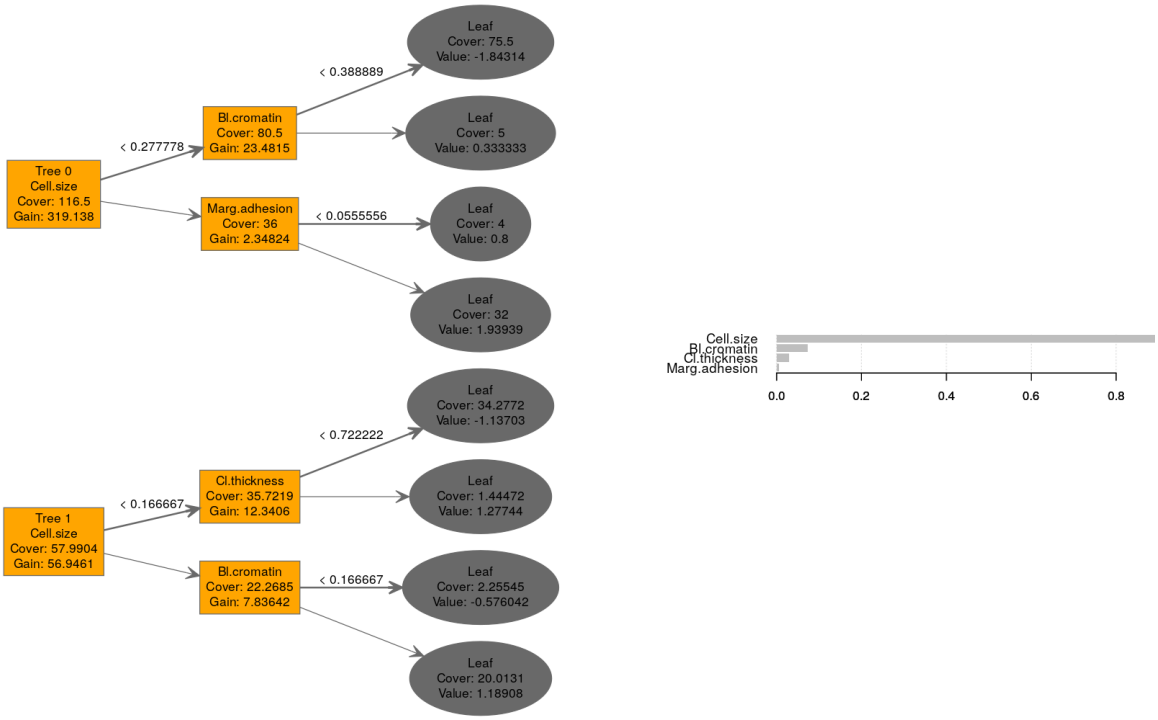

Figure 10: Architecture for 2-depth XGBoost tree

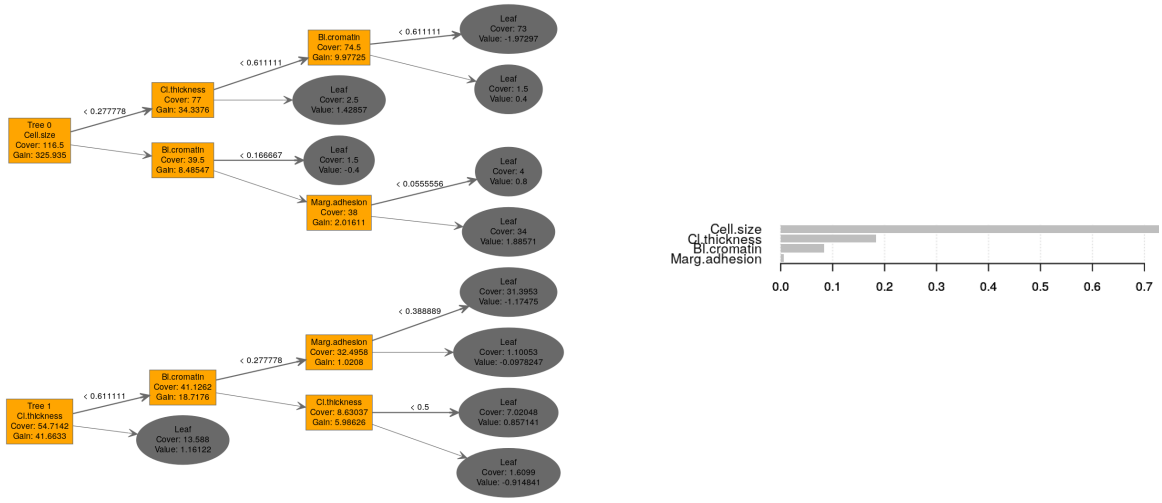

Figure 11: Architecture for 3-depth XGBoost tree

Basically, we focus on the Gain attribute of the importance measurement. The Gain attribute indicates the contribution of each feature when training the model (for each tree in a single updating iteration). The higher the Gain is, the more important the feature is (means more importance when predicting).

5. Results and discussions

The following tables are summary of our experiments. First, we reproduced very closed results shown in the original paper. Second, We applied three methods in original paper on a new breast cancer dataset. Third, We proposed the results of other machine learning algorithms applied on both datasets. Finally, we show that XGBoost performs the best over all other methods that we used.

Table 5: Reproduce results of the original approaches

	AR1+NN	AR2+NN	NN
Original Dataset	**0.968**	0.962	0.953
Diagnosis Dataset	**0.970**	0.951	0.961

Table 6: Results of new approaches

	PCA+NN	AR1+SVM	SVM	XGBoost
Original Dataset	0.969	0.966	0.965	**0.9957**
Diagnosis Dataset	0.979	0.973	0.968	**0.9894**

Table 7: Cumulative Results

	PCA+NN	AR1+NN	AR1+SVM	SVM	XGBoost
Original Dataset	0.969	0.968	0.966	0.965	**0.9957**
Diagnosis Dataset	0.979	0.970	0.973	0.968	**0.9894**

5.1 Some observations

5.1.1 Comparison of AR and PCA

When we process datasets with less attributes, we can use both AR and PCA to do feature selection. However, the computation will be more expensive if we still use AR because of the number of combinations. Different from market basket analysis where transactions can be easily represented by binary vectors, feature reduction using AR would involve multiple steps like discretization, interval generation, binarization if values are general real numbers, which would largely increased the algorithm complexity. So we recommend use PCA to reduce the number of features for datasets with many features.

5.1.2 More discussions on missing values

We noticed that there are some rules of thumb for handling missing values when we did this project.

1. Tree-based models are not sensitive to missing values, so it can be applied when there are many missing values in dataset.

2. When involving the distance measurement, the missing values can influence the model much more, such as SVM and KNN.

3. There is always distance computation in the loss function of linear models which leads to the problem calculation difference between the true value and prediction.

Works Cited

Agarwal, Rakesh, Ramakrishnan Srikant, et al. “Fast algorithms for mining association rules.” *Proc. of the 20th VLDB Conference*. 1994. 487–499. Print.

Boser, Bernhard E, Isabelle M Guyon, and Vladimir N Vapnik. “A training algorithm for optimal margin classifiers.” *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992. 144–152. Print.

Chen, Tianqi and Carlos Guestrin. “Xgboost: A scalable tree boosting system.” *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 2016. 785–794. Print.

Karabatak, Murat and M Cevdet Ince. “An expert system for detection of breast cancer based on association rules and neural network.” *Expert systems with Applications* 36.2 (2009): 3465–3469. Print.