Ensemble Learning in Diagnosing Breast Cancer

Bill Tang (bt294)| Lini Tan (lt398)

Cornell University

Author Note

Bill Tang, School of Engineering, Cornell University.

Lini Tan, School of Art and Science, Cornell University

Contact information:

Bill Tang: [bt294@cornell.edu](mailto:bt294@cornell.edu)

LIni Tan: [lt398@cornell.edu](mailto:lt398@cornell.edu)

Abstract

The study of applying ensemble learning in diagnosis of breast cancer probes usage of many fundamental machine learning techniques that are widely used in both industry and academia nowadays. Techniques include Decision Tree, Random forest, KNN, Naïve Bayes, Logistic Regression, Neural Network. The idea is to train various techniques with same training data set, then to test the results on the rest of the data set, and eventually to measure the goodness of fit between the real data and the expected data by looking at the cross-validation scores.

The main computer languages used in this study include Python, JavaScript, JQuery.

Introduction

The emergence of our thought on diagnosing breast cancer with machine learning method comes from Professor Bart, who taught Artificial intelligence at Cornell University. In 2017 fall, lecture slide 13, he mentioned how Decision Tree surpassed human experts and has been successfully diagnosing breast cancer. Out of curiosity, we wondered two questions: Why Decision Tree not other machine learning models is used? Is there a better machine learning technique that results the better prediction?

Following curiosity about this two questions, we conducted this case study. In this report, we will first briefly introduce each method we used. Then, we demonstrate the test methods followed by test results. In the end, we will compare the results from different method, and make a conclusion based on the results.

Introduction on Testing Methods

1. Decision Tree

Decision Tree, a form of supervised learning, is widely used to support making decisions by generating a tree-like graph.

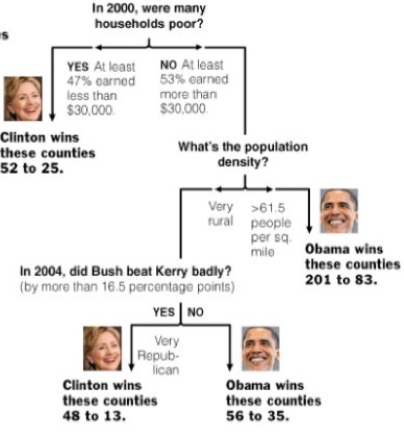


Figure 1.1 Decision Tree.

*This simple example regarding 2008 U.S. President election is decision tree is one of many quintessential cases of application of decision tree.*

The decision tree is very natural to human. A decision is made by walking down the tree from the root. At each level, a decision node specifies a choice of some attribute with two or more alternatives. Every decision node is part of a path to a leaf node which indicates the classification of a data set.

To construct a Decision Tree consistent with the training example, Decision Tree learning algorithm is used. The idea is to recursively choose the most significant attribute as root, and then use a top-down greedy search through the space of possible Decision Tree. The significance of the attribute is determined by something called “Information Gain” which is calculated by using the concept of “Information Entropy”.

1. Random Forest

Similar to decision tree, a Random Forest algorithm is a supervised classification algorithm working as a large collection of decorrelated decision trees. Instead of only one tree as with Decision Tree, many Decision Trees are used in Random Forest Algorithm. Also, instead of using information gain and gini index for calculating the root node, finding the root node and splitting the feature nodes will be randomized.

To construct a Random Forest from a sample, the concept of bagging is used. The bagging is to average noisy and unbiased models in order to create a model with low variance. The basic idea is to divide the sample into M subsets, and then create M decision trees with the corresponding data using Random Forest Algorithm.

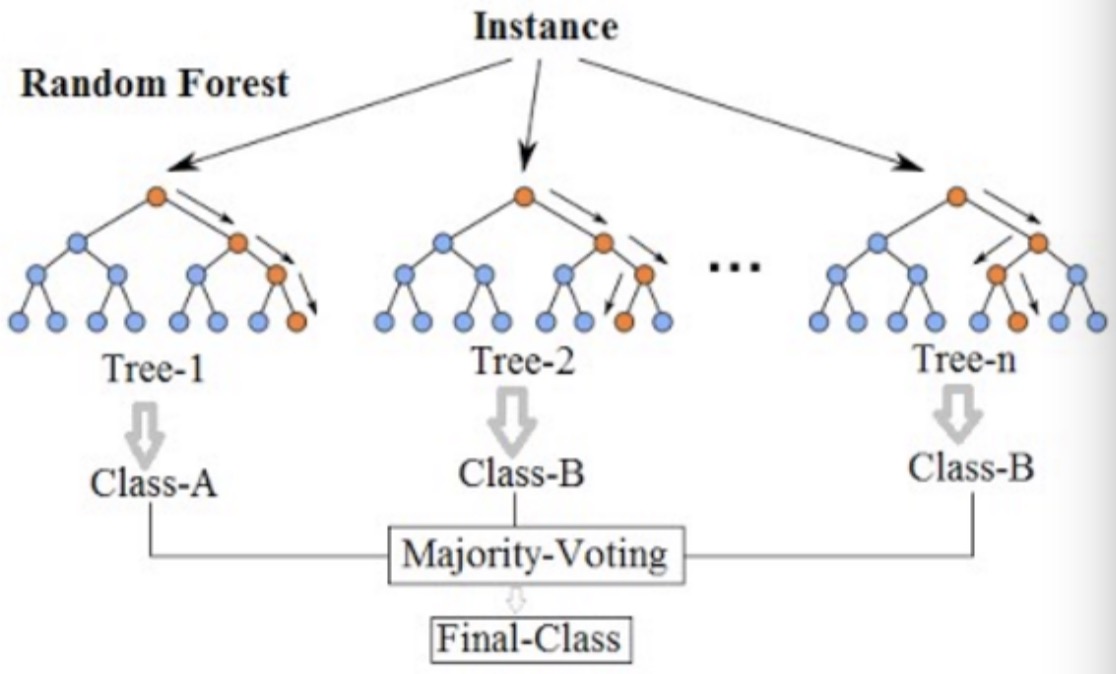


Figure 1.2 Random Forest

*There exist three decision trees in this random forest. Three classifications will be obtained from three trees to go into the final voting. The prediction is “class B” in this case.*

To make a decision using Random Forest, an instance is feed into the root of each sub-decision tree. Sub-trees will then produce classifications from each one of them. The final prediction is based on majority voting among the classifications of Decision Trees.

1. K-nearest Neighbor algorithm

The k-nearest neighbors algorithm is a supervised learning, non-parametric method used for classification. The idea of clustering is used to group a set of objects in a way that objects in the same group are more similar to each other than those in other groups.

In k-NN algorithm, the classification of a new data point is based on the “distance metric” to k nearest neighbors in the training data set. Then, this new object is classified by a majority vote of its neighbors.

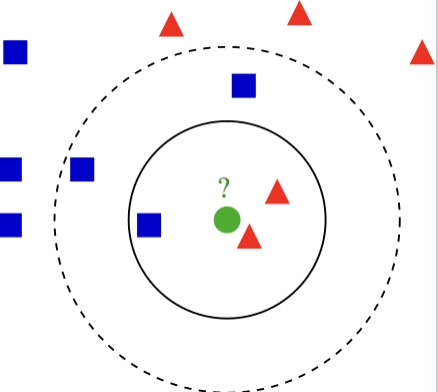


Figure 1.3 k-NN algorithm

*The new test object is colored in green.*

*If k=3 (solid line circle), the new object is assigned to the red triangle class since there are two triangles and only one square in the circle.*

*If k = 5 (dashed line circle), the new test object should be assigned to blue square class because there are three squares, but only two triangles.*

Correctly choosing the k factor determines the total accuracy of such algorithm. To gain optimal value of k, we can segregate the training set and validation set from the initial data set. Then, plot the validation error curve to obtain the optimal value of k at the minimal error point.

1. Naïve Bayes Method

In machine learning, Naïve Bayes method forms a group of simple probabilistic classifiers by applying Bayes’ theorem with strong independence assumptions between the features.

Given a small set of training data, Naïve Bayes classifiers can be trained very efficiently in a supervised learning setting. To estimate the parameters for naïve Bayes models, the idea of maximum likelihood is used, meaning one can work with the naïve Bayes model without accepting Bayesian probability or applying Bayesian methods.

1. Logistic Regression

Logistic regression was developed by David Cox, which is used to estimate the probability of a binary response based on one or more predictor variables. In the regression model, the dependent variable is categorical. It allows one today that the presence of a risk factor increases the odds of a given outcome by a specific factor.

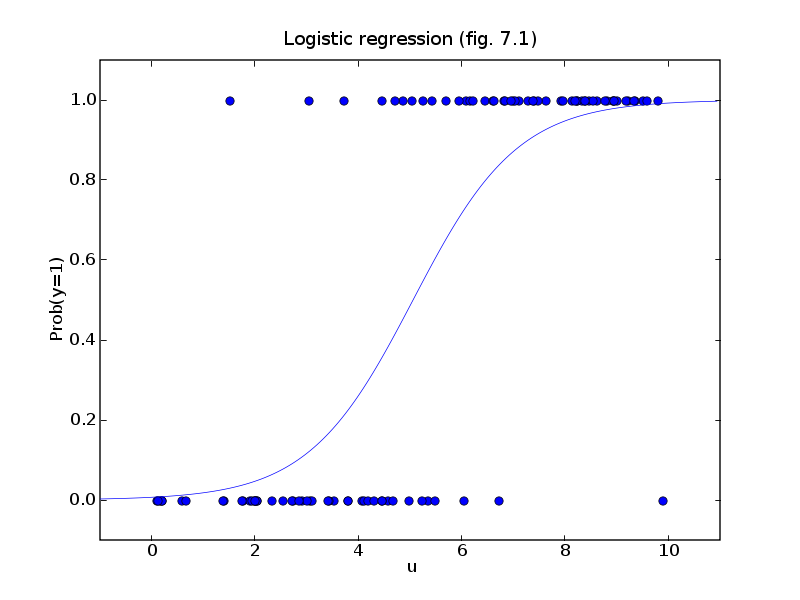


Figure 1.4 Logistic Regression

*Like the figure on the left, logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.*

The logistic regression is simple to implement and easy to compute. However, the decision boundary has to be linear (i.e. separable by a hyperplane). It could also end up being a feature selection problem as well. This is the problem we need to consider when using logistic regression.

1. Neural Network

Artificial neural networks (ANNs) or [connectionist](https://en.wikipedia.org/wiki/Connectionism) systems are computing systems inspired by the [biological neural networks](https://en.wikipedia.org/wiki/Biological_neural_network) that constitute animal brains. Such systems learn (progressively improve performance on) tasks by considering examples, generally without task-specific programming.

To construct a neural network, we consider it as a collection of connected units called artificial neurons. Each connection between neurons can transmit a signal from one to another. The receiving neuron can process the signals and then signal neurons connected to it.



Figure 1.5 Neural Network

*Here, the red nodes are input, the blue nodes are hidden layer and output are green nodes. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.*

In order to learn multi-layer neural nets, we consider the backpropagation algorithm for training such networks.

Testing Approaches and Results

**Data Searching Stage**

To answer two questions mentioned in the introduction of this report, we first searched raw data on Machine Learning Repository. We obtained the Mammographic Mass Data Set which is published by Image Processing and Medical Engineering from Fraunhofer Institute for Integrated Circuits in Germany.

As one of the most effective method for breast cancer screening today, Mammography, on the other side however, has low positive predictive value of breast biopsy. The reason for this is the approximately 70% unnecessary breast biopsies. Many computer-aid diagnosis (CAD) systems had been developed to decrease the such high rate. Data gathered by professionals can be used to train the systems, and to compare the performance of CAD systems to that of radiologists.

This data set contains 961 data entries, and each tuple has six attributes. They are:

1. BI-RADS assessment (from 1 to 5): an assessment of how confident the severity classifications. This is a subjective attribute assigned by human, but not a predictive feature used to train the system. Thus, we discarded this attribute.

2. Age: patient's age in years (integer)

3. Shape: mass shape: round=1, oval=2, lobular=3, irregular=4

4. Margin: mass margin: circumscribed=1, micro lobulated=2, obscured=3, ill-defined=4, speculated=5

5. Density: mass density high=1, iso=2, low=3, fat-containing=4

6. Severity (benign=0 or malignant=1):

Our goal is to apply six machine learning models to this data, and measure the success of

results based on the cross-validation value between each trained model and the data set. We want to see if our models can beat the human radiologists’ positive diagnosis which is round 70%, and to conclude which model works the best for the sample data.

**Data Processing Phase**

Before we feed our data into models, we need to process the data first. This includes data I/O, deleting BI-RADS column, taking care of missing data, normalizing the data, and extract the data from table frame in Panda into two Numpy arrays with just numerical data. Afterwards, the data can be used by scikit\_learn to feed into various models to be trained.

The result test frame will look like this:

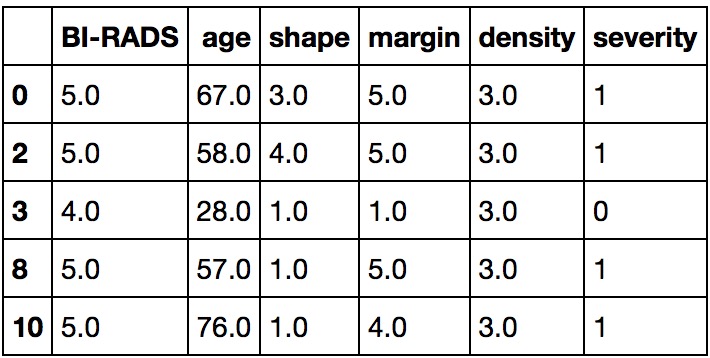


Figure 2.1 Test Frame

*This picture offers the head of the data table with six features.*

The result test Numpy arrays will look like this:

Features array example:

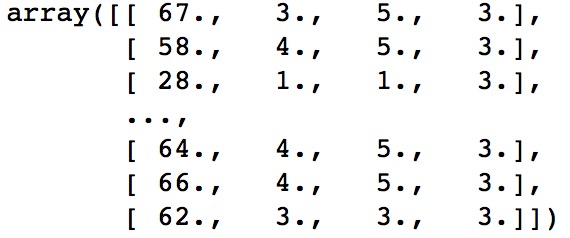


Figure 2.2 Features array

*This picture offers an example of what does feature array look like. In this array, only four features are included (age, shape, margin, density). BI-RADS, not predictive feature, and severity, target feature, are excluded.*

Target test array example:

Figure 2.3 Target Feature array

*Each integer represents the severity of breast cancer condition correspondent to each data tuple in above array. 0 = benign and 1 = malignant*



**Testing Phase**

1. Decision Tree

After having proper input, we can now start feeding data into Decision Tree model. This can be done by importing sklearn package into Jupiter, a python based language.

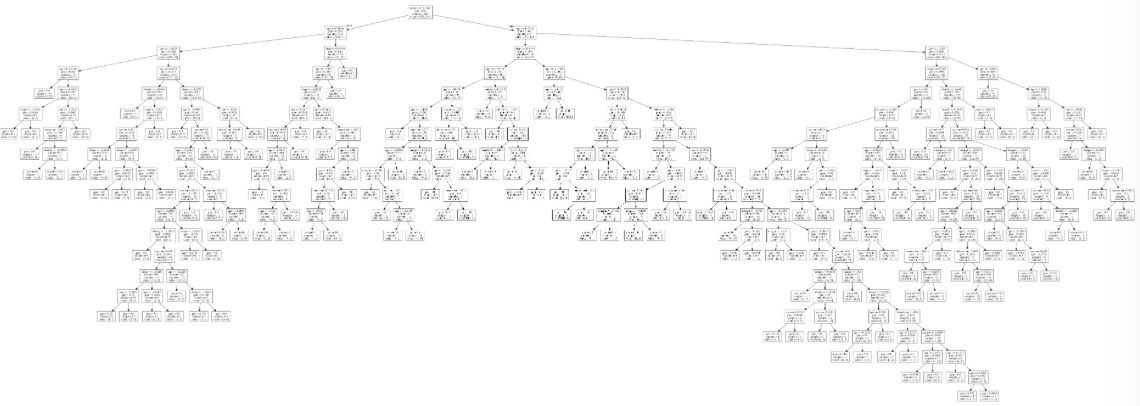
First of all, we split the data set into 75% training set and 25% testing set by applying “train-test-split” function. After this, we simply used “DecisionTreeClassifier()” function to generate a Decision Tree Classifier, and then simply call on the result of this function by using “fit (training\_features\_input, training\_target\_input)”. Finally, to show the result graphically, we simply call graphing functions to draw the decision tree: 

Figure 2.4 Decision Tree result

*This is the graphical result from our data. Each node represents a decision node, and the leaf is the classification. For clear graph, please see git code.*

With the decision tree constructed, it is very easy to measure the accuracy of this decision tree model by using test set data. The cross-validation score can be calculated by calling function “score( testing\_features\_input, testing\_targets)”. The result we got for this method is 0.735577.

1. Random forest

Following the same method as we did for Decision Tree model, we are capable of construct a Random Forest by simply calling function “RandomForestClassifier()” with different number of trees. The cross-validation score in this case can be calculated by first getting the score for each single tree, and then by calculating the mean of trees scores. For 10 trees, we got 0.75405. For 20 trees, we got 0.76614. for 100 trees, we got 0.7590.

1. KNN

For k-Nearest-Neighbors model, in the same package “sklearn”, the function “KNeighborsClassifier ()” can be used. And the score of this model can be calculated using the same method as before.

The hard part for KNN model is to find the optimal K value. To solve this problem, we looped from K=50 to K=120, we found the optimal K is 103, and the score is 0.80853.

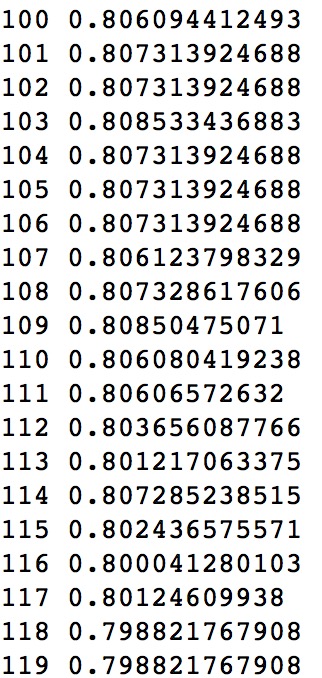


Figure 2.5 Cross-validation scores for KNN model

*The figure on the left shows the looping results on different K values. As we can see, the score drops around K =103.*

1. Naïve Bayes

To calculate the score of naïve bayes, we import sklearn package. We firstly calculate the min-max of the data and then use the MultinomialNB function to build the model.

The mean of cross\_value score of logistic regression is 0.784.

1. Logistic Regression

To calculate the score of logistic regression, we import sklearn package. We call the built-in logisticRegression function and then calculate the mean. It’s pretty straightforward.

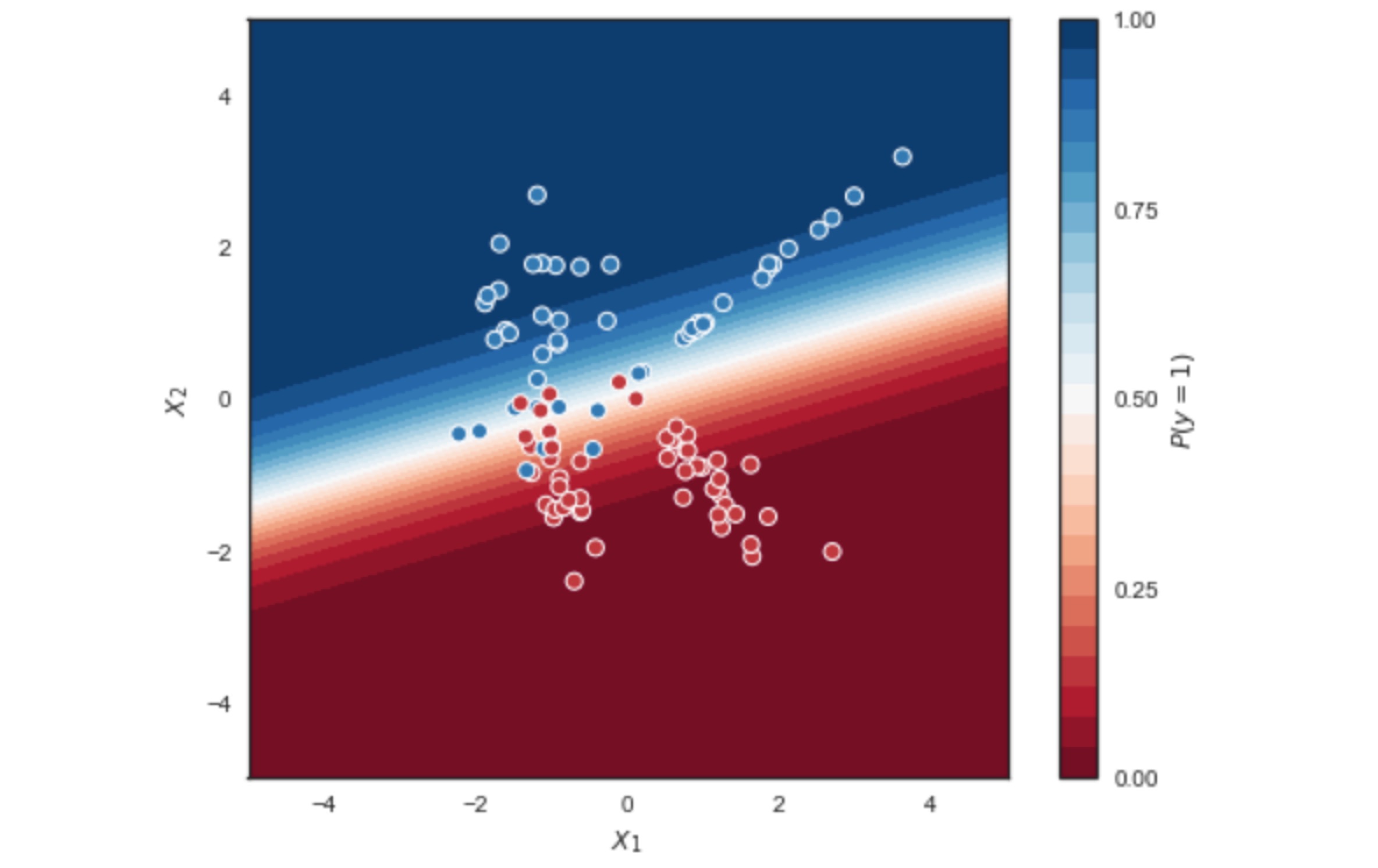


Figure 2.6 Logistic Regression

*In the figure, we can see the logistic regression model the data pretty well*

The mean of cross\_value score of logistic regression is 0.80736.

1. Neural Network

We import Keras package to compute the score of neural network. Firstly, we build a sequential model and add different layers. In this case, we find the hidden layer did not improve the performance so we use single layer here. By testing different variable, we find 7-unit input layers work best. So we choose this model.

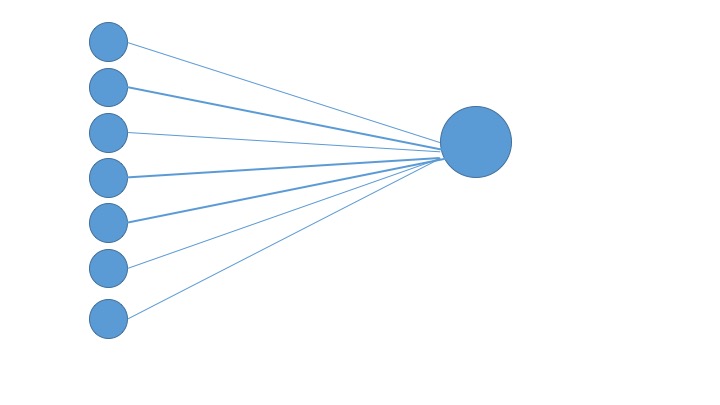


Figure 2.7 Neural Network

*We have 4 feature inputs going into a 7-unit layer and 1 output layer*

Then we create a binary classifier to estimate the score and output the mean score.

The mean of cross\_value of neural network is 0.801.

Conclusion

After a thorough test on six wildly known machine learning techniques, we got the test cross-validation scores with each different models. The scores are displayed on the right side of each column correspondent to the machine learning techniques labeled on the left.

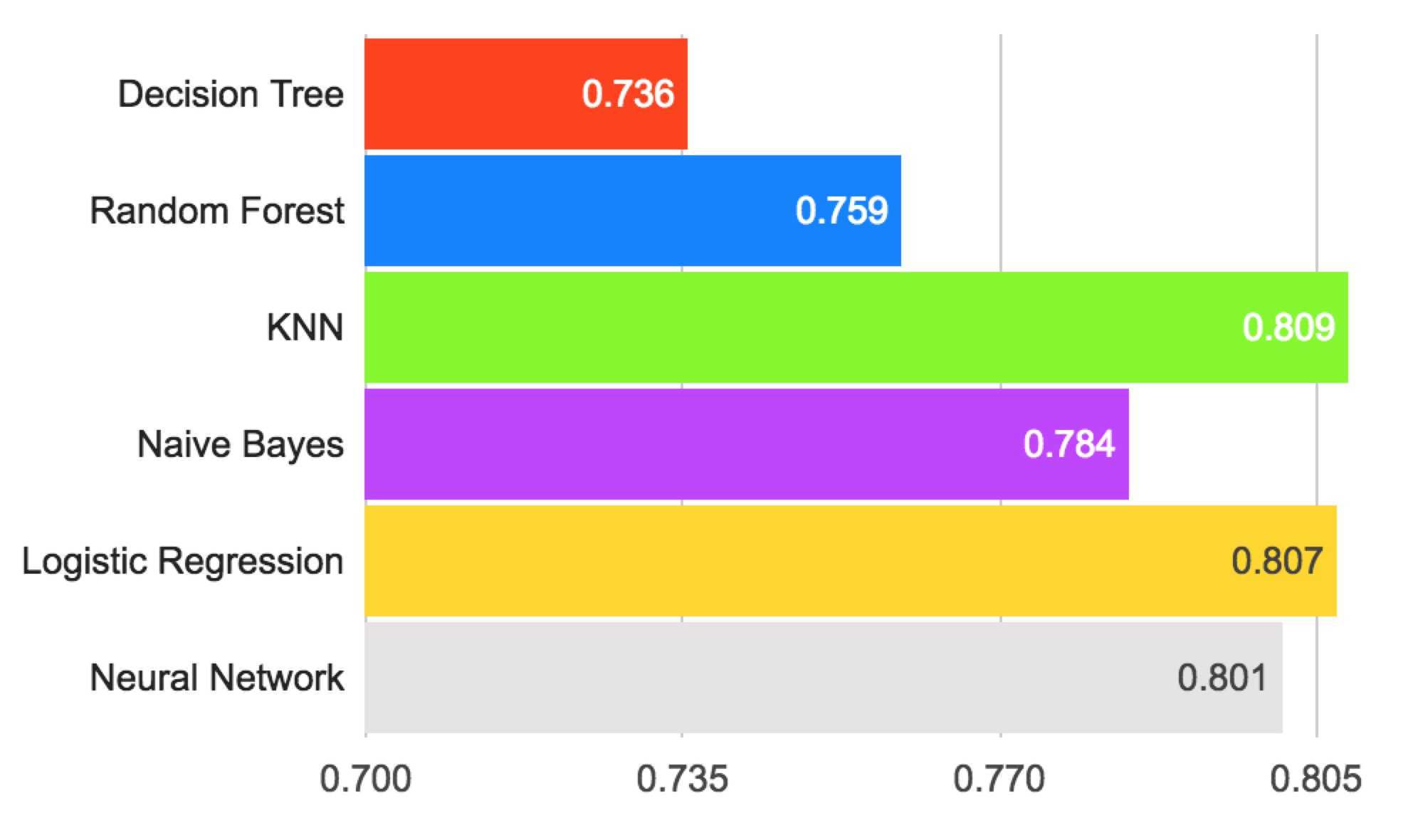


Figure 3.1 Cost Comparison

*The picture on the left shows the cost comparison among six machine learning techniques. The Decision tree scores the lowest, and the KNN method scores the highest.*

Surprisingly, decision tree which is wildly used in hospital seems to perform the worst among six methods. On the other hand, KNN model, neural Networks, and simple Logistic Regression all perform relatively well, beating human radiologists’ score of about 0.70.

Granted, more data sets should be fed into our models to rule out uncertainties. However, we believe our study still offers a glimpse of the performance of each different machine learning techniques on breast cancer diagnosis. Hopefully, a more accurate technique will be chosen to serve people, to save doctors’ time, and to save patients’ money.

Git

All of our codes stored in the github repository: <https://github.com/linitan/cs4701>

Reference

*CS4700 Introduction to Artificial Intelligence, Cornell, Lecture 13 p63 “how well does it work?”*

*How the random forest algorithm works in machine learning. (2017). Dataaspirant. Retrieved 14 December 2017, from* [*http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/*](http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/)

*Random Forest Template for TIBCO Spotfire® - Wiki page | TIBCO Community. (2017). Community.tibco.com. Retrieved 14 December 2017, from https://community.tibco.com/wiki/random-forest-template-tibco-spotfirer-wiki-page*

[*https://en.wikipedia.org/wiki/K-nearest\_neighbors\_algorithm*](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm)

*Introduction to KNN, K-Nearest Neighbors : Simplified. (2014). Analytics Vidhya. Retrieved 14 December 2017, from* [*https://www.analyticsvidhya.com/blog/2014/10/introduction-k-neighbours-algorithm-clustering/*](https://www.analyticsvidhya.com/blog/2014/10/introduction-k-neighbours-algorithm-clustering/)

[*Russell, Stuart*](https://en.wikipedia.org/wiki/Stuart_J._Russell)*;*[*Norvig, Peter*](https://en.wikipedia.org/wiki/Peter_Norvig)*(2003) [1995].*[Artificial Intelligence: A Modern Approach](https://en.wikipedia.org/wiki/Artificial_Intelligence:_A_Modern_Approach)*(2nd ed.). Prentice Hall.*[*ISBN*](https://en.wikipedia.org/wiki/International_Standard_Book_Number)[*978-0137903955*](https://en.wikipedia.org/wiki/Special:BookSources/978-0137903955)*.*

*Rish, Irina (2001).*[An empirical study of the naive Bayes classifier](http://www.research.ibm.com/people/r/rish/papers/RC22230.pdf)*(PDF). IJCAI Workshop on Empirical Methods in AI.*

*The Pros and Cons of Logistic Regression Versus Decision Trees in Predictive Modeling. (2017). HuffPost. Retrieved 14 December 2017, from* [*https://www.huffingtonpost.com/entry/the-pros-and-cons-of-logistic-regression-versus-decision\_us\_594330ffe4b0d188d027fd1d*](https://www.huffingtonpost.com/entry/the-pros-and-cons-of-logistic-regression-versus-decision_us_594330ffe4b0d188d027fd1d)

*David A. Freedman (2009). Statistical Models: Theory and Practice. Cambridge University Press. p. 128.*

*UCI Machine Learning Repository: Mammographic Mass Data Set. (2017). Archive.ics.uci.edu. Retrieved 14 December 2017, from* [*https://archive.ics.uci.edu/ml/datasets/M*](https://archive.ics.uci.edu/ml/datasets/M)

*Minsky, Marvin; Papert, Seymour (1969). Perceptrons: An Introduction to Computational Geometry. MIT Press. ISBN 0-262-63022-2.*