A comparison of Machine Learning Classifier Performance in a Binary Classification Domain

Gustavo Arismendi 1058672

Kunpeng Zhang 1623451

Abstract:

The purpose of this paper is to compare multiple machine learning classifiers in a binary classification domain represented by the wisconsin breast cancer dataset. We first compared classifiers by running them with chosen default parameters. Their ranking from best to worse based on accuracy was logistic regression, random forest, support vector machine, decision tree, and k-nearest neighbor. Next, we fine tuned the hyperparameters of each algorithm through multiple techniques and ran the algorithms to compare the new performances. Our final comparison show that logistic regression and random forest were tied for first place, support vector machine was second, and decision trees and k-nearest neighbor tied for last. We selected logistic regression as the best algorithm to use in this binary classification domain due to its high accuracy, ease of use, understanding, and implementation when compared to random forest.

Part I - Survey of Machine Learning Classifiers

1. Introduction

Over the past decade there has been a dramatic increase in the interest of machine learning models to specific application domains. One of the reasons for this increase is that academia and different industries see that machine learning has “the potential to become one of the key components of intelligent information systems, enabling compact generalizations, inferred from large databases of recorded information, to be applied as *knowledge* in various practical ways -such as being embedded in automatic processes like expert systems, or used directly for communicating with human experts and for educational purposes” [1].

As a subfield of artificial intelligence, machine learning has been divided into different types. There is supervised, unsupervised, reinforcement, and evolutionary learning. In this paper we will focus on the area of supervised learning, more specifically on

the performance that five supervised learning algorithms have on the application domain of breast cancer tumor classification. Our process to select the algorithms that we used for this domain was guided by two main criteria. The first criteria is our understanding of the usage, applications, advantages and disadvantages of each algorithm. The second criteria was an analysis of The 2017 State of Data Science and Machine Learning report by Kaggle. In this report, we noticed that most companies regarding of their size and industry (with the exception of military and security) used a set of main algorithms for their applications and experiments [2]. Using these two criteria we have decided to focus on comparing the performance of the logistic regression, support vector machines, decision trees, random forest, and k-nearest neighbor algorithms.

Now, we may be tempted to believe that once we have an existing dataset with correct classes labeled we can just go ahead and use our favorite algorithm every time to obtain the results we expect. However, to paraphrase the “no free lunch” theorem, there is no single algorithm that works best for all problems [3]. Because of 1) Bias-variance tradeoff. If we have available several different, but equally good, training data sets. A learning algorithm is biased for a particular input x if, when trained on each of these data sets, it is systematically incorrect when predicting the correct output for x. A learning algorithm has high variance for a particular input x if it predicts different output values when trained on different training sets. The prediction error of a learned classifier is related to the sum of the bias and the variance of the learning algorithm. A learning algorithm with low bias must be "flexible" so that it can fit the data well. But if the learning algorithm is too flexible, it will fit each training data set differently, and hence have high variance. 2) Function complexity and amount of training data. The amount of training data available relative to the complexity of the "true" function (classifier or regression function). If the true function is simple, then an "inflexible" learning algorithm with high bias and low variance will be able to learn it from a small amount of data. But if the true function is highly complex, then the function will only be learnable from a very large amount of training data and using a "flexible" learning algorithm with low bias and high variance.3) Dimensionality of the input space. If the input feature vectors have very high dimension, the learning problem can be difficult even if the true function only depends on a small number of those features. This is because the many "extra" dimensions can

confuse the learning algorithm and cause it to have high variance. Hence, high input dimensionality typically requires tuning the classifier to have low variance and high bias. 4) Noise in the output values. If the desired output values are often incorrect (because of human error or sensor errors), then the learning algorithm should not attempt to find a function that exactly matches the training examples. Attempting to fit the data too carefully leads to overfitting. You can overfit even when there are no measurement errors (stochastic noise) if the function you are trying to learn is too complex for your learning model. In such a situation, the part of the target function that cannot be modeled "corrupts" your training data - this phenomenon has been called deterministic noise. When either type of noise is present, it is better to go with a higher bias, lower variance estimator [7].

The five algorithms that we are going to use are all supervised learning. Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labeled training data consisting of a set of training examples.

The Advantage to use supervised learning is 1) You can get very specific about the definition of the classes, which means that you can train the classifier in a way which has a perfect decision boundary to distinguish different classes accurately. 2) You can specifically determine how many classes you want to have. 3) After training, you don't necessarily need to keep the training examples in a memory. You can keep the decision boundary as a mathematical formula and that would be enough for classifying future inputs.

We have decided to concentrate on breast cancer as our domain of application for multiple reasons. First, breast cancer is one of the main causes of death for women in the US, it represents 25% of all cancers in women [4]. Before using machine learning, "scientists applied different methods, such as screening in early stage, in order to find types of cancer before they cause symptoms. Moreover, they have developed new strategies for the early prediction of cancer treatment outcome. With the advent of new technologies in the field of medicine, large amounts of cancer data have been collected and are available to the medical research community. However, the accurate prediction of a disease outcome is one of the most interesting and challenging tasks for physicians" [5].

Because of this we are very interested in learning the ways in which machine learning can contribute to early diagnostic of diseases such as breast cancer. Second, there are a respectable and trusted number of datasets available to the public for machine learning analysis. In this paper we will be using scikit-learn's dataset based on the UCI ML Breast Cancer Wisconsin (Diagnostic) [6]. Finally, one of us has had close family members diagnosed with breast cancer and gone through treatment to survive the disease. We believe that being educated in a disease that is so common among women worldwide will be of benefit to us and our families now and in the future. In the next section we will begin by giving a quick summary of the each of the supervised learning classification algorithms that we have decided to focus on for this paper.

2. Machine Learning techniques

a. Logistic regression

"Logistic regression is a kind of statistical analysis that is used to predict the outcome of a dependent variable based on prior observations" [16]. In its simplest form, logistic regression uses a logistic function (sigmoid) that returns the probability of a particular sample belonging to a specific class given its features parameterized by the weights [8]. After we predict the probability of an event we can convert it to a binary outcome by using a unit step function [8].

There are many reasons why logistic regression is one of the most popular algorithms in machine learning. One reason is its simplicity, ease of use and effectiveness in classifying linear separable classes [8]. Another reason is that there are times where we are not only interested in the predicted class labels, but where we are also interested in estimating the probability that a sample belongs to a specific class (weather forecasting, medicine, finance) [8].

b. Support Vector Machine

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. For multi class problem, the SVM is also working. We can use two different methods which called one-against-one and 1-against-the rest. The intuition behind the support vector machine approach is that if a classifier is good at the most challenging comparisons (the

two points that are closest to each other), then the classifier will be even better at the easy comparisons (comparing points that are far away from each other). That's why SVM wants to find the maximum margins.

c. Decision tree

A decision tree is a classifier that predicts the label associated with an instance by traveling from a root node of a tree to a leaf node [10]. Each of the non-leaf nodes in a tree specify a test of some attribute of the instance and each branch represent a possible value for this attribute[10]. Decision trees are very popular because the learned trees can be represented and shown as a set of rules than can be easily read and understood by humans trying to interpret the results (Mitchell, 1997).

d. Random forest

Random Forest is an ensemble of unpruned classification or regression trees created by using bootstrap samples of the training data and random feature selection in tree induction. Prediction is made by aggregating (majority vote or averaging) the predictions of the ensemble [12] [13] [14].

The random forest algorithm can be summarized as follow:

"Draw a random bootstrap sample of size n (with replacement).

We then grow a decision tree from the bootstrap sample, and at each node we:

Randomly select d features without replacement.

Split the node using the feature that provides the best split (maximizing information gain).

Repeat the first two steps k times.

Assign the class label by doing a majority vote over all predictions" [8].

e. K-nearest neighbor

k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression [15]. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression:

In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most com-

mon among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

In k-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbors. k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.

The method why it works is very easy to understand, with certain definition of distance, a test point which is more close to k number of points, the majority class of k number of points can be assume as the class of the test point. The only problem is how to define distance. For different data type, the distance can be calculated with related knowledge domain[17].

Part II - Empirical Analysis

1. **Using default parameters**

In the second part of of this paper we will discuss the results or our empirical analysis, most specifically the results of running our five selected algorithms (with its default hyperparameter values) on our breast cancer wisconsin dataset. Before we get into a detail review of our results we will give a quick overview of the characteristics of our chosen dataset.

The scikit-learn dataset is binary classification dataset made up of a total of 569 samples (appendix 1, table 1), it has a dimensionality of 30 features where each and everyone of them is real and positive [6]. The data has the following features: 'mean radius', 'mean texture', 'mean perimeter', 'mean area','mean smoothness', 'mean compactness', 'mean concavity','mean concave points', 'mean symmetry', 'mean fractal dimension','radius error', 'texture error', 'perimeter error', 'area error','smoothness error', 'compactness error', 'concavity error', 'concave points error', 'symmetry error', 'fractal dimension error', 'worst radius', 'worst texture', 'worst perimeter', 'worst area', 'worst smoothness', 'worst compactness', 'worst concavity', 'worst concave points', 'worst symmetry', 'worst fractal dimension'.

Now that we have a general idea of the characteristics of the dataset we will show the results of running our selected algorithms with their default hyperparameter values. We will then compare the default results with each other. In this project we chose to use the scikit-learn implementation of the selected algorithms. We will begin with the logistic regression results.

a. Logistic regression

Selected parameters:	Results:
C=1.0	Accuracy score: 97.6608187134503
	Confusion matrix:
	[[60 3]
	[1 107]]

ROC Curve for logistic regression (appendix 2, figure 1).

b. Support Vector Machine

Selected parameters:	Results:
C=100	Accuracy score: 95.9064327485
kernel='linear'	Confusion matrix:
	[[62 1]
	[6 102]]

ROC Curve for SVM (appendix 3, figure 2).

c. Decision trees

Selected parameters:	Results:
criterion=’entropy’	Accuracy score: 95.32163742690058
max_depth=3	Confusion matrix:
random_state=0	[[61 2]
	[6 102]]

ROC Curve for decision tree (appendix 4, figure 3).

Final decision tree graph (appendix 5, figure 4).

d. Random forest

Selected parameters:	Results:
criterion=’entropy’	Accuracy score: 96.49122807017544
n_estimators=10	Confusion matrix:
random_state=1	[[61 2]
n_jobs=2	[4 104]]

Plotting ROC Curve for random forest (appendix 6, figure 5).

e. K-nearest neighbor

Selected parameters:	Results:
N_neighbors = 5	Accuracy score: 94.7368421053
Weights = ‘uniform’	Confusion matrix:
Algorithm = ‘auto’	[[59 4]
leaf_size=30	[5 103]]
p=2	
metric=’minkowski’	

Plotting ROC Curve for K-nearest neighbor (appendix 7, figure 6).

After running the algorithms with default parameters we can observe the following summary of accuracy scores: logistic regression=97.66, SVM=95.91, decision tree=95.32, random forest=96.49, and K-nearest neighbor=94.74. An initial comparison of the algorithms shows that logistic regression is the most accurate (97.66), followed by random forest (96.49), SVM (95.91), decision tree (95.32), and K-nearest neighbor (94.74) respectively.

Now that we have seen the results using our default selected parameters we will perform techniques to select hyperparameters that could help us improve our accuracy scores. Once again we’ll begin by looking at logistic regression and then we will proceed to see results for other algorithms.

2. Using researched parameters

a. Logistic Regression

In order to select the best possible value for the regularization hyperparameter we utilized scikit-learn’s pipeline and validation curve functions. In its most basic form we ran the logistic regression classifier several times automatically using different values for the regularization parameter C (in our case we used 8 different values for C, from 0.0001 to 1000.0). Our goal during this experiment was to plot a validation curve that would help us find the best value to use out of the eight values used for the hyperparameter C.

The resulting validation curve can be seen below (appendix 8, figure 7). By looking at the curve we can see than when C has a low values the accuracy is acceptable but not as good as it could be. As C increases in value we see an increase in the its accuracy as well, reaching its best accuracy at a value of C=1 while displaying a good trade-off between bias and variance. Interestingly enough it seems that during our de-

fault parameters we luckily picked a good value for the hyperparameter C. A value that thanks to the validation curve we can confirm that it's the best option we have out of the eight values that we tested.

b. Support Vector Machines

Using the same method from logistic regression, I modified my python codes to find the best hyperparameter for SVM. To make our problem easier, I used only the linear model of SVM, the comparing results are summarized in one image (appendix 9, figure 8) So, the C=0.1 is the best for test cases.

c. Decision Tree

When it comes to improving on the accuracy of a decision tree we decided to focus on the depth of the tree as the main hyperparameter. After creating a pipeline that takes advantage of the functionality provided by scikit-learn's GridSearchCV and cross_val_score we were able to execute ten fold cross validation to determine the tree depth that results in the highest accuracy. After processing the data in multiple occasions we were able to obtain a depth=3 as the best max depth for our tree. The best accuracy score that we could obtain was a 95.32 at this depth. Additionally we were able to see cross-validation accuracy scores as high 98.27 and others as low as 87.5. The results of these ten cross validations tests had an average accuracy of 93.5 with a standard deviation of +/- 3.3. The results of the validation curve can be seen below (appendix 10, figure 9).

d. Random Forest

For random forest we decided to focus on the n_estimators parameter (number of trees in the forest). Upon performing multiple test runs we see that as we continue to increase the number of estimators we see an increase in accuracy. One of the most accurate results that we obtained was an accuracy of 97.66 when n_estimators=1000. Although this was a small increase in accuracy for a significant increase in number of estimators, we have to note that testing n_estimators values as high as 5000 resulted in worse results than using 1000.

After looking at a validation curve (appendix 11, figure 10) for the random forest n_estimator we can see that our accuracy is very high on our training data but that if we keep increasing the number of trees we become very prone to overfitting or of having a

model that doesn’t generalize well to new data. However, when we fit the random forest with n_estimators=1000 we get the best accuracy on both the training and validation data. Below we can see the new results obtaining by selecting n_estimators= 1000.

e. K-nearest neighbors

For K-nearest neighbors algorithm, the parameter is only n_neighbors that stands for how many neighbors are you going to choose as representative. According to our validation image (appendix 12, figure 11), 7 neighbors is the best number for our test cases. Although the number 3 best fit training data, the test cases don't completely follow the same result.

Conclusion

We began our analytical part by giving a big picture overview of the Wisconsin breast cancer dataset which has 2 classes, 569 samples, and 30 dimensions with real positive values. We then proceeded to give a quick survey of the five algorithms that we selected: logistic regression, support vector machine (SVM), decision tree, random forest, and k-nearest neighbor. The results of executing each algorithm with selected default parameters can be seen in appendix 16 table 2.

Once we had the initial performances we proceeded to search for the best possible hyperparameters for each algorithm through validation curves, ten fold cross-validation, manual testing, grid search, and pipelines. During our experimentation we discovered parameters that in most cases increased the classification performance of the algorithm. The results of our hyperparameter search can be seen in appendix 17 table 3.

Finally, we compared the results of each algorithm before and after tuning the hyperparameters and ranked each one of them. We determined through different metrics and evaluation techniques that the best algorithm to use for a binary classification problem, such as the one presented by the Wisconsin breast cancer dataset, is logistic regression. However, random forest performed equally as well but its a much more complex algorithm when compared to logistic regression. Of the remaining algorithms their rankings were from best to worst: support vector machine (SVM), decision tree, and k-nearest neighbor.

References

1. Andrew Donkin et al. 1994: Weka: A Machine Learning Workbench. https://researchcommons.waikato.ac.nz/bitstream/handle/10289/1138/uow-cs-wp-1994-09.pdf?sequence=1
2. Kaggle, 2017: The 2017 State of Data Science and Machine Learning. https://www.kaggle.com/surveys/2017
3. David H. Wolpert and William G. Macready, 1997. No Free Lunch Theorems for Optimization. https://ti.arc.nasa.gov/m/profile/dhw/papers/78.pdf
4. Hiba Asri et al., 2016: Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis.The 6th International Symposium on Frontiers in Ambient and Mobile Systems (FAMS 2016) https://ac.els-cdn.com/S1877050916302575/1-s2.0-S1877050916302575-main.pdf?_tid=59b17b64-f0b5-40a4-ad46-bed60da263c2&acdnat=1543698050_8cbe05043674475bf06cd15f49b128bb
5. K. Kourou et al., 2015: Machine learning applications in cancer prognosis and prediction. Computational and Structural Biotechnology Journal. Volume 13, 2015, Pages 8-17 https://reader.elsevier.com/reader/sd/pii/S2001037014000464?token=48D3853580C063804B70E7DC7E9CE8EE0F0CF3F8617239E149638A36C15530E860C52D4E9E1993ABF1CFCE8A7CC2EE0A
6. Scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html
7. Wikipedia: https://en.wikipedia.org/wiki/Supervised_learning
8. Sebastian Raschka, 2016: Python Machine Learning. Packt Publishing Ltd. ISBN 978-1-78355-513-0.
9. Wilson & Lorenz, 2015: J.R. Wilson, K.A. Lorenz, Modeling Binary Correlated Responses using SAS, SPSS and R, ICSA Book Series in Statistics 9, DOI 10.1007/978-3-319-23805-0_2
10. Shalev-Shwartz & Ben-David, 2014: *Understanding Machine Learning: From Theory to Algorithms* by Shai Shalev-Shwartz and Shai Ben-David; Cambridge University Press, 2014.
11. J.R. Quinlan, 1985: Induction of Decision Trees. http://hunch.net/~coms-4771/quinlan.pdf
12. Culberson et al., 2003: Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. Journal of Chemical Information and Com-

puter Sciences 2003 43 (6), 1947-1958. DOI: 10.1021/ci034160g. https://pubs.acs.org/doi/pdf/10.1021/ci034160g

13. Tin Kam Ho, 1995: Random Decision Forests. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282. https://web.archive.org/web/20160417030218/http://ect.bell-labs.com/who/tkh/publications/papers/odt.pdf
14. Breiman, Leo 2001: Breiman, L. Machine Learning (2001) 45: 5. https://doi.org/10.1023/A:1010933404324
15. Wikipedia: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
16. Techopedia: Logistic Regression. https://www.techopedia.com/definition/32065/logistic-regression
17. Scholarpedia: http://www.scholarpedia.org/article/K-nearest_neighbor

Appendix

1. Wisconsin breast cancer dataset overview.

Classes	2
Samples per class	212(M), 357(B)
Samples total	569
Dimensionality	30
Features	real, positive

Table 1 dataset

2. Logistic regression ROC curve results (default and improved).

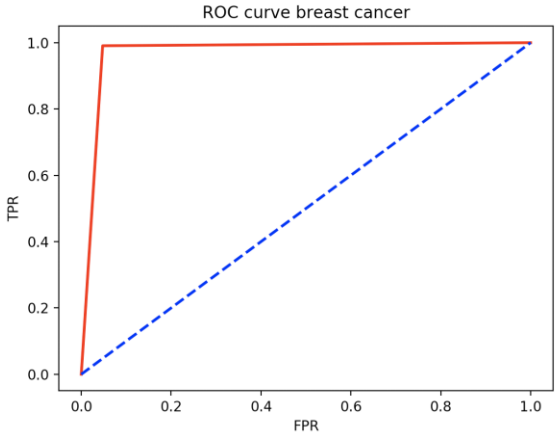

Figure 1 Logistic regression ROC

3. SVM ROC curve result (default and improved).

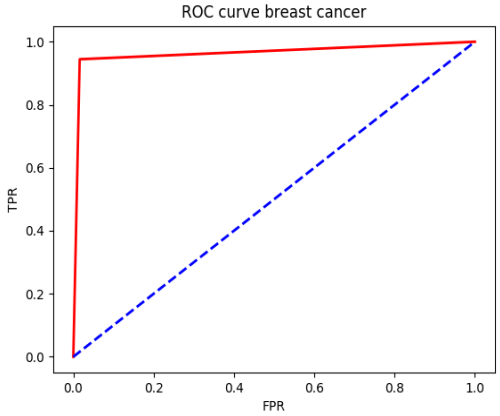

Figure 2 SVM ROC

4. Decision tree ROC curve result.

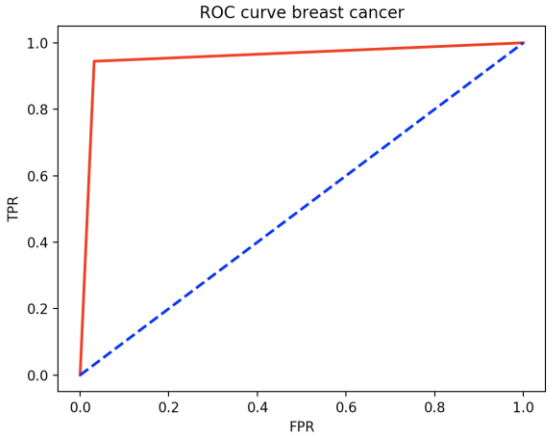

Figure 3 Decision tree ROC

5. Final decision tree after training.

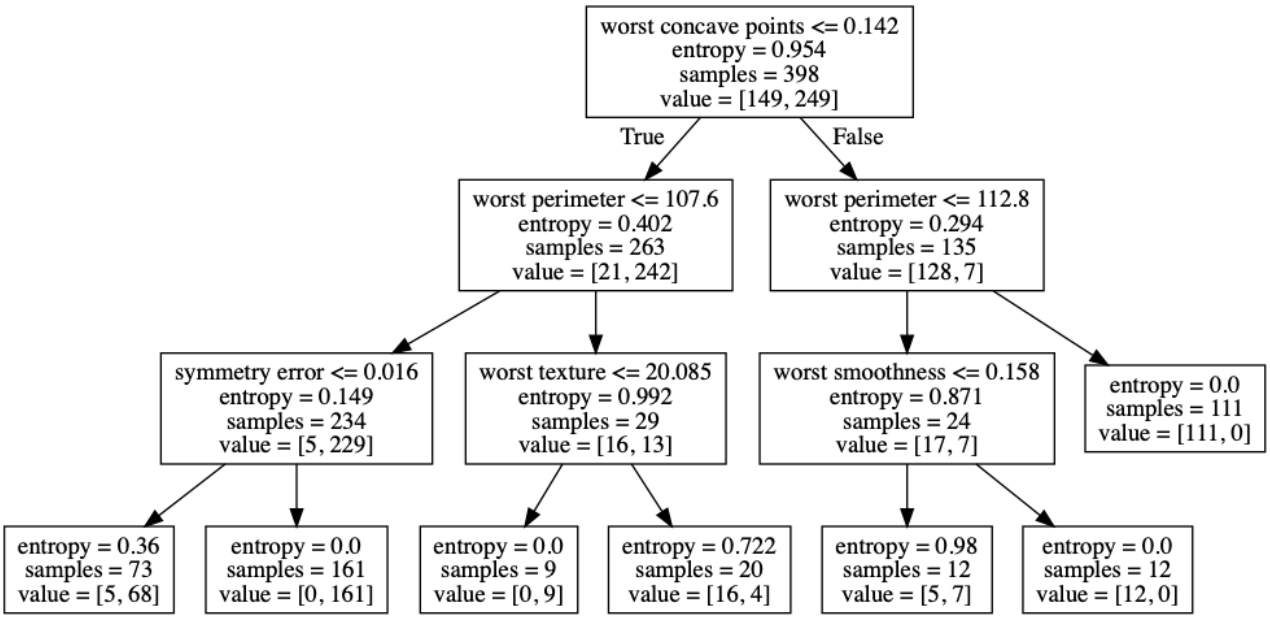

Figure 4 Final decision tree

6. Random Forest ROC curve result.

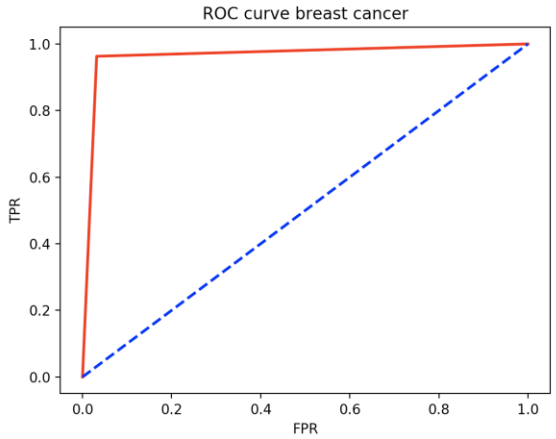

Figure 5 Random Forest ROC

7. K-nearest neighbors ROC curve result.

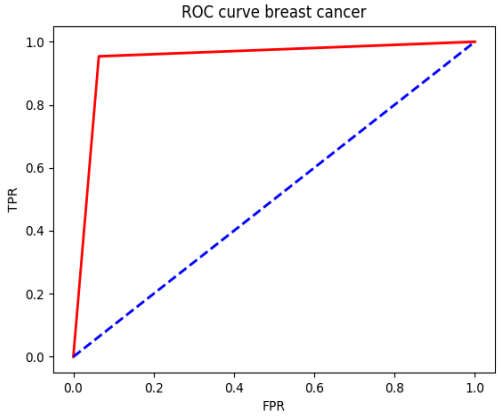

Figure 6 K-nearest neighbors ROC

8. Validation curve to find hyperparameter C for logistic regression.

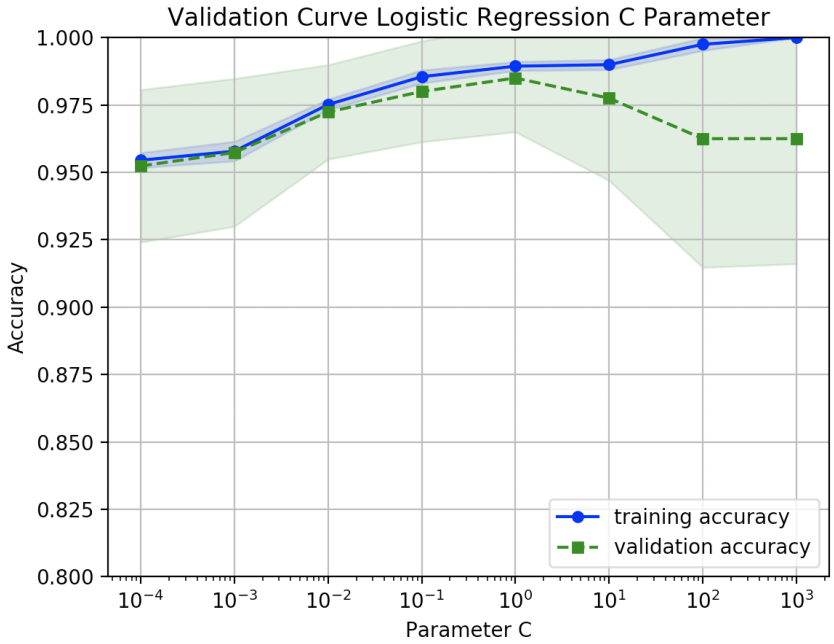

Figure 7 Logistic Regression validation curve

9. Validation curve to find hyperparameter C for SVM in linear model.

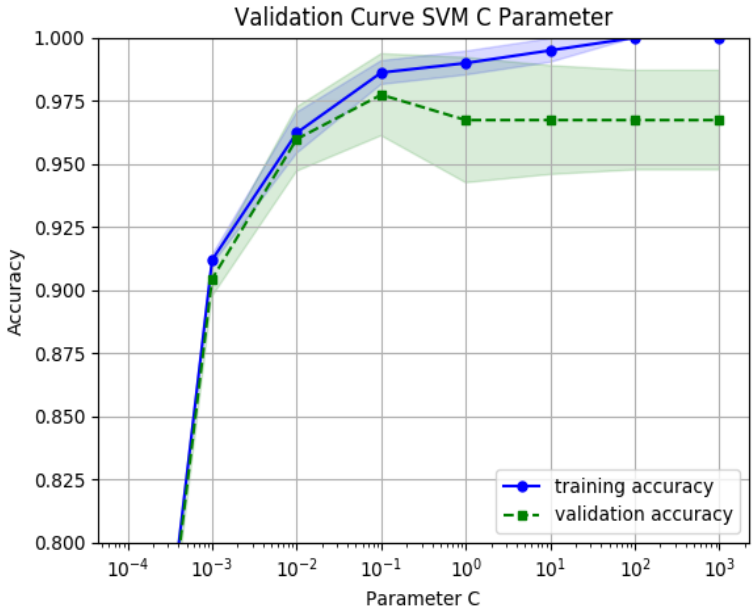

Figure 8 SVM validation curve in linear model

10. Validation curve to find hyperparameter max_depth for Decision Tree classifier.

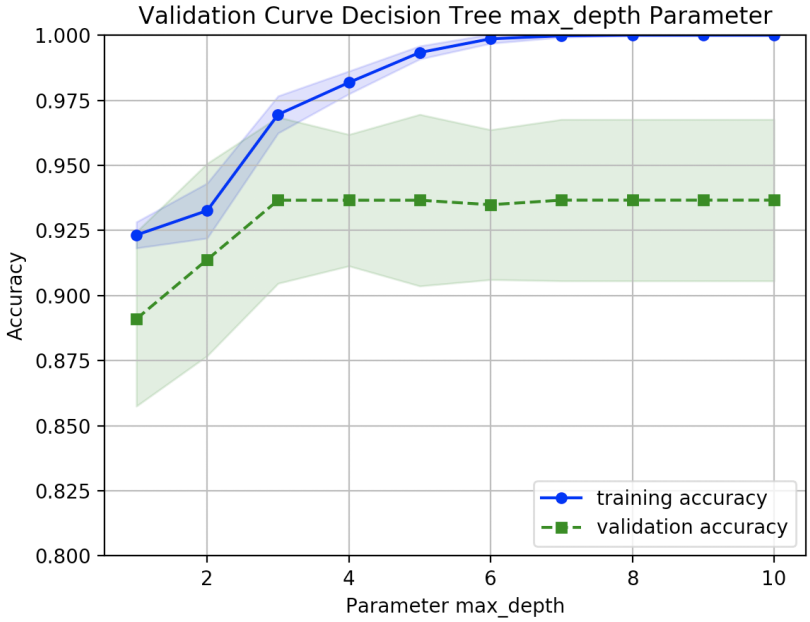

Figure 9 Decision Tree validation curve max_depth parameter

11. Validation curve to find hyperparameter n_estimators for Random Forest classifier

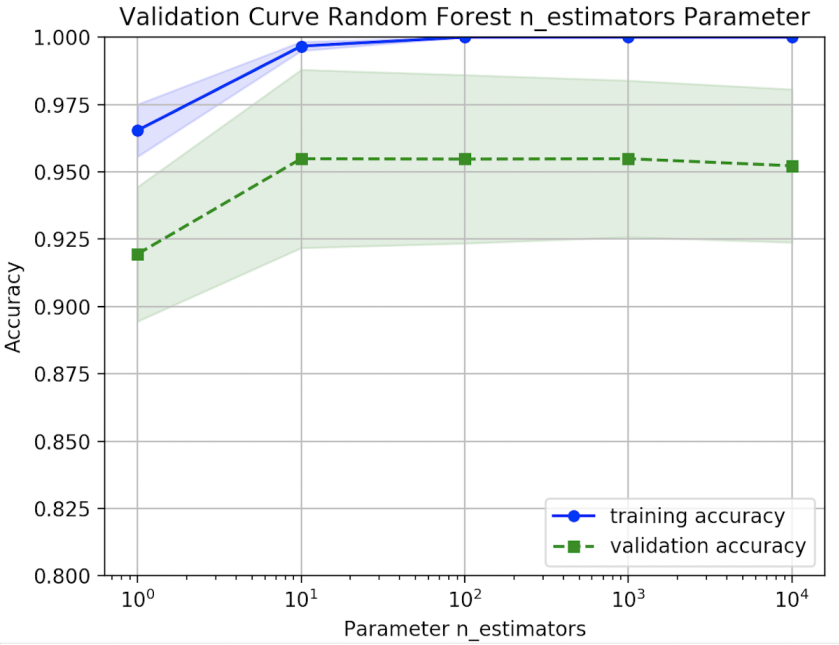

Figure 10 Random Forest validation curve

12. Validation curve to find hyperparameter n_neighbors for K nearest neighbors

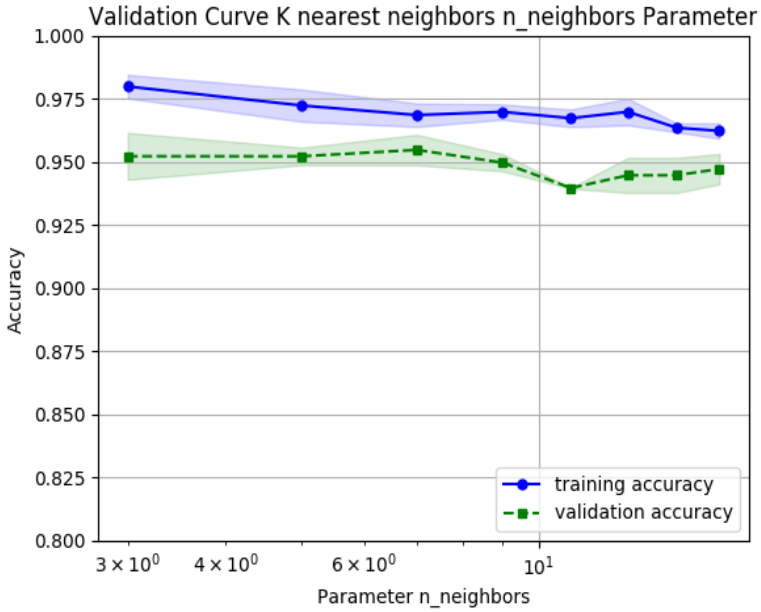

Figure 11 K nearest neighbors validation curve

13. Decision Tree improved ROC curve.

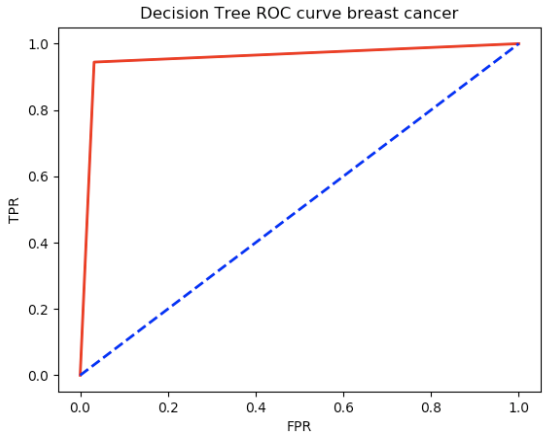

Figure 12 Improved Decision Tree ROC curve

14. Random Forest improved ROC curve.

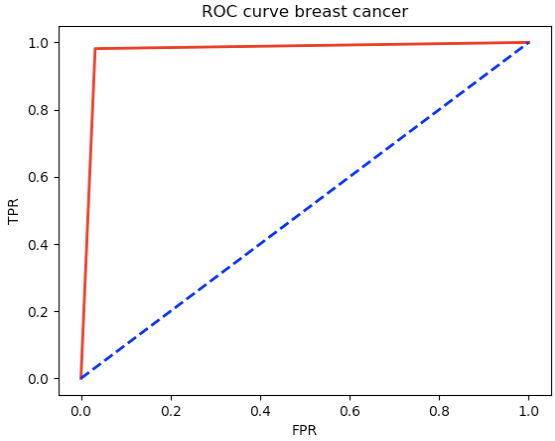

Figure 13 Improved Random Forest ROC curve

15. K-nearest neighbor ROC curve.

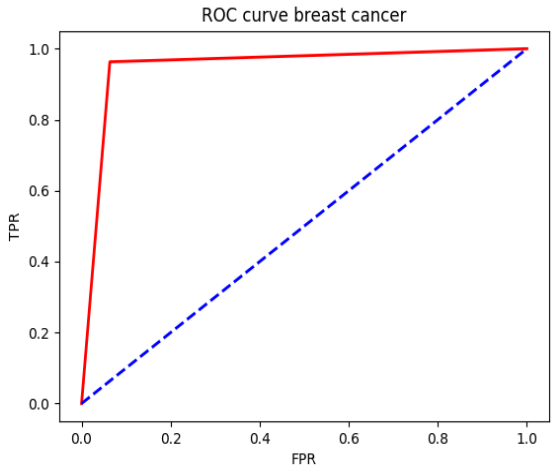

Figure 14 Improved K-nearest neighbor ROC curve

16. Accuracy score, confusion matrix, and ROC curve of each algorithm with default parameters.

Algorithm	**Default parameter**	**Accuracy**	**Confusion matrix**	**ROC Curve**	**Ranking**
Logistic regression	C=1	97.66	[[60 3] [1 107]]	Appendix 2, Figure 1	1
Support Vector Machine	C=100	95.91	[[62 1] [6 102]]	Appendix 3, Figure 2	3
Decision Tree	max_depth=4	95.32	[[61 2] [6 102]]	Appendix 4, Figure 3	4
Random Forest	n_estimators=10	96.49	[[61 2] [4 104]]	Appendix 6, Figure 5	2
K-nearest neighbor	k=5	94.74	[[59 4] [5 103]]	Appendix 7, Figure 6	5

Table 2 Default parameter comparison

17. Accuracy score, confusion matrix, and ROC curve of each algorithm with tuned parameters.

Algorithm	**Tuned parameter**	**Accuracy**	**Confusion matrix**	**ROC Curve**	**Ranking**
Logistic regression	C=1	97.66	[[60 3] [1 107]]	Appendix 2, Figure 1	1
Support Vector Machine	C=1	95.91	[[61 2] [5 103]]	Appendix 3, Figure 2	2
Decision Tree	max_depth=3	95.32	[[61 2] [6 102]]	Appendix 13, Figure 12	3
Random Forest	n_estimators=1000	97.66	[[61 2] [2 106]]	Appendix 14, Figure 13	1
K-nearest neighbor	k=7	95.32	[[59 4] [4 104]]	Appendix 15, Figure 14	3

Table 3 Improved parameter comparison