Applying Different Machine Learning Models to Predict Breast Cancer Risk

Ruolan Xu, *veraxrl@stanford.edu,* and Qiongjia Xu, *leediaxu@gmail.com*

***Abstract*—In this paper, we apply five machine learning models (Logistic Regression, Naive Bayes, LinearSVC, SVM with linear kernel and Random Forest) and three feature selection techniques (PCA, RFE and Heatmap) in one of the key procedures for breast cancer diagnosis. Using the biopsy cytopathology data with 30 numerical features, we achieve a high accuracy of 97.8%. We further compare performances of all models evaluated against various number of features, and examine the reasons behind their varying performances.**

***Keywords*—Breast cancer, Feature selection, Machine learning, Binary classification, SVM, Logistic regression, Random forest, Naive Bayes**

I. Introduction

With an estimated 252,710 new cases each year of invasive breast cancer diagnosed in women in the U.S., breast cancer is by far the most commonly diagnosed cancer among women worldwide. Much attention has been put into analyzing the risk factors pertaining to breast cancer, and what can be done to lower the risk. Besides widely known risk factors like age, family history, weight and lifestyle, medical professionals look into more specific metrics of the breast mass cells to determine the chances of malignant tumor.

Due to varying nature of breast cancers symptoms, patients are commonly subject to a series of tests, including but not limited to mammography, ultrasound and biopsy, to weigh their likelihoods of being diagnosed with breast cancer. Biopsy, which involves extraction of sample cells or tissues for examination, is the most indicative among these procedures.

The sample of cells is obtained from a breast fine needle aspiration (FNA) procedure and sent to a pathology laboratory to be examined under a microscope[1]. Various numerical features, such as radius, texture, perimeter and area, can be measured from microscopic images. Later on, data obtained from FNA are analyzed by physicians in combination with various imaging data to predict probability of the patient having malignant breast cancer tumor.

An automated system would be hugely beneficial in this scenario. It will likely expedite the process and enhance the accuracy of the doctor's predictions. In addition, if supported by abundance dataset and the automated system consistently performs well, it will potentially eliminate the needs for patients to go through various other tests, such as mammography, ultrasound, and MRI, which subject patients to significant amount of pain and radiation.

In this project, the input will be 30 numerical measurements based on the cytopathology features of the cells. Our approach is to apply the five machine learning models separately (Logistic Regress, Naive Bayes, LinearSVC, SVM with linear kernel and Random Forest) with or without feature selections, to understand the advantages and disadvantages of each model, and to select the best subset of features in our effort to generate the best binary classification result of either malignant or benign tumor.

II. Related Work

The breast cancer cytologic dataset was originally part of the study in 1994 "Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates"[2]. In this primary study, 30 numerical features was extracted from digital scan of the FNA sample. A decision tree algorithm named Multisurface Method(MSM) or MSM-Tree was applied for binary classification. More specifically, the MSM-Tree method places a series of separating planes in the feature space, and aims to minimize the number of planes and the number of features used. As a result, it successfully filtered down to three best features (mean texture, worst area, and worst smoothness) and achieved with a 95% confidence level that the true accuracy lies between 95.5% and 98.5%.

Since then, numerous studies has reused this dataset with various machine learning models and algorithms, most likely due to the cleanliness of the data and its large number of features to select from. Most following papers experiment with several machine learning algorithms and examine one or several specific metrics to compare performances. Most popular models include decision trees, neural network, SVM and perceptron, while the most commonly used metrics include area under ROC curve (AUC), overall accuracy and accuracy with confidence level.

As a continuation of these ongoing efforts, this paper will examine several traditional machine learning models and a more popular decision tree model, Random Forest using different metrics.

III. Dataset Features

A. Data

Our dataset is obtained from UCI database and collected from Wisconsin hospital. There are 569 entries in total, with 212 malignant cases and 357 benign cases. Each row contains 30 different features and the diagnosis of breast cancer (0 for benign and 1 for malignant). The 30 features represent the mean, standard deviation and the worst of 10 different cytopathology measurements, including radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry and fractal dimension.

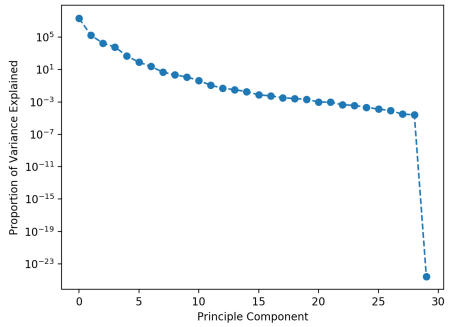
Due to small size of dataset, we only have training set and test set. 569 observations are split to 70% for training and 30% for testing.

B. Feature Selection

1) PCA: Principle Component Analysis (PCA) is a classical technique that rotationally transforms features of a dataset into a lower dimensional set of uncorrelated features called principal components (PCs). PCA is commonly used to reduce the dimension of a dataset, since too many dimensions is wasteful for learning algorithms and might lead to overfitting problems. As PCA tries to find orthogonal projects of the dataset, it makes the strong assumption that some of the variables in our the dataset is linearly correlated.

However, PCA did not end up improving our test results but made our performance worse for all five models. In exploring the reasons behind the bad performance, we plotted out the scree plot for PCA shown in Fig. 1. We can see from the plot that a large number of principle components are needed to explain the variances within the dataset. As expected, the dataset is not as linearly correlated, and thus is not a good candidate for PCA.

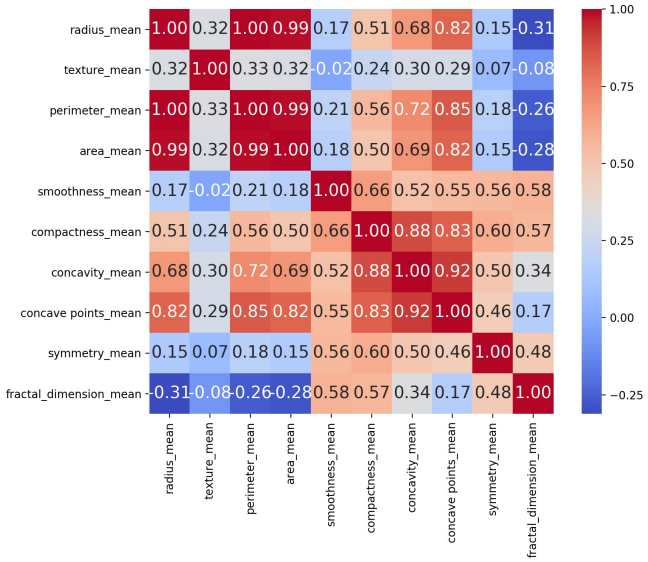
Fig. 1: Scree Plot For Principle Components Analysis(PCA)



2) Recursive Feature Elimination: Recursive Feature Elimination (RFE) starts with the initial set of features, and recursively remove one feature that is the least important until the desired number of features is reached. We run RFE algorithm in sklearn to reduce number of features down to 3, 5, 10, and 30 (original set). The selected features using different models are quite different, but radius and concavity related features appear to be more important as they are selected more often. For example, for logistic regression, [concavity_mean, radius_worst, concavity_worst] are selected if the number is set to 3. RFE is not applied to Naive Bayes because sklearn doesn't support such model.

3) Correlation Heat Map: The correlation matrix for all "mean" features is calculated and Fig.2 shows the correlation heat map for the matrix. A higher correlation index means two features are more closely related, and thus, including one of them in our selected features is enough. From this heat map, we notice that radius, perimeter and area can be grouped together, and concave_points and concavity can be grouped together. One valid feature selection strategy using the correlation heatmap could be [radius_mean, texture_mean, smoothness_mean, compactness_mean, concavity_mean, symmetry_mean, fractal_dimension_mean]. However, using the features selected by correlation heat map performs worse than the original feature set. So we will not include it in the later discussions.

Fig. 2: Correlation Heat Map of Mean Features



IV. Methods

Because the size of our dataset is relatively small, we use bootstrap and bagging technology in our implementation. Using resample in sklearn, roughly 30% of data are selected as testing set and 70% are selected as training set. The following four machine learning models are all implemented using the sklearn library.

A. Logistic Regression

Logistic Regression uses the following logistic function to make predictions:

$$h_\theta(x) = \frac{1}{1 + exp(-\theta^T x)}$$

The above logistic function utilizes sigmoid function, whose output approaches 1 as $z \to \infty$, and approaches 0 as $z \to -\infty$. The output $h_\theta(x)$ ranges between 0 and 1. With a selected threshold, for example, 0.5, the algorithm outputs 1 if $h_\theta(x) > 0.5$, outputs 0 otherwise.

The sklearn logistic regression package also includes L2-penalized regularization and minimizes the following cost function with coordinate descent (CD) algorithm [9].

$$\frac{1}{2}||\theta||_2^2 + \frac{C}{m}\sum_{i=1}^{m} log(1 + exp(-y_i(\theta^T x_i + c)))$$

B. Support Vector Machine

SVM has the underlying hinge loss function:

$$\varphi_{hinge}(z) = [1 - z]_+ = max\{1 - z, 0\}$$

where margin $z = yx\theta^T$. The loss remains zero for all clases that $z > 1$ (meaning y and $x^T\theta$ have the same signs and thus y predicts the right class). SVM is particularly good for linearly separable dataset that logistic regression is susceptible to.

In our analysis, we used both LinearSVC and SVM with Linear Kernel. Both belong to the linear SVM family, while the later uses the "kernel trick" in its implementation.

Kernel is an efficient method to get SVM to learn in high dimensional feature space. In feature mapping, our implementation used linear kernel function below. Linear kernel tends to perform well with large number of features.

$$K(x, z) = < x, x' >= \phi(x)^T\phi(z)$$

To improve the performance, both SVM models have L2-regularization with customized penalty parameter C:

$$\frac{1}{2}||\theta||_2^2 + \frac{C}{m}\sum_{i=1}^{m}[1 - y_i\theta^T x_i]_+$$

C. Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes theorem with the assumption that every pair of features are independent. Gaussian naive bayes model was applied to our dataset, which assumes a Gaussian distribution for the likelihood of the features:

$$P(x_i|y) = \frac{1}{2\pi\sigma_y^2}\ exp(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2})$$

Applying the Bayes theorem, the posterior probability can be expressed as:

$$P(y|x1, x2, ...x_n) = \frac{p(y)p(x1, x2, ...x_n|y)}{p(x1, x2, ...x_n)} = \frac{p(y)\prod_{i=1}^{n} p(x_i y)}{p(x1, x2, ...xn)}$$

To maximize the above probability, we need to find parameters that maximize $p(y)\prod_{i=1}^{n} p(x_i y)$. The maximum likelihood estimation of the parameters follows the two equations below. After fitting, we make predictions by calculating the posterior probability of each class and choose the class with highest probability.

$$\phi_{j|y=k} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = k\}}{\sum_{i=1}^{m} 1\{y^{(i)} = k\}}\ where\ k = 0, 1$$

$$\phi_y = \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}}{m}$$

D. Random Forest

Random Forest constructs multiple decision trees based on the subset of training examples and the subset of all given features at random. In each decision tree, the input enters at the root of the tree and traverses down the tree according to the split decision at each node. Along the way, data gets bucketed into smaller and smaller sets. In this research, we use Gini impurity as the split function that evaluates the quality of a split:

$$Gini(E) = 1 - \sum_{j=1}^{J} p_j^2$$

Note: J represents all possible labels (0 and 1 in our case), and p_j represents the possibility of labeled as j. Gini impurity measures how often a randomly chosen example would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

After the forest is trained, when a new input enters the system, it will run down all the trees and reach the leaf nodes of each tree. The final output is decided by the majority votes of the leaf node.

V. Results and Discussion

After fitting the above models with our training data and testing the performances with our testing data, we have generated the following results. We use accuracy, error rate ($1 - accuracy$), confidence interval, precision, recall, specificity and F1 score to evaluate the model's performance.

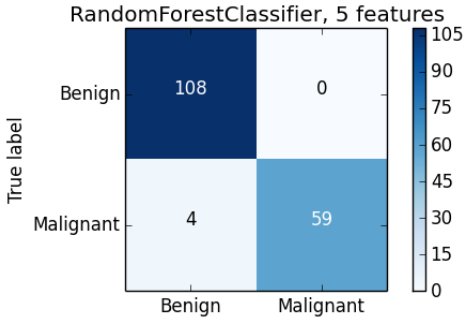
Among these evaluating metrics, accuracy, precision, recall, specificity and F1 score can be obtained from confusion matrices as shown in Fig. 3 below, and were used to evaluate different aspects of the performances. F1 score, the harmonic mean of precision and recall, can be interpreted as a weighted average of the two.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$Precision = \frac{TP}{TP + FP},\ \ Recall = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP},\ \ F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Fig. 3: An Example of Confusion Matrix



We first run five different models with all 30 available features using bootstrap strategy. Each model is run 100 times

with random samples at each run. In the end, the average error rate is computed. As shown in the tables below, Logistic Regression, Random Forest Classifier and SVM with linear kernel all perform relatively well. If we compare the difference between train and test error, Random Forest Classifier has a huge gap of 5%, which means that the model overfits the training set.

TABLE I: Train and Test Error for Different Models

Model	Train Error %	Test Error %
Logistic Regression	3.6	5.1
LinearSVC	9.4	10.2
Random Forest Classifier	0.2	5.2
Naive Bayes	5.5	6.2
SVM with linear kernel	2.7	4.9

Surprisingly, LinearSVC performs quite bad, while Naive Bayes performs relatively good. According to sklearn documentation, although LinearSVC and SVM with linear kernel both belong to the SVM family, they are implemented using two different libraries. As a result, LinearSVC is more suitable for larger dataset with smaller feature set, while SVM with linear kernel works better for smaller data set (the time complexity is high though). In our case, because the dataset size is considerably small and the feature set is large, LinearSVC doesn't fit our system well, even for the training set. Similarly, the unexpectedly good performance Naive Bayes could be due to the size of our dataset. Generally, we don't expect Naive Bayes to perform well when there are strong correlations between the features. According to our heat map, many features are strongly related. Thus, we are surprised to see the good error rate for Naive Bayes.

TABLE II: Confidence Interval for Test Accuracy

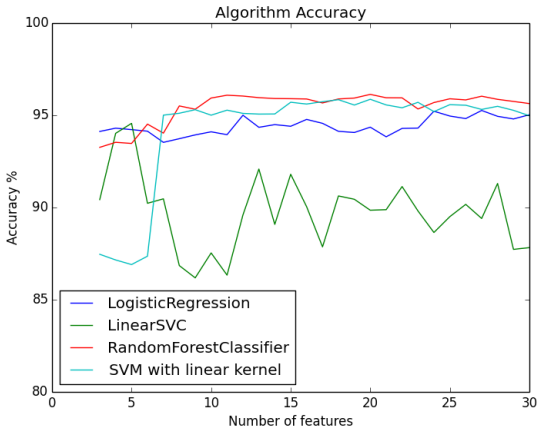
Model	Accuracy 95% Confidence Interval
Logistic Regression	92.3 - 96.5%
LinearSVC	73.5 - 94.0%
Random Forest Classifier	92.3 - 96.8%
Naive Bayes	91.6 - 95.8%
SVM with linear kernel	93.0 - 97.1%

Table I only shows the average error rate ($1 - accuracy$), while Table II shows the range where the accuracy of 95% iterations fall into. This gives a rough idea of how stable the model is. The result further proves that Logistic Regression, Random Forest Classifier and SVM with linear kernel are comparatively better models. LinearSVC is very unstable.

We also evaluate the effect of RFE feature selection on different models. We run RFE to select 3 to 30 features, and plot the accuracy of test set in Fig. 4.

There are some interesting observations we can see from this figure. First of all, we can see that Random Forest Classifier and SVM with linear kernel generally do slightly better then Logistic Regression, but all three of them should be considered as good model.

Fig. 4: Accuracy for Different Number of Features

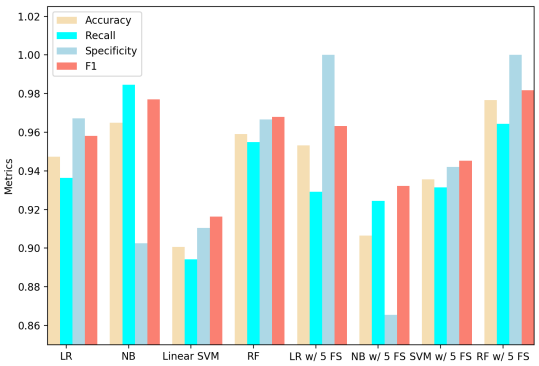


The peak of accuracy for different models appears at different number of features. For Random Forest Classifier, the accuracy is the highest at around 10 features, and it is around 16 for SVM with linear kernel. The accuracy for both models drops after the feature size is smaller than 7, especially for SVM with linear kernel. This shows that feature selection successfully reduce overfitting problem slightly for Random Forest Classifier and SVM with linear kernel. However, if the feature size gets too small, the accuracy will be hurt. For logistic regression, the accuracy is slightly going down as the number of features decreases, which means the model doesn't suffer from overfitting. Quite abnormal for LinearSVC, the accuracy actually rockets to almost the same as other models when the feature size is around 5. This proves the statement raised previously that LinearSVC works better for relatively larger dataset and smaller feature set. When the ratio of feature size versus dataset size becomes small, the performance improves. However, when the feature size is too small, like 3, the model is not good anymore due to insufficient features (underfitting).

After running different models with feature selection, we calculated different metrics based on the formulas for Logistic Regression, Naive Bayes, Linear SVM w/o Kernel and Random Forest. A comparison of these evaluation metrics before and after feature selections can be seen in Fig.5. All models show obvious improvements in all aspects of performance except Naive Bayes, which makes sense because Naive Bayes relies on the probability model where the joint likelihood can be represented as product of observation likelihoods. Smaller number of features will likely hurt its performance. It's also possible that strongly correlated features are selected in feature selection which worsens the performance of Naive Bayes.

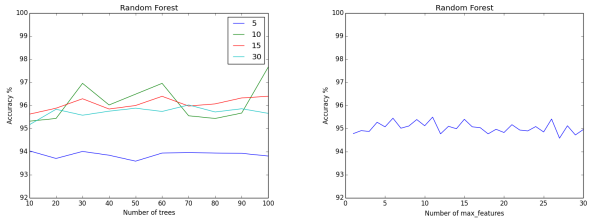
As discussed in the previous section, Random Forest tends to overfit the training set with almost zero training error and

Fig. 5: Evaluation Metrics Before and After Feature Selections



5% testing error. To reduce overfitting, there are three possible ways: feature selection, tuning the number of trees in each forest, and tuning the max number of features in each tree.

Fig. 6: Tuning Random Forest Classifier



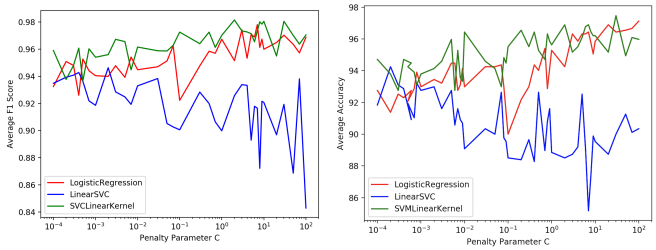
(a) Different number of trees for different number of features

(b) Different number of max_features for 30 features (entire feature set)

We can see from Fig. 6(a) that feature selection does have positive effect on accuracy, but the remaining feature set cannot be too small. If we reduce the feature set size to 5, the accuracy decreases obviously. Although the model isn't stable, we can still see a increasing trend of accuracy when the number of trees increases. When we have 100 trees, the accuracy almost reaches 98%. However, changing the maximum number of features in each tree doesn't seem to affect the accuracy a lot. There's a slightly decreasing trend when we increase the number of features. In general, to reduce overfitting for Random Forest Classifier, it is better to perform feature selection with the proper size of feature set, increase the number of trees in each forest, and decrease the maximum number of features in each tree.

Different penalty parameters, the coefficient of the regularization term, are applied to Logistic Regression, LinearSVC and SVM with linear kernel with 20 features to experiment the effect of regularization term on model performance. Despite a certain level of fluctuation, the overall trend is pretty obvious from Fig. 7 (a) and (b) that as we increase penalty parameter,

Fig. 7: Penalty Parameter for Regularization Term



(a) Penalty Parameter vs. F1 Score

(b) Penalty Parameter vs. Accuracy

the F1 score and accuracy of Logistic Regression and SVM with linear kernel improve, because higher penalty parameter decreases overfitting for them, while the performance of LinearSVC decreases, meaning that LinearSVC doesn't subject to overfitting, and the model itself doesn't fit well with our dataset.

VI. Conclusion & Future Work

In conclusion, Random Forest Classifier and SVM with linear kernel yield better prediction results than other models. These two models work better for small dataset. In particular, for Random Forest Classifier, if we have around 10 features selected, and use more trees, and less features in each tree to train the model, we can reduce overfitting and produce better accuracy. The highest accuracy we can get from Random Forest Classifier is about 98%. For SVM with linear kernel, with higher penalty parameter for the regularization term, the accuracy can reach 97% as well.

For future work, although we achieve relatively accurate prediction using several models, we would like to make sure the result is not biased due to the size of out dataset. We would like to find a bigger dataset and perform similar analysis and see if the results are the same. Futhermore, since our dataset is quite outdated (collected in the 90s), measurement of cytopathology data for breast cancer might be different nowadays. It would also be better if we can find some latest data and do the analysis. In addition, besides the above models we have tried, we would also like to try deep learning to train the data. We realize that neuron network with the right activation function might work well in our case, because we have a lot of correlated features.

VII. Contribution

Qiongjia Xu: Implemented RFE with different models, Random Forest comparison.

Ruolan Xu: Implemented PCA, metrics evaluation and regularization penalty comparison.

Both: Poster presentation, report writeup.

REFERENCES

[1] Fine Needle Aspiration Biopsy of the Breast. American Cancer Society, www.cancer.org/cancer/breast-cancer/screening-tests-and-early-detection/breast-biopsy/fine-needle-aspiration-biopsy-of-the-breast.html.

[2] Wolberg, William H., W. Nick Street, and O. L. Mangasarian. "Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates." Cancer letters 77.2-3 (1994): 163-171.

[3] Jele, ukasz, Thomas Fevens, and Adam Krzyak. "Classification of breast cancer malignancy using cytological images of fine needle aspiration biopsies." International Journal of Applied Mathematics and Computer Science 18.1 (2008): 75-83.

[4] Bradley, Andrew P. "The use of the area under the ROC curve in the evaluation of machine learning algorithms." Pattern recognition 30.7 (1997): 1145-1159.

[5] Benyamin, Dan. A Gentle Introduction to Random Forests, Ensembles, and Performance Metrics in a Commercial System. CitizenNet Blog, 9 Nov. 2012, blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics.

[6] Polamuri , Saimadhu . How the random forest algorithm works in machine learning. Dataaspirant, 2 May 2017, dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/.

[7] D. Cournapeau, et al., scikits.learn: machine learning in Python, http://scikit-learn.sourceforge.net

[8] Wolberg, WIlliam H. Breast Cancer Wisconsin (Original) Data Set. UCI Machine Learning Repository: Breast Cancer Wisconsin (Original) Data Set, 15 July 1994

[9] Friedman, Jerome, Trevor Hastie, and Rob Tibshirani. "Regularization paths for generalized linear models via coordinate descent." Journal of statistical software 33.1 (2010): 1.