Using Machine Learning for Breast Cancer Prediction

NURAN GOLBASI, nuran@live.unc.edu
EMILY TROUTMAN, emtrout@live.unc.edu
ROSHNI PASUPULA, roshnip@live.unc.edu

This report was created in association with the Intro to Machine Learning (COMP 562) course under Professor Jorge Silva of the University of North Carolina at Chapel Hill. Through the course of this paper, we applied several different machine learning models to classify our data set into malignant and benign diagnoses. We discovered that the SVM and Random Forest models classified the sample with the highest accuracy; the SVM Algorithm produced an accuracy of 96% while the Random Forest model produced an accuracy of 97%.

ACM Reference Format:
Nuran Golbasi, Emily Troutman, and Roshni Pasupula. 2020. Using Machine Learning for Breast Cancer Prediction. xx, x, Article x (November 2020), 5 pages. https://doi.org/http://dx.doi.org/10.1145/8888888.7777777

1 DATASET EXPLANATINON

For this final project, we have chosen a dataset that documents several features from a digitized image of a fine needle aspirate (FNA) of a breast mass in order to classify whether the image can be diagnosed as benign or malignant. Machine learning has been used successfully in fields such as Radiology to help with computer-aided detection and diagnosis for cases like pneumonia and lung cancer in CT and/or MRI images. We have elected to extend this capability in order to attempt to classify whether digitized images are indicative of breast cancer.

This dataset was found on Kaggle as "Breast Cancer Wisconsin (Diagnostic) Data Set." It is a CSV that contains 569 samples with 32 features each.

The following description is adapted from the description provided by UCI Machine Learning for the dataset on Kaggle [Learning 2016].

1.1 Description of Features

The first two features represent the following:

(1) ID number
(2) Diagnosis (M = malignant, B = benign)

1.1.1 3 through 32 are derived from the following procedure. Ten real-valued features are computed for each cell nucleus:

(1) radius (mean of distances from center to points on the perimeter)
(2) texture (standard deviation of gray-scale values)
(3) perimeter
(4) area
(5) smoothness (local variation in radius lengths)
(6) compactness ($\frac{perimeter^2}{area} - 1.0$)
(7) concavity (severity of concave portions of the contour)
(8) concave points (number of concave portions of the contour)
(9) symmetry
(10) fractal dimension ("coastline approximation" - 1)

"The mean, standard error and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius. All feature values are re-coded with four significant digits."

Missing attribute values: none

Class distribution: 357 benign, 212 malignant

2 APPLICATION AND MOTIVATION

2.1 Early Intervention In Cancer

The longer a cancer has been spreading inside a patient, the more difficult it is to treat. That is why early detection is vital in the treatment of cancers. A computer with a well trained model can process exponentially more patient files than the largest team of doctors and in a fraction of the time. Machine learning methods enable large-scale screening that would otherwise be infeasible for most hospitals. By applying our model to an existing patient database, high-risk individuals can be identified quickly and accurately, and any breast cancer can be identified sooner.

2.2 Decreasing Physician Burnout

In medical facilities, patient data is recorded and stored in electronic databases. Machine Learning models can then be easily applied to minimize the burden on the doctors and nurses. By having this model automatically classify patient scans as high/low probability of being malignant tumors, doctors can quickly perform further testing on the patients that are considered "likely" to have cancerous growths, and can do so well before a radiological analysis can be completed. With over 42% of physicians complaining of burnout in a 2020 survey, this can be one of the many comprehensive steps taken to minimize this prevalent issue.

2.3 Improving Cancer Research

Machine Learning could provide means to detect patterns within collected medical and patient information that the human eye may easily miss. Some data sets may have extremely large amounts of features, which could all be used to identify cancerous cells. Gathering as much information as possible could help to uncover correlations between features and indications of cancer. However, in

Authors' addresses: Nuran Golbasinuran@live.unc.edu; Emily Troutmanemtrout@live.unc.edu; Roshni Pasupularoshnip@live.unc.edu.

XXXX-XXXX/2020/11-ARTx $15.00
https://doi.org/http://dx.doi.org/10.1145/8888888.7777777

data sets with numerous features, the breadth of these features may be unmanageable or at the very least severely difficult for medical staff to manage. Using machine learning could help to remedy these issues and uncover new correlations between collected data and cancer classification.

3 RELATED WORK

The application of machine learning in medical diagnosis is a growing field. In 2012, G. Parthiban and S.K.Srivatsa investigated the potential for machine learning methods in diagnosing heart disease for diabetic patients [Parthiban et al. 2011]. In 2015, an article published by Kourou et. al. explored the potential of machine learning in cancer prognosis and prediction[Kourou et al. 2015]. A study in 2017 by Lahmiri, Dawson, and Shmuel explored the performance of machine learning models in diagnosing Parkinson's disease using measures of dysphonia [Lahmiri et al. 2018]. Another study in 2018 by Abdulhaya et. al. looked into applying machine learning to Parkinson's diagnosis, but based on gait measures [Abdulhay et al. 2018].

4 APPROACH

Before we attempted to train this data, we first did our best to clean and optimize the data to keep features that provided meaningful input and omit those that were not likely to contribute relevant values. Just by looking at our dataset, we observed that the very last column titled "Unnamed" consisted only of "NaN" values, and that the first column "id" contained different numerical identification numbers for each entry. Because neither of these columns provided meaningful data for us to test, we promptly removed them from our data set.

Next, we inspected the remaining features for potential extraneous values that could cause skewed data, and kept an eye out for collective groups of outliers by using the pandas .describe() function which generated the mean, standard variation, minimum and maximum values for each feature as well as visualizations such as histograms.

Using these methods, we did find a few outliers; however, we made the executive decision to keep them just in case an outlier could indicate a Malignant tumor. As we also had a fairly small data set with only 569 entries, we wished to keep as many data points as possible to produce better training and test data results.

After inspecting the data, we divided the samples into testing and training sections so that we can get a sense of how accurate the model would be in a practical setting.

We chose to test and train this data set with cross validation and K-folds using Logistic Regression, SVM, Naive Bayes for Gaussian and a new algorithm called Random Forest in order to determine which of these machine learning methods would be able to classify the digital images with the best accuracy.

Once we trained our models, we selected the two best performing models and validated their accuracy in making predictions using their test data. This was an important step in producing a more concrete estimate of the accuracy, since the test data was unseen and independent of the data used to train the model.

4.1 Logistic Regression

Logistic Regression is a classification model that can be used when the data falls into exactly two classes. For this set, the classes are Benign = 0 and Malignant = 1. It attempts to create a logistic model to describe the data.

It uses the following probability function:

$$p(y|x, \beta_0, \beta) = \frac{exp\{y(\beta_0 + x^T\beta)\}}{1 + exp\{\beta_0 + x^T\beta\}} \tag{1}$$

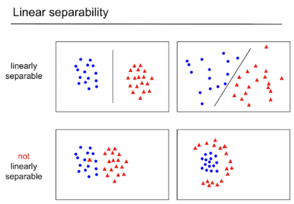
From which it derives the following log likelihood function:

$$\mathcal{LL}(\beta_0, \beta|y, x) = \sum_i y_i(\beta_0 + {x_i}^T\beta) - \log\{1 + exp\{\beta_0 + {x_i}^T\beta\}\} \tag{2}$$

Which will be maximized. Then the weights vector β that resulted in this max value will be saved as the logistic model.

4.2 SVM Algorithm

SVM, otherwise known as Support Vector Machines, can be used for binary classification if the dataset given is linearly separable. We have included a slide from lecture below to help demonstrate whether the given dataset is linearly separable dataset versus not linearly separable:



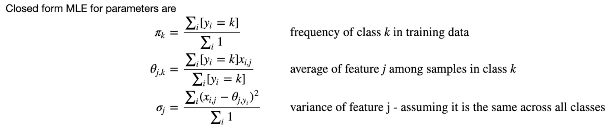
The objective of the Support Vector Machines is to find the optimal hyperplane or a decision boundary that separates the points, thereby classifying them in this case into two distinct classes. There are many possible hyperplanes that could be chosen, but the optimal hyperplane would be the one with the largest margin between the two different data point classes. As the name suggests, we can use Support Vectors in order to maximize this distance.

4.3 Naive Bayes for Gaussian

Of the generative models for classification, we chose to classify the data using Naive Bayes for Gaussian. The Naive Bayes model itself has many forms, including Gaussian, Bernoulli, Binomial, Categorical and Multinomial, but in this case, since our dataset sample contained more than two features that would be needed to classify it, we decided to use its Gaussian form.

Naive Bayes itself is called "Naive" because it simplifies the calculations of the probabilities. Instead of explicitly calculating the values of each feature, it is assumed that they are conditionally independent from each other.

For the Gaussian Naive Bayes model, the parameters required for learning the model include the frequency of each class in the training data, the average value for each feature across the data set sample in each class, and the variance of each feature across the data set. The Gaussian Naive Bayes model assumes that the value of the variance for features is the same across all classes, which is



Closed form MLE for parameters are

$$\pi_k = \frac{\sum_i [y_i = k]}{\sum_i 1} \qquad \text{frequency of class } k \text{ in training data}$$

$$\theta_{j,k} = \frac{\sum_i [y_i = k] x_{i,j}}{\sum_i [y_i = k]} \qquad \text{average of feature } j \text{ among samples in class } k$$

$$\sigma_j = \frac{\sum_i (x_{i,j} - \theta_{j,y_i})^2}{\sum_i 1} \qquad \text{variance of feature j - assuming it is the same across all classes}$$

where it earns its “Naive” nickname. Using these three parameters, we can find the predicted class by using this equation:

$$y^* = \underset{k}{\operatorname{argmax}} \log \pi_k - \underbrace{\sum_j (x_{j,i} - \theta_{j,k})^2}_{\text{distance to class center}} + \text{const.}$$

4.4 K-fold Cross Validation

Cross Validation is a statistical method used to approximate how accurate a Machine Learning model is when applied to a particular dataset. Using k-fold Cross Validation helps to provide a less biased or skewed estimate of how accurate a machine learning model is. Following the process of k-fold Cross Validation, the dataset is divided into a specific number of “folds” or groups. The number of groups varies depending on the dataset and is commonly indicated by the letter k which is where the name k-folds comes from.

Once the k parameter is defined, the data sample is promptly broken down into the number of groups defined by k’s value. For example, if k = 10, then the model would use 10-fold cross validation. In most cases, the value of k is typically either 5 or 10, but there are cases where the value of k is assigned to the constant number n, which is chosen specifically for the data set.

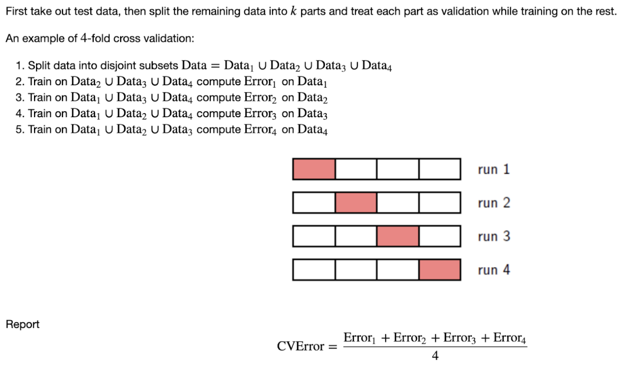
Choosing the correct k value is important for datasets. The larger that the k value becomes, the less bias, but the greater the variance. K-values of 5 and 10 have been proven to have a good balance between risking a high bias or a high variance.

The process of k-fold cross validation is as follows. First, the data sample is reshuffled from its original order into a new, random order. Next, the reshuffled data set is split as mentioned earlier into the number of groups indicated by the k value. Out of these k groups or “folds” the first is kept separate from the rest and is used later for validation. The remaining groups are fitted with the machine learning model. After this first round of training, the second group is kept separate to be used as the validation for the rest of the groups, which are fitted with the model. This process occurs until all groups have at some point been used to validate the rest.

The Stratified K Fold is a commonly used variation of the k fold cross validation method that ensures each fold has the same proportion, and is what we used for our dataset.

4.5 Random Forest

The Random Forest is a supervised learning algorithm that creates decision trees on random samplings of the data in order to select the best solution. It primarily utilizes multiple decision trees or series of true/false questions that evaluate our data and lead to a prediction. For each of these questions, which are represented as nodes, only a subset of features are selected. Once the model constructs a decision tree for the random samples, it performs a vote on the predicted results. The prediction result with the most votes is the final output.



First take out test data, then split the remaining data into k parts and treat each part as validation while training on the rest.

An example of 4-fold cross validation:

1. Split data into disjoint subsets $\text{Data} = \text{Data}_1 \cup \text{Data}_2 \cup \text{Data}_3 \cup \text{Data}_4$
2. Train on $\text{Data}_2 \cup \text{Data}_3 \cup \text{Data}_4$ compute Error_1 on Data_1
3. Train on $\text{Data}_1 \cup \text{Data}_3 \cup \text{Data}_4$ compute Error_2 on Data_2
4. Train on $\text{Data}_1 \cup \text{Data}_2 \cup \text{Data}_4$ compute Error_3 on Data_3
5. Train on $\text{Data}_1 \cup \text{Data}_2 \cup \text{Data}_3$ compute Error_4 on Data_4

Report

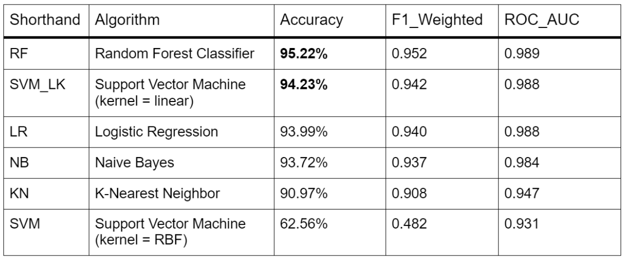
$$\text{CVError} = \frac{\text{Error}_1 + \text{Error}_2 + \text{Error}_3 + \text{Error}_4}{4}$$

5 RESULTS AND DISCUSSION

In order to compare our models, we used the following measures as defined below.

- Accuracy: the percentage of correctly predicted data points
- F-score: the harmonic mean of the precision and recall of the data set
- ROC_AUC_score: the area under the receiving operating characteristic (ROC) curve

The results of our analysis can be found in Table 1, in order of accuracy. We trained our original SVM model with different implementations and got very different results. First, we trained the SVM model using the default kernel value “RBF” as given by the example code for the model, which gave us an accuracy of 62.56%. We noticed that this was particularly low in contrast to the rest of the models. In an attempt to improve this result, we trained the SVM model with the kernel set to “linear”, which resulted in the model’s accuracy to increase to 94.23%. We attributed this difference to nonoptimal parameters.



Shorthand	Algorithm	Accuracy	F1_Weighted	ROC_AUC
RF	Random Forest Classifier	**95.22%**	0.952	0.989
SVM_LK	Support Vector Machine (kernel = linear)	**94.23%**	0.942	0.988
LR	Logistic Regression	93.99%	0.940	0.988
NB	Naive Bayes	93.72%	0.937	0.984
KN	K-Nearest Neighbor	90.97%	0.908	0.947
SVM	Support Vector Machine (kernel = RBF)	62.56%	0.482	0.931

Fig. 1. Model Comparison

After we improved upon our SVM model, the two most accurate algorithms from the cross validation was the Random Forest Classifier and the Support Vector Machine with a Linear Kernel. Since the Random Forest Classifier model does not suffer from overfitting the way Support Vector machines can, we predicted that the RF could

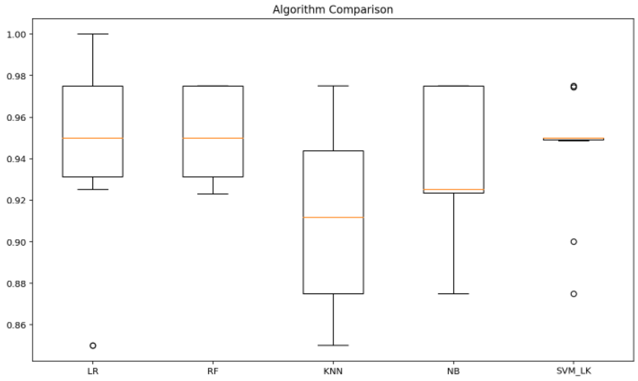


Fig. 2. Box Plots

be the better model. To further validate these models, we used the independent test data to predict tumor outcomes, and evaluated their accuracy. The results were very close; SVM2 had an accuracy of 96.491% while Random Forest had an accuracy of 97.076%. As such, we concluded that the Random Forest Classifier was the most accurate model.

Below is the ROC curve and confusion matrix of the Random Forest Classifier:

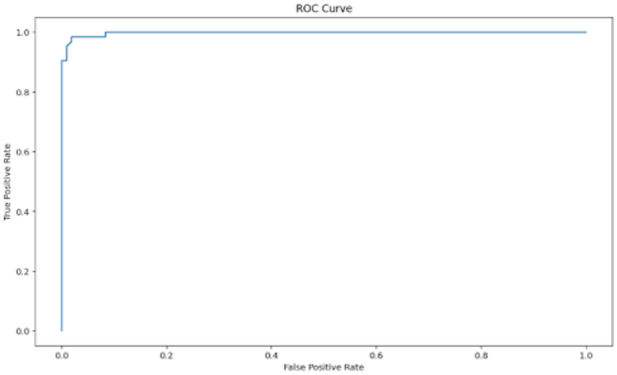
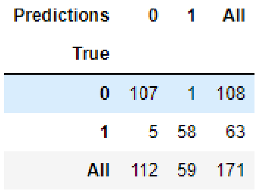


Fig. 3. ROC Curve



Predictions	0	1	All
True			
0	107	1	108
1	5	58	63
All	112	59	171

Fig. 4. Confusion Matrix

The receiver operating curve of our model validates its accuracy, since the area under the curve is very close to 1. This is further supported by the confusion matrix. There are very few false positives (1) and false negatives (4), indicating our model should be fairly reliable. Minimizing the number of false negatives is particularly important when attempting to classify cancerous cells, because each false negative would be a missed diagnosis of a malignant tumor. Even a number as low as four would mean four patients' tumors were misclassified as harmless. Therefore, validation is necessary; however, with such high accuracies and low false positive and false negative values, this would be a useful tool for physicians and doctors to support their diagnoses.

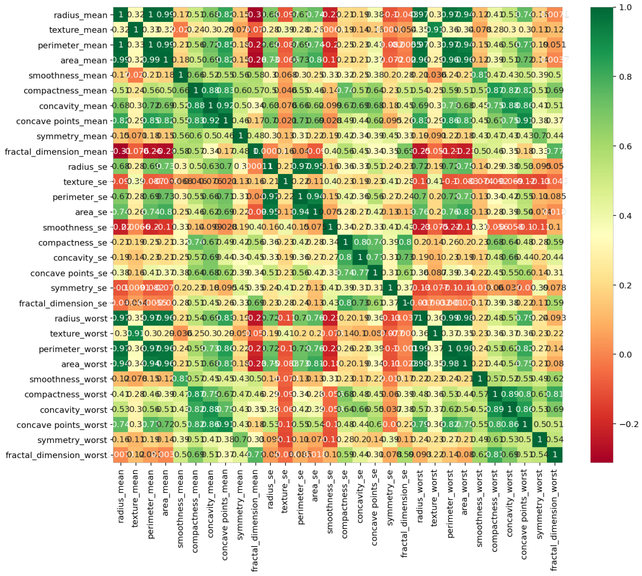
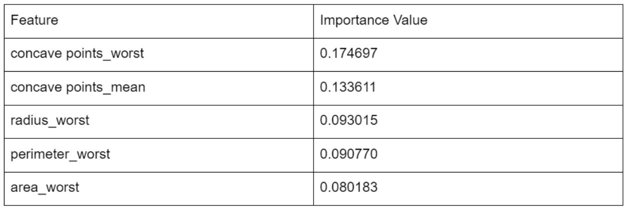


Fig. 5. Heat Map



Feature	Importance Value
concave points_worst	0.174697
concave points_mean	0.133611
radius_worst	0.093015
perimeter_worst	0.090770
area_worst	0.080183

Fig. 6. Feature Importance

One great aspect of the Random Forest Classifier is that it can provide us with feature importance, thereby giving us more insight into the dataset. Below is a table of the top values of importance for the first five features: concave points_worst, concave points_mean, radius_worst, perimeter_worst, and area_worst. Since four of the five important features are the worst selection from these metrics, this demonstrates how important it is to look for the "worst" abnormalities or extremities in these images. The heatmap we generated earlier as a visualization to more easily view outliers in the data confirms the importance of these features since they have higher correlations. This is valuable for physicians, since it indicates which features they should pay attention to during diagnosis.

REFERENCES

Enas Abdulhay, N. Arunkumar, Kumaravelu Narasimhan, Elamaran Vellaiappan, and V. Venkatraman. 2018. Gait and tremor investigation using machine learning techniques for the diagnosis of Parkinson disease. *Future Generation Computer Systems* 83 (2018), 366 – 373. https://doi.org/10.1016/j.future.2018.02.009

Konstantina Kourou, Themis P. Exarchos, Konstantinos P. Exarchos, Michalis V. Karamouzis, and Dimitrios I. Fotiadis. 2015. Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal* 13 (2015), 8 – 17. https://doi.org/10.1016/j.csbj.2014.11.005

S. Lahmiri, D. A. Dawson, and A. Shmuel. 2018. Article: Performance of machine learning methods in diagnosing Parkinson's disease based on dysphonia measures. *Biomedical Engineering Letters* 8, 1 (February 2018), 29–39.

UCI Machine Learning. 2016. Breast Cancer Wisconsin (Diagnostic) Data Set. https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

G. Parthiban, A. Rajesh, and S.K.Srivatsa. 2011. Article: Diagnosis of Heart Disease for Diabetic Patients using Naive Bayes Method. *International Journal of Computer Applications* 24, 3 (June 2011), 7–11. Full text available.