**I.**          **INTRODUCTION**

**1.1 Description of the project**

Breast cancer is the most commonly occurring cancer in women and the second most common cancer overall.(World Cancer Research Fund), what makes it a significant health problem for today’s society. According to clinical statistics, 1 out of 8 women is diagnosed with breast cancer during their lifetime. However, periodic clinical analysis examinations and self-examinations help in its early detection and thus it can facilitate timely clinical management of patients and help greatly increase the chances of survival. The method, employed worldwide for Breast Cancer detection consists of combination of 3 medical tests. After these, the clinicians flag the patients as having Breast Cancer if at least one of them indicates positive malignancy and as benign if all of them indicate benign. Therefore, the correct diagnosis of Breast Cancer and the classification of patients into malignant and benignant groups has become subject of multiple studies.

**1.2 Related work**

Major part of these studies use the Breast Cancer datasets from the University of California Irvine (UCI). Asri et al. (2016) investigate four different classifiers like Decision Tree, K-Nearest Neighbor (k-NN), NB, SVM and find that SVM gives the best classification accuracy of 97.13% and outperforms, therefore, all other algorithms. Borges (2015) achieved 97.80% by using the Bayesian Networks algorithm. Kharya et al., (2014) uses Naive Bayes (NB) algorithm for breast cancer detection and demonstrates the accuracy results of 93%. Nithya and Santhi (2014)  achieved 97.8% accuracy using an ensemble algorithm called multiboost Sequential Minimal Optimization. Maglogiannis et al. (2009) used SVM based classifiers and obtained the classification accuracy of 97%. Liu and Zheng (2006) obtained 92.90% accuracy, using filtered and supported sequential forward feature, which is based on support vector machine.

**II.**        **DATA DESCRIPTION**

**2.1**

We use the Breast Cancer Wisconsin (Diagnostic) Data Set, which is available from the online UCI database. This data set was created by Dr. William H. Wolberg, physician at the University Of Wisconsin Hospital at Madison, Wisconsin,USA. To create the dataset Dr. Wolberg used fluid samples, taken from patients with solid breast masses and an easy-to-use graphical computer program called Xcyt[1], which is capable of performing the analysis of cytological features based on a digital scan. It contains 569 instances and 32 attributes. It includes 30 features and 1 explanatory variable. The output variable corresponds to the diagnosis of breast tissue, which is either “M” for malignant or “B” for benignant.   Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nucleus present in the image. All the inputs are continuous and real-valued and computed for each cell nucleus. The main features are:

1. Radius mean: mean of distances from center to points on the perimeter

2. Texture mean: standard deviation of gray-scale values

3. Perimeter mean: mean size of the core tumor

4. Area mean

5. Smoothness mean: mean of local variation in radius lengths

6. Compactness mean: mean of perimeter^2 / area - 1.0

7. Concavity mean: mean of severity of concave portions of the contour

8. Concave points mean:mean for number of concave portions of the contour

9. Symmetry mean

10.  Fractal dimension mean

         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.

 (See if we can make a box plot for the features)

Approach –

Given the number of training model and techniques at our disposal, we would like to understand which algorithms can be best suited according to our classification needs.

In this project we take gradual steps in building our understanding towards different classification model which encompasses the following –

i)           A comparison of efficacy of various Machine Learning Methods

ii)          Developing understanding on feature selection

iii)         Finally, optimizing the hyperparameters to tune the algorithm to provide best possible classification

Th motivation is not just to be able to train a Model to provide better accuracy but we would also like to optimize the number of features that are required. Given the complex nature of Medical Sciences we require best possible prediction with minimum number of features and selected.

**Disclaimer:** The tools and techniques used in this project are being extensively taught in the lecture course. So our focus majorly will be on building the concept to achieve better classification results. For this we assume the reader has equal and/or more knowledge of the concepts from the lecture. Hence, we would only dive deeper into mathematical details whenever it's explicitly required. Though we would certainly mention the key differences in various models as we proceed further

**MODEL SELECTION:**

We have divided the Model Selection into 3 sub sections -



Part 1: Preliminary Comparison - This is the first step our analysis where we train various Machine Learning Models to make an initial comparison on the performance of various models. We use all the information available to us - essentially we use all the features to train the algorithms. This gives us a starting point into the efficacy of various models.

Part 2: Subset Selection: The most fundamental expectation from an algorithm is to provide efficient results with the minimum number of required features. In this section we explore three basic approaches to have reduced dimensionality in order to achieve required classification results.

Part 3: Hyper-paramater Optimization: This section covers tuning of hyper-paramater for our few chosen algorithms.

**Model Selection - Part 1: Preliminary Model Comparison**

In this section we train a few Machine Learning Algorithms to gives us a basic understanding of classification efficiency associated with the corresponding model. This gives us a starting point for our further and more extensive analysis.

**Methodology - Modet Training:**

We have tried to pick up various classification methods from fundamental Classification Classes. We have tried to represent models from Generative Classification class, Discriminative Classification Class, SVMs, Ensemble Methods etc.

The Models trained uses a 5 fold cross - validation approach to ensure that the resulting accuracy of the model gives us a fare sense of associated statistical error.

**Different Models Used**

**LDA and QDA**

'linear' (default) — Estimate one covariance matrix for all classes.

'quadratic' — Estimate one covariance matrix for each class.

 Discriminant analysis and nearest neighbor algorithms do not analyze data that contains both numeric and categorical variables. But in our case since we dont have categorical features we go ahead with LDA and QDA

**Gaussian Naive Bayesian:**

The fitcdiscr function has other two other types, 'DiagLinear' and 'DiagQuadratic'. They are similar to 'linear' and 'quadratic', but with diagonal covariance matrix estimates. These diagonal choices are specific examples of a naive Bayes classifier, because they assume the variables are conditionally independent given the class label.

**Logistic regression**

The classifier models the class probabilities as a function of the linear combination of predictors. LC assumes specific parametric form (logistic) of P (Y|X)

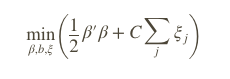
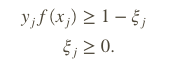
**Support Vector Machines**

In this section we discuss the methodology we used with respect to our SVM analysis. We will use basic three models from SVM -

1. Linear SVM
2. Quadratic SVM
3. Gaussian SVM

To explain further we would like to discuss a few intricacies related to SVM that we have incorporated in our analysis.

We have not assumed linear separability in our dataset and hence we train our Model for Soft Margin SVM. The optimization problem is -

such that 

In this case our eta variable or the slacked variable ensures Soft Margin SVM and its zero for the case of Hard Margin. C is the penalty parameter and we will discuss the optimization of this hyper parameter in subsequent section.

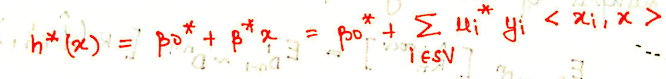
Just to provide a quick context - we know from theory that the Dual optimization problem derived for Hard and Soft Margin SVM is similar, the only difference is the constraints.

**How to classify ?**

The decision for any given x is constructed using



which essentially translates to finding the inner product  -



Now to produce better result we transform our feature space to a higher dimension Hilbert Space. Now instead of working the above solution with the inner product of xi we calculate the inner product of  <*φ*(*x*1),*φ*(*x*2)>. And we obtain this inner product of  <*φ*(*x*1),*φ*(*x*2)> using Kernel Trick

**Advantage**

The beauty of the techniques lies in obtaining a non linear decision boundary while still working with linear classification.

### Decision Tree

**Another classification algorithm is based on a decision tree. A decision tree is a set of simple rules, such as "if the sepal length is less than 5.45, classify the specimen as setosa." Decision trees are also nonparametric because they do not require any assumptions about the distribution of the variables in each class.**

Now we apply the tree-based methods for our classification problem. These are useful for interpretation and give a simple and flexible way to model interactions. Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets and gradually constructs the decision tree. The final result is a tree with decision nodes and leaf nodes. Therefore, this method includes segmenting the feature space in different simple regions. In order to make a prediction for a given observation, they typically use the mean or the mode of the training observations in the region to which it belongs.

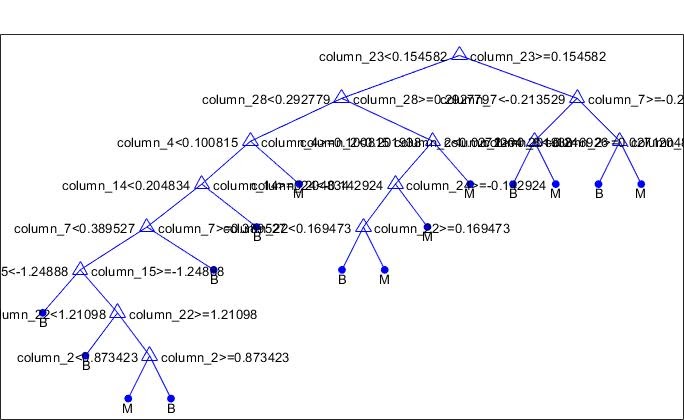
**Classification trees**

For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs. We use recursive binary splitting to grow a classification tree. There are 3 splitting criterions with which we can make a binary split:

* Misclassification error
* Gini index
* Cross-Entropy/Deviance/Log-Score

Formulas?

Matlab uses the Gini index by default. In the Classification learner we can make a choice between the “coarse tree”, “medium tree” and “fine tree”. Their difference is in the maximum number of splits used to grow a tree. Here we use the “fine tree”, which uses 100 as the maximum number of splits. After applying  to our testing data  the model obtained from the Classification Learner, we get the accuracy of 93,81%. And the tree obtained is shown in the tree below.



As we can see that only 10 features are used in this tree, It is due to the fact that

the decision tree has implicit feature selection during the model building process. When it builds the tree, it only splits features that cause the greatest increase in the node purity.

(Do you think we should describe the names of the features or better directly during the presentation)

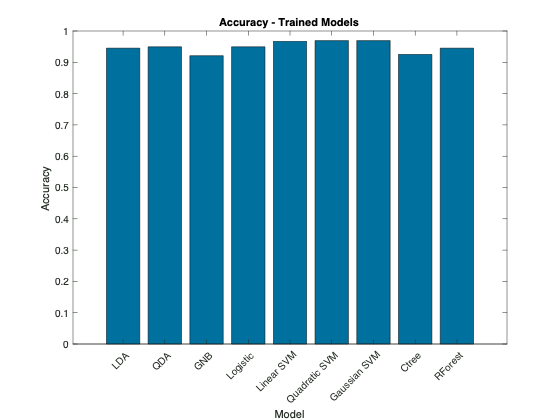
Let see once we are done with everything else if we can rename the features otherwise we’ll just add a discaliamer saying refer to the names or something in appendix

**Random forests**

The decision trees suffer from high variance, that is why we use here the Random Forest Classifier.It is an ensemble algorithm, which uses multiple classification trees on the bootstrapped training samples. When building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors (typically the number of predictors considered at each split is approximately equal to the square root of the total number of predictors) .This random choice of predictors induces the low correlation between the individual tress, what results in lower error.*The final predictions of the random forest are made by averaging the predictions of each individual tree.* (one decide the final class of the test object via combining the votes from different decision trees). As we can see from our results the accuracy of Random Forest model applied on the testing data is of 98,23%, what is 4,42% more then when we use a simple classification tree.

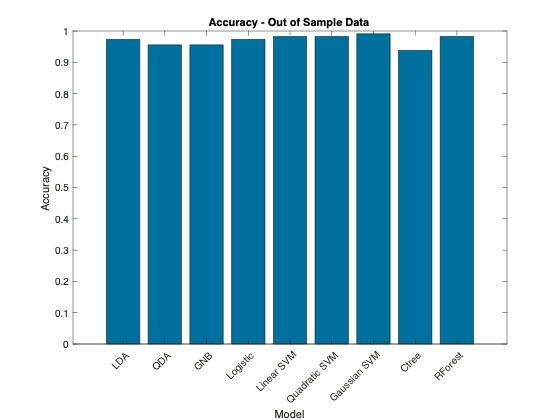
**Comparison Results:**

Post training the model we observe following  -



All the trained models had an accuracy of more than 90%. We can observe that the class of classifiers from Support Vector Machines perform consistently for the given three types of classifiers.

Subsequently, we compare the accuracy for the given testing data to observe out of sample performance of the trained models. We obtain the following -



The above graph provides following observations -

1. We see an increased value for accuracy for almost all the trained models. Usually we would not expect the accuracy on the testing sample to be higher than training dataset.
2. Secondly, we see that all three models for SVMs are still providing consistent result
3. For the testing dataset, Random Forest is also providing better results

**Feature Selection**

Post our preliminary analysis of comparing various Models in terms of their prediction accuracy, we now turn our focus to obtain the best set of features which can provide us with good prediction accuracy.

In this section we have tried to present how the analysis evolves in terms of feature selection. As a result we divide our analysis in three parts -

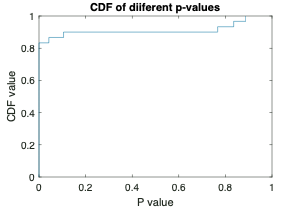
1. Filtering
2. Forward Subset Selection
3. Principal Component Analysis

**Filtering**

<https://de.mathworks.com/help/stats/examples/selecting-features-for-classifying-high-dimensional-data.html>

For Filtering, the method does not actually involve which training algorithm we will be using as it generally relies on the characteristics of the data to select features.

We have two classes of data (Benign and Malignant) and we apply a two sample t-test and compare p-values for each feature as a measure of how effectively it separates the two classes.



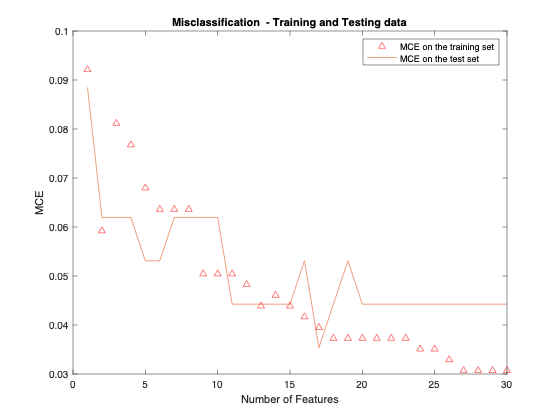
If we draw a cumulative distribution of the p-values we observe that almost 80% of features have their p-values less than 5%. So we reject the null hypothesis and we say that a large amount of features have strong discrimination power.

When we sort the features with respect to their p-values we get the following ranking -

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ranking of Features | | | | | | | | | |
| 28 | 23 | 8 | 21 | 3 | 1 | 27 | 7 | 24 | 4 |
| 6 | 26 | 11 | 22 | 13 | 14 | 2 | 25 | 18 | 5 |
| 29 | 9 | 30 | 16 | 17 | 20 | 15 | 10 | 12 | 19 |

This ranking gives us first step in exploring feature selection.

We then go ahead and try to try to train a basic Quadratic Disc Model just to observe how this feature ranking performs. In the next segment we start with the so called best feature, as obtained using t-test and we keep on adding the next subsequent important feature in each step. We train our algorithm with one extra feature each time and then we plot the Misclassification error for training and testing data.

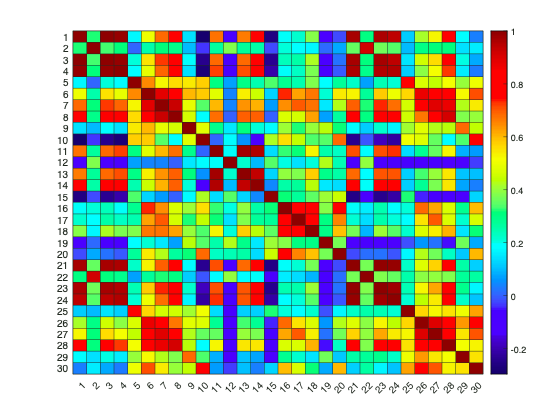


The figure gives a few interesting observations –

1. We see a gradual decrease in the training error as we increase the number of observations (except for second observation). We would also expect a drop in the training error as we increase the number of features as this is not a cross validation error.

2. The drop in Misclassification for second observation (i.e feature 23) hints that 23rd feature may be an important feature in our classification criteria. Or maybe our third observation which is 3rd feature might be giving us wrong classification thereby increasing our error. We will further explore on this part in the report.

3. Lastly we do observe a drop in error with our testing data initially and then it doesn’t change much, which may be likely that the additional features do not play significant role in classification further.



**Drawback:**

A drawback with this approach may be that it does not account for correlation within the features

As we can observe their are features that are correlated with values above 85%. We will deal with this issue in coming section on Principal Component Analysis

**Forward Selection**

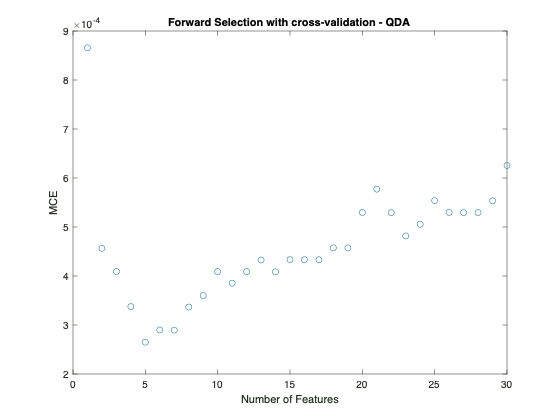
In the last section we derived a feature ranking based on the p-value obtained from the t-test and subsequently we saw how Misclassification error evolves when we increase the number of features.

Now we instead of fixing which feature enters first in our analysis, as we did earlier based on p-value ranking, this time around we allow the algorithm to select best feature for itself in each round and calculate the corresponding error.

We proceed with Forward feature selection approach in which the algorithm chooses the feature corresponding to lowest error and in the subsequent round we keep on adding features to our already accumulated features till now.

**Cross-validation approach**

To ensure that our error calculations are robust we use a 5-Fold cross validation approach to obtain the corresponding error while training our algorithm



**Key Inference**

We observe that the algorithm keeps on selecting the best subsequent feature to keep on lowering the error. However, once top 5 features are added, the next subsequent addition to the features actually gives a model with higher error of decreased precision.

So we infer that top 5 features selected using forward selection provide us with relevant features to predict our model.

**NOTE**

Since the forward selection algorithm is Greedy in the sense that it tries to achieve the best result with limited number of iterations. We can not surely say that Forward Selection is the Universally best method to give us the least error model. There are essentially 2K-1+1 (including only intercept model) possible and its probable that due to greedy setup an omitted feature combination might still provide with a better model.

**NOTE**

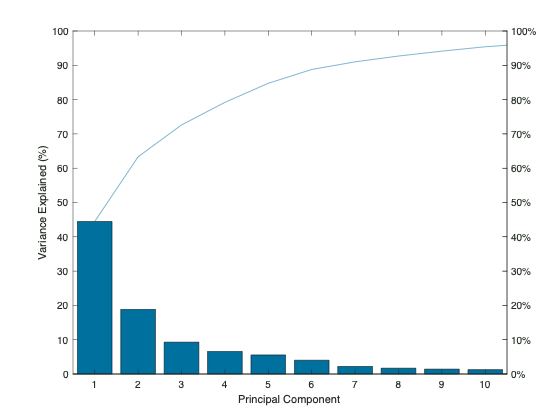
Also we need to note that the Y axis is not exactly the error term. We have used Matlab function ‘sequentialfs’, which in turn uses cross validation approach and tries to minimize the criterion function. In our case we have used loss from classification and hence the criteria is to reduce the classification loss in each step. However, the ‘sequentialfs’ function sums up all the criterion ‘scalar’ value and divides it by the number of test observations. Hence the y-axis indirectly represents a comparison of classification error only that it is divided by number of test data by the function.

But for our purpose since we only require the ranking of the features we can consider these values as a proxy for the prediction error.

**Feature Selection - Principal Component Analysis**

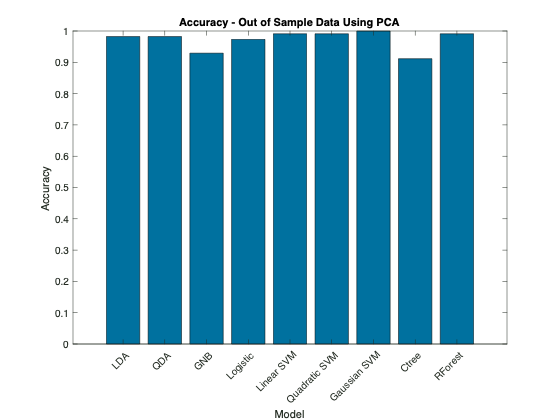
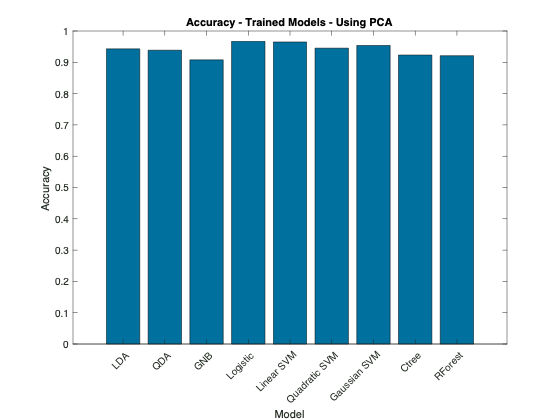
Before we proceed on the analysis, we would like to acknowledge that essentially Principal Component Analysis is considered as a dimensionality reduction technique and in our case we have put it under a bigger head - Feature Selection, we acknowledge that this may be slightly abusing the notation. However, we consider Principal Component analysis to lower dimensions that still explain the variability in data significantly.

We use the standardised training set to analyse the basic variability explained by the corresponding Principal Component in the graph below -



Plotting a curve we observe that 95% of the variability can be explained by the first 10 Principal components.

We now proceed further and train our model with these 10 Principal Components and observe the following -



We observe very similar result in terms of prediction accuracy when we compare both the above graphs to the one that we generated earlier, training models using all 30 features.

Also if there are any inputs for this section

Principal Component Analysis is an unsupervised machine learning approach, which finds the low-dimensional (high dimensionality means that the dataset has a large number of features) representation of a data set that contains as much as possible of the variation. Reducing dimensionality can create classification models which help to prevent the overfitting.PCA linearly transforms predictors in order to remove redundant dimensions.

It tries to keep the important parts that have more variation of the data, and remove the non-essential parts with less variation.PCA is such a technique which groups the different variables in a way that we can drop the least important feature. All the features that are created are independent of each other. In the Classification Learner we set the PCA so that it keeps only the components that explain 95% of the variance.

* Ensure that the input features are standardized.

**Hyperparameter Optimization**

Once we have discussed the methods on subset selection, we proceed further to optimize the hyper parameters in our trained Models. We will allow for optimization of hyper parameters based on overall training data that we have, i.e. considering all the features. Due to computational issues we are only considering a few of the training algorithms in this section.

**Approach**

Different machine learning algorithms consider different methods to classify data and in this section we will briefly describe the hyperparameters which we intend to optimize.

1. **Polynomial SVM**

Our very first comparison earlier considered Linear and Quadratic Support Vector Machines. In this section we take a combined approach and put them under the polynomial SVM case which essentially uses the Polynomial Kernel- *G*(*x*1,*x*2) = (1 + *x*1′*x*2)^p.

We use Bayesian Optimization form the Built-in Matlab function to optimize the Box Constraint level or the soft-margin penalty known as C in the primal equations.

1. **Gaussian SVM**

Our next step in optimizing the hyper parameters is to optimize using kernel function - Gaussian type. The Gaussian Kernel used is - *G*(*x*1,*x*2) = exp(–∥*x*1–*x*2)∥2).

1. **Classification Tree**

When growing a classification tree different hyper-parameters can affect its complexity, including maximum number of splits, the splitting criterion and the minimum number of observations in a leaf node. Using the Matlab Classification Learner app we optimized all of them The optimization shows that the optimal maximum number of splits is of 24, the splitting criterion is the maximum deviance and the minimum leaf size is of 17.

 The Box Constraint parameter is the soft-margin penalty known as C in the primal equations.

(*Mention stuff and explain a little bit of Soft Margin SVM)*

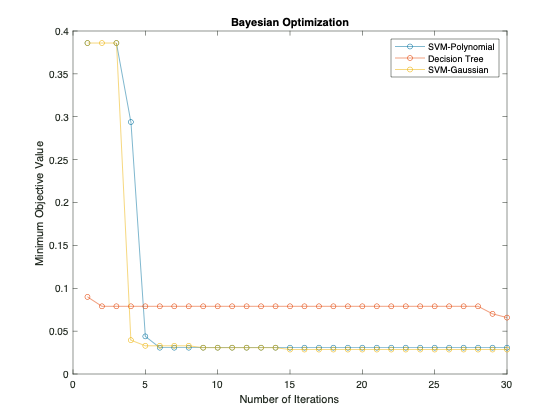
*Link* [*https://de.mathworks.com/help/stats/choose-a-classifier.html#buwh8ek*](https://de.mathworks.com/help/stats/choose-a-classifier.html#buwh8ek)

An alternative way to manage support vectors is to reduce their numbers during training by specifying a larger box constraint, such as 100. Though SVM models that use fewer support vectors are more desirable and consume less memory, increasing the value of the box constraint tends to increase the training time.

<https://de.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html#bsr5o1q>

**Optimizing hyper parameters:**

We train the above-mentioned Models to tune their hyper-parameters and we obtain the following plot when we run a Bayesian Optimization algorithm -

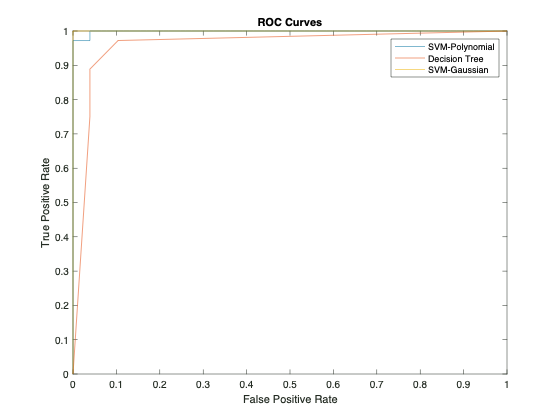


The hyperparameter sets are tuned over several iterations to improve the performance of the Model. The plot above shows that both the Models for SVM have better prediction accuracy when compared with to that of Decision Trees.

We further create a comparison between their respective confusion matrix to give further understanding of prediction of individual cases of Benign and Malignant Cases.



The Gaussian SVM predicts with a perfect accuracy. We observe that SVM are again providing consistent predictions. Another observation is that Gaussian Model is a bit better than the Polynomial one, which is what we have observed in earlier cases too. The same fact is again validated from the ROC curve as below -



**Optimizing Hyper-parameters with PCA**

This brings us to our final analysis leg. In this section we optimize the hyper parameters for our ML Models with Principal Component Analysis.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Linear SVM | Classification Tree | Gaussian SVM |
| Prediction Accuracy | 0.99 | 0.91 | 0.99 |

And once again we observe better performance of Support Vector Machines. However, in order to make one last final observation on choosing between linear or Gaussian Model, we finally look at the F measure and hence we consider their confusion matrix -



In this concluding part we can observe that there are no false negatives generated by the Gaussian SVM in contrast with one False Negative for Linear SVM case. Since we are dealing with bio-medical data here, the cost of Flase negatives are instrinsically higher than False Positives and hence we can say that Gaussian SVM model with PCA performs better than the other models that we compared

APPENDIX

Table 1 - ACCURACY for training and testing data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | LDA | QDA | GNB | Logistic | Linear SVM | Quadratic SVM | Gaussian SVM | Ctree | RForest |
| Training | 0.95 | 0.95 | 0.92 | 0.95 | 0.97 | 0.97 | 0.97 | 0.93 | 0.95 |
| Testing | 0.97 | 0.96 | 0.96 | 0.97 | 0.98 | 0.98 | 0.99 | 0.94 | 0.98 |

Table 2 - Area Under Curve for training and testing data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | LDA | QDA | GNB | Logistic | Linear SVM | Quadratic SVM | Gaussian SVM | Ctree | RForest |
| Training | 1.00 | 0.99 | 0.99 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Testing | 1.00 | 1.00 | 0.99 | 0.97 | 1.00 | 1.00 | 1.00 | 0.94 | 1.00 |

References

University of California Irvine. UCI Machine Learning Repository; 2019. Available from

Nithya R, Santhi B. A Data Mining Techniques for Diagnosis of Breast Cancer Disease. World Applied Sci J 2014:18-23.

Liu, Y. and Y.F. Zheng, 2006. FS-SFS: A novel feature selection method for support vector machines. Pattern Recognition, 39(7): 1333-1345.

Maglogiannis, I., E. Zafiropoulos and I. Anagnostopoulos, 2009. An intelligent system for automated breast cancer diagnosis and prognosis using SVM based classifiers. Applied Intelligence, 30(1): 24-36

Borges, L.R. Analysis of the Wisconsin Breast Cancer Dataset and Machine Learning for Breast Cancer Detection. Group 1989, 1, 369.

Kharya S, Agrawal S, Soni S (2014).Naive Bayes classifiers: A probabilistic detection model for breast cancer. Int J Comput Appl, 92, 26-31.

[1] The program uses a curve-fitting algorithm, to compute ten features from each one of the cells in the sample, then it calculates the mean value, extreme value and standard error of each feature for the image, returning a 30 real-valuated vector.