Introduction

Motivation 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.

Hence, we are motivated to apply our acquired knowledge from the Course 'Applied Econometrics and Machine Learning' at the University of Konstanz, Germany, to develop predictive classification model that provide us with high prediction accuracy by using the optimum/least number of features.

Related work

Major part of the studies use the Breast Cancer datasets from the University of California Irvine (UCI). Asri et al. (2016) investigate four different classifiers like Decision Tree, Discriminant Classifiers, Naïve Bayes, 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) uses 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.

Data Description

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.

The Rest of the features are mentioned below –

Feature 11	radius_se	Feature 21	radius_worst
Feature 12	texture_se	Feature 22	texture_worst
Feature 13	perimeter_se	Feature 23	perimeter_worst
Feature 14	area_se	Feature 24	area_worst
Feature 15	smoothness_se	Feature 25	smoothness_worst
Feature 16	compactness_se	Feature 26	compactness_worst
Feature 17	concavity_se	Feature 27	concavity_worst
Feature 18	concave points_se	Feature 28	concave points_worst
Feature 19	symmetry_se	Feature 29	symmetry_worst
Feature 20	fractal_dimension_se	Feature 30	fractal_dimension_worst

Methodology

General

Given the number of training models and techniques at our disposal, we would like to understand which algorithms can be best suited to provide us with best predictive classification model.
In this project we take gradual steps in building our understanding towards different classification model which encompasses the following –

- A comparison of efficacy of various Machine Learning Methods
- Developing understanding of feature selection
- Finally, optimizing the hyperparameters to tune the algorithm to provide the best possible classification

The 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 were being extensively taught in the lecture course. So our focus majorly is on building better predictive classification models. For this we assume the reader has equal and/or more knowledge of the concepts from the lecture. Hence, we would dive deeper into explicit mathematical details only whenever it's required. Though we would certainly mention the key differences in various models as we proceed further

Dataset Selection

We have used 80-20% split to segregate training and testing data. Since some features differ in their scales so we have Normalized our data using centring and standardisation.

Model Selection

Model Selection and Optimization forms the core of this paper. We discuss different approaches that are used in developing the predictive models. The section is divided into following 3 subsections

1. **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.
2. **Part 2: Subset Selection –** The most fundamental expectation from an algorithm is to provide efficient results with minimum number of required features. In this section we explore three basic approaches to have reduced dimensionality in order to achieve required classification results. –
   a. Filtering
   b. Forward Selection
   c. Principal Component Analysis
3. **Part 3:Hyper-parameter Optimization:** This section forms the concluding part of our Model Selection and it covers tuning of hyper-parameter for our chosen model

Part 1: Preliminary Model Comparison

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

Methodology - Model 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

1. **Generative Classifiers** – Our first group of classifier consists of LDA, QDA and Naïve Bayes Classifier. For this group we obtain the Posterior probabilities using Bayes Rule. So we obtain $P_{Y|X}$ using Bayes rule and express it using $P_{X|Y}$ and P_X

   The key differences for the three classifiers is as follows –
   - **LDA** – We estimate one covariance matrix for all classes
   - **QDA** – Individual covariance matrix is estimated for each class
   - **Naïve Bayes** – It assumes conditional independence of features given the class
2. **Logistic Classifier** – The classifier models the class probabilities as a function of the linear combination of predictors. LC assumes specific parametric form (logistic) of $P_{Y|X}$
3. **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 -
   a. Linear SVM
   b. Quadratic SVM
   c. 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 –

$$\min_{\beta,b,\xi}\left(\tfrac{1}{2}\beta'\beta + C\textstyle\sum_i \xi_i\right) \qquad \text{such that} \quad y_i f(x_j) \geq 1-\xi_j \;\;\&\;\; \xi_j \geq 0$$

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 the 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 in the constraints.

Classification criteria - The decision for any given x is constructed using

$$f^*(x) = sign(h^*(x))$$

which essentially translates to finding the inner product -

$$h^*(x) = \beta_0^* + \beta^* x = \beta_0^* + \sum_{i\in sv} \mu_i^* y_i < xi, x >$$

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 x_i we calculate the inner product of $<\varphi(x_1),\varphi(x_2)>$. And we obtain this inner product of $<\varphi(x_1),\varphi(x_2)>$ using Kernel Trick.

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

4. **Classification Trees** – 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. The final result is a tree with decision nodes and leaf nodes. Therefore, this method includes segmenting the feature space in different sample regions.

   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:

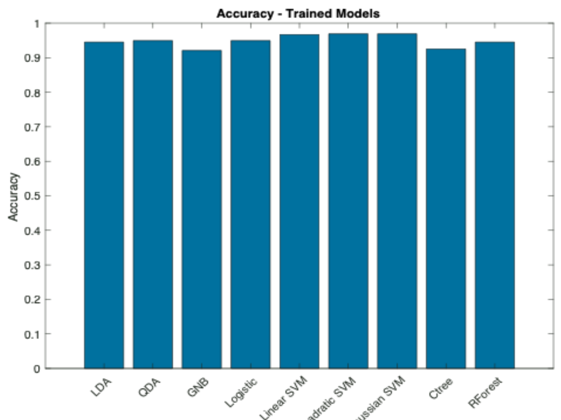
Misclassification error	$\frac{1}{n_m}\sum_{x_i \in R_m} \mathbb{1}(y_i \neq r) = (1-\hat{p}_{mr})$
Gini index	$\sum_{r \neq r\prime} \hat{p}_{mr}\hat{p}_{mr}{}' = \sum_{r=1}^{k} \hat{p}_{mr}(1-\hat{p}_{mr})$
Cross-Entropy/Deviance/Log-Score	$-\sum_{r=1}^{k} \hat{p}_{mr}\, ln(1-\hat{p}_{mr})$

   Our model uses Gini Index to train the algorithm. Of the available option we used ‘fine tree’ in the Matlab which allow a higher number of splits.

5. **Random Forest** – 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). To apply the random forest algorithm we use the Bagged Trees option form the Classification learner which Breiman's (2001) Random forest algorithm by default.

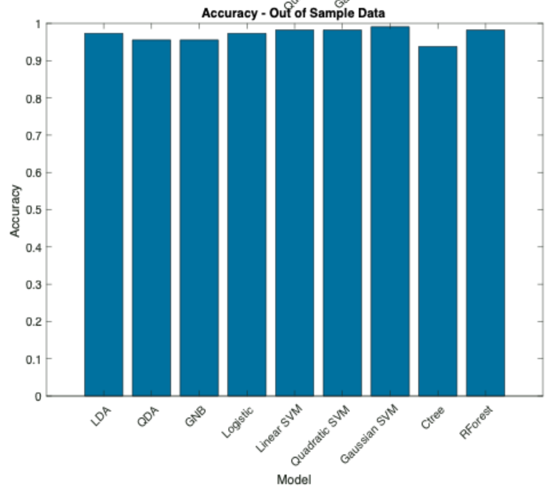
Performance Comparison

All the trained models have an accuracy of more than 90%. We observe that the class of classifiers from Support Vector Machines perform consistently for the given three types of classifiers. Comparatively the discriminant classifiers along with Random Forest are giving good prediction accuracy too.



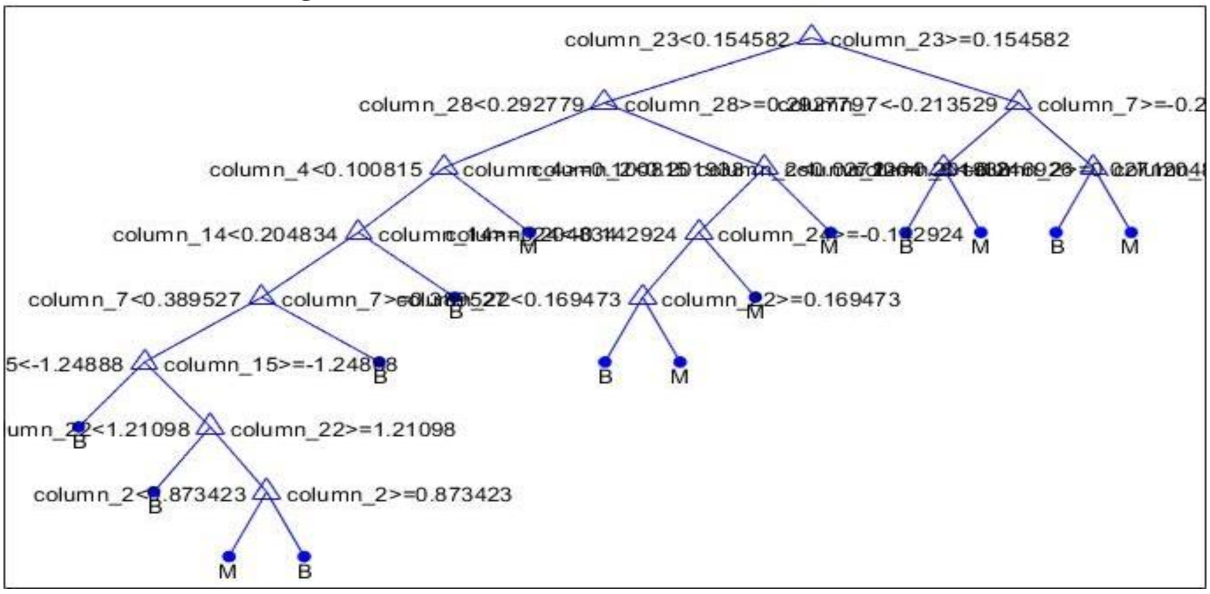
Subsequently, we compare the accuracy for the given testing data to observe out of sample performance of the trained models. The above graph provides following observations -

- 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.
- Secondly, we see that all three models for SVMs are still providing consistent result
- For the testing dataset, Random Forest is also providing better results



Comparison between Classification Tree and Random Forest

Applying the trained Classification Tree on the testing data, we get an accuracy of 93,81% which is 4.42% short when compared to that of the Random Forest.
We obtain the following Classification Tree –



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 node purity.

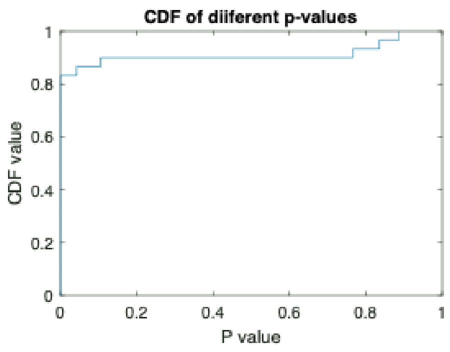
Part 2: Feature Selection

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 to balance the trade-off between model complexity and 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 into three parts -

- Filtering
- Forward Subset Selection
- Principal Component Analysis

A. Filtering

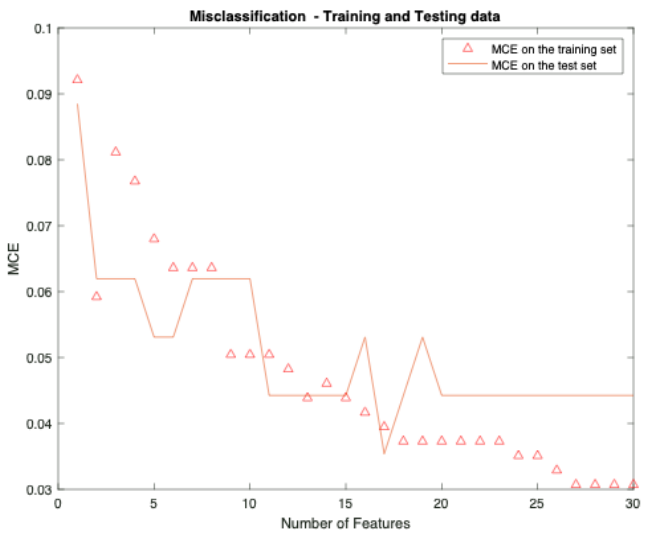
We have two classes of data (Benign and Malignant) and we apply a two-sample t-test and compare p-values for each feature as a measure of how effectively it separates the two classes.
If we draw a cumulative distribution of the p-values we observe that almost 80% of features have their p-values less than 5%. So we reject the null hypothesis and we say that a large number of features have strong discrimination power.



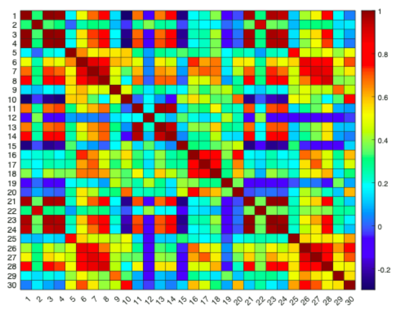
Sorting the p-values we get a ranking of features (see Appendix – T3) and we further use this feature ranking to train a QDA model to observe how adding features in this ranking order is related to misclassification error. The premise is to use the best-ranked feature (lowest p-value) and subsequently add the next best feature.

The figure gives a few interesting observations –

- We see a gradual decrease in the training error as we increase the number of observations. We also expect a drop in the training error as we increase the number of features as this is not a cross-validation error.
- Lastly we do observe a drop in error with our testing data initially and then it doesn't change much. It might hint towards having approx. 10 features to predict train the model efficiently



Drawback: with this approach may be that it does not account for correlation within the features. From the correlation heatmap we can see that there are features that are correlated with values above 85%.
We will deal with this issue in coming section on Principal Component Analysis



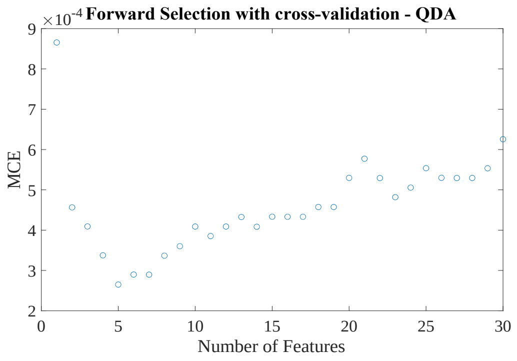
B. Forward Selection

Now 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



We train a QDA Model and observe that the algorithm selects the best subsequent feature to keep on lowering the error. However, once top 5 features are added, the next successive addition to the features actually gives a model with higher error or decreased precision.
So we infer that top 5 features selected using forward selection provide us with relevant features to predict our model. The features are mentioned in Appendix A1

NOTE

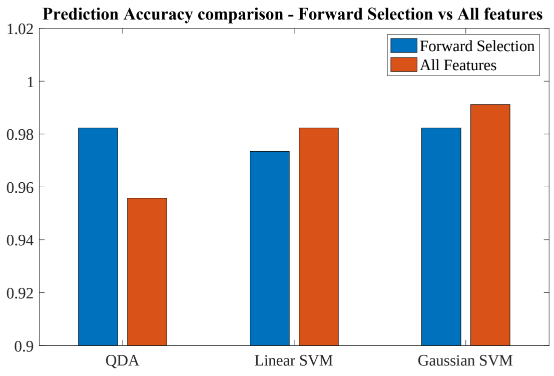
Since the forward selection algorithm is Greedy in the sense that it tries to achieve the best result with limited number of iterations. We can not surely say that Forward Selection is the universally best method to give us the least error model. There are essentially 2K-1+1 (including only intercept 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', that uses cross validation approach to minimize the criterion function. In our case, we have used loss from classification and 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. Hence the Y-axis serves as a proxy for Misclassification error.

Model Validation

We train our Models based on the features being selected by the Forward Subset Selection approach and then we validate our model using out of sample data. We compare these results with their earlier counterparts of algorithms which included all 30 features



When we train our model using the best features selected from forward subset selection method we observe that QDA gives us good results with using lesser features while both the SVMs perform better using all the features. This gives us a good basic understanding of features that stand out as all the three models include similar features in their best feature subset (the ranking of features is mentioned in Appendix: T4 -A/B/C.

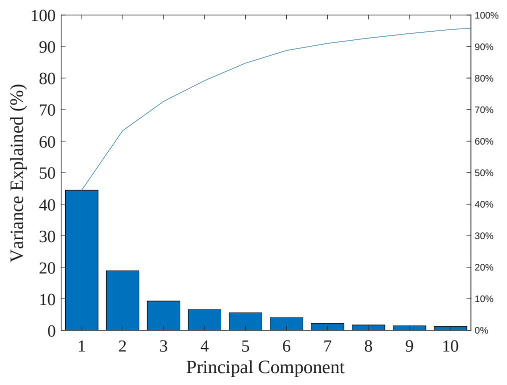
C. Principal Component Analysis

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.

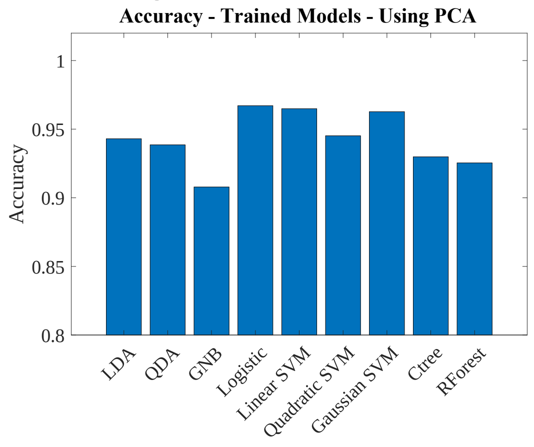
We use the standardised training set to analyse the basic variability explained by the corresponding Principal Component in the graph below

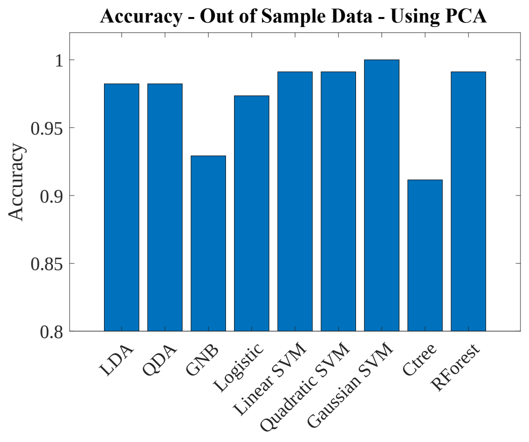
Plotting a curve we observe that first Principal Component explains approximately 45% of variability in data.

Another key observation is 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.
A comparison between trained data prediction accuracy and out of sample accuracy shows that Out of Sample results are better, this fact may be attributed to random sampling in the testing data. As we would expect the training accuracy to be better than the out of sample one.

Part 3A: 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.

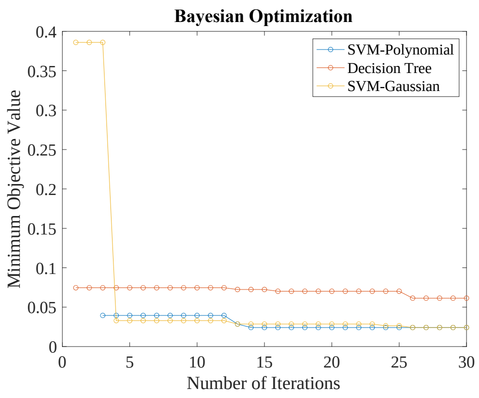
Different Models Used

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.
2. **Gaussian SVM** – Our next step in optimizing the hyper parameters is to optimize using kernel function - Gaussian type. The Gaussian Kernel used is - $G(x_1, x_2) = \exp(-\|x_1 - x_1)\|_2)$.

   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
3. **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.

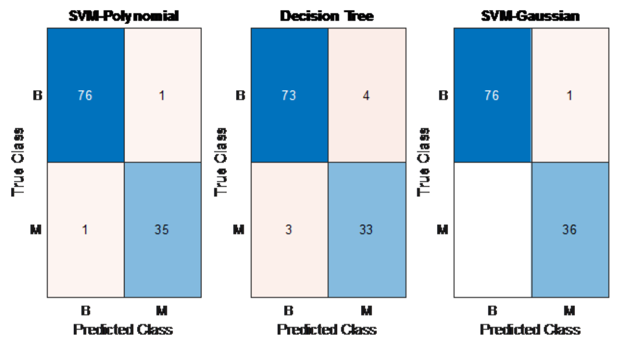
Optimizing Hyper Parameters

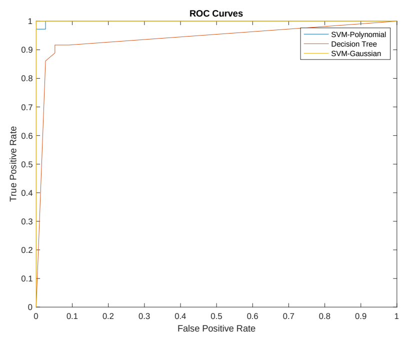
We train the above-mentioned Models to tune their hyper-parameters and we obtain the following plot when we run a Bayesian Optimization algorithm –



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.

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



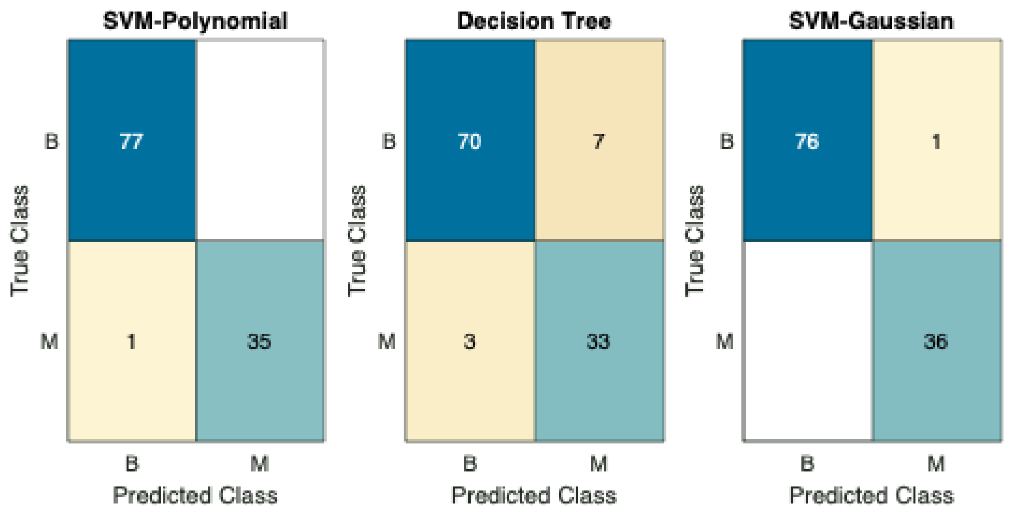


Part 3B: Hyperparameter Optimization with PCA

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 False negatives are intrinsically higher than False Positives and hence we can say that Gaussian SVM model with PCA performs better than the other models that we compared

Conclusion

Potential Opportunities

- Deep Learning: A possible improvement in the trained models could be possible addition of Neural Network which have deeper layers of learned algorithm to potentially provide even more precise classification
- Regularization: Since we are dealing with a topic from the field of bio-statistics, which usually has wide data, we can possibly think of better shrinkage techniques to improve classification models. Regularization may not only provide us with sparser model but also shift our focus towards more important features by shrinking/eliminating less important features
- Another possible addition in a step towards training better model would be to possibly have further addition to features set. Currently the dataset has only real values, however we can also expect more categorical data in regards to data collection when it comes to Breast Cancer
- Biased Cost Function – Since we are dealing with a severe medical condition, the cost for misclassification can be modified accordingly such that it reflects the true cost of flase negatives

Summary

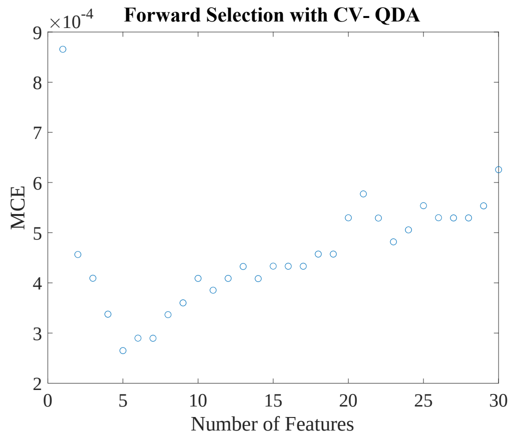
Throughout the paper we were able to see continuous improvement in various trained models by tweaking some parameter in each subsequent step. From basic model training to fine tuning the hyperparameters we did observe how the prediction efficiency changed.
We would like to give a dual approach in regards to our conclusion –

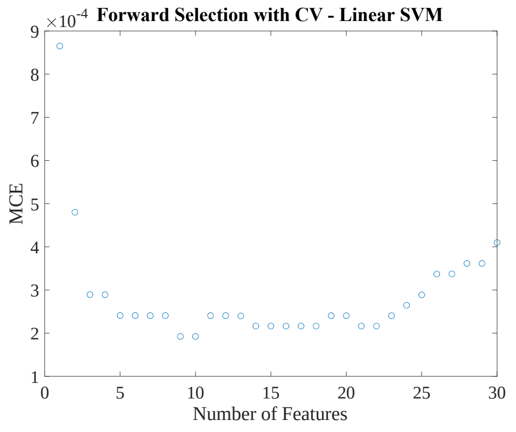
A. Recognising Important Features – The focus is just not finally receiving a label for a corresponding set of features, but more importantly, we would like to figure out which features are key to better classification models. And from our feature selection section we were able to draw a list of features that were constantly ranked higher in terms of their explanation towards model accuracy. We were able to group ‘feature 23’, ‘feature 25’, ‘feature 22’ ‘feature 2’ as being recurring as most important feature in multiple algorithms.

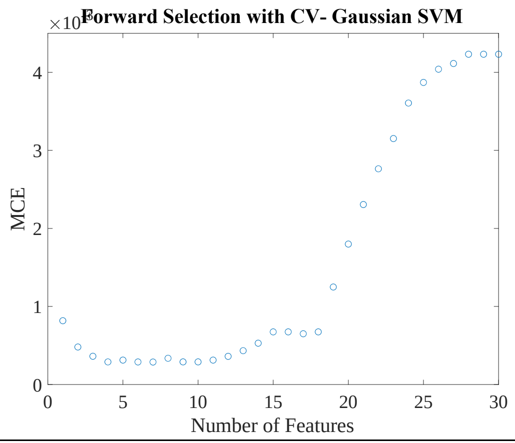
B. Accurate Prediction – Without any second thoughts, this was the metric that we constantly gauged for. We were able to observe that classifiers from SVM family were able to produce consistently out perform their peer classes of models in almost every scenario. Even though our dataset wasn’t very ‘wide’, however effective use of Principal Component Analysis led us to highly trained SVM models.
Lastly, the choice within the SVM class was going for Gaussian Kernel, which not only gave better results in terms of accuracy but also it was consistently best performer when calculating the F-measure. Considering a severe situation such as Breast Cancer, the cost of false negatives can be huge and Gaussian SVM was able to work really well in this regard.

APPENDIX

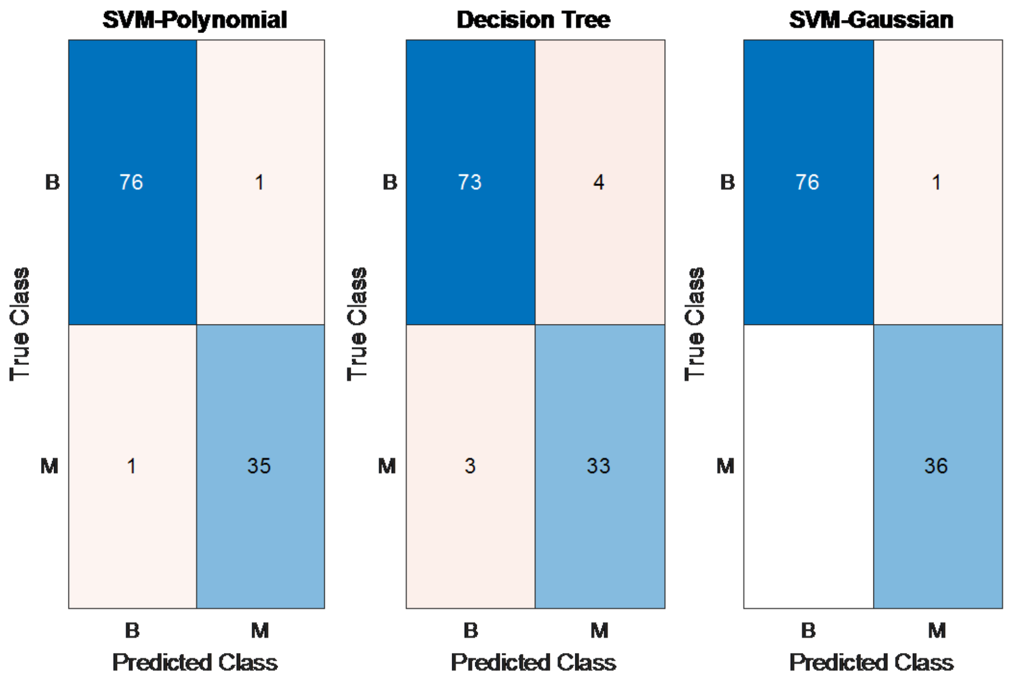
A1: Forward selection – MCE with increasing number of features



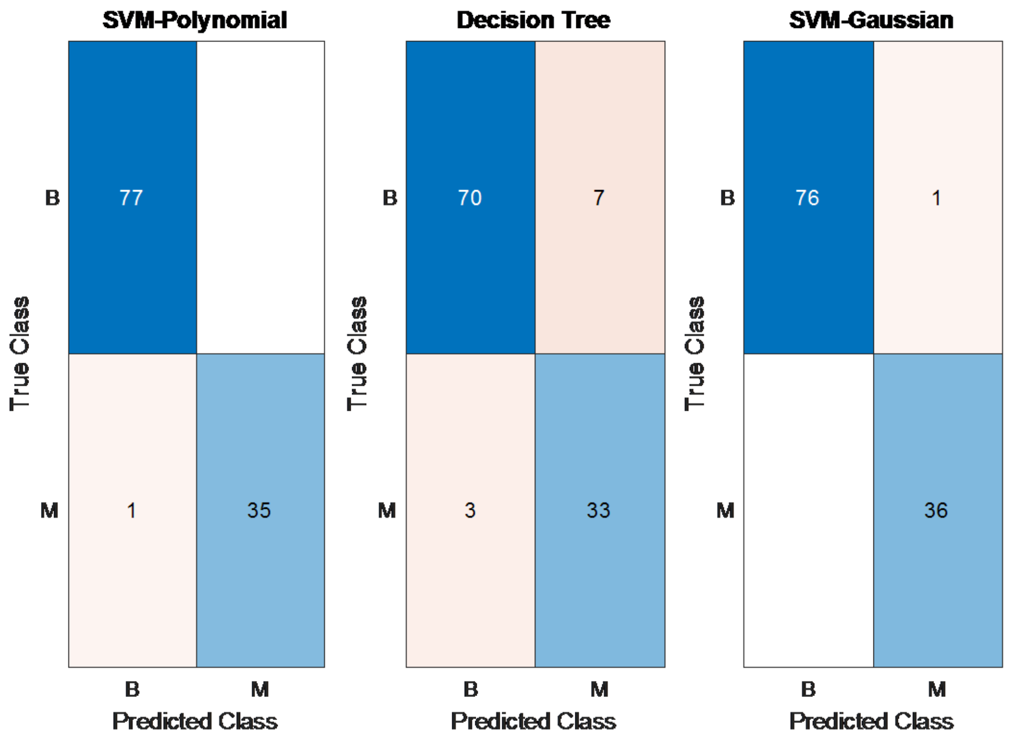




A2: Confusion Matrix – Hyper-parameter Optimization (All feature)



A2: Confusion Matrix – Hyper-parameter Optimization (using PCA)



Tables

T1: Accuracy for Training & Testing Data (All Features)

	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	0.94
Testing	0.97	0.96	0.96	0.97	0.98	0.98	0.99	0.94	0.99

T2: AUC for Training & Testing Data (All Features)

	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

T3: Ranking of Features for QDA model basis p values

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

T4 A: Feature Ranking – QDA Forward Subset Selection

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

T4 B: Feature Ranking – Linear SVM Forward Subset Selection

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

T4 C: Feature Ranking – Gaussian SVM Forward Subset Selection

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

T4: Accuracy for Training & Testing Data (with PCA)

	LDA	QDA	GNB	Logistic	Linear SVM	Quadratic SVM	Gaussian SVM	Ctree	RForest
Training	0.94	0.94	0.91	0.97	0.96	0.95	0.96	0.93	0.93
Testing	0.98	0.98	0.93	0.97	0.99	0.99	1.00	0.91	0.99

T5: AUC for Training & Testing Data (with PCA)

	LDA	QDA	GNB	Logistic	Linear SVM	Quadratic SVM	Gaussian SVM	Ctree	RForest
Training	0.99	0.99	0.97	1.00	1.00	1.00	1.00	1.00	1.00
Testing	1.00	1.00	0.98	0.99	1.00	1.00	1.00	0.93	1.00