Predicting the Breast Cancer Status
by applying
Support Vector Machine, Random Forests, Balanced Random Forests

Full Name : **Melike Savas**
Student ID : **2752688**

Adviser : **Richard Yiying Fan**
Course Number : **STA 696**
Department : **Mathematics with Applied Statistics**

Semester of Graduation : **Spring - 2022**

ABSTRACT

Breast Cancer is one of the most commonly diagnosed cancers among American women. Generally, Breast Cancer is diagnosed in women who do not have symptoms. There are two types of Breast cancer that are benign, and malign.[2] If I can have a predictive model depending on some future, that would help me to predict malign type breast cancer that gives a chance to take precautions and decrease the death caused because of breast cancer.

Support Vector Machine and Random Forest are some of the useful methods in machine learning. Support vector machine is creating a boundary that is separating classes. This boundary could be either linear or radial based on the separation shape. When we have a boundary we are naming the closest units to the boundary from both classes as "support vectors". [6]

Random forest is a supervised machine learning algorithm that is commonly used to solve classification and regression problems. It creates decision trees from several samples, using the majority vote for classification and the average for regression[5]

My data set is imbalanced, and there is a new method RandomForestSRC that is expected to give better results for imbalance data sets.

I am going to use the "Breast Cancer" data set. I know that data is collected from the patients who have tended to have breast cancer, however, the origin of the data is not given, since the patient's information is confidential. I am planning to obtain a predictive model by using a support vector machine method, random forest, and balanced random forests(BRF), then compare them to see which machine learning method would give me better predictive results for the breast cancer data set.

INTRODUCTION

Breast cancer is one of the most debilitating diseases and a leading cause of mortality among women. Late discovery of Breast Cancer can significantly lower survival chances; as a result, the automatic disease detection system assists the medical sector in diagnosing and analyzing, providing fast reaction, dependability, efficacy, and a reduction in the risk of mortality. Generally, Breast Cancer is diagnosed in women who do not have symptoms. There are two types of Breast cancer that are benign, and malign.[2]

Because the SVM technique is data-driven and model-free, it has a lot of discriminative potential for classification, especially when sample sizes are small and there are a lot of variables (high-dimensionality space) [3].The Support Vector Machine (SVM) is a supervised machine learning approach for pattern detection and classification. The SVM method conducts classification by generating a multidimensional hyperplane that maximizes the margin between two data clusters to best differentiate between two classes. This approach provides strong discriminative power by transforming the input space into a multidimensional space using specific nonlinear functions known as kernels[2]. SVM: Support Vector Machine is a supervised classification algorithm where we draw a line between two different categories to differentiate between them. SVM is also known as the support vector network[4].

Random forest is a supervised machine learning algorithm that is commonly used to solve classification and regression issues. It creates decision trees from several samples, using the majority vote for classification and the average for regression. One of the most essential characteristics of the Random Forest Algorithm is that it can handle data sets with both continuous

and categorical variables, as in regression and classification. For classification difficulties, it produces superior results.[5]

Classification of class imbalanced data sets has been identified as a top problem in machine learning. There is an existing new method called BRF which is an alternative to solve this problem. It defines the problem of class imbalance and breaks down the strategies that have been utilized to address the issue. Among the many strategies proposed, one of the most prominent is undersampling the majority class so that its cardinality equals that of the minority class. Because it was implemented in Breiman's original Fortran code used by the random forest R-package, undersampling the majority class improves classification performance compared to the minority class in the balanced random forests method. It appears to be the most common approach when using random forests to learn imbalanced data.[7]

PROJECT GOALS, METHODS, AND BACKGROUND

My goal is to use Support Vector Machine, Standard Random Forest, and Balanced Random Forest to predict diagnosis. I am looking for which method gives me the best results. What are the advantages and disadvantages of each method?

Support Vector Machine

Support Vector Machine(SVM) is a generalization of a simple and intuitive classifier called a maximal margin classifier. The maximal margin classifier is simple, however, it can not be applied to most data sets, since it requires that the classes should be separable by a linear boundary. The extension of a maximal margin classifier is called a support vector classifier which could be applicable in a larger range of cases. And SVM is a further extension of the support vector classifier in order to accommodate non-linear class boundaries.[kitap advanced]

Classification With Non-Linear Decision Boundaries

When the decision boundary is not linear SVM does not have an absolute theory to set non-linear decision boundaries. SVM is the result of a specific method of extending the feature space with kernels. [kitap].Our main idea is by enlarging our feature space to set a non-linear boundary between classes.

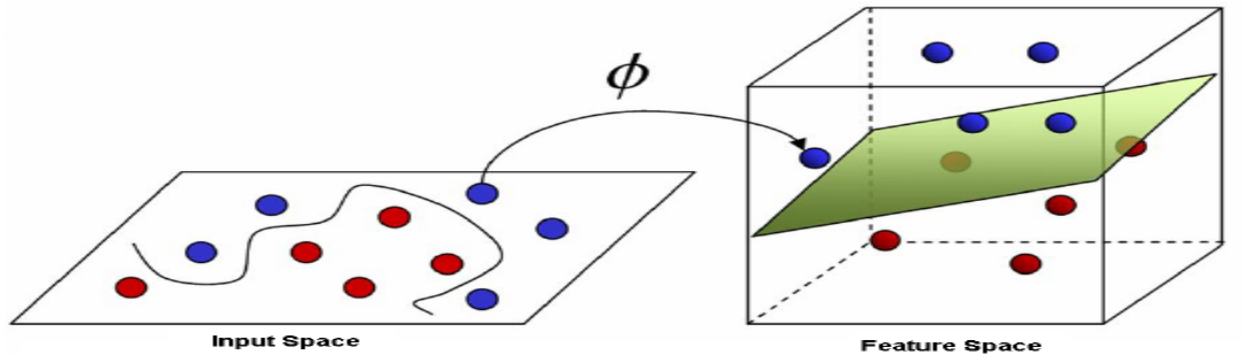

Figure 1.1 Nonlinear decision boundary

The linear support vector classifier can be represented as

$$f(x) = \boldsymbol{\beta}_0 + \sum_{i=1}^{n} \boldsymbol{\alpha}_i \prec x, x_i \succ \quad \ldots (9.1)$$

where there are n parameters alpha i, i=1,2,3,...,n one per training observations.
To estimate the parameters alpha 1, … alpha n, and beta0, all we need are the n.(n-1)/2 inner products $\prec x_i, x_{i'} \succ$ between all pairs of training observations .

Now we replace the inner product (9.1) with a generalization of the inner product of the form

$$K \prec x_i, x_{i'} \succ \quad \ldots.(9.3)$$

where K is some function that we will refer to as a kernel. A kernel is a function that quantifies the similarity of two observations.

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j} \qquad \ldots. (9.2)$$

that give us back the support vector classifier. Equation 9.2 is known as the linear kernel since the support vector classifier is linear in the features; the linear kernel essentially quantifies the similarity of a pair of observations using Pearson(standard) correlation.

When we choose another form for 9.3;

One could replace every instance of $\sum_{j=1}^{p} x_{ij} x_{i'j}$ with the quantity

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^{d} \qquad \ldots (9.4)$$

This(9.4) is known as polynomial kernel of degree d is a positive integer. With a kernel d>1 support vector classifier algorithm leads to a much more flexible decision boundary. It essentially amounts to fitting a support vector classifier in a high-dimensional space involving polynomials of degree d, rather than in the original feature space.

When the support vector classifier is combined with a non-linear kernel such as(9.4) the resulting classifier is known as a support vector machine. In this case, the function has the form

$$f(x) = \beta_0 + \sum_{i=1}^{p} \alpha_i K(x_{ij} x_{i'j}) \qquad \ldots (9.5)$$

The polynomial kernel is shown in (9.4) is one example of a possible non-linear kernel, but alternatives abound. Another popular choice is the radial kernel which takes the form

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{i=1}^{p} (x_{ij} - x_{i'j})^2 \quad \ldots (9.6)$$

γ *is a positive constant.*

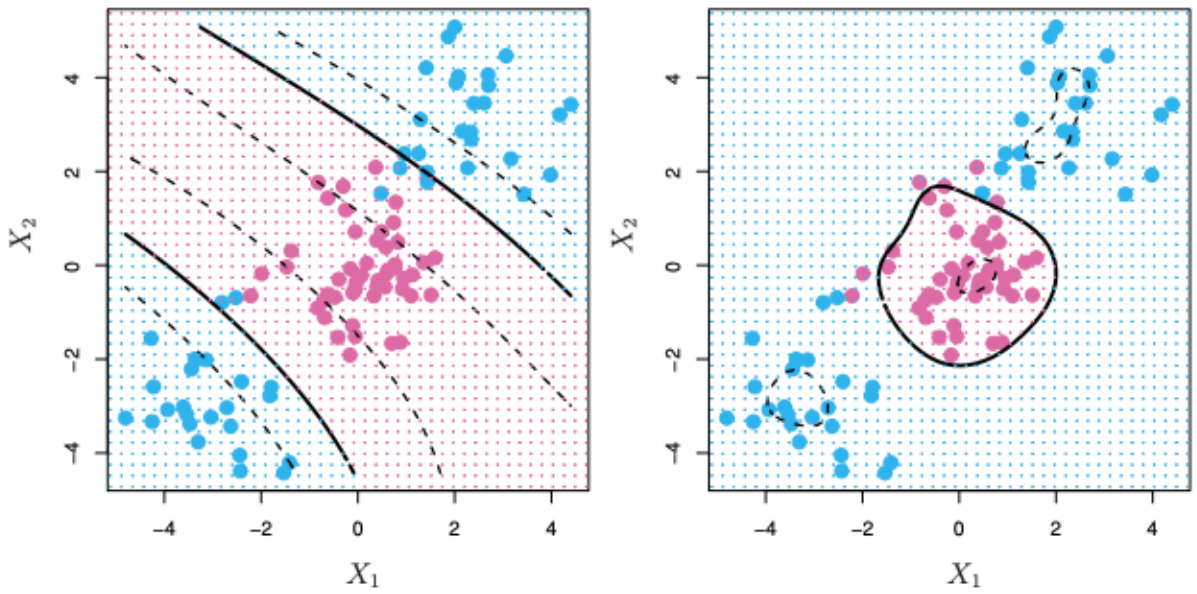

Figure 1.2 Radial kernel and Poly kernel Left is an SVM with a polynomial kernel of degree 3 is applied to the non-linear data. Right, an SVM with a radial kernel is applied.

Random Forests:

Bagging, also known as Bootstrap aggregating, is an ensemble learning strategy that helps machine learning algorithms increase their performance and accuracy. It decreases the variance of a prediction model by dealing with bias-variance trade-offs.

Random forests outperform bagged trees, because of a modest modification that decorrelates the trees. On bootstrapped training samples, we create n number of decision trees, similar to bagging. When creating these decision trees, however, a random sample of m predictors from the whole set of p predictors is picked as split candidates for each split in the tree.

When creating a random forest, the algorithm is not even permitted to evaluate a majority of the available predictors at each split in the tree.

In this case, bagging does not allow us to reduce variance much across a single tree; however, random forests solve this problem by forcing each split to evaluate just a subset of the predictors. This method may be thought of as decorrelating the trees since the average of the generated trees becomes less variable and hence more reliable.

Using a small value of m in building a random forest will be useful while we have a large number of correlated predictors.

Random Forest BRF(Balanced Random Forests):

Rather than selecting a bootstrap sample of size N, a sample of size 2N1 is used where the probabilities of minority and majority class instances are selected for the bootstrap sample that it must in

order to balance the data, thus satisfying the balancing condition.BRF uses the Bayes rule for classification.

STATISTICAL ANALYSIS and RESULTS in

1. Descriptive Statistics:

I am going to use the “breast cancer” data set. This dataset includes information about patients that can be used to predict whether a patient is likely to get breast cancer depending on the given future. I know that data is collected from the patients who have tended to have strokes, however, the origin of the data is not given, since the patient's information is confidential.

I have 32 variables and 569 observations in the data set. My response variable is “diagnosis”, my predictors are "diagnosis","radius_mean","texture_mean","perimeter_mean","area_mean" "smoothness_mean","compactness_mean","concavity_mean","concave.points_mean", "symmetry_mean" ,"fractal_dimension_mean","radius_se" ,"texture_se","perimeter_se", "area_se" "smoothness_se","compactness_se","concavity_se","concave.points_se" , "symmetry_se" ,"fractal_dimension_se","radius_worst","texture_worst","perimeter_worst" , "area_worst" "smoothness_worst" , "compactness_worst" , "concavity_worst" , "concave.points_worst" "symmetry_worst" , "fractal_dimension_worst". My response variable is categorical, it gives “B”, and “M” which means Benign and Malignant.

2. Statistical Analysis:

All of my variables are quantitative in my dataset as predictors. I do not have any missing values. The class of the “diagnosis” variable is a character, I change it as a factor. I also removed the “id” variable which is not useless. (Table 2.1)

Table2.1 Internal structure of the data after manipulation

```
'data.frame':   569 obs. of  31 variables:
 $ diagnosis              : Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...
 $ radius_mean            : num  18 20.6 19.7 11.4 20.3 ...
 $ texture_mean           : num  10.4 17.8 21.2 20.4 14.3 ...
 $ perimeter_mean         : num  122.8 132.9 130 77.6 135.1 ...
 $ area_mean              : num  1001 1326 1203 386 1297 ...
 $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
 $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
 $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
 $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
 $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
 $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
 $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
 $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
 $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
 $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
 $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
 $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
 $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
 $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
 $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
 $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
 $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
 $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
 $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
 $ area_worst             : num  2019 1956 1709 568 1575 ...
 $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
 $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
 $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
 $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
 $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
 $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...
```

When I checked the number of "B'' 's and "M" 's for the "diagnosis" variable, I obtained the ratio of two values as 1.68, thus meaning I have an imbalanced dataset. (Figure 2.1)

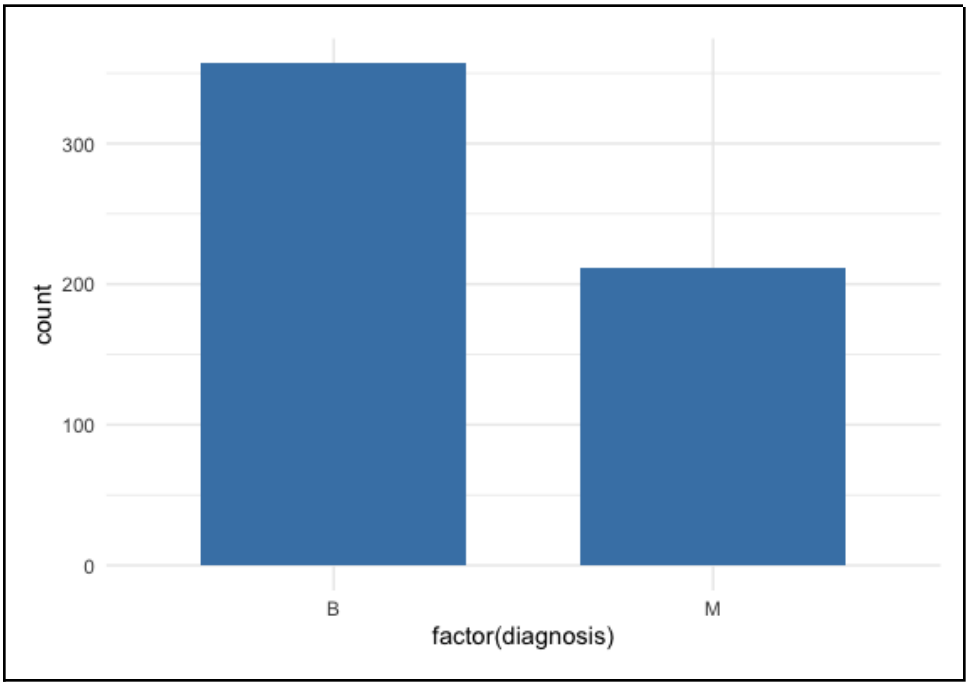

Figure 2.1 Visualization for imbalanced dataset

I am going to use the "Support Vector Machine" method to predict "diagnosis" status by using given predictors. I will try to fit SVM with the Linear kernel, SVM with the Radial kernel, and SVM with the kernel. Then I will pick the one that has best accuracy with the highest sensitivity.

I used random sampling to split the data as a train and test set. I used 399 units for the train set and the rest for the test set. (Table 2.3). I also created a train control group by using random sampling with repeated cross-validation.

	Train set	Test set
Number of units	399	170

Table2.3 Dimension of the train and test set

SVM with Linear kernel:

When I fit the model I need to look for the best accuracy depending on the cost. For the support vector machine with a linear kernel, I chose cost=0.21, since it gives me the highest accuracy and Kappa. (Table 2.4)

```
Support Vector Machines with Linear Kernel

399 samples
 30  predictor
```

```
 2  classes: 'B', 'M'

Pre-processing: centered (30), scaled (30)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 359, 359, 359, 360, 359, 359, ...
Resampling results across tuning parameters:

 C          Accuracy   Kappa
 0.0000000       NaN       NaN
 0.1052632  0.9740812  0.9433309
 0.2105263  0.9749145  0.9453302
 0.3157895  0.9732479  0.9418240
 0.4210526  0.9724145  0.9400716
 0.5263158  0.9724145  0.9400236
 0.6315789  0.9732479  0.9418728
 0.7368421  0.9715812  0.9382692
 0.8421053  0.9682051  0.9311577
 0.9473684  0.9665171  0.9276358
 1.0526316  0.9682051  0.9314336
 1.1578947  0.9673718  0.9299584
 1.2631579  0.9665385  0.9282496
 1.3684211  0.9657051  0.9264958
 1.4736842  0.9648718  0.9247882
 1.5789474  0.9657051  0.9266381
 1.6842105  0.9665385  0.9284879
 1.7894737  0.9648504  0.9249134
 1.8947368  0.9656838  0.9266659
 2.0000000  0.9648504  0.9249583

   Accuracy was used to select the optimal model using the largest value.
The final value used for the model was C = 0.2105263.
```

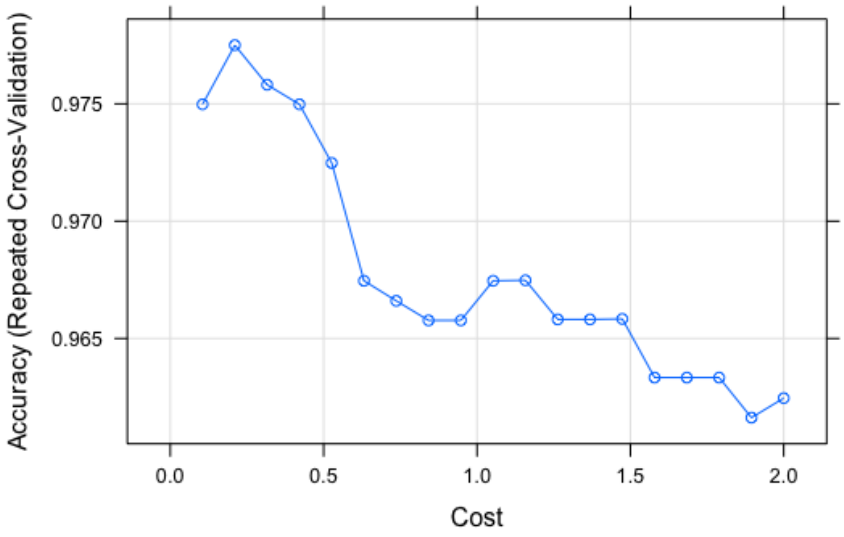

Table2.4 Choosing the most efficient cost for linear SVM

I will fit the model with cost= 0.21 to the test set to predict the status of "diagnosis" . The accuracy is 0.97, and the sensitivity is 0.972 for the SVM with linear kernel model. (Table 2.5)

```
Confusion Matrix and Statistics

test_pred_l      B   M
          B    104   2
          M      3  61

               Accuracy : 0.9706
                 95% CI : (0.9327, 0.9904)
    No Information Rate : 0.6294
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9372

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9720
            Specificity : 0.9683
         Pos Pred Value : 0.9811
         Neg Pred Value : 0.9531
             Prevalence : 0.6294
         Detection Rate : 0.6118
   Detection Prevalence : 0.6235
      Balanced Accuracy : 0.9701
```

'Positive' Class : B

Table 2.5 Fit the SVM linear model for test set

SVM with Radial kernel:

When I fit the model I need to look for the best accuracy depending on the cost. For the support vector machine with a radial kernel, I chose cost=2, since it gives me the highest accuracy and Kappa. (Table 2.6)

Support Vector Machines with Radial Basis Function Kernel

399 samples
30 predictor
2 classes: 'B', 'M'

Pre-processing: centered (30), scaled (30)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 359, 359, 359, 359, 359, 359, ...
Resampling results across tuning parameters:

C	Accuracy	Kappa
0.25	0.9548718	0.9033006
0.50	0.9699573	0.9359987
1.00	0.9749786	0.9463301
2.00	0.9799786	0.9570005
4.00	0.9783120	0.9534450
8.00	0.9791453	0.9553429
16.00	0.9766239	0.9500538
32.00	0.9732692	0.9429035
64.00	0.9732692	0.9429035
128.00	0.9732692	0.9429035

Tuning parameter 'sigma' was held constant at a value of 0.04192088
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were sigma = 0.04192088 and C = 2.

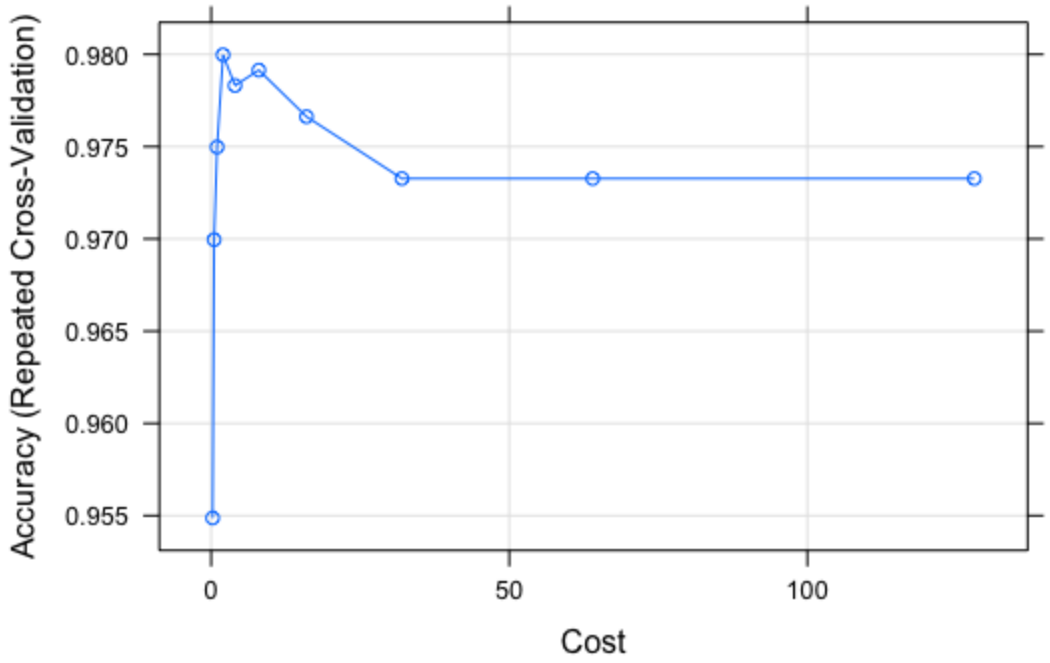

Table 2.6 Choosing the most efficient cost for SVM with radial kernel

I will fit the model with cost= 2 to the test set to predict the status of "diagnosis" . The accuracy is 0.96, and the sensitivity is 0.972 for the SVM with radial kernel model. (Table 2.7)

```
Confusion Matrix and Statistics

test_pred_r     B   M
          B   104   3
          M     3  60

          Accuracy : 0.9647
            95% CI : (0.9248, 0.9869)
   No Information Rate : 0.6294
   P-Value [Acc > NIR] : <2e-16

             Kappa : 0.9243

 Mcnemar's Test P-Value : 1

         Sensitivity : 0.9720
         Specificity : 0.9524
      Pos Pred Value : 0.9720
      Neg Pred Value : 0.9524
```

```
          Prevalence : 0.6294
      Detection Rate : 0.6118
Detection Prevalence : 0.6294
   Balanced Accuracy : 0.9622

    'Positive' Class : B
```

Table 2.7 Fit the SVM radial model for test set

SVM with poly kernel:

When I fit the model I need to look for the best accuracy depending on the cost. For the support vector machine with a poly kernel, I chose cost=0.5, since it gives me the highest accuracy and Kappa.The final values used for the model were degree = 2, scale = 0.01 and C = 0.5. (Table 2.8)

```
Support Vector Machines with Polynomial Kernel

399 samples
 30 predictor
  2 classes: 'B', 'M'

Pre-processing: centered (30), scaled (30)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 359, 359, 359, 359, 360, 359, ...
Resampling results across tuning parameters:

  degree  scale  C     Accuracy   Kappa
  1       0.001  0.25  0.7719444  0.4416331
  1       0.001  0.50  0.8964316  0.7636957
  1       0.010  0.25  0.9482265  0.8853159
  1       0.010  0.50  0.9641239  0.9213037
  2       0.001  0.25  0.8955983  0.7612999
  2       0.001  0.50  0.9398504  0.8656460
  2       0.010  0.25  0.9616026  0.9156491
  2       0.010  0.50  0.9741453  0.9438077

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were degree = 2, scale = 0.01 and C = 0.5.
```

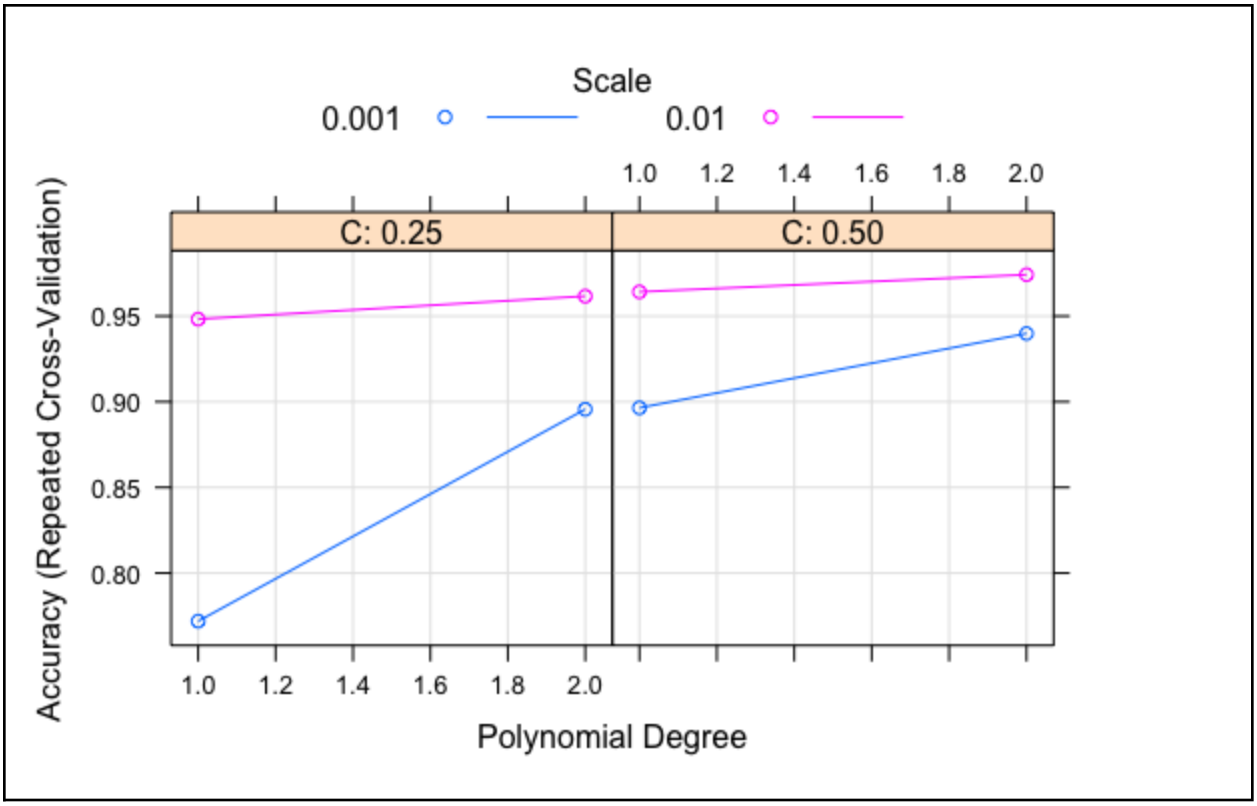

Table 2.8 Choosing the most efficient cost for SVM with poly kernel

I will fit the model with cost= 0.5 to the test set to predict the status of "diagnosis" . The accuracy is 0.97, and the sensitivity is 1 for the SVM with radial kernel model. (Table 2.9). When I see sensitivity is 1 I also checked the specificity that is 0.92.

```
Confusion Matrix and Statistics

test_pred_p      B   M
         B     107   5
         M       0  58

          Accuracy : 0.9706
            95% CI : (0.9327, 0.9904)
   No Information Rate : 0.6294
   P-Value [Acc > NIR] : < 2e-16

             Kappa : 0.9359

 Mcnemar's Test P-Value : 0.07364

        Sensitivity       : 1.0000
        Specificity       : 0.9206
     Pos Pred Value       : 0.9554
     Neg Pred Value       : 1.0000
```

```
          Prevalence        : 0.6294
      Detection Rate        : 0.6294
Detection Prevalence : 0.6588
   Balanced Accuracy : 0.9603

    'Positive' Class : B
```

Table 2.9 Fit the SVM poly kernel model for test set

When I compare all three SVM models I decided to go with support vector machine with linear kernel since it gives me the best accuracy and Kappa with the optimum sensitivity and specificity rate.

I will go on applying the Random Forest model to predict "diagnosis" status.

Random Forests:

I am going to use the standard RandomForest method to predict "diagnosis" status by using given predictors.

I split the data as train and test sets in the same ratio as in Support Vector Machine methods.

When I apply standard random forest to predict the status "diagnosis", OOB estimate of the error rate will be 3.77%. The classification error for each class will be very low (0.029, 0.051). (Table 2.10)

```
Call:
 randomForest(formula = diagnosis ~ ., data = traincancerb, mtry = 5,      importance = TRUE)

               Type of random forest  : classification
                     Number of trees       : 500
No. of variables tried at each split: 5

      OOB estimate of  error rate: 3.77%

Confusion matrix:
          B      M       class.error
B     237       7       0.02868852
M       8    146       0.05194805
```

Table 2.10 Standard Random Forest

Random forests method is also giving me an opportunity of variable importance. According to standard random forests " concave.points_worst", " concavepairs_worst", "area_worst", "radius_worst", "concave.points_mean", "perimeter_mean", and "concavity_mean" are the most important seven variables to designate the "diagnosis" status. (Table 2.1)

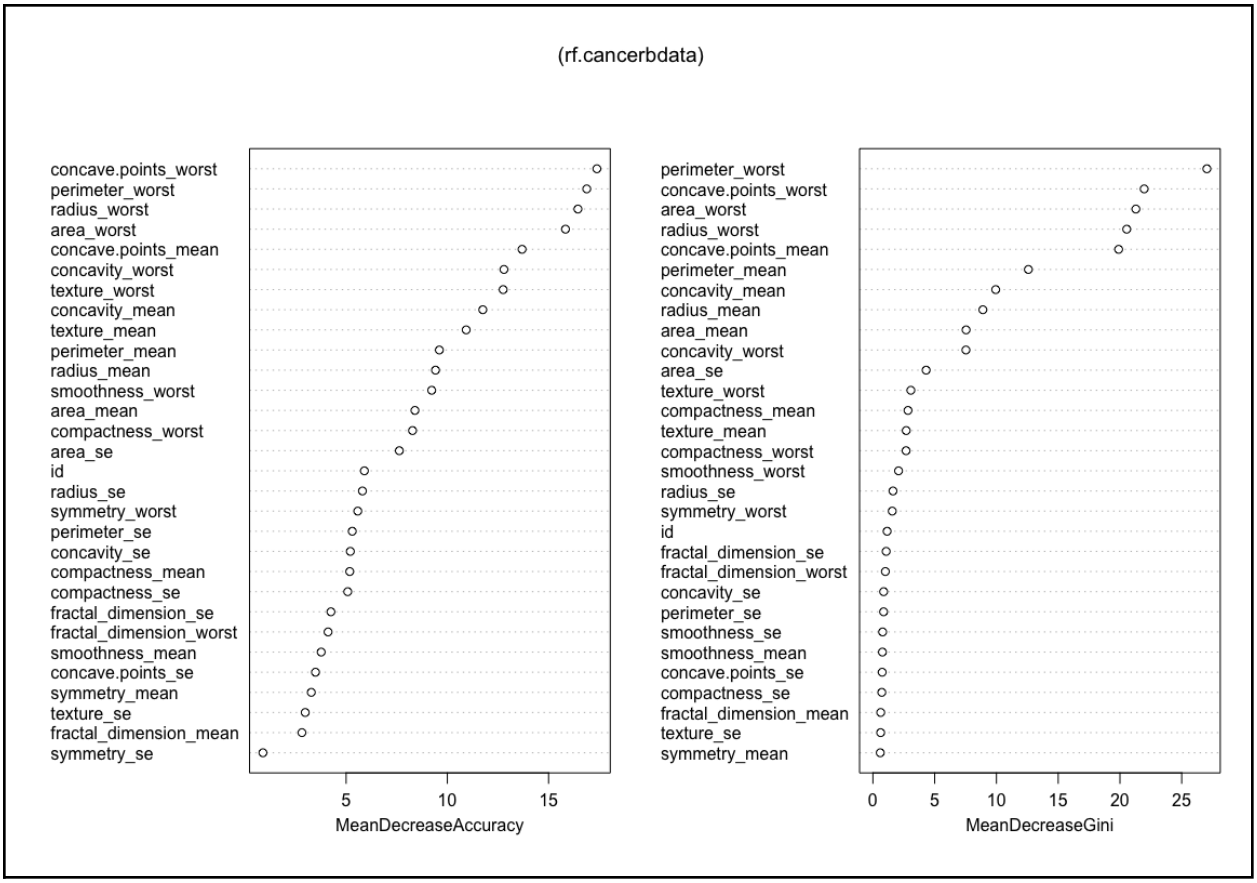

Table 2.11 Variable importance for standard Random Forests

My data set is an imbalance, I can also use new package RandomForestSRC that includes "brf" balanced random forests method for imbalanced data sets. BRF is classifying imbalance problems using a density-based approach. This results in a classifier that can be seen to be an example of a quantile classifier, which classifies samples based on whether the conditional probability of the minority class exceeds 0<q<1. Thus we need a specific q* classifier that maximizes TPR and TNR. q*- classifier can achieve near-zero risk in highly imbalanced data while optimizing the TPR and TNR.

```
                                Sample size: 398
                  Frequency of class labels: 244, 154
                                Number of trees: 500
                      Forest terminal node size: 1
                  Average no. of terminal nodes: 19.768
No. of variables tried at each split: 3
                             Total no. of variables: 31
                  Resampling used to grow trees: swr
Resample size used to grow trees: 308
                                        Analysis: RF-C
                                          Family: class
                                Splitting rule: gini *random*
```

```
        Number of random split points: 10
                      Imbalanced ratio: 1.5844
                (OOB) Brier score: 0.03364068
     (OOB) Normalized Brier score: 0.13456274
                        (OOB) AUC: 0.99276134
                    (OOB) PR-AUC: 0.99146625
                    (OOB) G-mean: 0.9653210
(OOB) Requested performance error: 0.03467892

Confusion matrix:

                predicted
observed         B     M      class.error
       B       235    9       0.0369
       M         5    149     0.0325

   (OOB) Misclassification rate: 0.03517588
```

Table 2.12 BRF (Balanced Random Forests)

Overall the error rate goes little bit down. But the more important thing is it is giving a better prediction for the minority class which means decreasing in minority class classification error is higher than increasing in majority class classification error compared to standard RF results. Even if the Random Forest BRF method is not creating a big difference, it is clear that it is making a better prediction for my imbalanced data set.

This method has also variable importance. " concave.points_worst", "perimeter_worst", "radius_worst"," texture_worst", "area_mean", "perimeter_mean", and "radius_mean" are the most important variables to designate the "diagnosis" status. (Table 2.13)

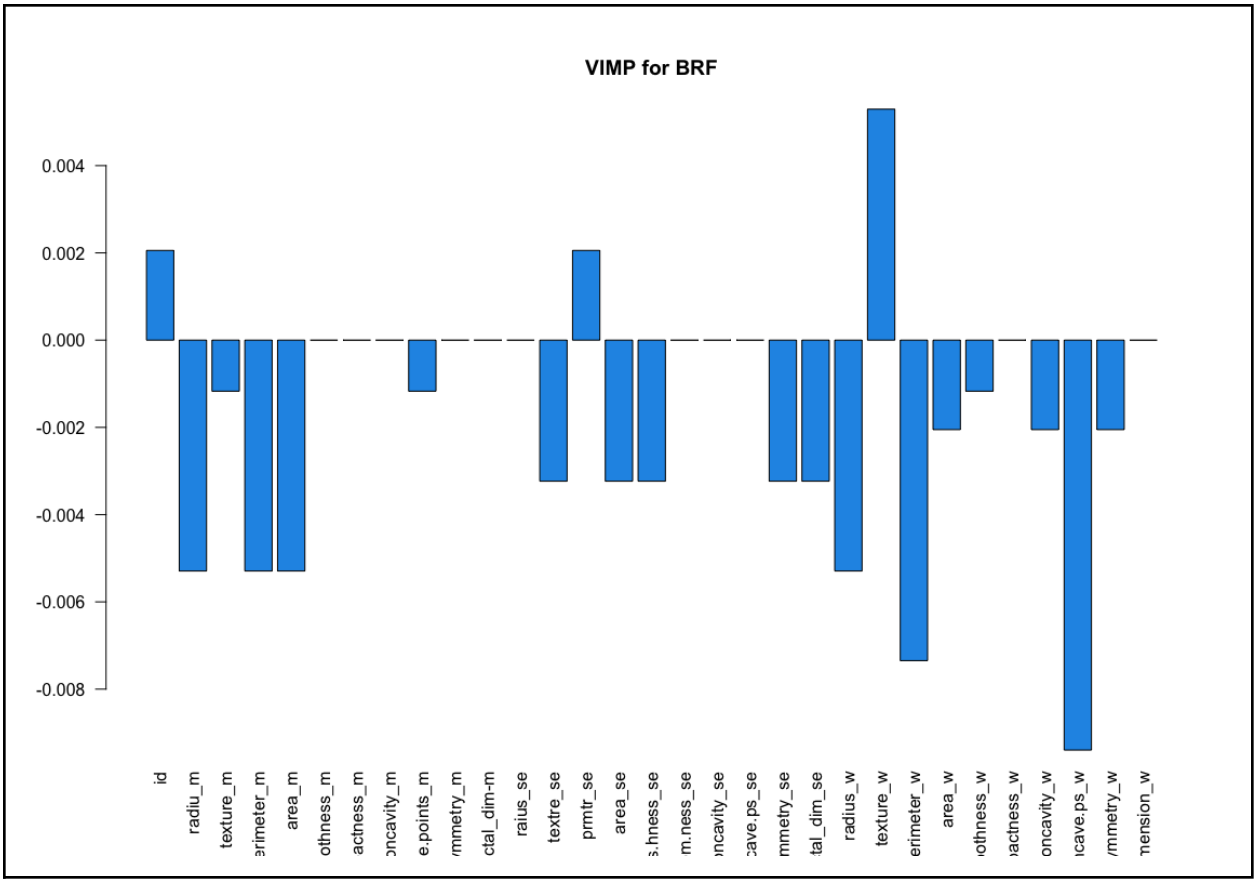

Table 2.13 Variable importance for BRF

Comparing methods(SVM, standard RF, BRF):

According to the results in Table 2.14 I can compare the results for SVM, standard RF, and BRF methods.

Method	Misclassification Error Rate	Majority class Misclassification error rate	Minority Class Misclassification Error Rate	Total misclassified units
SVM linear	0.0294	0.028	0.0317	5
Standard RF	0.0377	0.028	0.0519	15
BRF	0.0352	0.0369	0.0325	14

Table 2.14 Comparing SVM, standard RF, BRF methods

CONCLUSIONS

★ SVM method is giving me the smallest misclassification error rate. It has the least misclassified units.

- ★ For the imbalanced data sets, the misclassification rate for the minority class is as important as the total accuracy rate. SVM also has the least minority class misclassification error rate.
- ★ . If I need variable importance I would say BRF is giving me better prediction than standard RF.

As conclusion, SVM method is giving me the best prediction. In the case of variable importance is not needed, I would prefer SVM method to predict "diagnosis" status.

REFERENCES

1) https://www.kaggle.com/yasserh/breast-cancer-dataset/download
2) *Breast cancer statistics: How common is breast cancer?* American Cancer Society. (n.d.). Retrieved March 26, 2022, from https://www.cancer.org/cancer/breast-cancer/about/how-common-is-breast-cancer.html
3) https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/1472-6947-10-16
4) https://towardsdatascience.com/svm-support-vector-machine-for-classification-710a009f6873
5) *Random Forest: Introduction to random forest algorithm*. Analytics Vidhya. (2021, June 24). Retrieved March 26, 2022, from https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/
6) Chen, L. (2019, January 7). *Support Vector Machine - simply explained*. Medium. Retrieved March 26, 2022, from https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496
7) O'Brien, R., & Ishwaran, H. (2019, January 29). *A random forests quantile classifier for class imbalanced data*. Pattern Recognition. Retrieved March 26, 2022, from https://www.sciencedirect.com/science/article/abs/pii/S0031320319300536
8) James, G., Witten, D., Hastie, T. J., & Tibshirani, R. J. (2013). *An introduction to statistical learning: With applications in R*. Springer.
9) *SVM: Support Vector Machines*. RDocumentation. (n.d.). Retrieved January 14, 2022, from https://www.rdocumentation.org/packages/e1071/versions/1.7-9/topics/svm
10) James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). *An introduction to statistical learning: With applications in R*. Springer.

APPENDIX

```
#################################
##### Breast Cancer Dataset#####
#################################
cancerb<-read.csv(file="breast-cancer.csv", header=TRUE)
head(cancerb)
dim(cancerb)          ### Breast cancer data set 569 units 32 variables
table(cancerb$diagnosis)
names(cancerb)
```

```
#diagnosis is my response variable Malignant :( or Benign :)
#missing value?
anyNA(cancerb)                  #no missing value
str(cancerb)                    #diagnosis is target variable M or B
cancerb$id<-NULL                #id variable is not needed
cancerb$diagnosis<-as.factor(cancerb$diagnosis)       # for classification
##################
library(ggplot2)##
##################
#visualization of imbalanced data set
ggplot(cancerb, aes(x=factor(diagnosis)))+
  geom_bar(stat="count", width=0.7, fill="steelblue")+ theme_minimal() #350 units B, 200 units M data
imbalanced!
###################################
##### SUPPORT VECTOR MACHINE #####
###################################
##### "caret" package needed for SVM
install.packages("caret")
#################
library(caret) ##          package to apply SVM
#################
cancerb[["diagnosis"]]<-factor(cancerb[["diagnosis"]])   #in caret package use response as factor!
summary(cancerb)
#random sampling set train and test set
set.seed(583)
traincreate<-createDataPartition(y=cancerb$diagnosis, p=0.7,list=FALSE)
training<-cancerb[traincreate,]        #train set
testing<-cancerb[-traincreate,]        #test set
dim(training) ; dim(testing)           #check dimentions of sets to make sure split

traincont<-trainControl(method="repeatedcv",number=10,repeats=3) #random sampling, repeated cross
validation
###### SVM use Linear classifier
svm_linear<-train(diagnosis~., data=training,method="svmLinear",trControl=traincont,
                  preProcess=c("center","scale"), tuneLength=10)    #fit model for train set
svm_linear
svm_linear2<-train(diagnosis~., data=training,method="svmLinear",trControl=traincont,
                  preProcess=c("center","scale"), tuneGrid = expand.grid(C=seq(0,2,length=20)))
svm_linear2              # looking for most accuracy depending on cost
plot(svm_linear2)
svm_linear2$bestTune      #cost 0.21 is best tune
#test for the linear model with best tune, cost=0.21
test_pred_l<- predict(svm_linear2, newdata=testing)   # test the model with best tune model
test_pred_l
```

```
CM1<- confusionMatrix(table(test_pred_l,testing$diagnosis))
CM1                    # accuracy 0.97

##### SVM Radial
svm_radial <-train(diagnosis~., data=training,method="svmRadial",trControl=traincont,
                preProcess=c("center","scale"), tuneLength=10)
svm_radial
plot(svm_radial)
svm_radial$bestTune          # cost=2 is best accuracy!
#test for radial model with best tune, cost =2
test_pred_r<- predict(svm_radial, newdata=testing)
test_pred_r
CM2<-confusionMatrix(table(test_pred_r,testing$diagnosis))
CM2                          # accuracy 0.96.47

##### SVM with Poly kernel
svm_poly <-train(diagnosis~., data=training,method="svmPoly",trControl=traincont,
                 preProcess=c("center","scale"), tuneLength=2)
svm_poly
plot(svm_poly)
svm_poly$bestTune          # cost= 0.5 is best accuracy!
#test for the poly model with best tune cost =0.5
test_pred_p<- predict(svm_poly, newdata=testing)
test_pred_p
CM3<-confusionMatrix(table(test_pred_p,testing$diagnosis))
CM3                          # accuracy 0.97

###############################################
##### RANDOM FOREST & RANDOM FOREST SRC #####
###############################################
cancerb<-read.csv(file="breast-cancer.csv", header=TRUE)
dim(cancerb)
str(cancerb)
cancerb$diagnosis<- as.factor(cancerb$diagnosis)
##### Split data terain and test sets
set.seed(583)
train = sample(1:nrow(cancerb), nrow(cancerb)*7/10)
traincancerb<-cancerb[train,]
testingcancerb<-cancerb[-train,]
dim(traincancerb) ; dim(testingcancerb)   #check dimesntions
########################
library(randomForest) ##
########################
###### RF tree
```

```
set.seed(1)
rf.cancerbdata = randomForest(diagnosis~., data=traincancerb,mtry=5, importance=TRUE)  #fit model
for train set
print(rf.cancerbdata)

par(mfrow=c(1,1))
plot(rf.cancerbdata)    # OOB estimate of error 5.99%

# green M black OOB red B

yhat.rf=predict(rf.cancerbdata, newdata=testingcancerb)
mean(yhat.rf != cancerb$diagnosis[-train])

#Variable importance
importance(rf.cancerbdata)
varImpPlot((rf.cancerbdata))

# RandomForestSRC for imbalanced dataset method="BRF"
#######################################
library(randomForestSRC)          ##
install.packages("randomForestSRC") ##
#######################################
set.seed(123)
c.brf<-imbalanced(diagnosis~., data=traincancerb, ntree=500,mtry=3, method=c("brf"),
                  importance="permute", pref.type="g.mean")
print(c.brf)                    # OOB misclassification rate is 3.952

pred.c.brf= predict(object=c.brf, newdata = testingcancerb)

### variable importance for BRF
vpcb<- c.brf$importance[,1]
print(vpcb)
plot(vpcb)
nms <- c("id","radiu_m","texture_m","perimeter_m","area_m","smoothness_m","compactness_m",

"concavity_m","concave.points_m","symmetry_m","fractal_dim-m","raius_se","textre_se","prmtr_se",

"area_se","s.hness_se","com.ness_se","concavity_se","concave.ps_se","symmetry_se","fractal_dim_se",

"radius_w","texture_w","perimeter_w","area_w","smoothness_w","compactness_w","concavity_w",
        "concave.ps_w","symmetry_w","fractal_dimension_w")
barplot(vpcb,main="VIMP for BRF",col=4,horiz=TRUE,las=2, names.arg=nms)
barplot(vpcb,main="VIMP for BRF",col=4,horiz=FALSE,las=2, names.arg=nms)
```