17.05.2020

Big Data final report
Oskar Ratus
75279

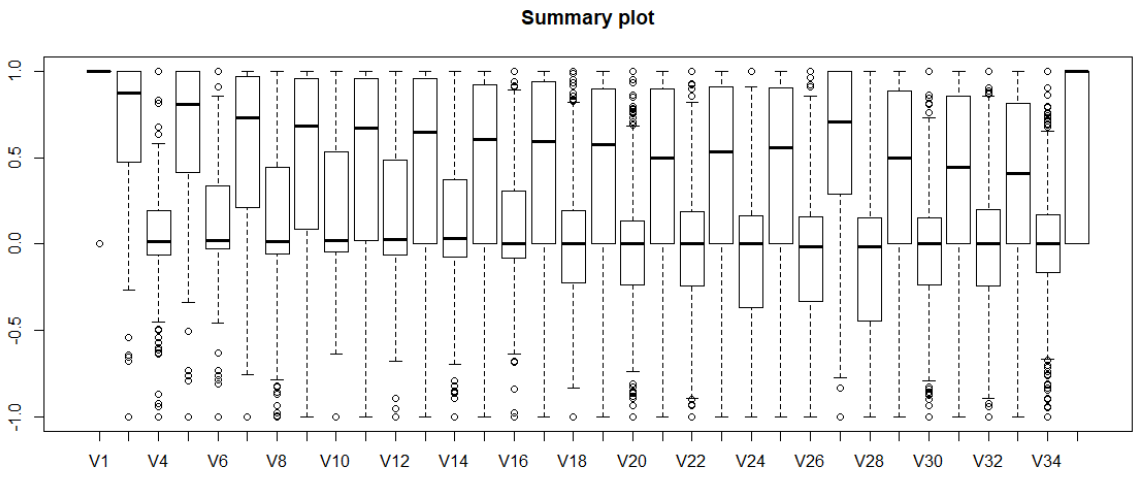
1. Introduction

The dataset presented in this project is called Johns Hopkins University Ionosphere database and can be reached through UCI Machine Learning Repository. This radar data was collected by a system in Goose Bay, Labrador in 1989. The first use of it was to classify radar returns from inosphere using neural networks (Sigillito, V. G., Wing, S. P., Hutton, L. V., \& Baker, K. B. 1989). David Aha briefly investigated this database. He found that nearest neighbour attains an accuracy of 92.1%, that Ross Quinlan's C4 algorithm attains 94.0% (no windowing), and that IB3 (Aha \& Kibler, IJCAI-1989) attained 96.7% (parameter settings: 70% and 80% for acceptance and dropping respectively). Further research will be made to find even better algorithm for this job. In this paper it will be tested weather CART algorithm implemented in classification tree, boosting and random forests can reach similar performance.

2. Descriptive stats

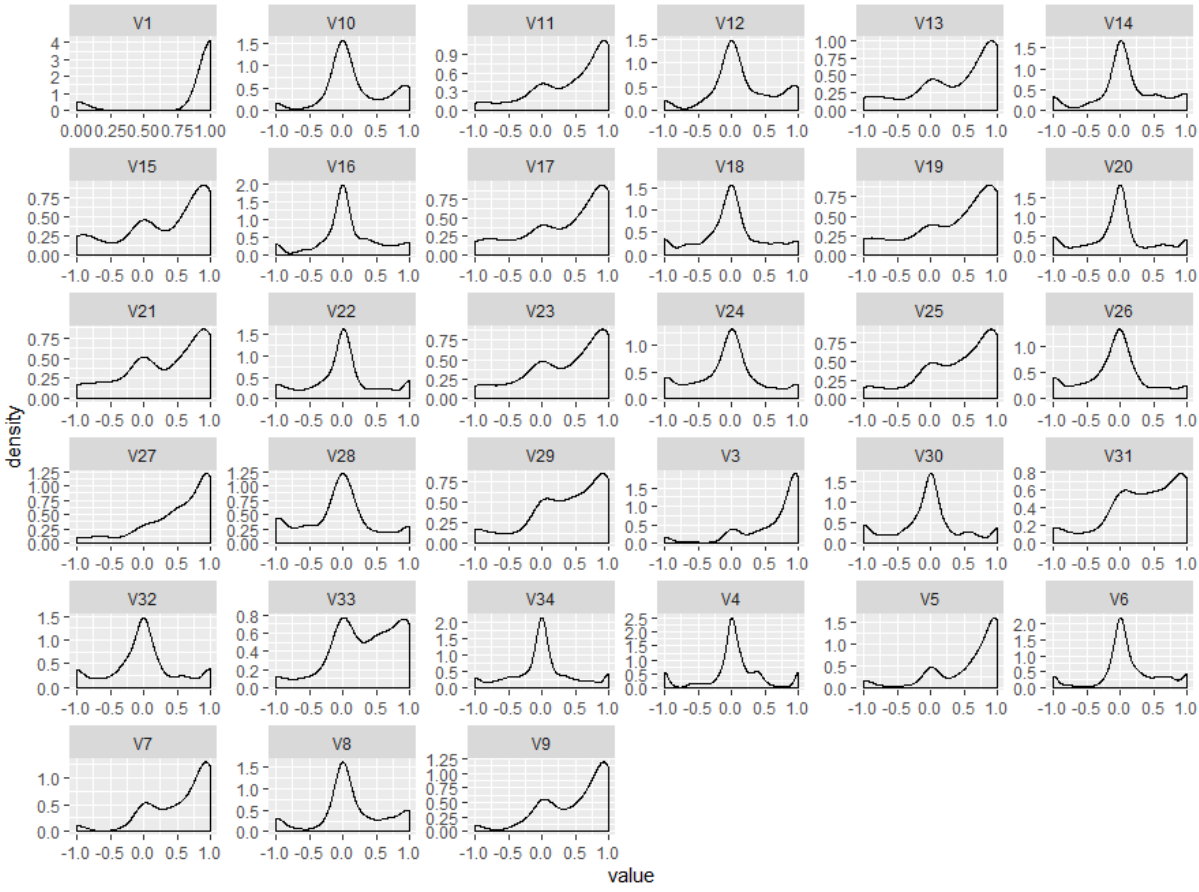
This radar data was collected by a system in Goose Bay, Labrador. System consists of a phased array of 16 high-frequency antennas with a total transmitted power on the order of 6.4 kilowatts. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not, their signals pass through the ionosphere. The total number of observations this dataset contains is 351 with 34 attributes each. All of 34 predictor attributes are continuous. The 35th attribute is the target value either “good” or “bad” transformed to binary values in pre-processing. There are no missing values in the dataset.

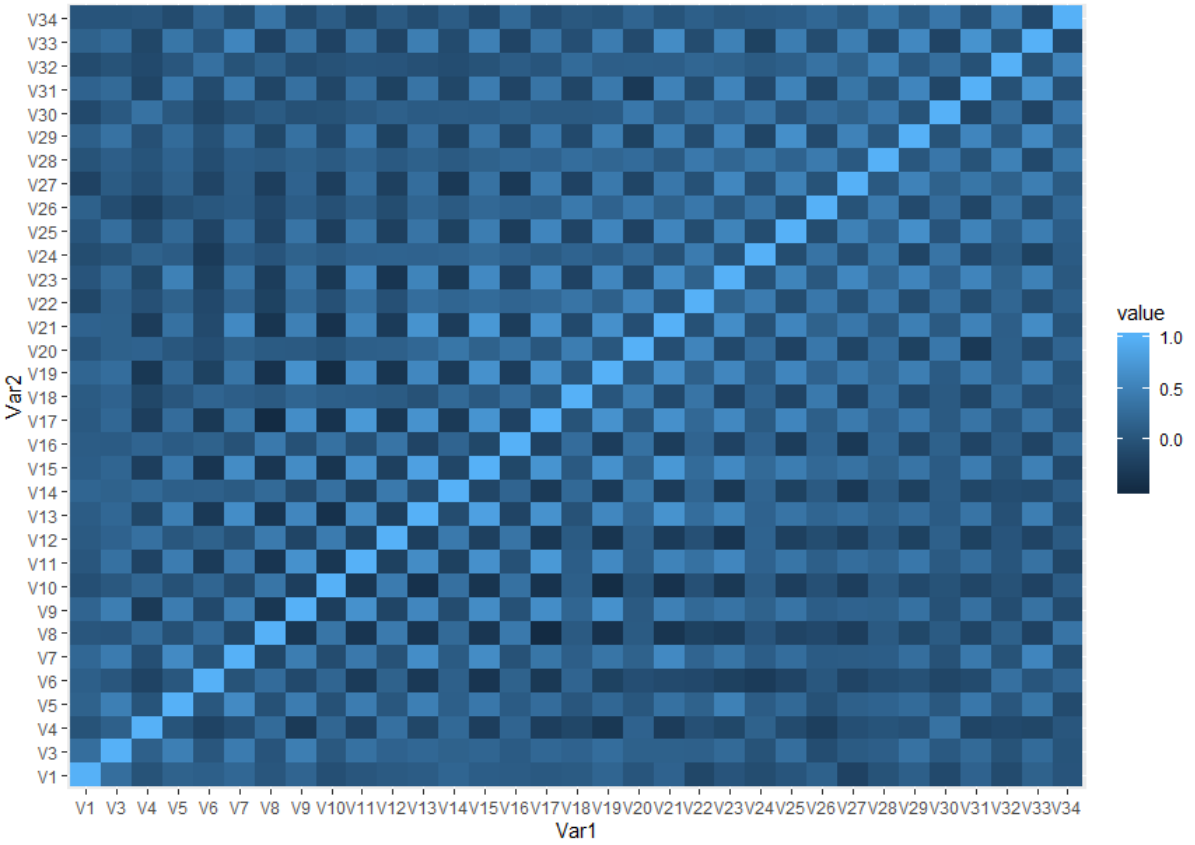
```
"Min.   :0.000  "
 "1st Qu.:0.000  "
 "Median :1.000  "
 "Mean   :0.641  "
 "3rd Qu.:1.000  "
 "Max.   :1.000  "
```



As it can be seen in the table above the dataset is slightly skewed with the mean value of the target attribute 0.64. This means 225 “good” returns and 126 “bad’ returns from the inosphere. The rest of

the data is continuous values varying from -1 to 1. The graph shows that the data divides into two groups one having its mean around 0 and the second one with mean values around 0.5.





Besides that, it is hard to perform any summary or graphical analysis that will carry any information that would be easily interpretable by human being without inside knowledge of that problem. No significant correlation or dependencies can be observed in the data.

In the next step an attempt will be made to fit several models such as classification tress and neural networks to the above dataset. The evaluation of the model performance will be made with the use of confusion matrix and accuracy score.

		Reference	
		N = Bad	P = Good
Predicted	N = Bad	TN	FN
	P = Good	FP	TP

Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class, where: P = positive; N = Negative; TP = True Positive; FP = False Positive; TN = True Negative; FN = False Negative. The Accuracy score is calculated with the equation given below and means the ratio of correctly predicted values to all predictions.

$$ACC = \frac{TP + FN}{P + N}$$

Having in mind that any further processing of the data can be made with “good” returns form the inosphere the main goal is to select model with the lowest number of false predictions, this means the Accuracy score for this model will be the highest.

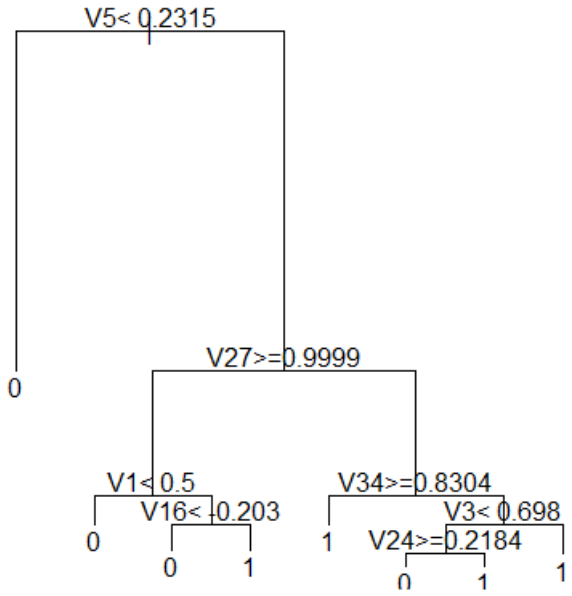
The data was randomly spitted into training and testing set with the ratio of 80% training and 20% test data.

3. Tree based models

Recursive partitioning is a fundamental tool in data mining. It helps us explore the structure of a set of data, while developing easy to visualize decision rules for predicting a categorical (classification tree). While the dataset used in this paper is hard in interpretation the outcome of tree models can help to find optimal hyper-parameters in neural net.

1. CART model

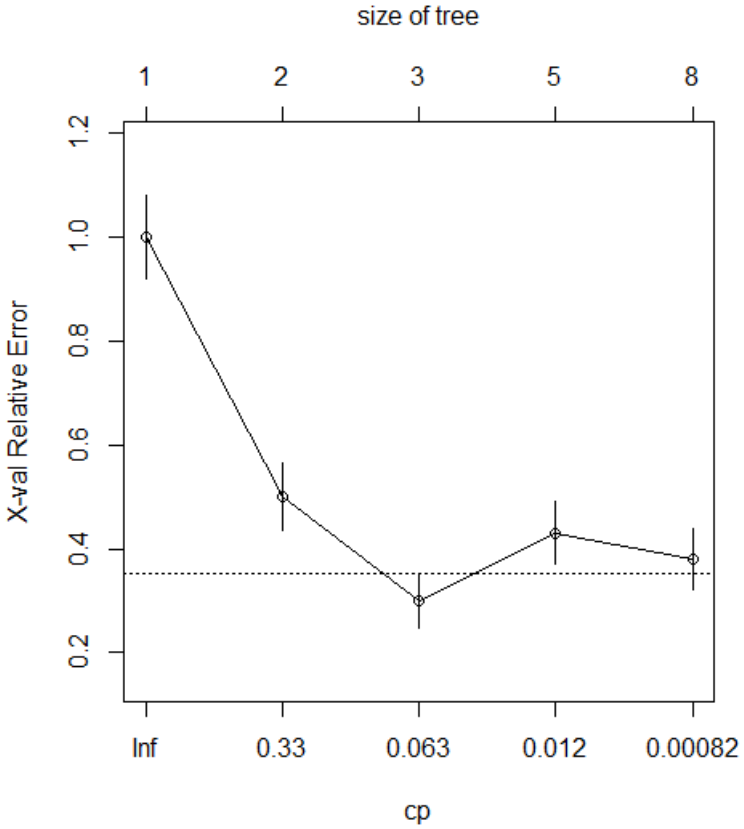
Classification And Regression Models (CART) can be generated with the use of *rpart* package. This package is an implementation of most of the functionality of the 1984 book by Breiman, Friedman, Olshen and Stone. In classification trees, the split point is defined so that the population in sub-partitions are pure as much as possible. Two measures of purity are generally used, including the Gini index and the entropy (or information gain). Trees generated by this package needs to be pruned manually. As a result, this package gives a lot of control to user.



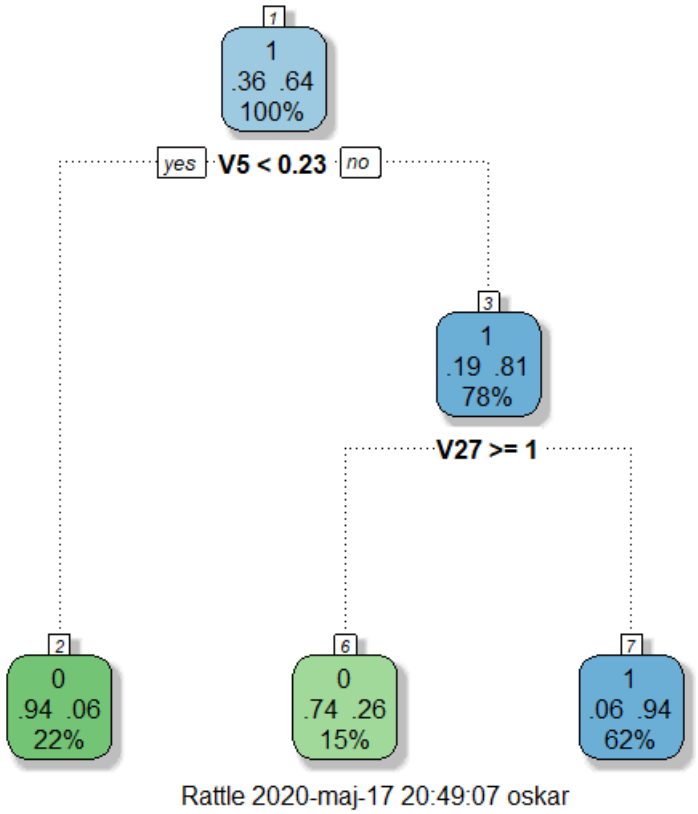

```
$ConfusionMatrix
          Reference
Prediction  0  1
         0 17  1
         1  9 44

$Accuracy
[1] 0.8591549
```

The initial tree fully tree is presented above. However, this tree is very complex and is likely to overfit the data which may lead to poor predictions on test data (Accuracy = 86%). In the next step the tree will be pruned to prevent overfitting the data. The goal here is to see if a smaller subtree can give us comparable results to the fully grown tree. If yes, we should go for the simpler tree because it

reduces the likelihood of overfitting. With the use of *plotcp()* the set of possible cost-complexity pruning of a tree can be tested. A good choice of cp for pruning is often the leftmost value for which the mean lies below the horizontal line, which in this case is 0.063 as presented below.



After pruning, the tree looked like on the graph below, it can be clearly seen that the tree is simpler and less complex with only root one node. Besides simpler model misclassified values can be found in each leaf. The accuracy score is on the level of 91%, so it is a big improvement on test data when compared to 85% of fully grown tree.



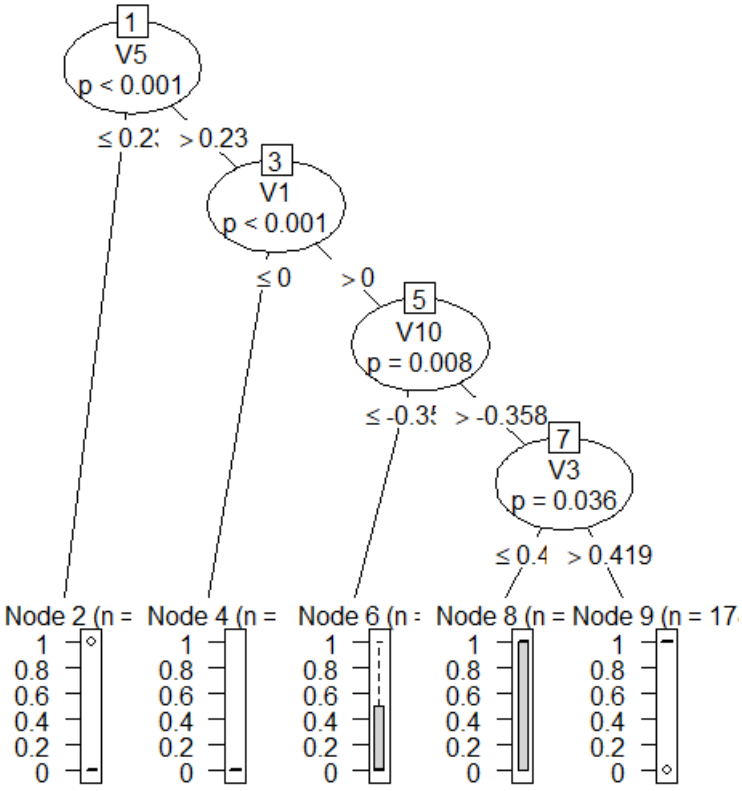

```
n= 280

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 280 100 1 (0.35714286 0.64285714)
  2) v5< 0.23154 63   4 0 (0.93650794 0.06349206) *
  3) v5>=0.23154 217  41 1 (0.18894009 0.81105991)
    6) v27>=0.999945 42  11 0 (0.73809524 0.26190476) *
    7) v27< 0.999945 175  10 1 (0.05714286 0.94285714) *


$ConfusionMatrix
          Reference
Prediction  0  1
         0 22  2
         1  4 43

$Accuracy
[1] 0.915493
```

Changing splitting hyper-parameter form Gini index to information ratio did not make a change in accuracy of the model, so this step is omitted in this paper.

2. Conditional inference tree

Conditional Inference trees, also referred as unbiased recursive partitioning is a non-parametric class of decision trees that uses a statistical theory (selection by permutation-based significance tests) in order to select variables instead of selecting the variable that maximizes an information measure (Gini coefficient or Information Gain). This can limit overfitting compared to the classical rpart

algorithm. At each splitting step, the algorithm stops if there is no dependence between predictor variables and the outcome variable. Otherwise the variable that is the most associated to the outcome is selected for splitting. *Ctree()* function in party package will be used to estimate this model. It is worth noting that this implementation does not need pruning of the tree.




```
        Conditional inference tree with 5 terminal nodes

Response:  target
Inputs:  V1, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13, V14, V15, V16,
V17, V18, V19, V20, V21, V22, V23, V24, V25, V26, V27, V28, V29, V30, V31,
V32, V33, V34
Number of observations:  280

1) V5 <= 0.23; criterion = 1, statistic = 73.845
  2)*  weights = 63
1) V5 > 0.23
  3) V1 <= 0; criterion = 1, statistic = 78.814
    4)*  weights = 17
  3) V1 > 0
    5) V10 <= -0.35818; criterion = 0.992, statistic = 13.541
      6)*  weights = 7
    5) V10 > -0.35818
      7) V3 <= 0.41932; criterion = 0.964, statistic = 10.651
        8)*  weights = 15
      7) V3 > 0.41932
        9)*  weights = 178


$ConfusionMatrix
          Reference
Prediction  0  1
         0 19  3
         1  7 42

$Accuracy
```

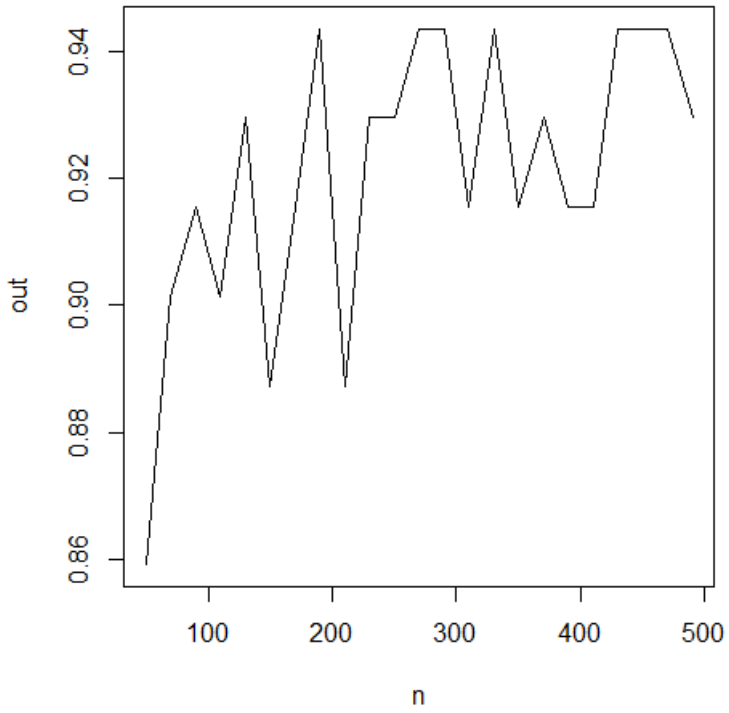
```
[1] 0.8591549
```

Tree produced by ctree algorithm is very different and far more complex than the previous one. Except root node with V5 variable which is the same as in CART algorithm the rest is different. As stated, before having more complex tree leads easily to overfitting the data and this can be the case in above tree. The accuracy score which equals to 86% is lower than pruned rpart tree. As a rule of thumb, the less complex tree the better. Some adjustments were made to improve overall performance of the model. The range of *mincriterion* (which is 1 – p-value that must be exceeded in order to implement a split) was tested, but the default value was the best in terms of accuracy of the model. *Ctree()* function allows to change test statistic that is used to evaluate the split, but both of the available statistics performed equally good on this dataset generating similar trees.

3. Bagging

Bagging in other words bootstrap aggregating is a meta algorithm for classification trees to improve the stability and accuracy. The method of bootstrapping the data and creating new subsets reduces variance and helps to avoid overfitting. Bagging was first suggested by Breiman (1996a, 1998. For the sue of this paper b*agging()* function form ipred package will be used to generate trees for this model. This package implements rpart trees described earlier on in this paper.

At first 23 different bagging models were build to explore possible accuracy range which can be accessed through this method with different sample size. The results are presented below.



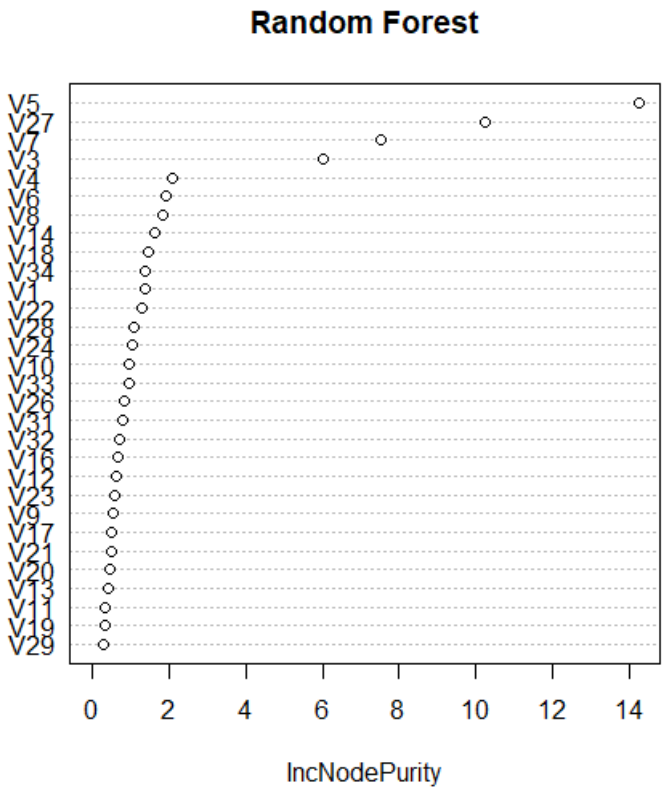
As it can be seen bagging method peaked its performance at around sample size of 300 and 500 with 94% of accuracy and the lowest slightly below 200 and slightly above with the score of 88%. The worst result is better than ctree algorithm and it peaks above the score of single pruned rpart tree.

```
$ConfusionMatrix
          Reference
Prediction  0  1
         0 22  0
         1  4 45

$Accuracy
[1] 0.943662
```

The final model was build with 420 of bootstrap replications which seemed to perform best and reached 94% of Accuracy on test data.

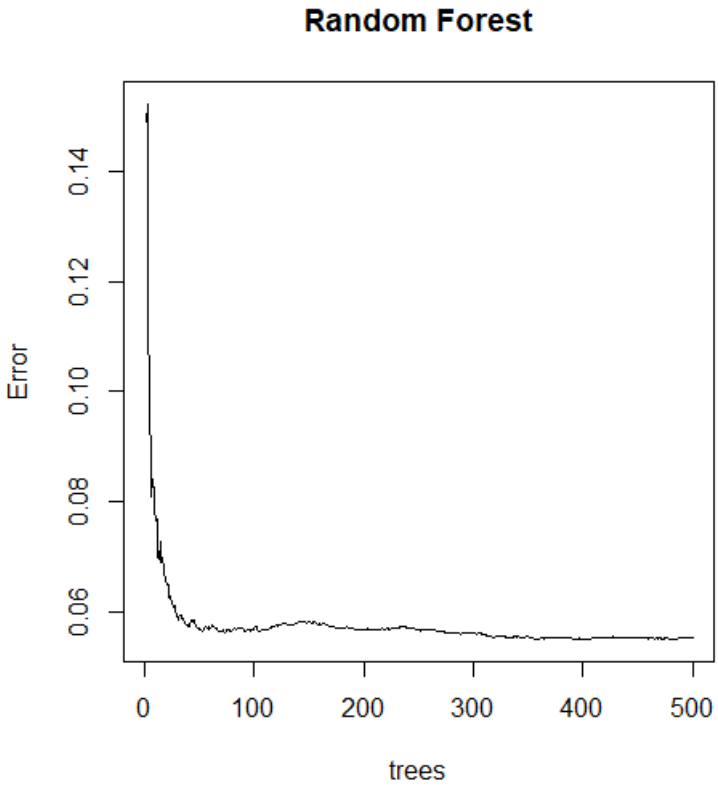
4. Random Forest

In this part random forests algorithm will be introduced. The main idea behind that is to aggregate the predictions made by multiple decision trees of varying depth. Every decision tree in the forest is trained on a subset of the dataset called the bootstrapped dataset.

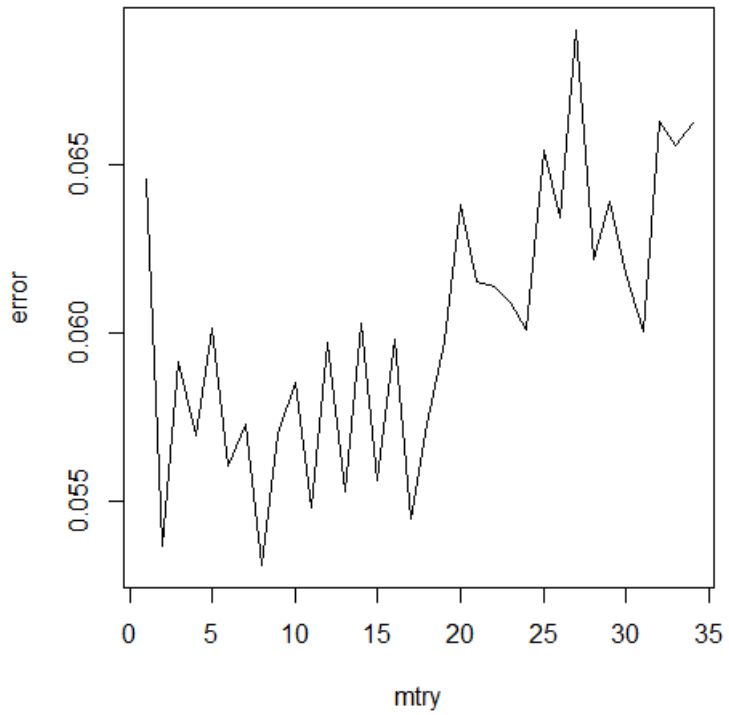


The first graph shows a dotchart of variable importance as measured by a Random Forest. The most important variable is V5 and V27 which was also mentioned in previous trees.

The next step is to find optimal forest size. This was evaluated by calculating MSE error, the reason for that was that this value is calculated automatically by *randomForest().* The curve drops fast till 100 as the size of the forest grows, then a slight upward trend cab be observed around 150. After that MSE slowly converges to its minimal value.



At this point optimal value of *mtry* parameter was calculated. Mtry means the number of variables randomly sampled as candidates at each split. The graph below is very rough, but for the purpose of this paper it can be assumed that the lowest values can be found between 7 and 17.



Having optimal parameters values chosen and passed to the final model it performed as presented below. The accuracy on the level of 92% and only 5 misclassified values is second best in this paper.

```
$ConfusionMatrix
          Reference
Prediction  0  1
         0 22  3
         1  2 44

$Accuracy
[1] 0.9295775
```

4. Summary

Model	Accuracy
Rpart (full)	0.8591549
Rpart (pruned)	0.915493
Ctree	0.8591549
Bagging	0.943662
Random Forest	0.9295775

As the table above shows, the best method turned out to be bagging which reached 94% of accuracy, second best was another aggregating method with the score of 92% of correct predictions. Bootstrapping the data and averaging the outcome reduce variance and overfitting tendency of the models so it is not surprising that those models reached the highest score. It is also worth mentioning that adjusted rpart model performed only slightly worse than aggregation methods and the difference of 1- 2 % can be caused by one misclassification more.