Statistical Learning, Homework 2

Virginia Leombruni 247160

2024-03-05

Introduction

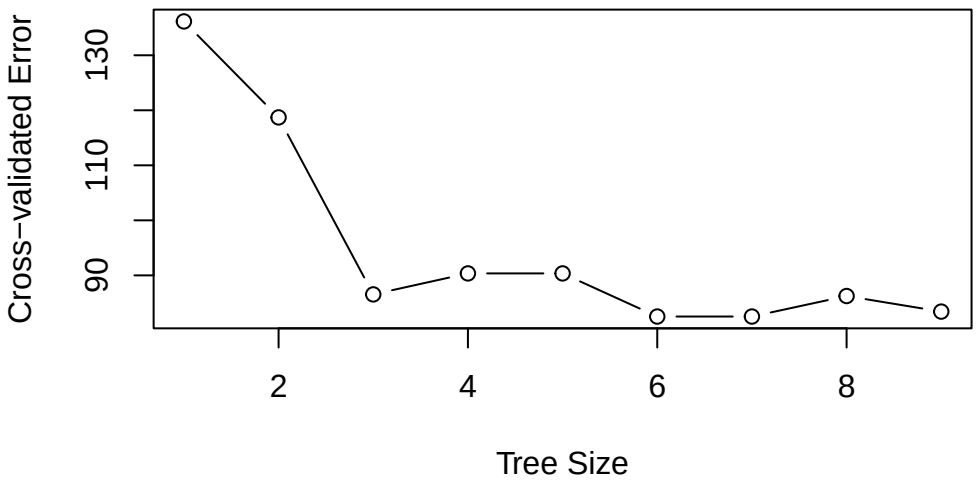
The data in this work are data-related to study the association between the level of prostate-specific antigen (lpsa, in ng/ml and log scale) and several clinical measures, measured in 97 men who were about to receive radical prostatectomy. The explanatory variables are:

- `lcavol`: log(data volume in cm3)
- `lweight`: log(prostate weight in g)
- `age` in years
- `lbph`: log(amount of benign prostatic hyperplasia in cm2)
- `svi`: seminal vesicle invasion (1 = yes, 0 = no)
- `lcp`: log(capsular penetration in cm)
- `gleason`: Gleason score for prostate data (6,7,8,9)
- `pgg45`: percentage of Gleason scores 4 or 5, recorded over their visit history before their final current Gleason score

```
head(data)
str(data)
sum(is.na(data))
data <- (na.omit(data))
```

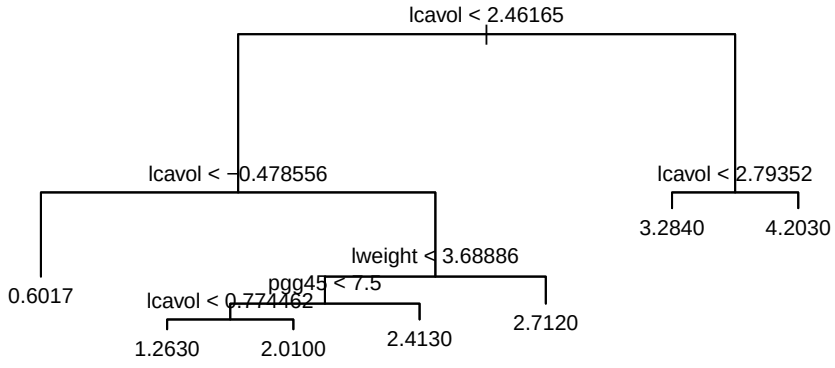
Decision tree

```
# Fit decision tree using cross-validation
tree_cv <- cv.tree(tree(lpsa ~ ., data = data), FUN = prune.tree)
```

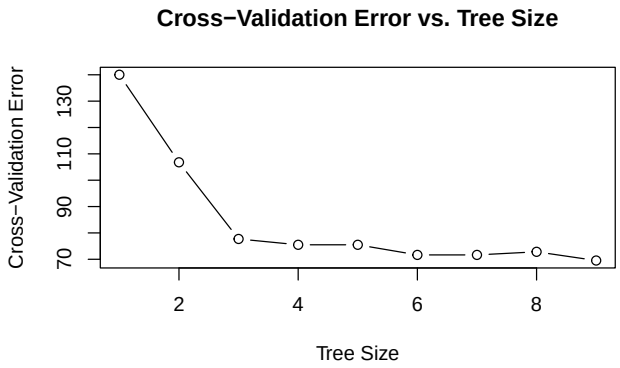


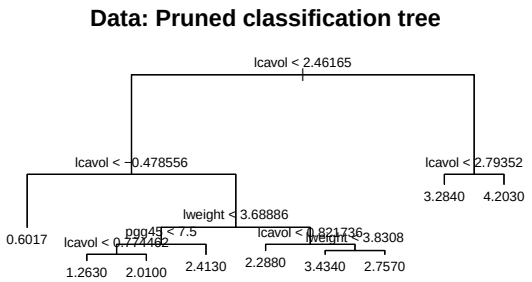
Get the optimal tree size

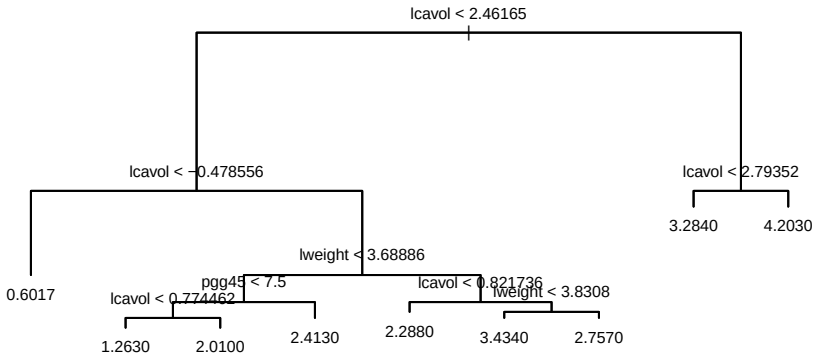
Data: Pruned classification tree with the optimal size



The constructed regression tree model has a relatively low residual mean deviance, indicating a good fit for the data. However, further analysis and validation may be needed to accurately assess the model's predictive performance.

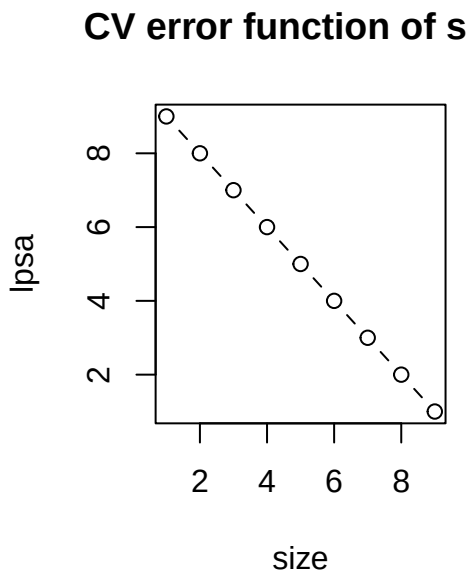


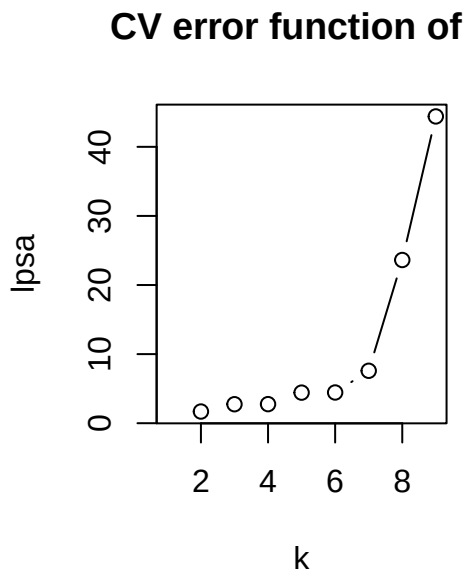





```
cv.data <- cv.tree(object = tree.data)
# names(cv.data) cv.data
```

This output provides insights into the performance and complexity of the decision tree model at different levels of pruning. It is possible to use this information to select the optimal tree size based on model fit and complexity.



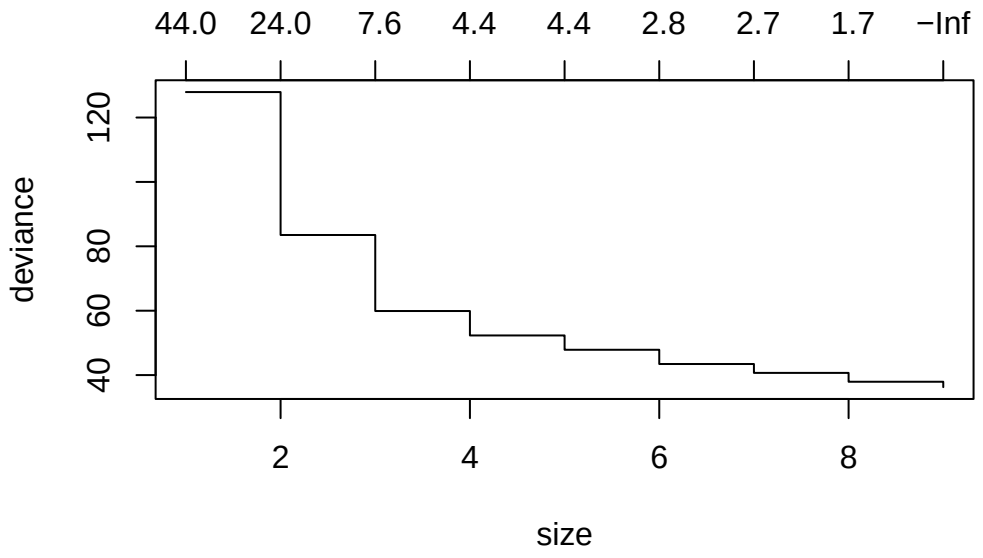



```
opt.size <- cv.data$size[which.min(cv.data$dev)]
```

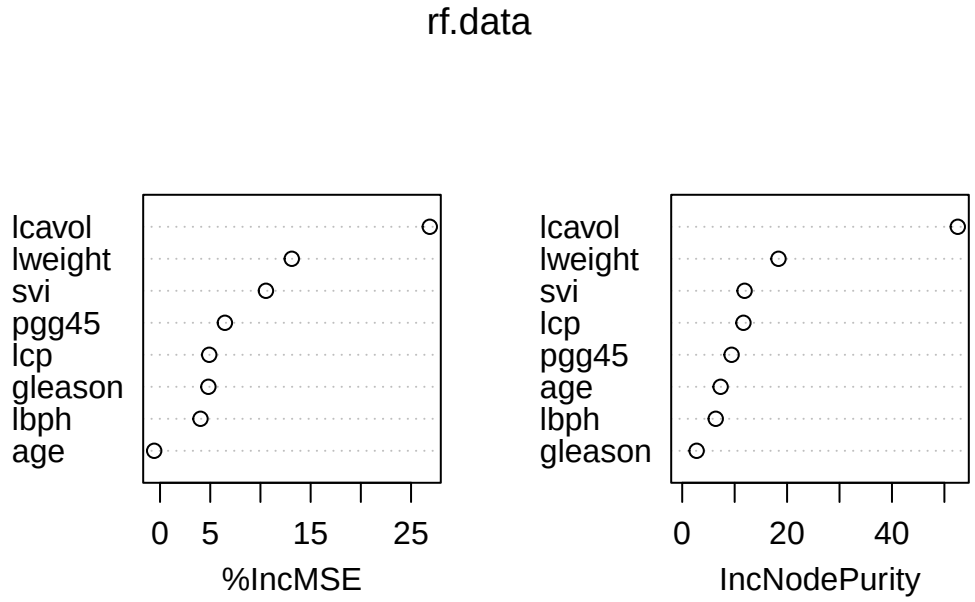
```
[1] "The optimal size: 3"
```

The optimal size selected during cross-validation is 3. This means that the decision tree with 3 terminal nodes (leaves) had the best performance based on the chosen evaluation metric (in this case, likely deviance).

```
# Prune the decision tree using the 'deviance' method
prune.data <- prune.tree(tree.data, method = "deviance")
```



Random Forest



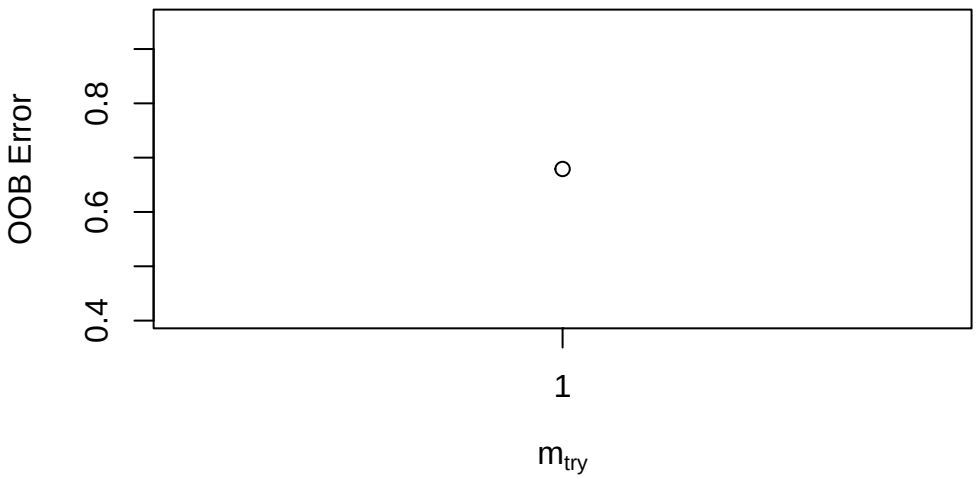

```
rf <- randomForest(lpsa ~ ., data = data, importance = TRUE)
oob <- tail(rf$mse, 1)
```

```
plot(oob)
#plot(rf, main = "Random Forest")
```

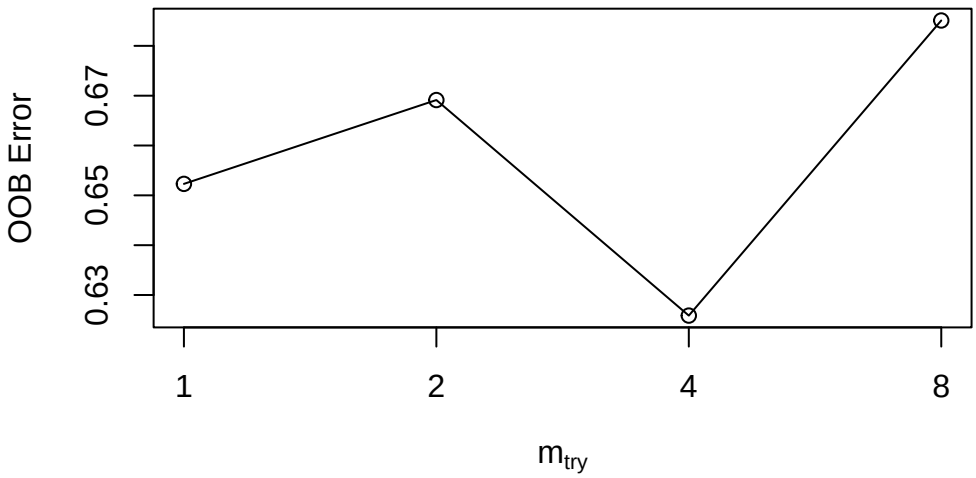
The value of the out-of-bag (OOB) error, is approximately 0.61. The OOB error is a measure of the prediction error of a random forest model. It is estimated using the observations that were not included in the bootstrap samples used to train each individual tree in the random forest.

Therefore, based on the given output, the best value of **mtry** is 4, which resulted in the lowest OOB error among the values considered.

```
mtry = 1  OOB error = 0.6791526
Searching left ...
Searching right ...
```




```
mtry = 2  OOB error = 0.6691261
Searching left ...
mtry = 1    OOB error = 0.6523007
0.02514531 0.05
Searching right ...
mtry = 4    OOB error = 0.6258694
0.06464648 0.05
mtry = 8    OOB error = 0.685091
-0.09462283 0.05
```



2. Random Forest

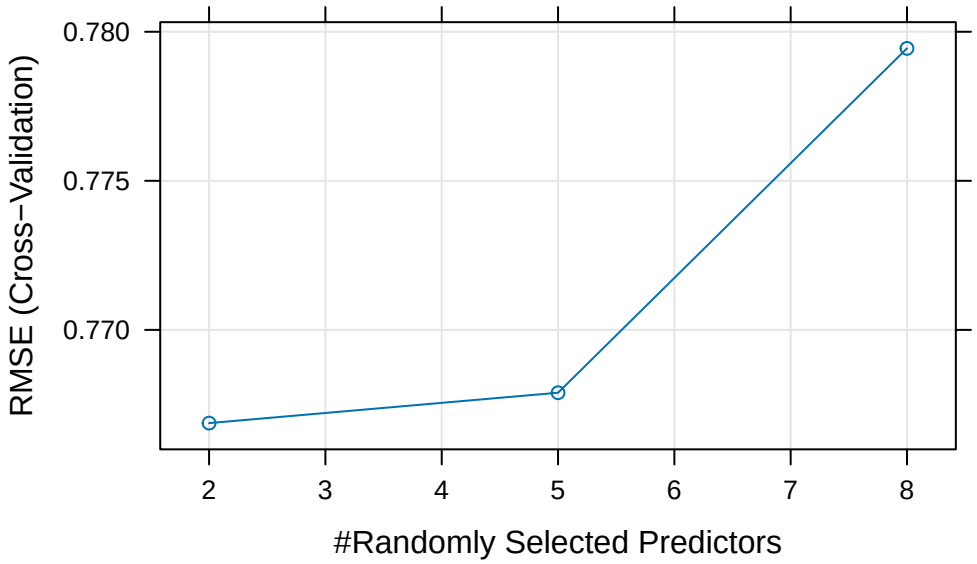
Fit random forest with cross-validation

```
rf_model <- train(lpsa ~ ., data = data, method = "rf", trControl = trainControl(method = "cv"))
```

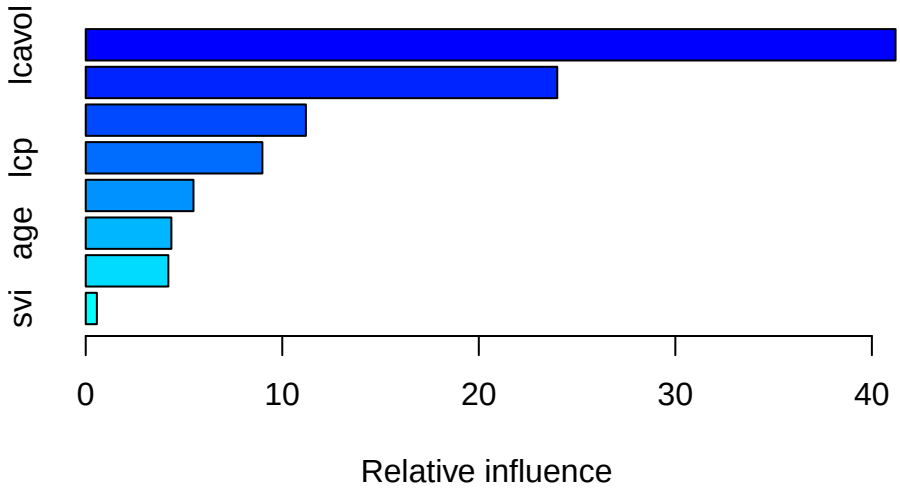
```
# Interpret the optimal model
rf_model
```

```
Random Forest

97 samples
 8 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 89, 87, 86, 88, 88, 87, ...
Resampling results across tuning parameters:

  mtry  RMSE       Rsquared   MAE
  2     0.7668721  0.5657187  0.6242453
  5     0.7678909  0.5739755  0.6247204
  8     0.7794426  0.5646524  0.6341039

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 2.
```

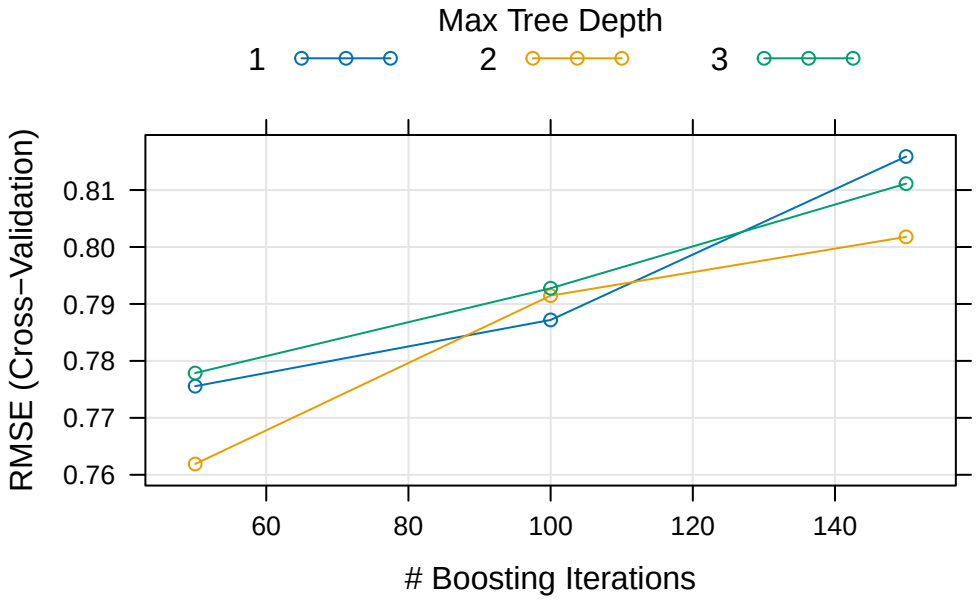
```
plot(rf_model)
```



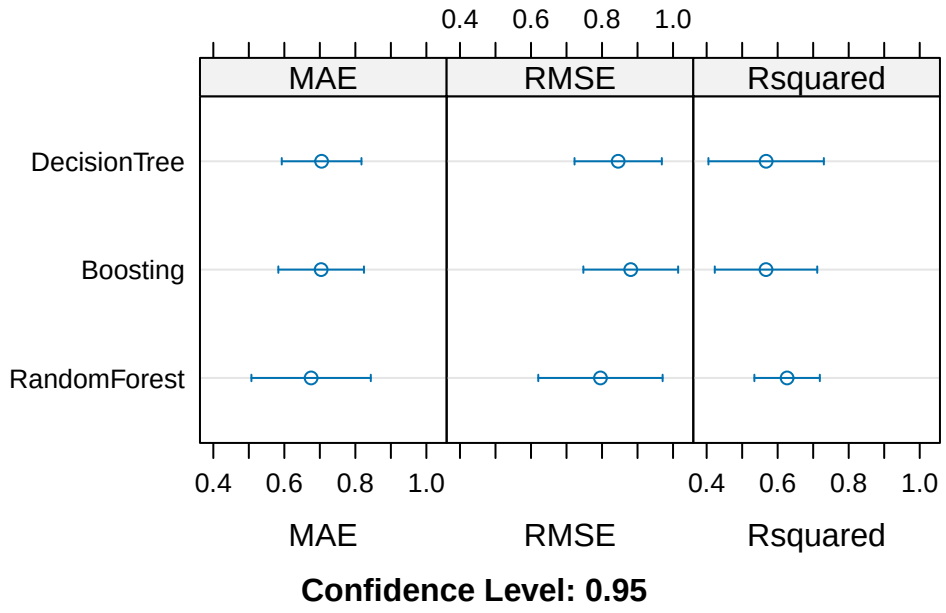
Boosted regression trees



The `gbm()` function is used to fit a boosted regression model, which aims to predict the `lpsa` variable based on the other variables in the dataset, using a Gaussian distribution for the response variable and constructing an ensemble of 50 decision trees with a maximum interaction depth of 4.



Cross-validation



Compare the performance using nested cross-validation

```
# Define cross-validation settings
cv <- trainControl(method = "cv", number = 5)
outer <- trainControl(method = "cv", number = 5)

tree_fit <- train(lpsa ~ ., data = data, method = "rpart", trControl = cv)
```

Train decision tree with cross-validation

```
tree_fit <- train(lpsa ~ ., data = data, method = "rpart", trControl = cv)
```

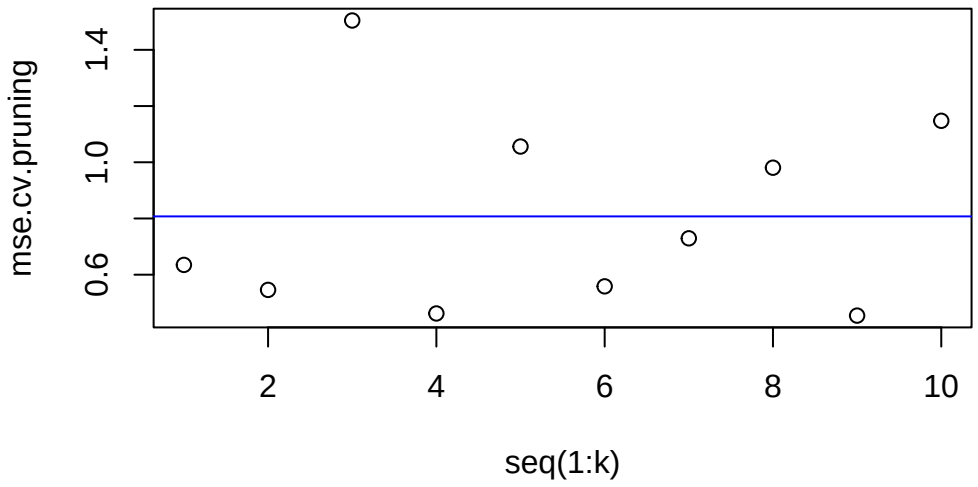
```
tree_fit <- train(lpsa ~ ., data = data, method = "rpart", trControl = cv)
tree_nested <- train(lpsa ~ ., data = data, method = "rpart", trControl = outer)
```

```
rf_fit <- train(lpsa ~ ., data = data, method = "rf", trControl = cv)
rf_nested <- train(lpsa ~ ., data = data, method = "rf", trControl = outer)
```

Compare nested cross-validation

```
nested_results <- resamples(list(Tree = tree_nested, RF = rf_nested, Boosting = boost_nested))
# summary(nested_results)
```

Based on these observations, overall, the Random Forest model shows the best performance among the three models for this particular dataset and task, even when using nested cross-validation.



These MSE values provide quantitative indications of the effectiveness of pruning in improving model performance. Lower MSE values generally indicate better predictive accuracy.

CONCLUSION

Based on these observations, overall, the Random Forest model demonstrates the best performance among the three models for this particular dataset and task.