Classification

Blake Pappas

2023-12-19

Part III. Classification in R

```r
library(dplyr)
library(caret)
library(rpart)
library(rpart.plot)
library(class)
library(e1071)
```

Import the "breast_cancer.csv" data file into R:

```r
cancer = read.csv("breast_cancer.csv")
```

After you import the data, convert "Class" variable into a factor, so that R treats it as a categorical variable, instead of a numeric variable:

```r
cancer$Class = factor(cancer$Class)
```

1. Split the dataset into 80% training and 20% testing

```r
library(caret)

train_rows = createDataPartition(y = cancer$Class,
                                 p = 0.7991, list = FALSE)

cancer_train = cancer[train_rows, ]

cancer_test = cancer[-train_rows, ]
```

2. Build a decision tree model

```r
library(rpart)

tree = rpart(Class ~ ., data = cancer_train,
             method = "class",
             parms = list(split = "information"))
```

3. Plot the tree, and then answer the following questions:

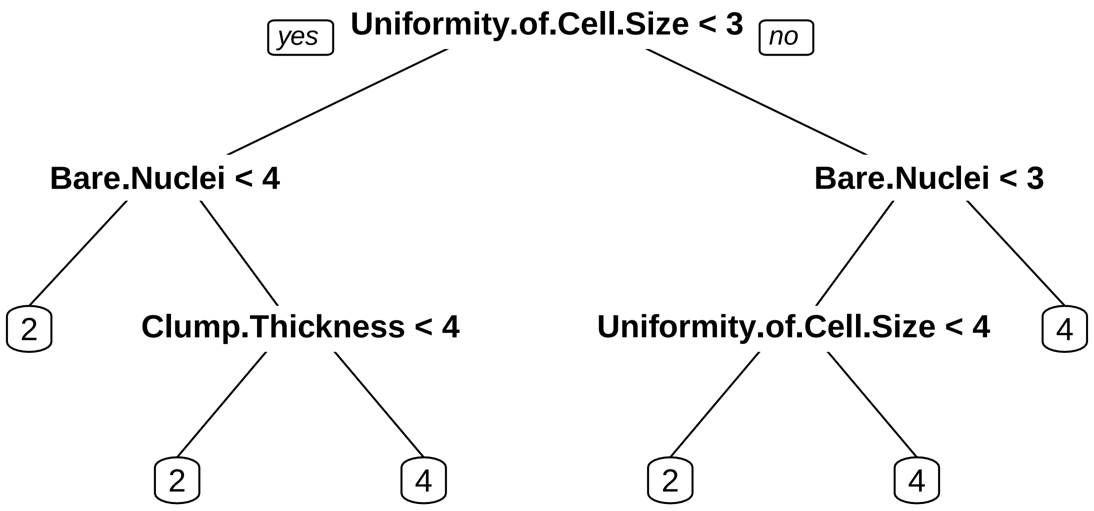
3.1. How many decision nodes are there in your tree?

Answer: There are ___ decision nodes in the decision tree.

3.2. Pick one decision rule from your tree and interpret it.

Answer:

```r
library(rpart.plot)

prp(tree, varlen = 0)
```



4. Evaluate the performance of the decision tree. Specifically, report the following metrics: (1) confusion matrix; (2) accuracy; (3) precision, recall, f-measure for "malignant" class; (4) AUC for "malignant" class.

```
# Make Categorical Predictions
pred_tree = predict(tree, cancer_test, type = "class")

# Make Class Probability Predictions
prob_pred_tree = predict(tree, cancer_test, type = "prob")

# Evaluate Model Performance
confusionMatrix(pred_tree, cancer_test$Class,
                mode = "prec_recall", positive = "4")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  2  4
##          2 83  1
##          4  6 47
```

```
##
##                Accuracy : 0.9489
##                  95% CI : (0.8976, 0.9792)
##     No Information Rate : 0.6496
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8904
##
##  Mcnemar's Test P-Value : 0.1306
##
##               Precision : 0.8868
##                  Recall : 0.9792
##                      F1 : 0.9307
##              Prevalence : 0.3504
##          Detection Rate : 0.3431
##    Detection Prevalence : 0.3869
##       Balanced Accuracy : 0.9559
##
##        'Positive' Class : 4
##
```

```r
# Make ROC Curve for Class "malignant"
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
roc_tree = roc(response = ifelse(cancer_test$Class == "4", 1, 0),
             predictor = prob_pred_tree[, 2])
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
# Plot the ROC Curve
# plot(roc_tree)

# Calculate AUC
auc(roc_tree)
```

```
## Area under the curve: 0.957
```

5. Now, consider using K-NN to do the classification. Is there any need to normalize the data? Why or why not? If you think normalization is needed, write the code below to do so. If you think normalization is not necessary, explain why.

Answer: There is no need to normalize the data because all nine attributes already take values from the same range: one to ten.

6. Build a K-NN model with your own choice of k value and evaluate the performance of your K-NN model. Does it have higher or lower AUC than the decision tree model?

```
knn = knn3(Class ~ ., data = cancer_train, k = 3)

# Make Categorical Predictions
pred_knn = predict(knn, cancer_test, type = "class")

# Make Class Probability Predictions
prob_pred_knn = predict(knn, cancer_test, type = "prob")

# Evaluate Model Performance
confusionMatrix(pred_knn, cancer_test$Class,
                mode = "prec_recall", positive = "4")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  2  4
##          2 83  0
##          4  6 48
##
##                Accuracy : 0.9562
##                  95% CI : (0.9071, 0.9838)
##     No Information Rate : 0.6496
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9065
##
##  Mcnemar's Test P-Value : 0.04123
##
##               Precision : 0.8889
##                  Recall : 1.0000
##                      F1 : 0.9412
##              Prevalence : 0.3504
##          Detection Rate : 0.3504
##    Detection Prevalence : 0.3942
##       Balanced Accuracy : 0.9663
##
```

```
##        'Positive' Class : 4
##
```

```r
# Make ROC Curve for Class "malignant"
library(pROC)

roc_knn = roc(response = ifelse(cancer_test$Class == "4", 1, 0),
             predictor = prob_pred_knn[, 2])
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```r
# Plot the ROC Curve
# plot(roc_knn)

# Calculate AUC
auc(roc_knn)
```

```
## Area under the curve: 0.9747
```

Answer: The K-NN model has a ___ AUC than the decision tree model.

7. Try several different k values, report the AUC of each one you tried. Also, report which k value provides the highest AUC. Try using a for loop for this task.

```r
# Cross-Validation
cv = createFolds(y = cancer$Class, k = 5)

# Make a Vector to Store AUC from Each Fold
auc = c()

# Loop Through Each Fold
for (test_rows in cv) {
  cancer_train = cancer[-test_rows, ]
  cancer_test = cancer[test_rows, ]

  # Then, Train Your Model and Evaluate Its Performance
  knn = knn3(Class ~ ., data = cancer_train)

  pred_knn = predict(knn, cancer_test, type = "class")

  prob_pred_knn = predict(knn, cancer_test, type = "prob")

  roc_knn = roc(response = ifelse(cancer_test$Class == "4", 1, 0),
                 predictor = prob_pred_knn[, 2])

  # Add the AUC of the Current Fold
  auc = c(auc, auc(roc_knn))
  print(auc)
}
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## [1] 0.9923059

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## [1] 0.9923059 0.9866125

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## [1] 0.9923059 0.9866125 0.9992978

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## [1] 0.9923059 0.9866125 0.9992978 0.9940309

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## [1] 0.9923059 0.9866125 0.9992978 0.9940309 0.9852528
```

```r
# Average Accuracy Across Folds
print(mean(auc))
```

```
## [1] 0.9915
```

8. Build a naive bayes model and evaluate its performance on the same testing data. Does it have higher or lower AUC than your best decision tree and K-NN models?

```r
library(e1071)

NB_model = naiveBayes(Class ~ ., data = cancer_train)

# Make Categorical Predictions
pred_nb = predict(NB_model, cancer_test, type = "class")

# Make Class Probability Predictions
prob_pred_nb = predict(NB_model, cancer_test, type = "raw")

# Evaluate Model Performance
confusionMatrix(pred_nb, cancer_test$Class,
                mode = "prec_recall", positive = "4")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  2  4
##          2 85  2
##          4  4 46
##
##                Accuracy : 0.9562
##                  95% CI : (0.9071, 0.9838)
##     No Information Rate : 0.6496
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9047
##
##  Mcnemar's Test P-Value : 0.6831
##
##               Precision : 0.9200
##                  Recall : 0.9583
##                      F1 : 0.9388
##              Prevalence : 0.3504
##          Detection Rate : 0.3358
##    Detection Prevalence : 0.3650
##       Balanced Accuracy : 0.9567
##
##        'Positive' Class : 4
##
```

```r
# Make ROC Curve for Class "malignant"
library(pROC)

roc_nb = roc(response = ifelse(cancer_test$Class == "4", 1, 0),
             predictor = prob_pred_nb[, 2])
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

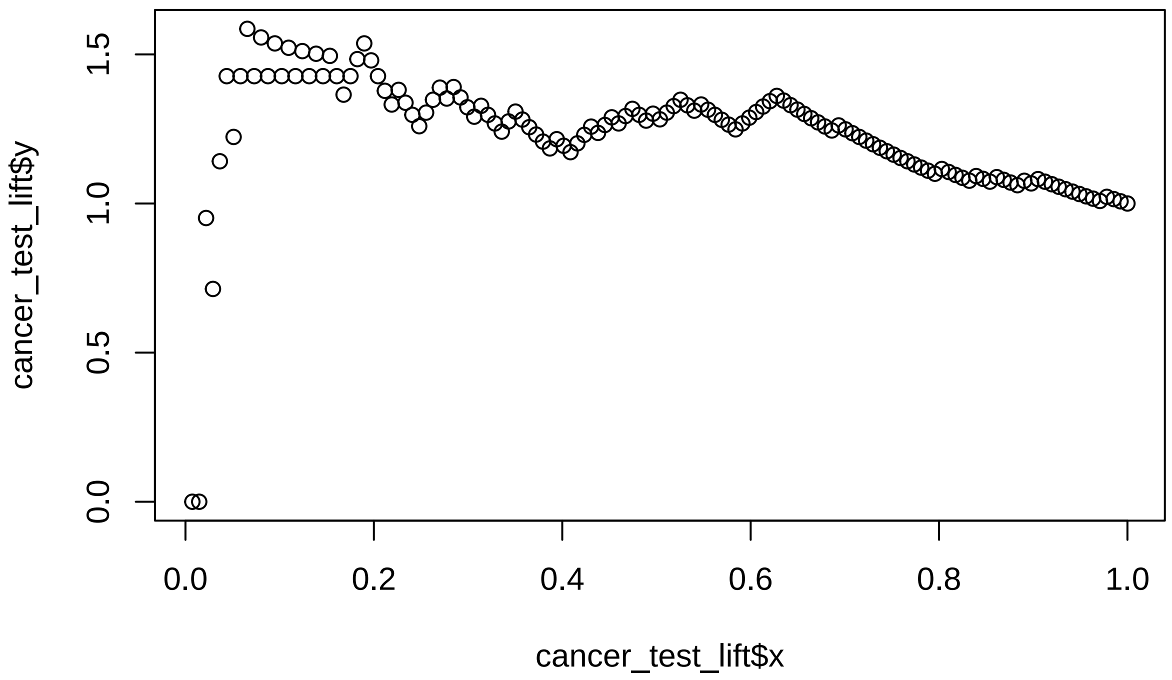
```r
# Plot the ROC Curve
# plot(roc_nb)

# Calculate AUC
auc(roc_nb)
```

```
## Area under the curve: 0.9807
```

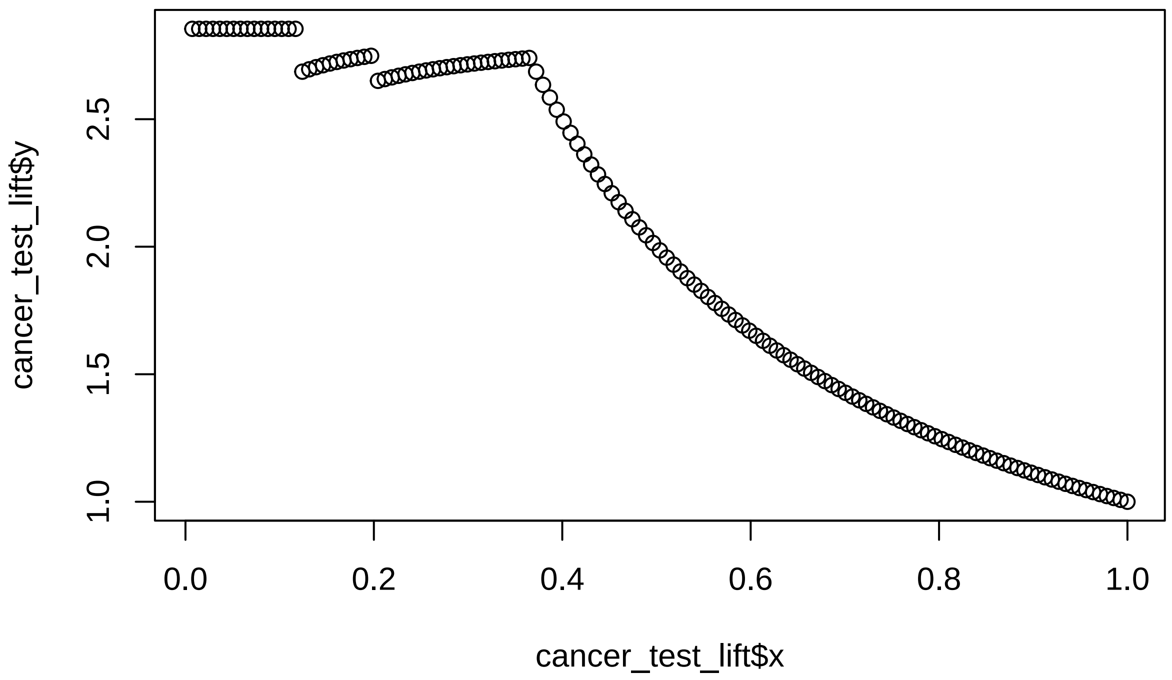
Answer: The naive bayes model has a ___ AUC than the best decision tree and K-NN models.

9. Take the best model in terms of AUC and plot the lift curve. What is the lift ratio at the top 10% of cases with highest "malignant" probability as predicted by the model? Interpret the meaning of the lift ratio.

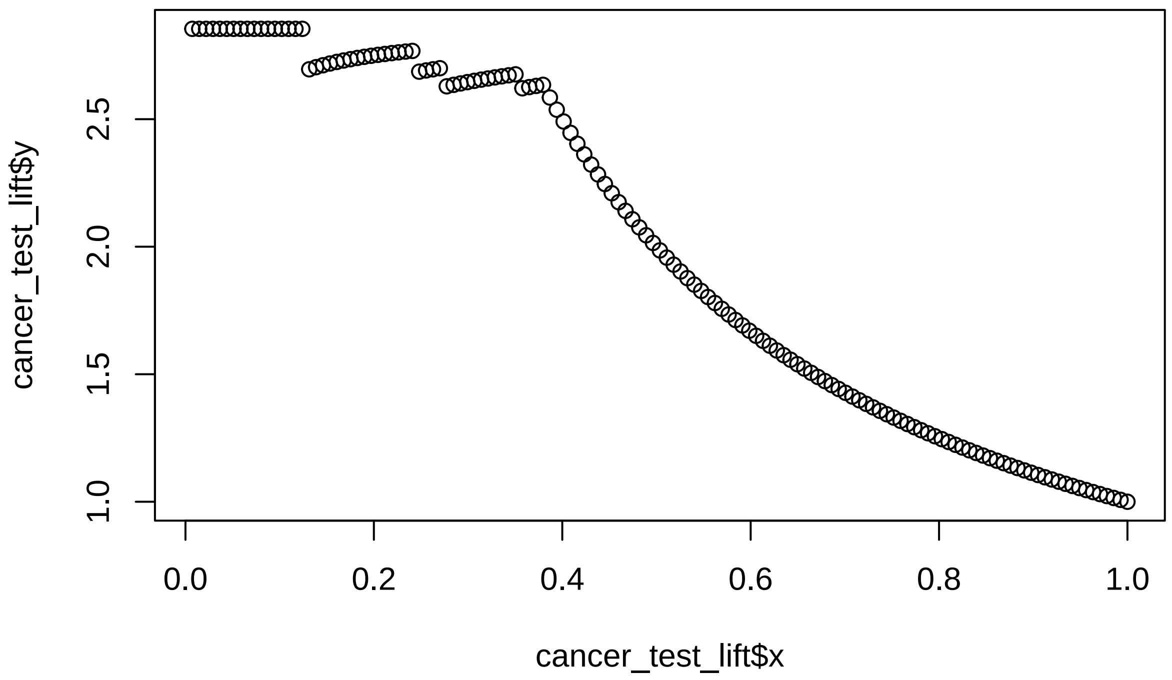
```r
cancer_test_lift = cancer_test %>%
  mutate(prob = prob_pred_tree[, 2]) %>%
  arrange(desc(prob)) %>%
  mutate(malignant_yes = ifelse(Class == "4", 1, 0)) %>%
  # The Following Two Lines Make the Lift Curve
  mutate(x = row_number() / nrow(cancer_test),
         y = (cumsum(malignant_yes) / sum(malignant_yes)) / x)

plot(cancer_test_lift$x, cancer_test_lift$y)
```



```r
cancer_test_lift = cancer_test %>%
  mutate(prob = prob_pred_knn[, 2]) %>%
  arrange(desc(prob)) %>%
  mutate(malignant_yes = ifelse(Class == "4", 1, 0)) %>%
  # The Following Two Lines Make the Lift Curve
  mutate(x = row_number() / nrow(cancer_test),
         y = (cumsum(malignant_yes) / sum(malignant_yes)) / x)

plot(cancer_test_lift$x, cancer_test_lift$y)
```



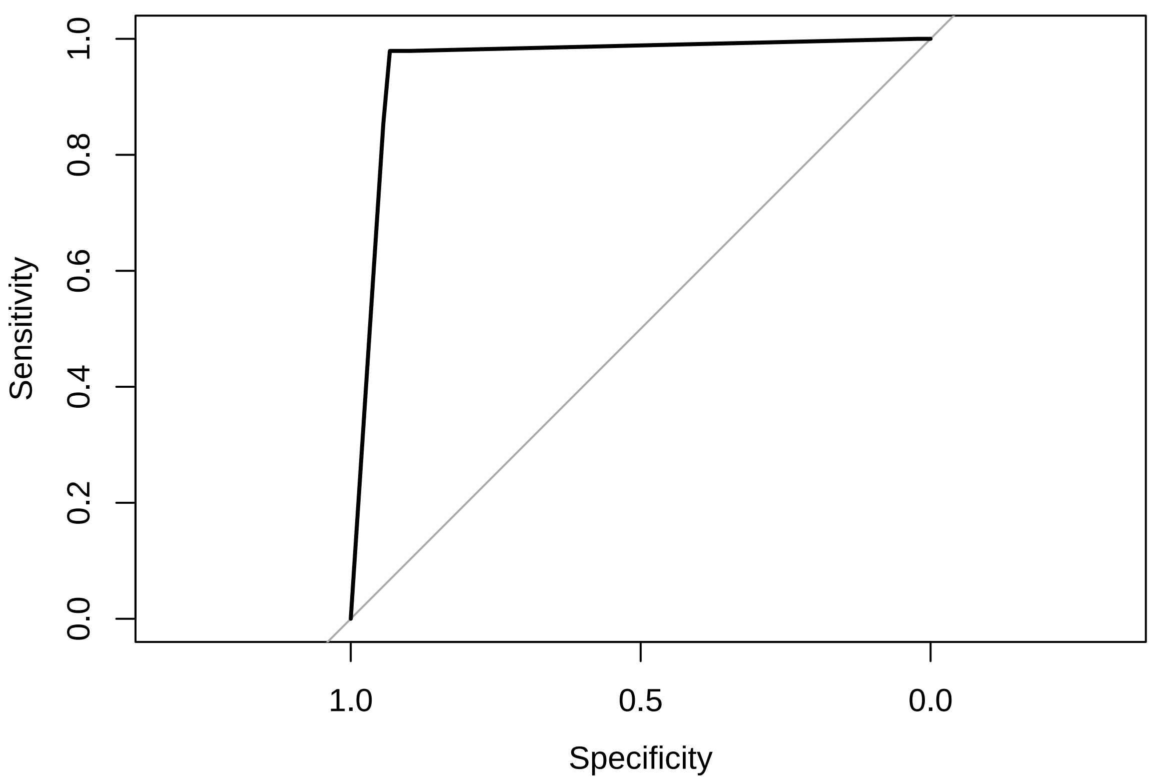
```
cancer_test_lift = cancer_test %>%
  mutate(prob = prob_pred_nb[, 2]) %>%
  arrange(desc(prob)) %>%
  mutate(malignant_yes = ifelse(Class == "4", 1, 0)) %>%
  # The Following Two Lines Make the Lift Curve
  mutate(x = row_number() / nrow(cancer_test),
         y = (cumsum(malignant_yes) / sum(malignant_yes)) / x)

plot(cancer_test_lift$x, cancer_test_lift$y)
```



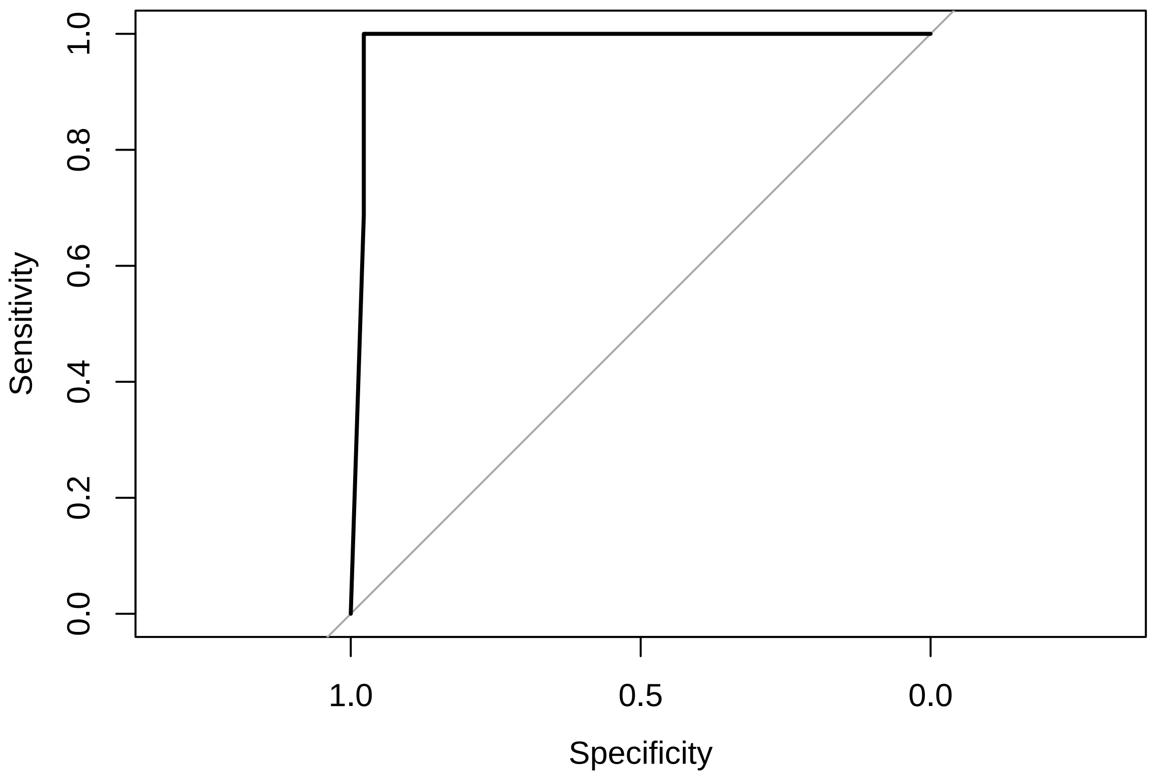
Answer: The lift ratio for the top 10% of cases with the highest "malignant" probability is about . This means that the model's top 10% of cases with the highest "malignant" probability are about times as good at predicting the outcome than random guesses.

10. Again, take the best model in terms of AUC and plot the ROC curve for class "malignant".

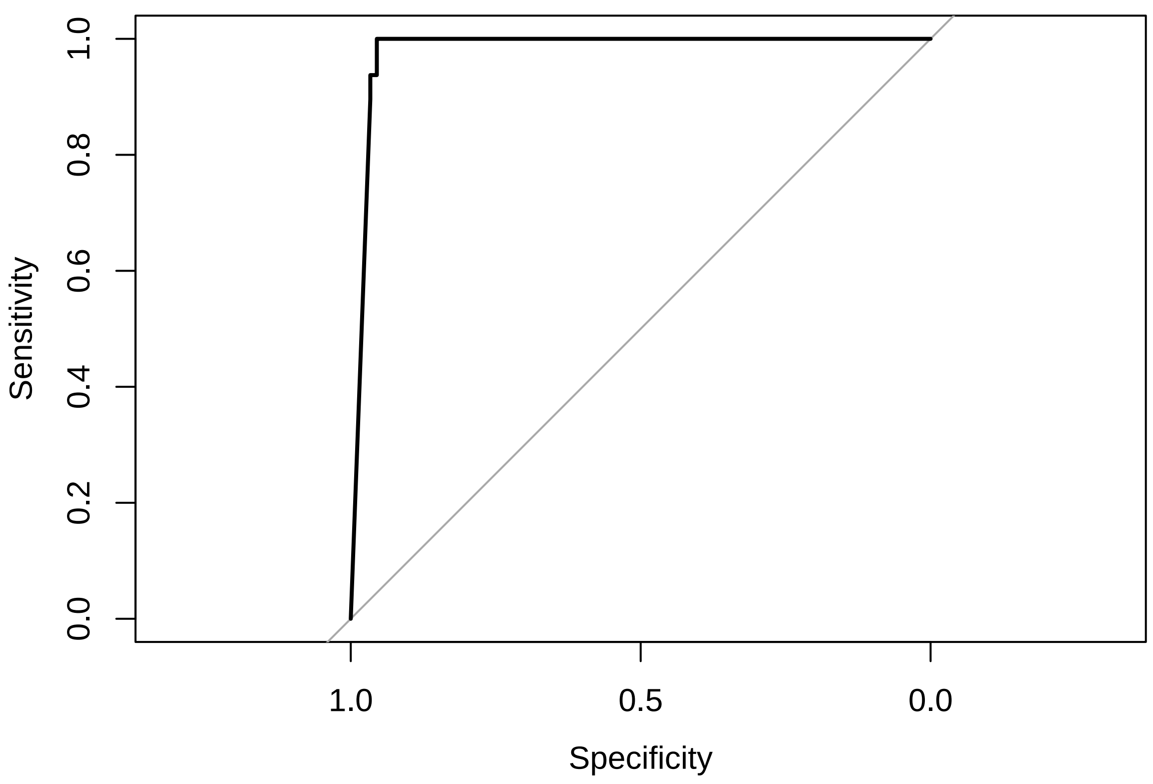
```r
plot(roc_tree)
```



```
plot(roc_knn)
```



```r
plot(roc_nb)
```


1.0
0.8
0.6
0.4
0.2
0.0
Sensitivity
1.0
0.5
0.0
Specificity