Assignment 2: Classification for the Detection of Opinion Spam

Anouk van der Lee (6620590), Shu Zhao (6833519), Fleur Petit (5583837)

20, October, 2020

Contents

Dataset for training, validation and test

Dataset split

```
# 1 fold = 80 samples
# Fold 1-4 for training = 1:320 from true + 1:320 from false
index.train <- c(c(1:320), 400+c(1:320))

# Training document-term matrix
train.dtm <- DocumentTermMatrix(reviews.all[index.train])
# Test document-term matrix
test.dtm <- DocumentTermMatrix(reviews.all[-index.train], list(dictionary = dimnames(train.dtm)[[2]]))
```

Feature selection

```
# Remove terms that occur in less than 5% of the documents
# Training document-term matrix
train.dtm <- removeSparseTerms(train.dtm, 0.95)
# Test document-term matrix
test.dtm <- DocumentTermMatrix(reviews.all[-index.train], list(dictionary = dimnames(train.dtm)[[2]]))
```

```r
# Convert document term matrix to binary (term present/absent)
train.dtm.bin <- as.matrix(train.dtm) > 0

# Compute mutual information of each term with class label
train.mi <- apply(as.matrix(train.dtm.bin),
                  2,
                  function(x,y){ mi.plugin(table(x,y)/length(y), unit="log2")},
                  labels[index.train])

# Sort the indices from high to low mutual information
train.mi.order <- order(train.mi, decreasing = TRUE)

# Show the top50 terms with highest mutual information
train.mi[train.mi.order[1:50]]
```

```
##     chicago    location       smell      luxury     decided    recently
## 0.126280238 0.047409953 0.045492572 0.045453470 0.037337886 0.034615562
##     finally  millennium      seemed       great     cleaned  experience
## 0.031413717 0.030255024 0.026663642 0.026539428 0.025712453 0.025586068
##        open     smelled        rude        star     arrived    elevator
## 0.024431835 0.023444423 0.022280388 0.021355574 0.020805631 0.018345946
## comfortable        many       floor       found        make        wait
## 0.016417030 0.016314485 0.016037563 0.015707192 0.015445026 0.014807508
##     website       smoke      suites        like       clerk      sheets
## 0.014636104 0.014273082 0.013792167 0.013256517 0.013136065 0.012546660
##       ready        took        cant        food        hour     reviews
## 0.012214705 0.011980107 0.011911600 0.011313915 0.011060284 0.010898259
##      coffee      towels     staying    expected   recommend        need
## 0.010894875 0.010891371 0.010781899 0.010582660 0.010335039 0.010120674
##       hours        rate        walk       hotel      street         now
## 0.009781835 0.009683180 0.009683180 0.009301503 0.008779978 0.008733504
##        room         day
## 0.008658183 0.008584388
```

```r
# Training document-term matrix
train.dtm.top50 <- train.dtm[, train.mi.order[1:50]]
# Test document-term matrix
test.dtm.top50 <- test.dtm[, train.mi.order[1:50]]

# Training document-term matrix
train.dtm.top100 <- train.dtm[, train.mi.order[1:100]]
# Test document-term matrix
test.dtm.top100 <- test.dtm[, train.mi.order[1:100]]
```

Classifiers

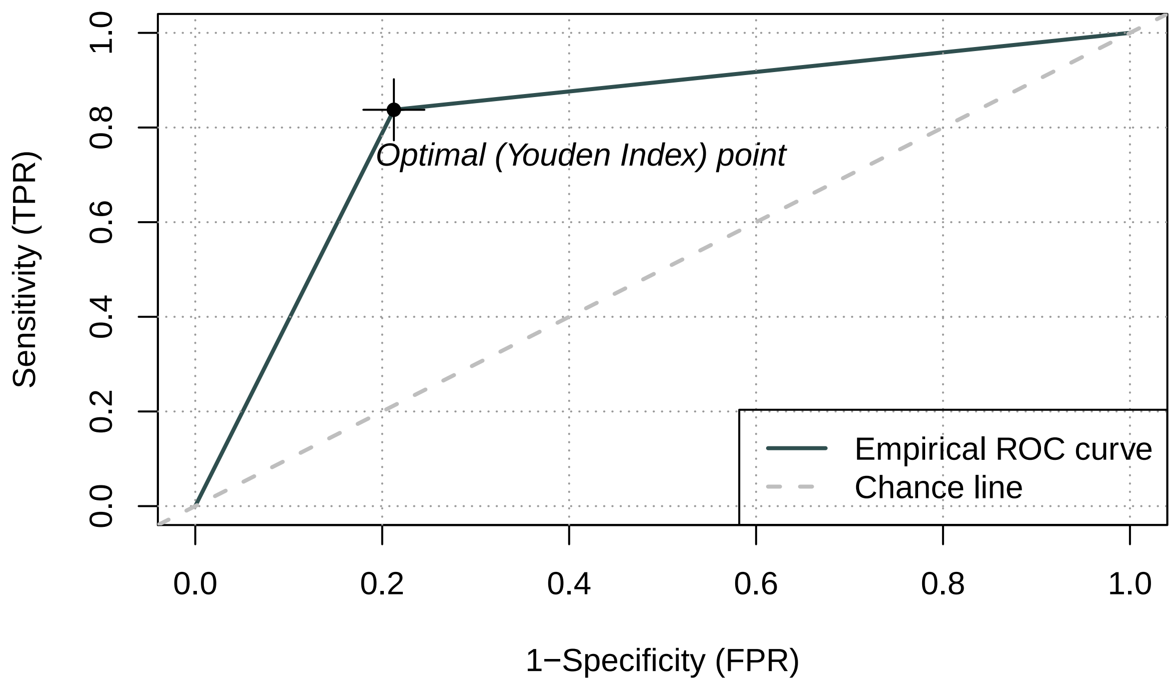
Multinomial naive Bayes

```r
# Train the model with priors and conditional probabilities
reviews.mnb <- train.mnb(as.matrix(train.dtm), labels[index.train])
```

```r
# Make predictions
reviews.mnb.pred <- predict.mnb(reviews.mnb, as.matrix(test.dtm))

# Confusion matrix
conf.mat.mnb <- table(labels[-index.train], reviews.mnb.pred, dnn = c("actual", "predicted"))

perf.mnb <-performance(conf.mat.mnb)
perf.mnb %>% kable()
```

metric	value
recall	0.8375000
miss-rate	0.1625000
fall-out	0.2125000
selectivity	0.7875000
prevalence	0.5000000
precision	0.7976190
false omission rate	0.1710526
pos likelihood ratio	3.9411765
neg likelihood ratio	0.2063492
accuracy	0.8125000
false discovery rate	0.2023810
neg predictive value	0.8289474
diagnostic odds ratio	19.0995475
F1	0.8170732

```r
# ROC
roc.mnb <- rocit(score=as.numeric(reviews.mnb.pred), class=labels[-index.train])
plot(roc.mnb)
```

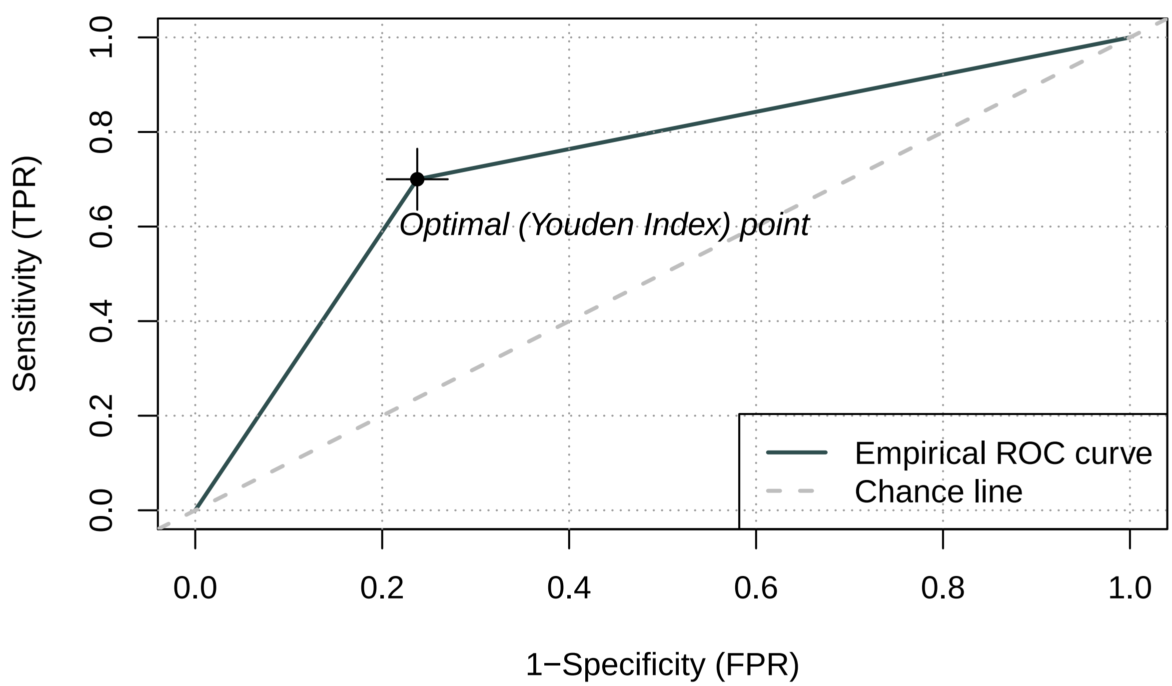



```r
# Train the model with priors and conditional probabilities
reviews.mnb.top50 <- train.mnb(as.matrix(train.dtm.top50), labels[index.train])
# Make predictions
reviews.mnb.top50.pred <- predict.mnb(reviews.mnb.top50, as.matrix(test.dtm.top50))

# Confusion matrix
conf.mat.mnb.top50 <- table(labels[-index.train], reviews.mnb.top50.pred, dnn = c("actual", "predicted"

perf.mnb.top50 <-performance(conf.mat.mnb.top50)
perf.mnb.top50 %>% kable()
```

metric	value
recall	0.7000000
miss-rate	0.3000000
fall-out	0.2375000
selectivity	0.7625000
prevalence	0.5000000
precision	0.7466667
false omission rate	0.2823529
pos likelihood ratio	2.9473684
neg likelihood ratio	0.3934426
accuracy	0.7312500
false discovery rate	0.2533333
neg predictive value	0.7176471
diagnostic odds ratio	7.4912281

metric	value
F1	0.7225806

```
# ROC
roc.mnb.top50 <- rocit(score=as.numeric(reviews.mnb.top50.pred), class=labels[-index.train])
plot(roc.mnb.top50)
```

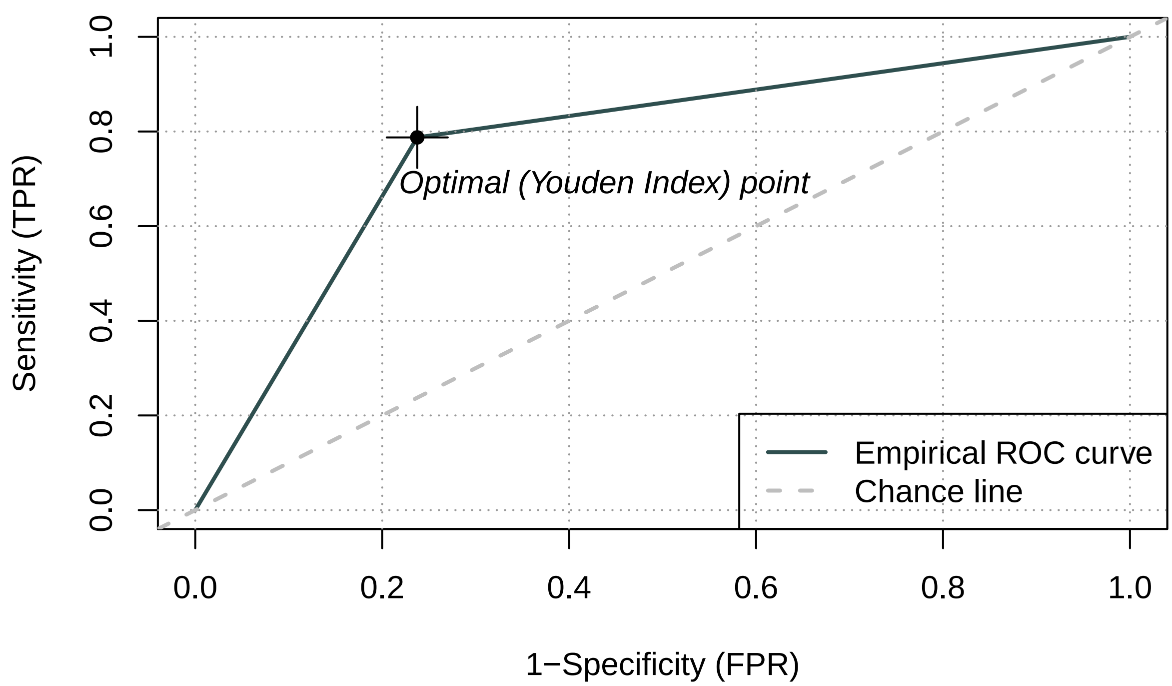



```
# Train the model with priors and conditional probabilities
reviews.mnb.top100 <- train.mnb(as.matrix(train.dtm.top100), labels[index.train])
# Make predictions
reviews.mnb.top100.pred <- predict.mnb(reviews.mnb.top100, as.matrix(test.dtm.top100))

# Confusion matrix
conf.mat.mnb.top100 <- table(labels[-index.train], reviews.mnb.top100.pred, dnn = c("actual", "predicte

perf.mnb.top100 <-performance(conf.mat.mnb.top100)
perf.mnb.top100 %>% kable()
```

metric	value
recall	0.7875000
miss-rate	0.2125000
fall-out	0.2375000
selectivity	0.7625000

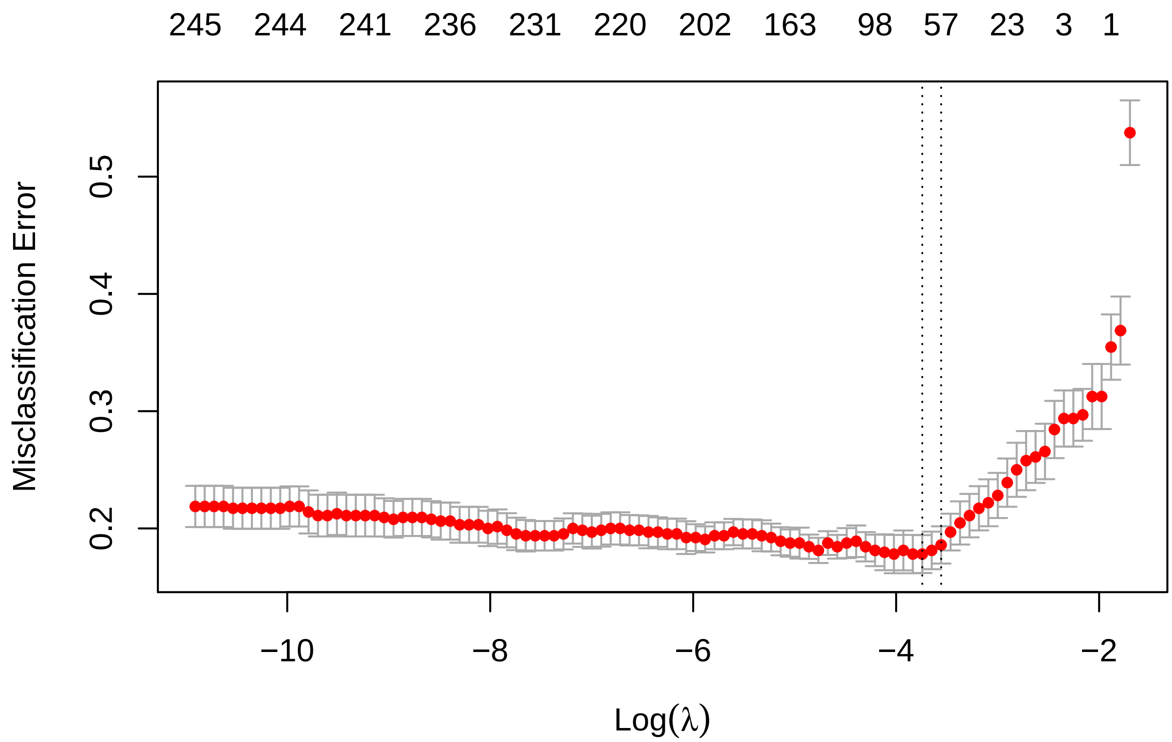
metric	value
prevalence	0.5000000
precision	0.7682927
false omission rate	0.2179487
pos likelihood ratio	3.3157895
neg likelihood ratio	0.2786885
accuracy	0.7750000
false discovery rate	0.2317073
neg predictive value	0.7820513
diagnostic odds ratio	11.8978328
F1	0.7777778

```
# ROC
roc.mnb.top100 <- rocit(score=as.numeric(reviews.mnb.top100.pred), class=labels[-index.train])
plot(roc.mnb.top100)
```



Regularized logistic regression

```
# Logistic regression with lasso penalty
reviews.glmnet <- cv.glmnet(as.matrix(train.dtm), labels[index.train], family="binomial", type.measure=
plot(reviews.glmnet)
```

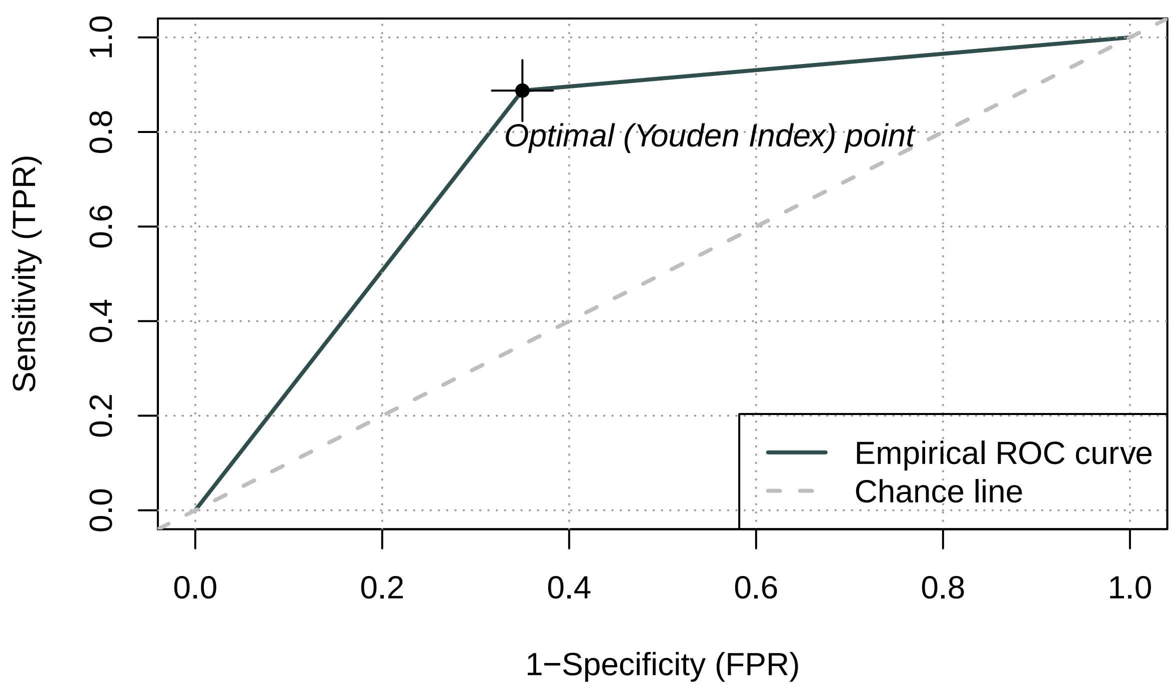



```r
# coef(reviews.glmnet, s="lambda.1se")

# Make predictions on the test set
reviews.logreg.pred <- predict(reviews.glmnet, newx=as.matrix(test.dtm), s="lambda.1se", type="class")

# Confusion matrix
conf.mat.logreg <- table(labels[-index.train], reviews.logreg.pred, dnn = c("actual", "predicted"))

perf.logreg <-performance(conf.mat.logreg)
perf.logreg %>% kable()
```

metric	value
recall	0.8875000
miss-rate	0.1125000
fall-out	0.3500000
selectivity	0.6500000
prevalence	0.5000000
precision	0.7171717
false omission rate	0.1475410
pos likelihood ratio	2.5357143
neg likelihood ratio	0.1730769
accuracy	0.7687500
false discovery rate	0.2828283
neg predictive value	0.8524590
diagnostic odds ratio	14.6507937

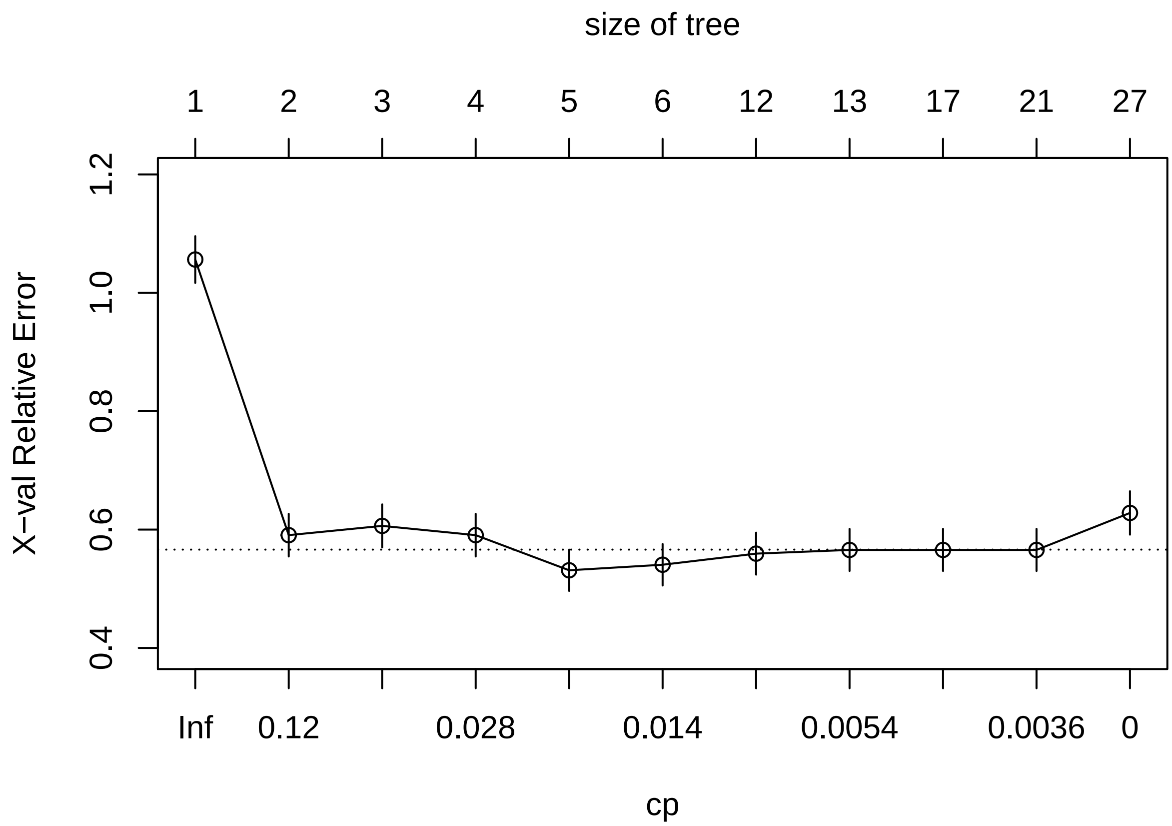
metric	value
F1	0.7932961

```
# ROC
roc.logreg <- rocit(score=as.numeric(reviews.logreg.pred), class=labels[-index.train])
plot(roc.logreg)
```

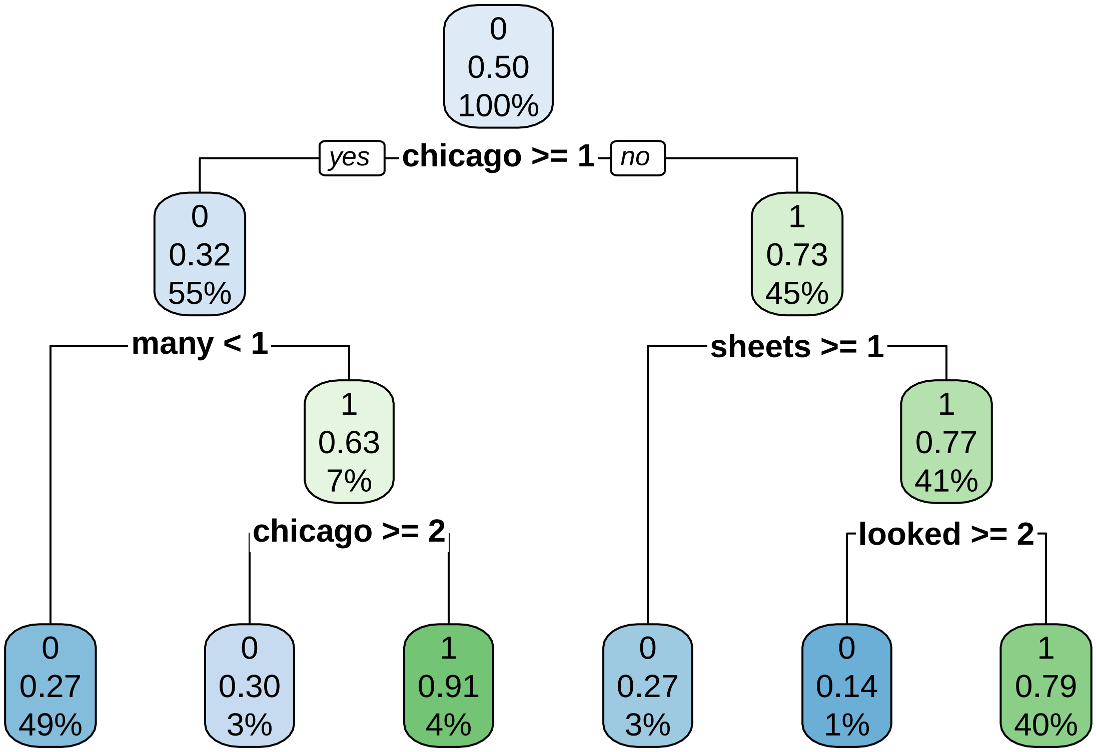


Classification trees

```
# Grow the tree
reviews.rpart <- rpart(label ~ .,
                       data = data.frame(as.matrix(train.dtm), label = labels[index.train]),
                       cp = 0,
                       method = "class")

# Plot cv-error of pruning sequence
plotcp(reviews.rpart)
```




```
# Tree with lowest cv error
reviews.rpart.pruned <- prune(reviews.rpart, cp=0.014)
# Plot the tree
rpart.plot(reviews.rpart.pruned)
```

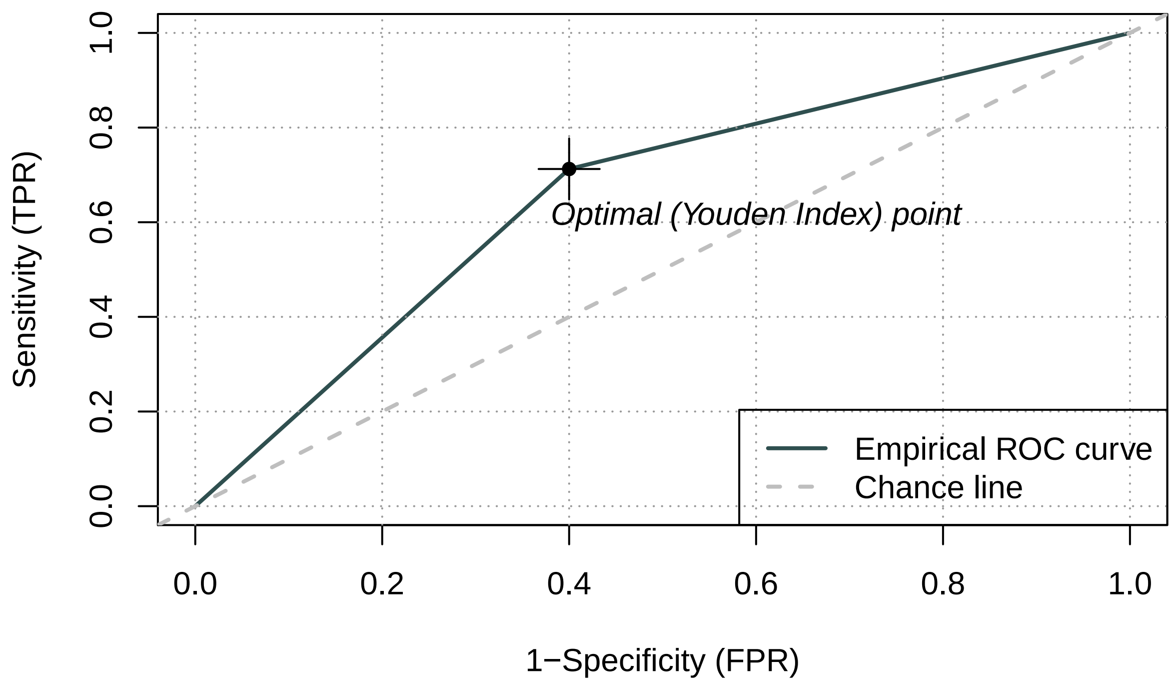



```r
# Make predictions on the test set
reviews.rpart.pred <- predict(reviews.rpart.pruned,
                              newdata = data.frame(as.matrix(test.dtm)),
                              type = "class")

# Confusion matrix
conf.mat.rpart <- table(labels[-index.train], reviews.rpart.pred, dnn = c("actual", "predicted"))

perf.rpart <-performance(conf.mat.rpart)
perf.rpart %>% kable()
```

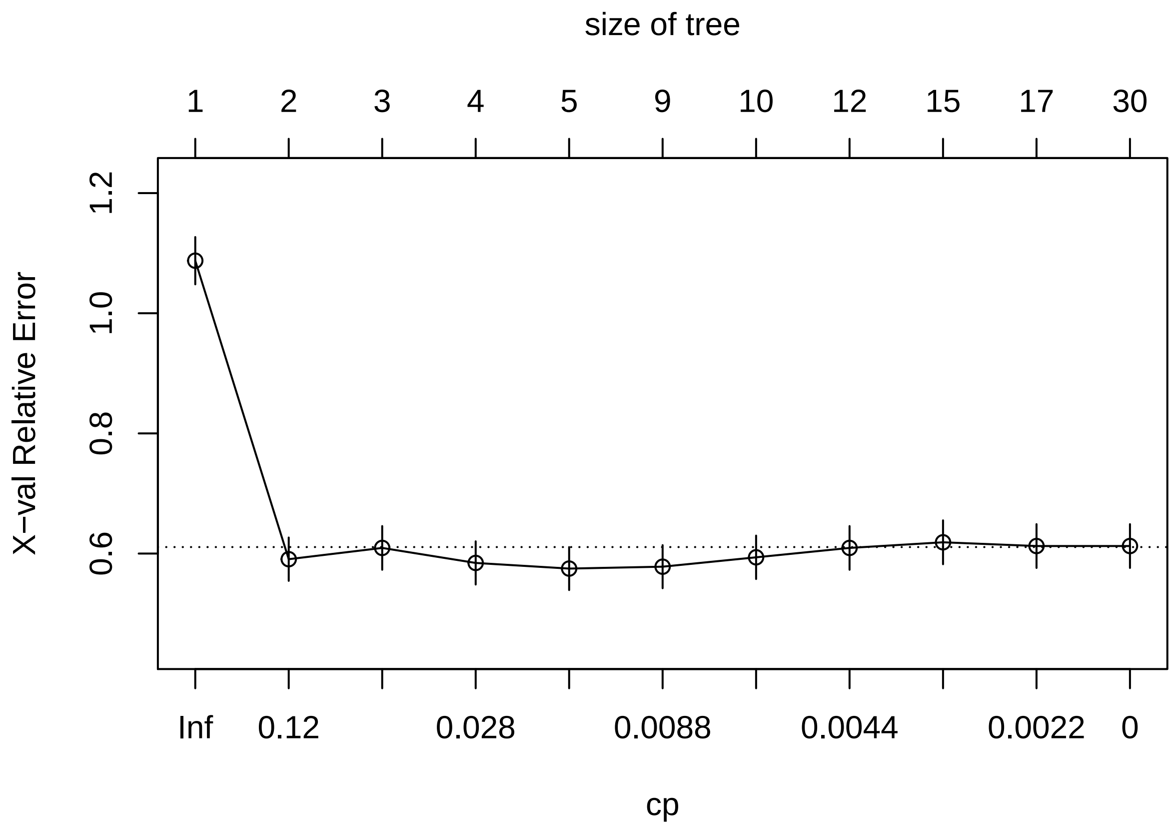
metric	value
recall	0.7125000
miss-rate	0.2875000
fall-out	0.4000000
selectivity	0.6000000
prevalence	0.5000000
precision	0.6404494
false omission rate	0.3239437
pos likelihood ratio	1.7812500
neg likelihood ratio	0.4791667
accuracy	0.6562500
false discovery rate	0.3595506
neg predictive value	0.6760563
diagnostic odds ratio	3.7173913

metric	value
F1	0.6745562

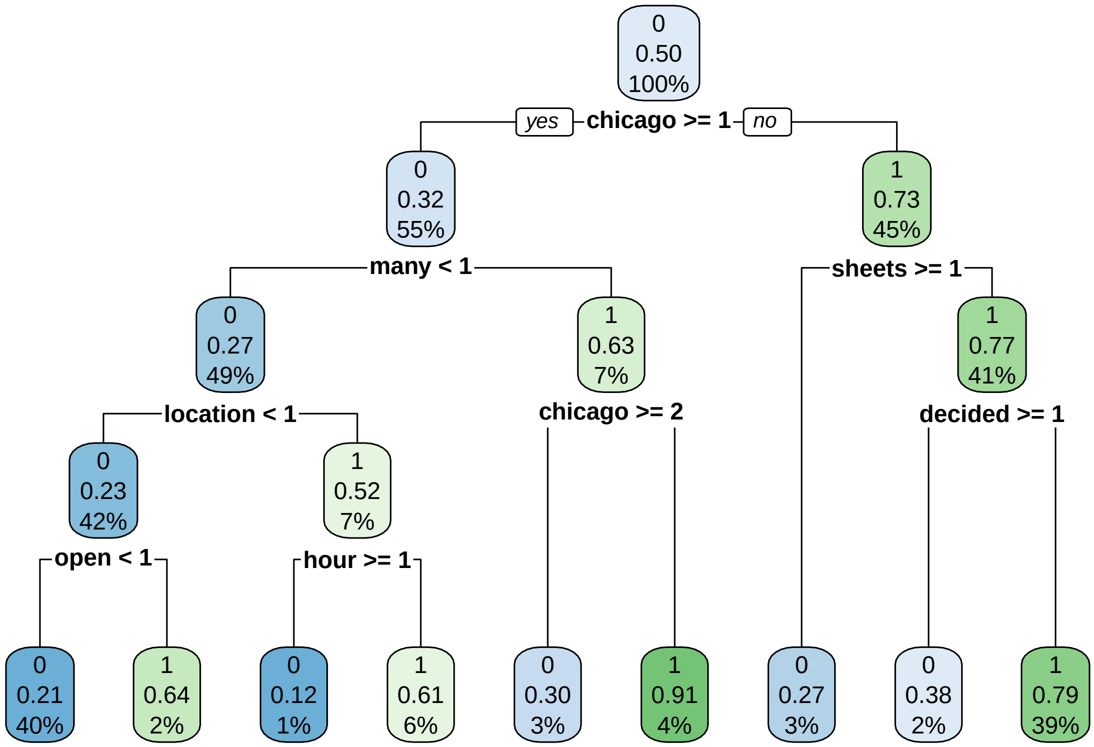
```
# ROC
roc.rpart <- rocit(score=as.numeric(reviews.rpart.pred), class=labels[-index.train])
plot(roc.rpart)
```




```
# Grow the tree
reviews.rpart.top50 <- rpart(label ~ .,
                             data = data.frame(as.matrix(train.dtm.top50), label = labels[index.train]),
                             cp = 0,
                             method = "class")

# Plot cv-error of pruning sequence
plotcp(reviews.rpart.top50)
```




```
# Tree with lowest cv error
reviews.rpart.top50.pruned <- prune(reviews.rpart.top50, cp=0.012)
# Plot the tree
rpart.plot(reviews.rpart.top50.pruned)
```

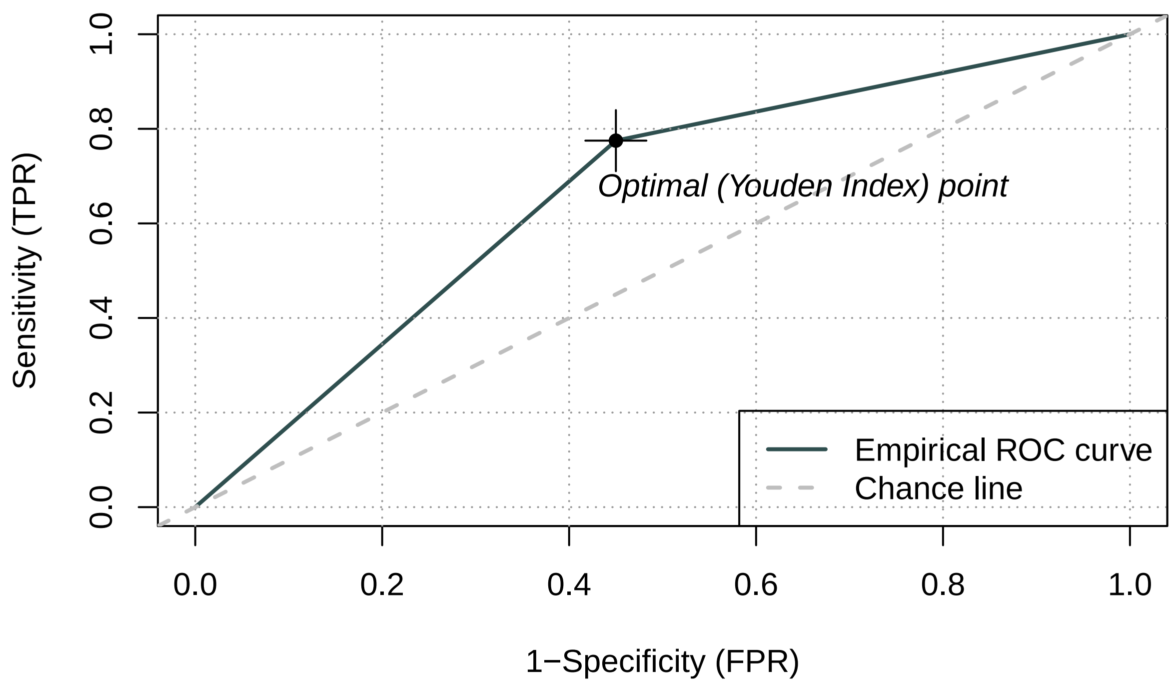



```r
# Make predictions on the test set
reviews.rpart.top50.pred <- predict(reviews.rpart.top50.pruned,
                                    newdata = data.frame(as.matrix(test.dtm.top50)),
                                    type = "class")

# Confusion matrix
conf.mat.rpart.top50 <- table(labels[-index.train], reviews.rpart.top50.pred, dnn = c("actual", "predic

perf.rpart.top50 <-performance(conf.mat.rpart.top50)
perf.rpart.top50 %>% kable()
```

metric	value
recall	0.7750000
miss-rate	0.2250000
fall-out	0.4500000
selectivity	0.5500000
prevalence	0.5000000
precision	0.6326531
false omission rate	0.2903226
pos likelihood ratio	1.7222222
neg likelihood ratio	0.4090909
accuracy	0.6625000
false discovery rate	0.3673469
neg predictive value	0.7096774
diagnostic odds ratio	4.2098765

metric	value
F1	0.6966292

```
# ROC
roc.rpart.top50 <- rocit(score=as.numeric(reviews.rpart.top50.pred), class=labels[-index.train])
plot(roc.rpart.top50)
```

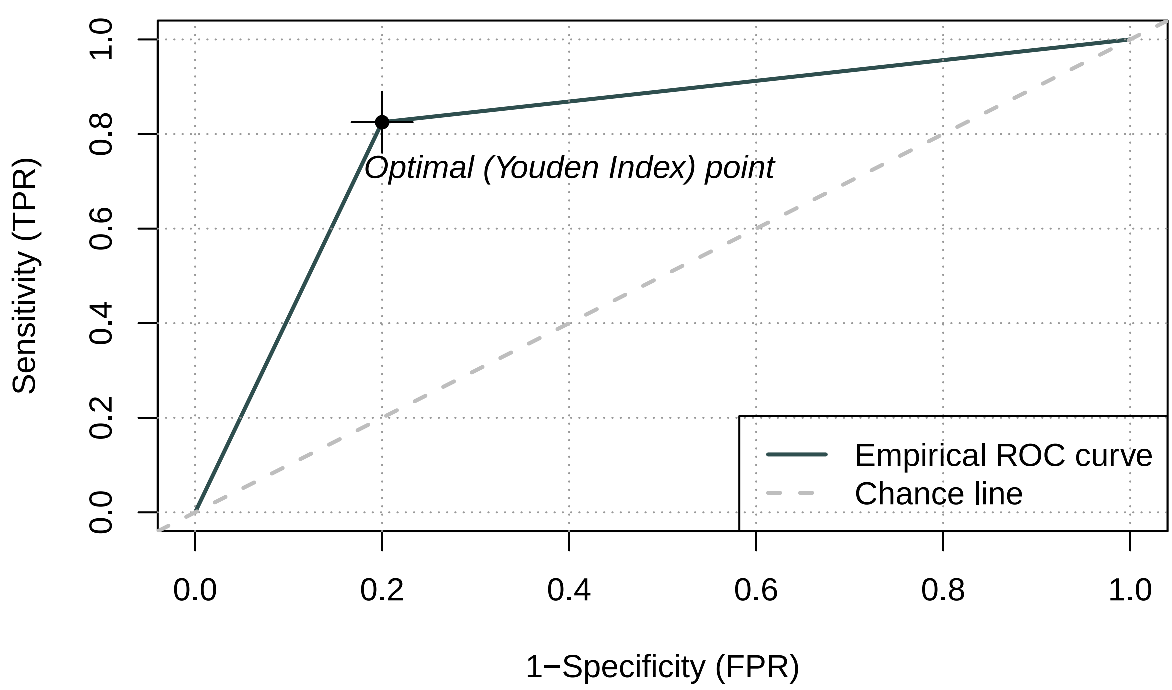


Random forests

```
# Train random forest with default settings: 500 trees and mtry = 17
reviews.rf <- randomForest(as.factor(label) ~ .,
                           data = data.frame(as.matrix(train.dtm), label = labels[index.train]))
# Make predictions
reviews.rf.pred <- predict(reviews.rf,
                           newdata = data.frame(as.matrix(test.dtm), label = labels[-index.train]))

# Confusion matrix
conf.mat.rf <- table(labels[-index.train], reviews.rf.pred, dnn = c("actual", "predicted"))

perf.rf <-performance(conf.mat.rf)
perf.rf %>% kable()
```

metric	value
recall	0.8250000
miss-rate	0.1750000
fall-out	0.2000000
selectivity	0.8000000
prevalence	0.5000000
precision	0.8048780
false omission rate	0.1794872
pos likelihood ratio	4.1250000
neg likelihood ratio	0.2187500
accuracy	0.8125000
false discovery rate	0.1951220
neg predictive value	0.8205128
diagnostic odds ratio	18.8571429
F1	0.8148148

```r
# ROC
roc.rf <- rocit(score=as.numeric(reviews.rf.pred), class=labels[-index.train])
plot(roc.rf)
```



Hyper-parameters

Questions

1. For a single classification tree, the impurity reduction is not equal to mutual information?
2. How to use cross validation for Multinomial naive Bayes or Classification tree, and use out-of-bag evaluation for Random forest?
3. Bi-gram for Multinomial naive Bayes