Homework 3

Xinyi Lin

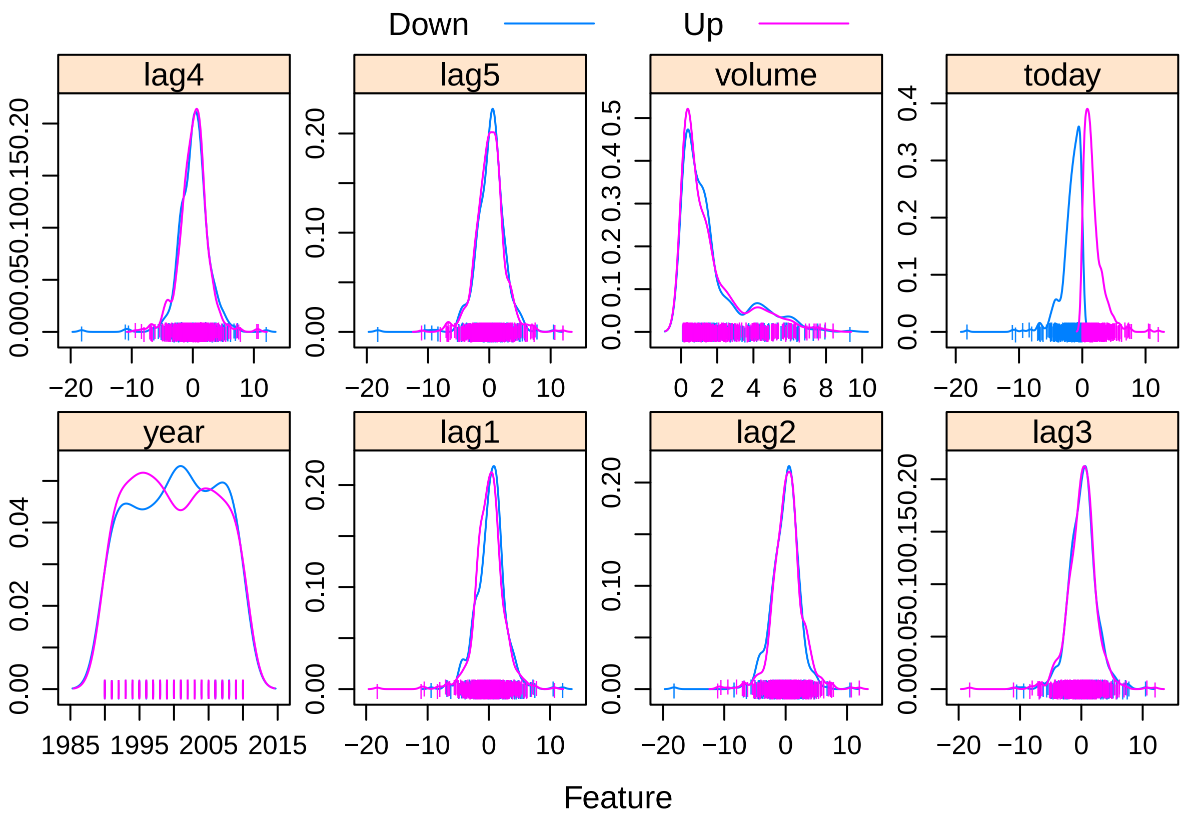
4/1/2019

```r
library(ISLR)
library(tidyverse)
library(ggplot2)
library(patchwork)
library(MASS)
library(caret)
library(pROC)
```

Problem 1

```r
weekly = Weekly %>%
  janitor::clean_names()
```

```r
#transparentTheme(trans = .4)
featurePlot(x = weekly[, 1:8],
            y = weekly$direction,
            scales = list(x=list(relation="free"),
                          y=list(relation="free")),
            plot = "density", pch = "|",
            auto.key = list(columns = 2))
```



Problem 2

The logistic regression are shown below:

```
set.seed(123)
glm.fit <- glm(direction~lag1 + lag2 + lag3 + lag4 + lag5 + volume,
               data = weekly,
               family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = direction ~ lag1 + lag2 + lag3 + lag4 + lag5 +
##     volume, family = binomial, data = weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## lag1        -0.04127    0.02641  -1.563   0.1181
## lag2         0.05844    0.02686   2.175   0.0296 *
## lag3        -0.01606    0.02666  -0.602   0.5469
## lag4        -0.02779    0.02646  -1.050   0.2937
## lag5        -0.01447    0.02638  -0.549   0.5833
## volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

Lag2 is statistically significant.

Problem 3

Using 0.5 as cutoff for Bayes classifier and evaluating its performance. Confusion matrix is as following:

```
full.pred.prob  <- predict(glm.fit, newdata = weekly,
                           type = "response")
full.pred <- rep("Down", length(full.pred.prob))
full.pred[full.pred.prob>0.5] <- "Up"


# confusionMatrix(data = as.factor(test.pred),
#                  reference = dat$diabetes[-rowTrain])

res = confusionMatrix(data = as.factor(full.pred),
                reference = weekly$direction,
                positive = "Up")
res$table
```

```
##           Reference
## Prediction Down  Up
##       Down   54  48
##       Up    430 557
```

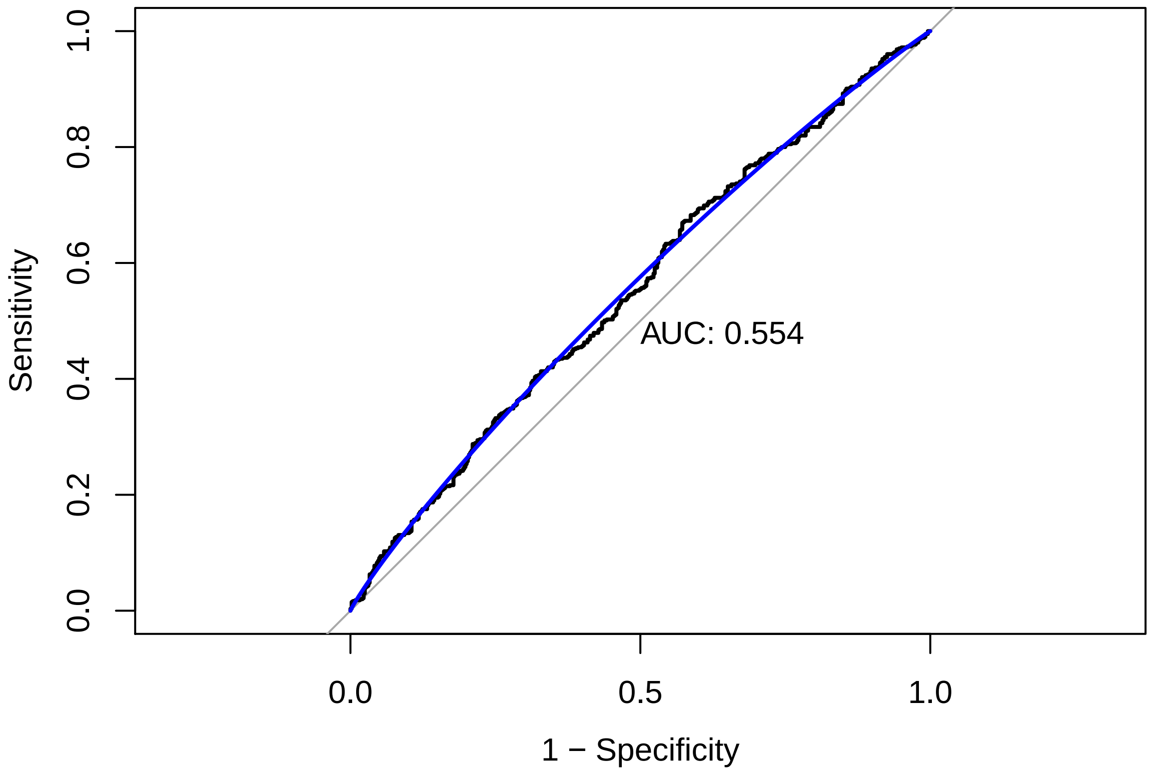
The overall fraction of correct predictions is: $\frac{54+557}{54+48+430+557} = 0.561$

According to the confusion matrix, we can find that only 54 "Down" direction were predicted properly, but 557 "Up" direction were predicted properly and only 48 "Up" direction were predicted to be "Down", but 430 "Down" direction were predicted to be "Up" direction. This means more direction are predicted as "Up" than "Down". This classifier have high sensitivity and low specificity.

Problem 4

Plot the ROC curve and calculate AUC

```
roc.glm <- roc(weekly$direction, full.pred.prob)
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```



The black curve is ROC curve and AUC equals to 0.554. According to this plot, ROC curve is very close to stright line and value of AUC is close to 0.5, we can know that this classifier doesn't perform well.

Problem 5

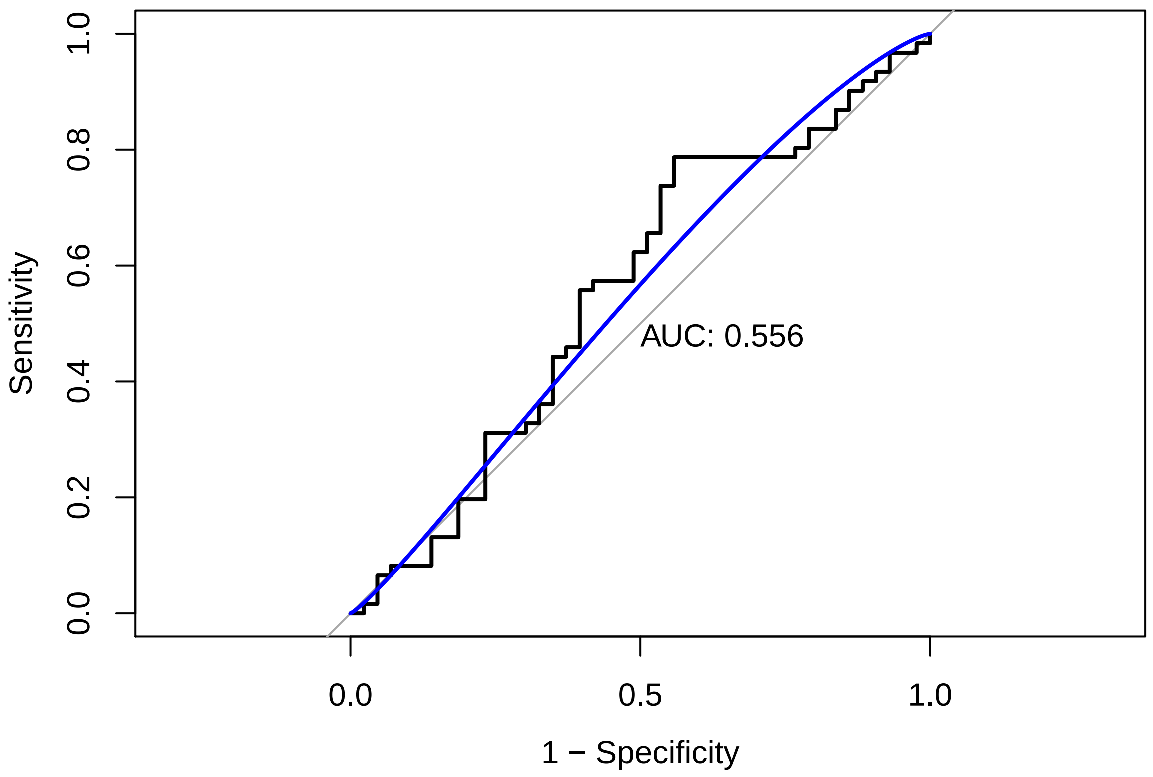
Fit logistic regression model

```r
rowTrain = as.vector(c(1:985))  # get training data
set.seed(123)
glm.fit <- glm(direction~lag1 + lag2,
               data = weekly,
               subset = rowTrain,
               family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = direction ~ lag1 + lag2, family = binomial, data = weekly,
##     subset = rowTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6149  -1.2565   0.9989   1.0875   1.5330
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.21109    0.06456   3.269  0.00108 **
## lag1        -0.05421    0.02886  -1.878  0.06034 .
## lag2         0.05384    0.02905   1.854  0.06379 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1347.0  on 982  degrees of freedom
## AIC: 1353
##
## Number of Fisher Scoring iterations: 4
```

Plot the ROC curve and calculate AUC

```r
test.pred.prob  <- predict(glm.fit, newdata = weekly[-rowTrain,],
                           type = "response")
test.pred <- rep("Down", length(test.pred.prob))
test.pred[test.pred.prob>0.5] <- "Up"
roc.glm <- roc(weekly$direction[-rowTrain], test.pred.prob)
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```

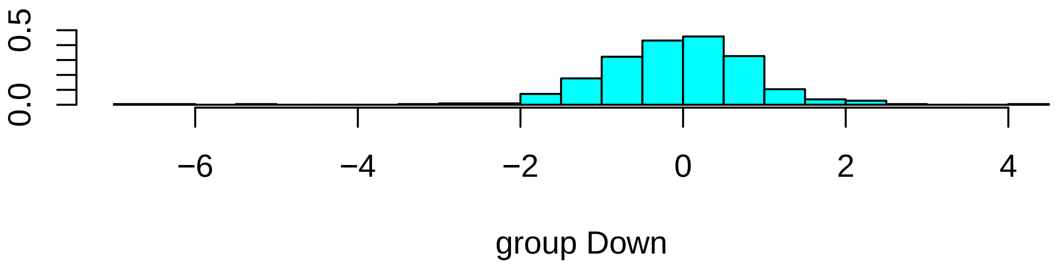


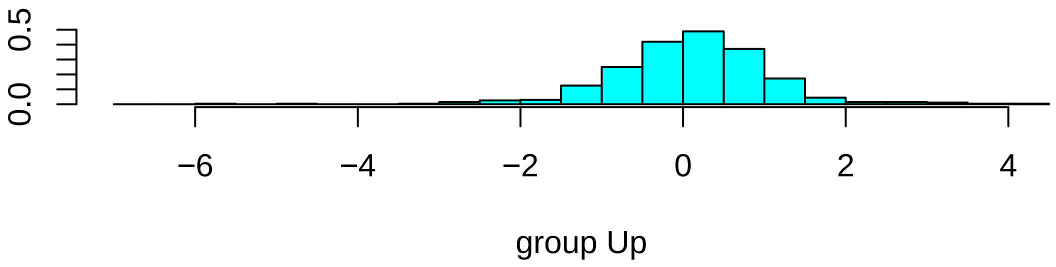
The black curve is ROC curve and AUC equals to 0.556. According to this plot, the value of AUC is close to 0.5, we can know that this classifier doesn't perform well.

Problem 6

LDA

```
lda.fit <- lda(direction~lag1 + lag2, data = weekly,
               subset = rowTrain)
plot(lda.fit)
```



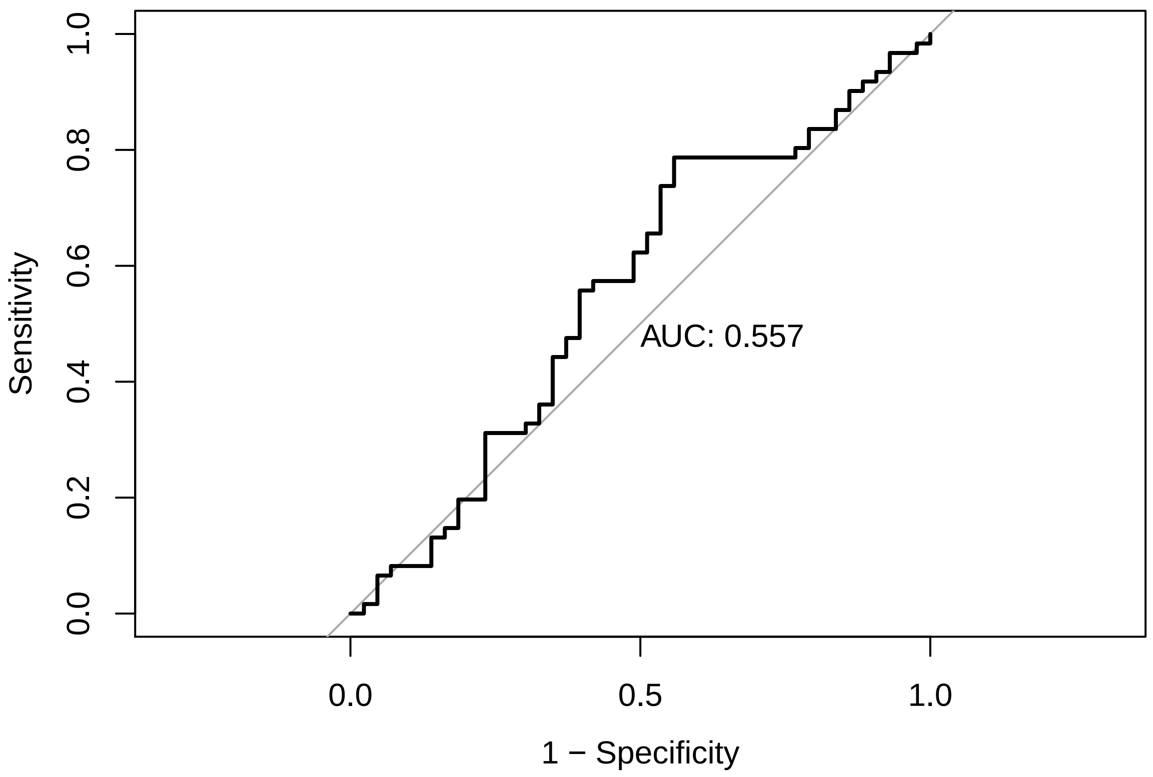


Evaluate the test set performance using ROC.

```r
lda.pred <- predict(lda.fit, newdata = weekly[-rowTrain,])
#head(lda.pred$posterior)

roc.lda <- roc(weekly$direction[-rowTrain], lda.pred$posterior[,2],
               levels = c("Down", "Up"))

plot(roc.lda, legacy.axes = TRUE, print.auc = TRUE)
```



The black curve is ROC curve and AUC equals to 0.557.

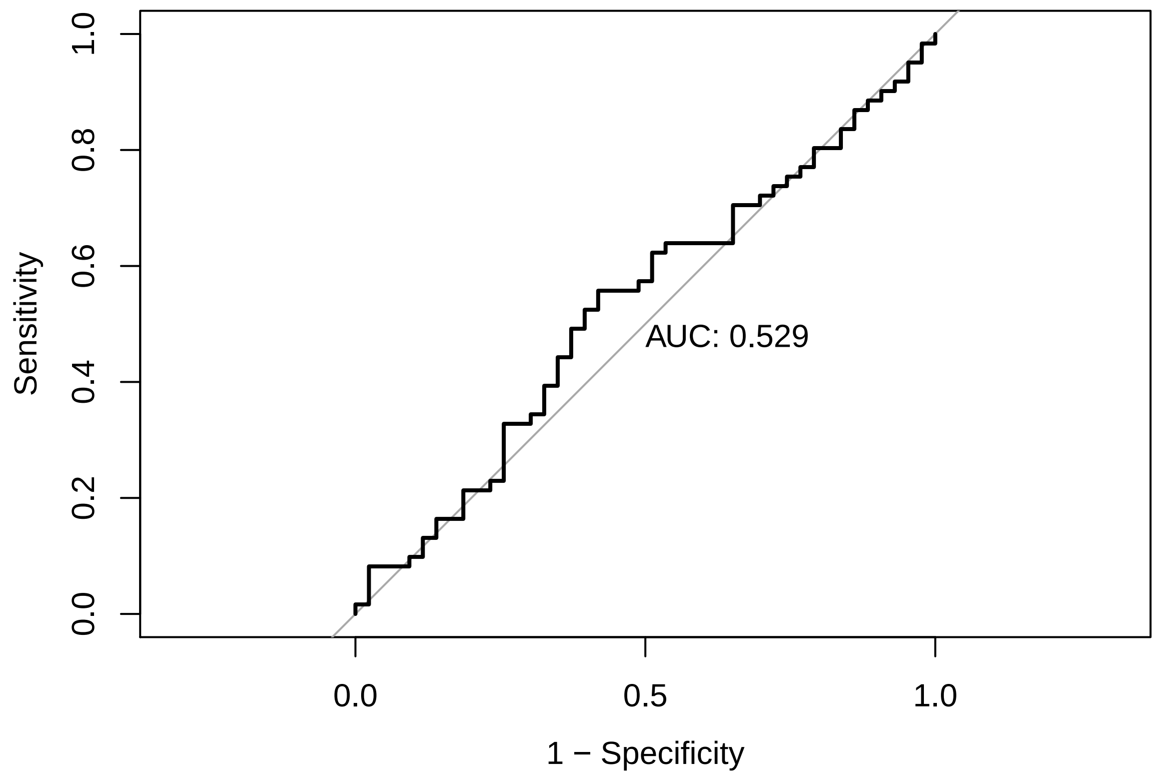
QDA

```
qda.fit <- qda(direction~lag1 + lag2, data = weekly,
               subset = rowTrain)
```

Evaluate the test set performance using ROC.

```
qda.pred <- predict(qda.fit, newdata = weekly[-rowTrain,],type="prob")
#head(qda.pred$posterior)
roc.qda <- roc(weekly$direction[-rowTrain], qda.pred$posterior[,2],
               levels = c("Down", "Up"))

plot(roc.qda, legacy.axes = TRUE, print.auc = TRUE)
```



The black curve is ROC curve and AUC equals to 0.529.

Problem 7

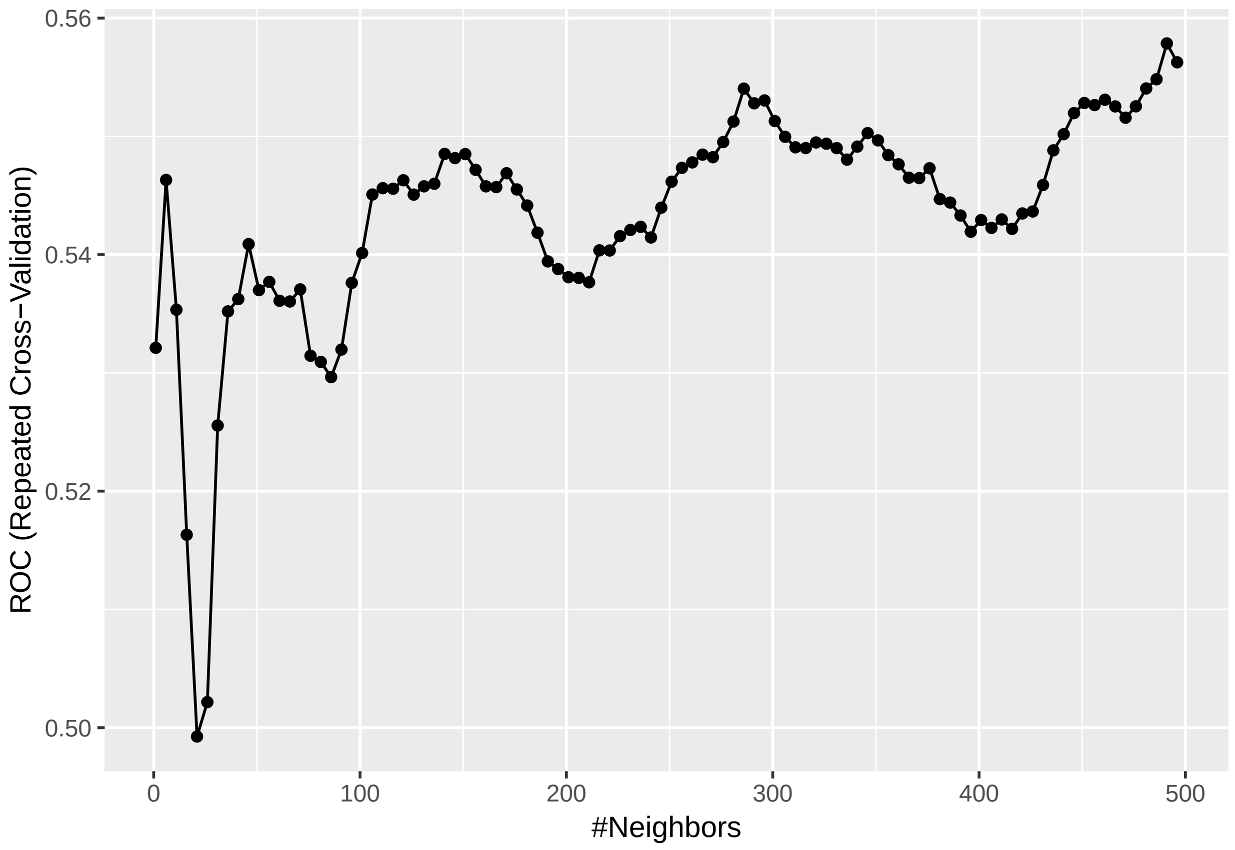
```r
set.seed(123)

ctrl <- trainControl(method = "repeatedcv",
                     repeats = 5,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

model.knn <- train(direction ~ lag1 + lag2,
                 data = weekly,
               subset = rowTrain,
                 method = "knn",
               trControl = ctrl,
               preProcess = c("center","scale"),
               tuneGrid = data.frame(k = seq(1,500,by=5)))
```

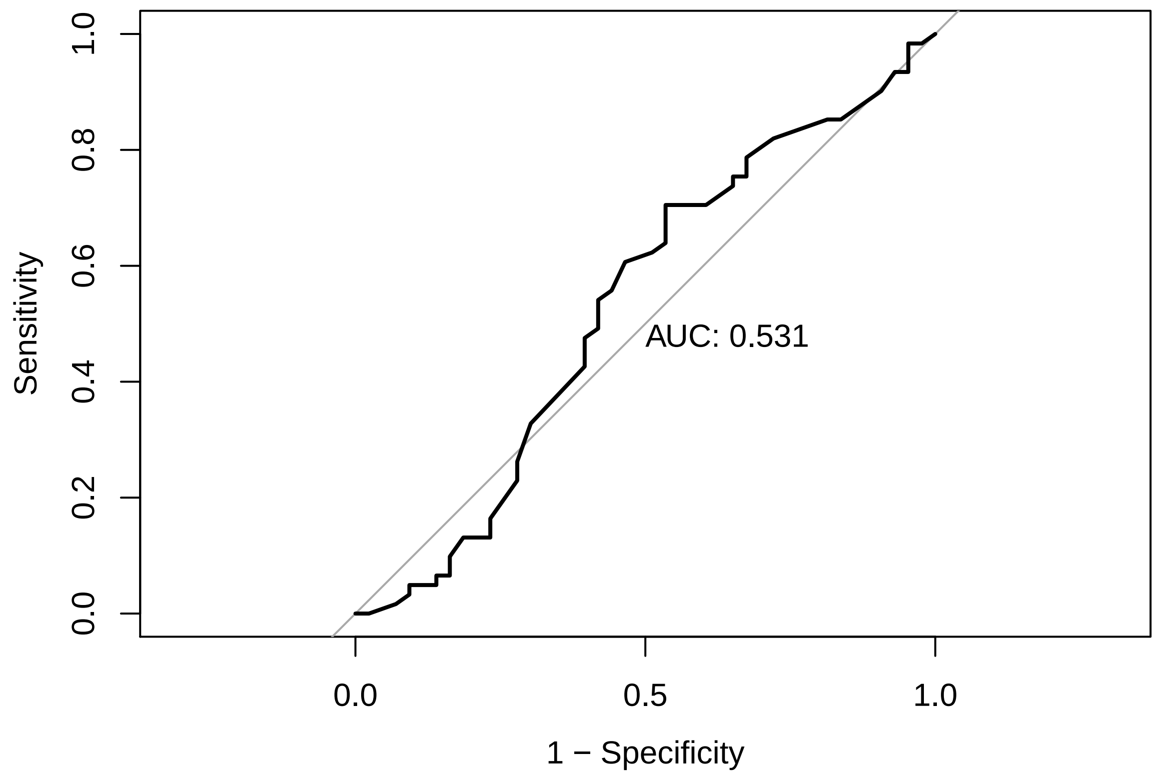
```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was
## not in the result set. ROC will be used instead.
```

```r
ggplot(model.knn)
```



In R, the range of k cannot larger than 500. Even though according to the plot, best k seems to be larger than 500, the knn model is not stable.

```
knn.pred <- predict.train(model.knn, newdata = weekly[-rowTrain,],type="prob")
roc.knn <- roc(weekly$direction[-rowTrain], knn.pred[,"Down"], levels = c("Down", "Up"))
plot(roc.knn, legacy.axes = TRUE, print.auc = TRUE)
```



The black curve is ROC curve and AUC equals to 0.531.

```
set.seed(123)
model.lda <- train(x = weekly[rowTrain,2:3],
                   y = weekly$direction[rowTrain],
                   method = "lda",
                   metric = "ROC",
                   trControl = ctrl)
model.qda <- train(x = weekly[rowTrain,2:3],
                   y = weekly$direction[rowTrain],
                   method = "qda",
                   metric = "ROC",
                   trControl = ctrl)
```

```
res <- resamples(list(LDA = model.lda, QDA = model.qda,KNN = model.knn))
summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: LDA, QDA, KNN
## Number of resamples: 50
##
## ROC
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.4154040 0.5099747 0.5429293 0.5492488 0.5906987 0.6491736    0
## QDA 0.4053030 0.4904748 0.5233586 0.5261994 0.5500397 0.6515152    0
## KNN 0.4330808 0.5190446 0.5609825 0.5578588 0.6026410 0.6545455    0
##
```

```
## Sens
##            Min.    1st Qu.     Median         Mean   3rd Qu.       Max.
## LDA 0.00000000 0.06818182 0.09090909 0.0957171717 0.1136364 0.22727273
## QDA 0.08888889 0.11931818 0.18181818 0.1869191919 0.2272727 0.38636364
## KNN 0.00000000 0.00000000 0.00000000 0.0009090909 0.0000000 0.02272727
##     NA's
## LDA    0
## QDA    0
## KNN    0
##
## Spec
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.7962963 0.8767677 0.9259259 0.9172323 0.9629630 1.0000000    0
## QDA 0.6727273 0.8148148 0.8348485 0.8371246 0.8888889 0.9814815    0
## KNN 0.9818182 1.0000000 1.0000000 0.9989091 1.0000000 1.0000000    0
```

According to ROC, KNN model performs better than LDA and QDA model, but according to specificity and sensitivity, knn models give abnormal performance. According to the ROC and specificity, LDA model performs better than QDA model. So we can choose different models based on requirements of high sensitivity or high specificity.