

# P8106 HOMEWORK 3

xc2474 Xinlei Chen

3/28/2019

## Problem

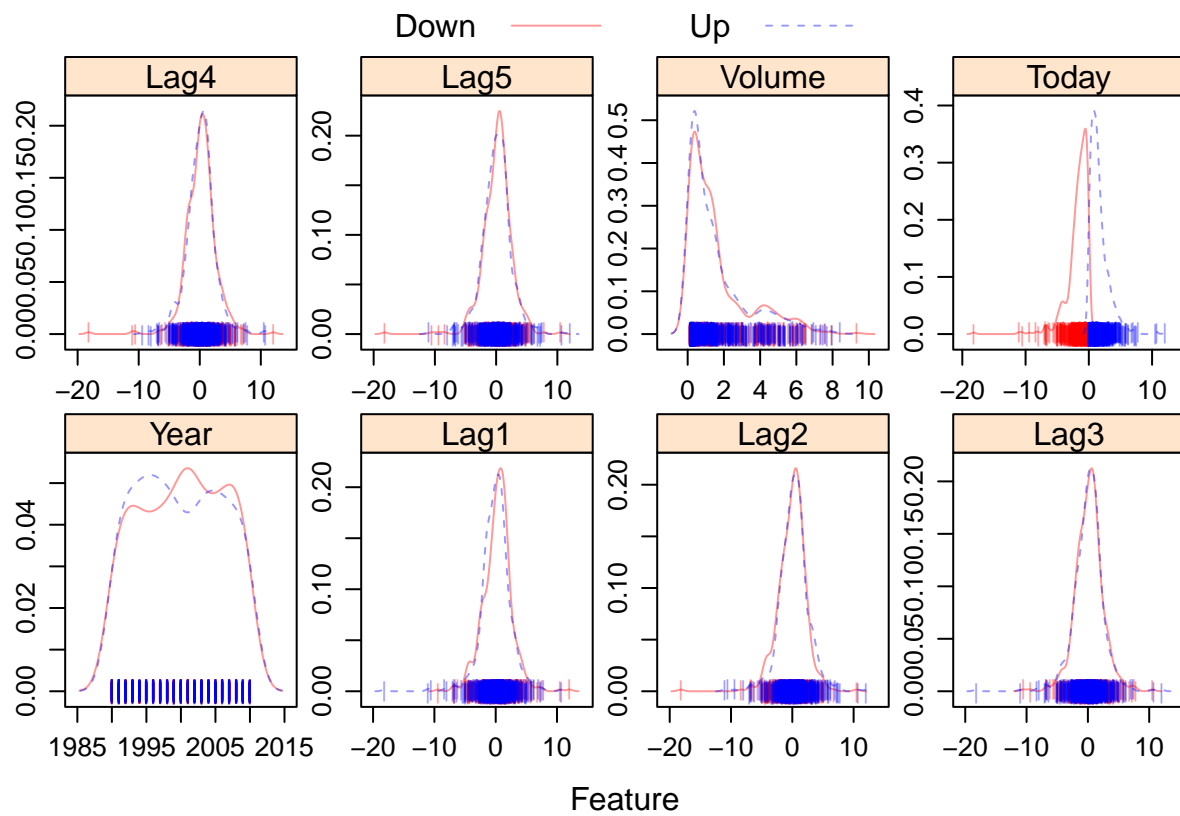
*This questions will be answered using the Weekly data set, which is part of the ISLR package. This data is similar in nature to the Smarket data on the textbook except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010. A description of the data can be found by typing ?Weekly in the Console. (Note that the column Today is not a predictor.)*

```
# load packages
library(tidyverse)
library(ISLR)
library(caret)
library(AppliedPredictiveModeling)
library(pROC)
library(MASS)
library(class)
#import data
data("Weekly")
dat = Weekly
head(dat)
```

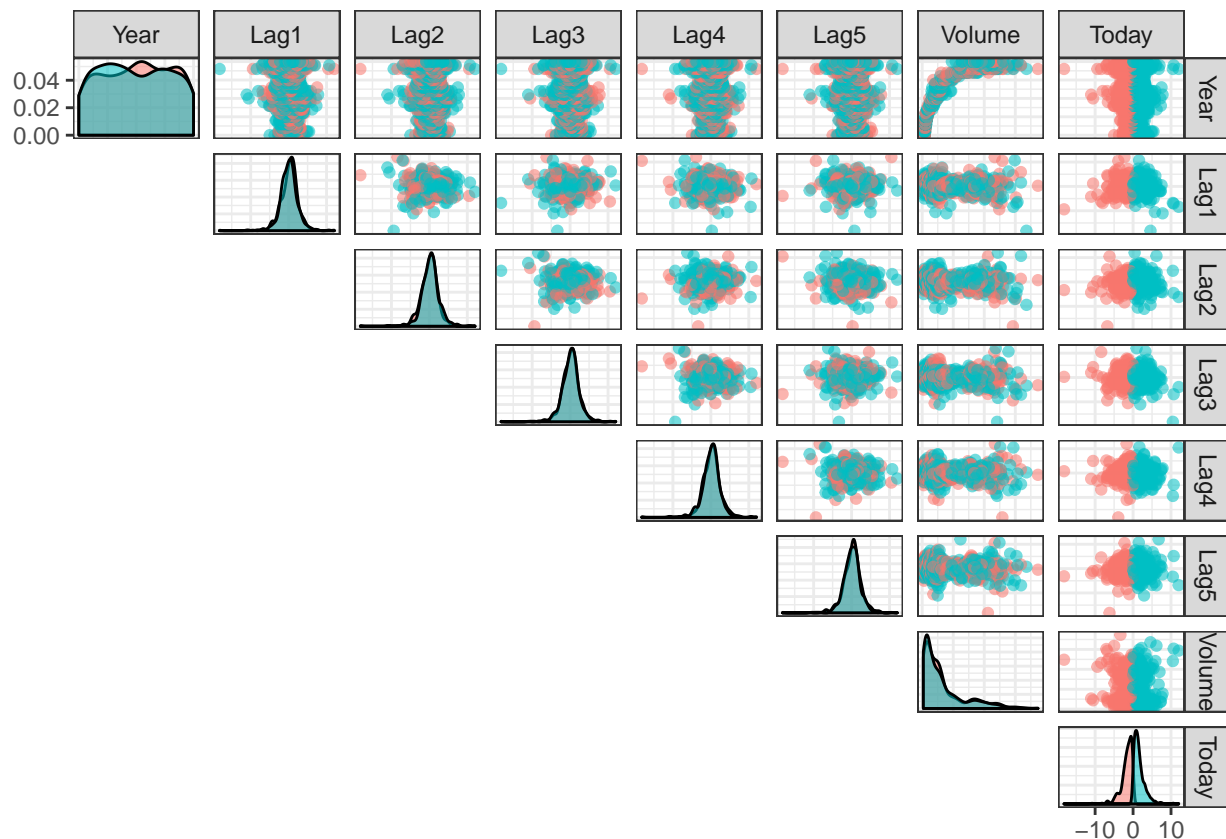
##	Year	Lag1	Lag2	Lag3	Lag4	Lag5	Volume	Today	Direction
## 1	1990	0.816	1.572	-3.936	-0.229	-3.484	0.1549760	-0.270	Down
## 2	1990	-0.270	0.816	1.572	-3.936	-0.229	0.1485740	-2.576	Down
## 3	1990	-2.576	-0.270	0.816	1.572	-3.936	0.1598375	3.514	Up
## 4	1990	3.514	-2.576	-0.270	0.816	1.572	0.1616300	0.712	Up
## 5	1990	0.712	3.514	-2.576	-0.270	0.816	0.1537280	1.178	Up
## 6	1990	1.178	0.712	3.514	-2.576	-0.270	0.1544440	-1.372	Down

(a) Produce some graphical summaries of the Weekly data.

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



```
GGally::ggpairs(dat, columns = 1:8,
  upper = list(continuous = "points"), lower = "blank",
  aes(colour = Direction, alpha = 0.3) ) +
  theme_bw()
```



For each predictors, there is no obvious different pattern of distribution between different direction; even we combine predictors, there is still no clear separated pattern.

(b) Use the full data set to perform a logistic regression with Direction as the response and the five Lag variables plus Volume as predictors. Do any of the predictors appear to be statistically significant? If so, which ones?

```
glm.fit <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
               data=dat, family="binomial")
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##      Volume, family = "binomial", data = dat)
##
## 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 appears to be statistically significant with p-value 0.0296, which is less than 0.05.

(c) Compute the confusion matrix and overall fraction of correct predictions. Briefly explain what the confusion matrix is telling you.

```
test.pred.prob <- predict(glm.fit, type = "response")
test.pred <- rep("Down", length(test.pred.prob))
test.pred[test.pred.prob>0.5] <- "Up"
confusionMatrix(data = as.factor(test.pred), reference = dat$Direction, positive = "Up")
```

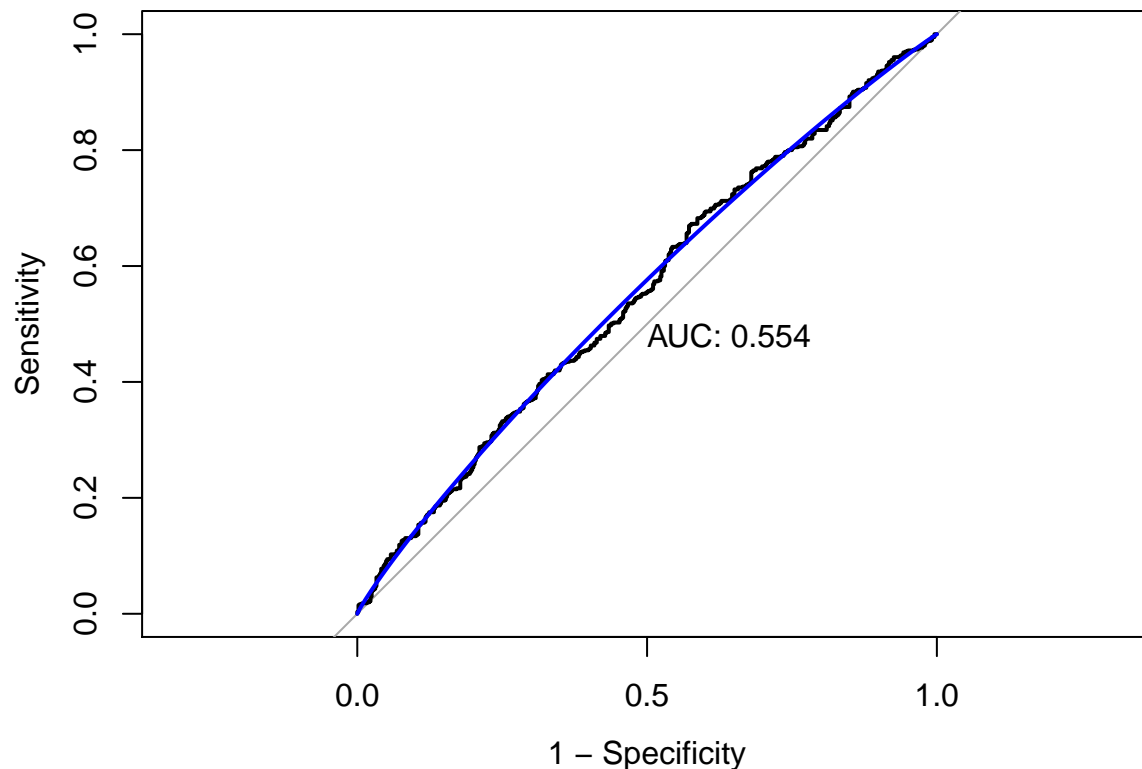
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down  Up
##      Down   54  48
##      Up    430 557
##
##           Accuracy : 0.5611
##           95% CI : (0.531, 0.5908)
##      No Information Rate : 0.5556
##      P-Value [Acc > NIR] : 0.369
##
##           Kappa : 0.035
##
## Mcnemar's Test P-Value : <2e-16
##
##           Sensitivity : 0.9207
##           Specificity : 0.1116
##      Pos Pred Value : 0.5643
##      Neg Pred Value : 0.5294
##           Prevalence : 0.5556
##      Detection Rate : 0.5115
##      Detection Prevalence : 0.9063
##      Balanced Accuracy : 0.5161
##
##      'Positive' Class : Up
##
```

A confusion matrix is a table that is often used to describe the performance of a classification model (or

“classifier”) on a set of test data for which the true values are known. This confusion matrix tells us that (1) the percentage of correct predictions on the training data is 56.11%, or say, the training error rate is 43.89%. (2) For weeks when the market goes up, the model is right 92.07% of the time; for weeks when the market goes down, the model is right 11.16% of the time. (3) The kappa, which represents the agreement between the predictive value and the true value is 0.035.

(d) Plot the ROC curve using the predicted probability from logistic regression and report the AUC.

```
roc.glm <- roc(dat$Direction, test.pred.prob)
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```

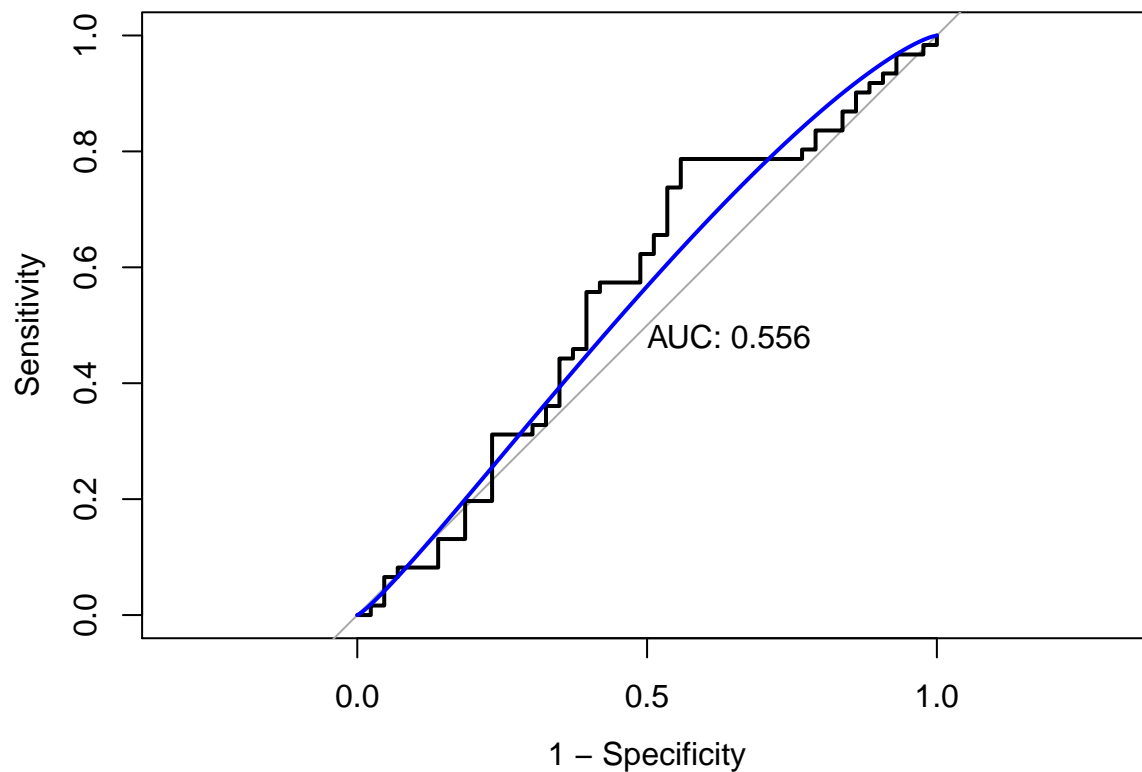


The AUC is 0.554.

(e) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag1 and Lag2 as the predictors. Plot the ROC curve using the held out data (that is, the data from 2009 and 2010) and report the AUC.

```
trainset = (dat$Year<=2008)
testset = dat[!trainset,]

glm.fit.d <- glm(Direction ~ Lag1 + Lag2, data=dat, subset=trainset, family="binomial")
glm.probs.d <- predict(glm.fit.d, type="response", newdata=testset)
roc.glm <- roc(testset$Direction, glm.probs.d)
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```



The AUC is 0.556.

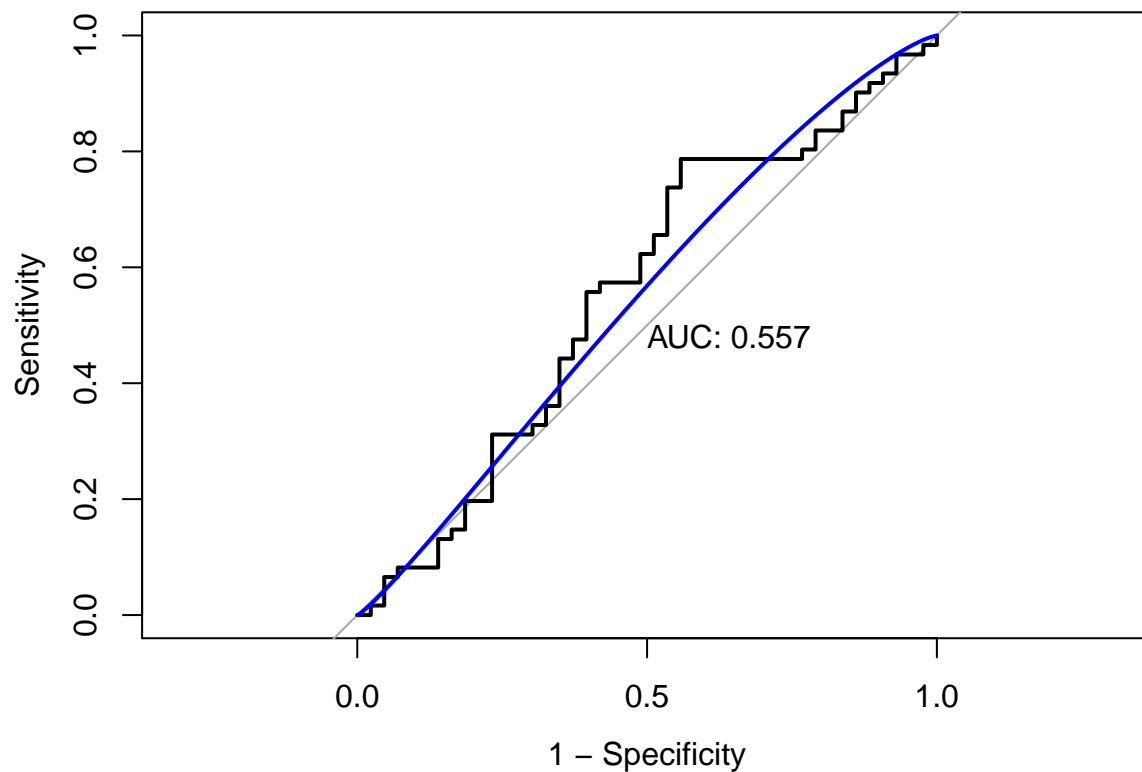
(f) Repeat (e) using LDA and QDA.

```
# LDA
lda.fit <- lda(Direction ~ Lag1 + Lag2, data=dat, subset=trainset)
lda.pred <- predict(lda.fit, newdata = testset)
head(lda.pred$posterior)
```

```
##          Down      Up
## 986 0.5602039 0.4397961
## 987 0.3079163 0.6920837
## 988 0.4458032 0.5541968
## 989 0.4785107 0.5214893
## 990 0.4657943 0.5342057
## 991 0.5262907 0.4737093
```

```
roc.lda <- roc(testset$Direction, lda.pred$posterior[,2],
               levels = c("Down", "Up"))
```

```
plot(roc.lda, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.lda), col = 4, add = TRUE)
```



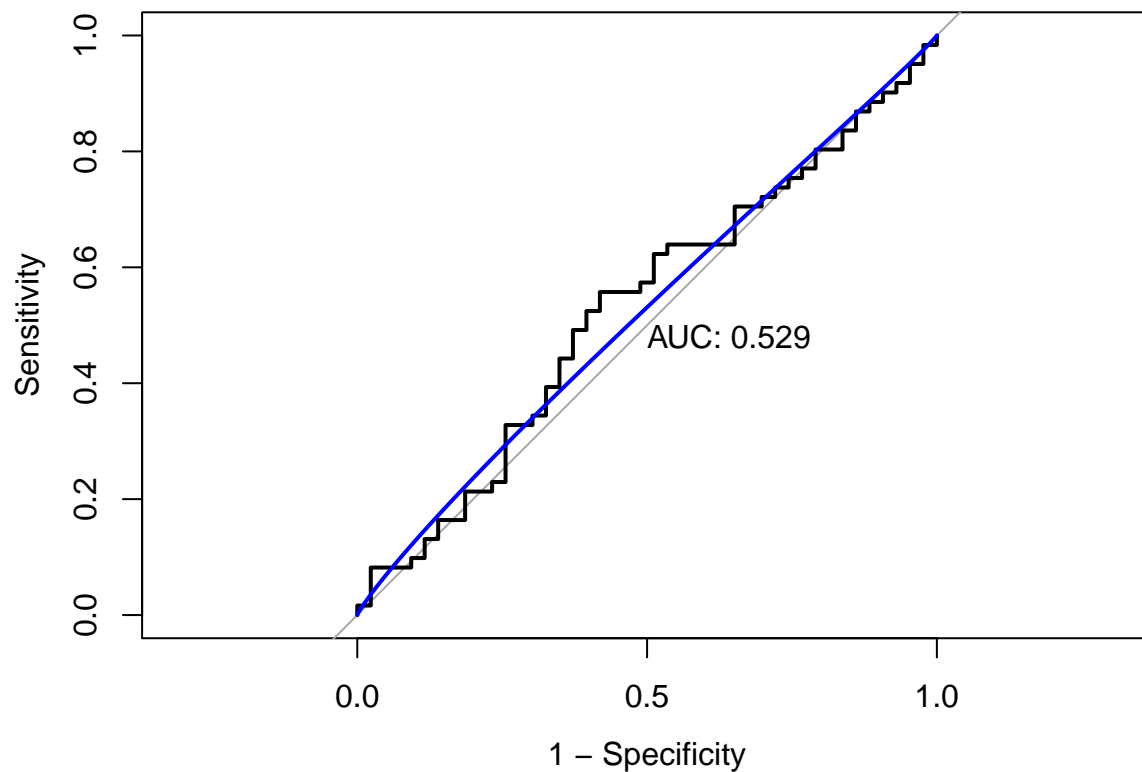
For LDA, the AUC is 0.557.

```
# QDA
qda.fit <- qda(Direction ~ Lag1 + Lag2, data=dat, subset=trainset)
qda.pred <- predict(qda.fit, newdata = testset)
head(qda.pred$posterior)
```

```
##           Down           Up
## 986 0.5436205 0.4563795
## 987 0.3528814 0.6471186
## 988 0.2227273 0.7772727
## 989 0.3483016 0.6516984
## 990 0.4598550 0.5401450
## 991 0.5119613 0.4880387
```

```
roc.qda <- roc(testset$Direction, qda.pred$posterior[,2],
               levels = c("Down", "Up"))

plot(roc.qda, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.qda), col = 4, add = TRUE)
```



For QDA, the AUC is 0.529.

(g) Repeat (e) using KNN. Briefly discuss your results.

```
# choose the best K
train = dat %>%
  filter(Year<=2008)

test = dat %>%
  filter(Year>2008)

set.seed(123123)

ctrl <- trainControl(method="repeatedcv",repeats = 3,classProbs=TRUE,
  summaryFunction = twoClassSummary)

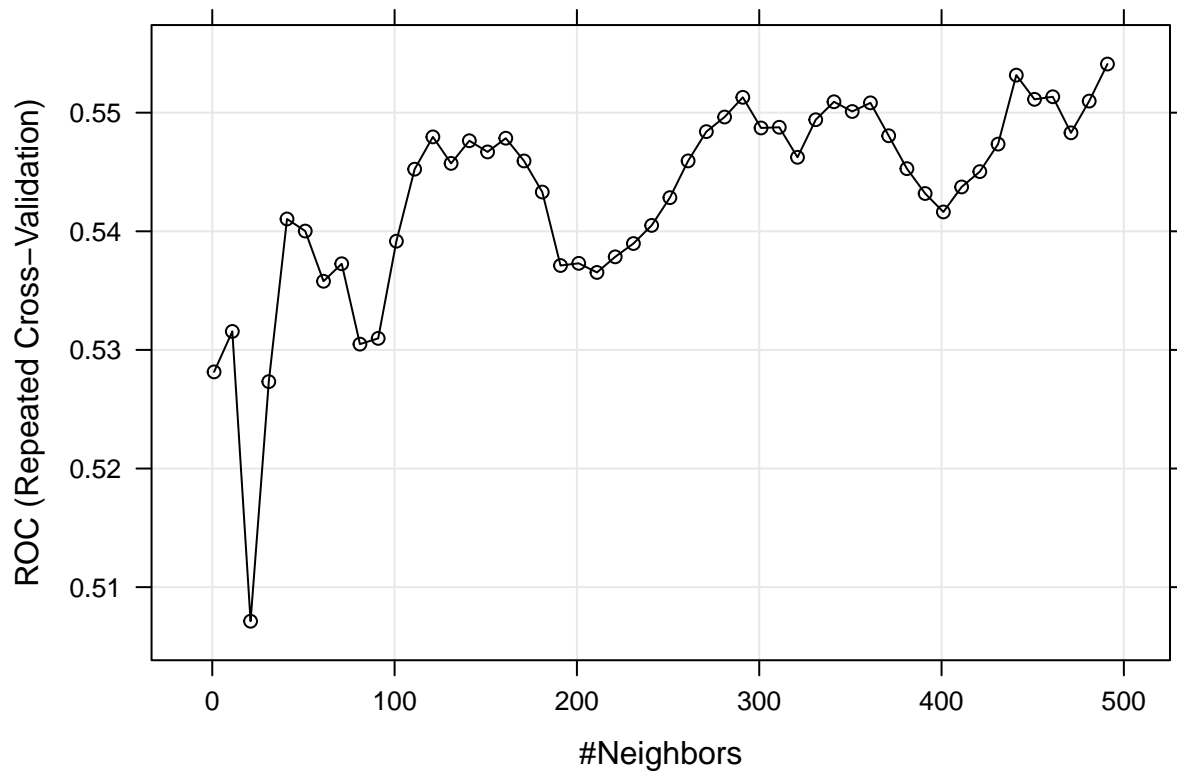
knnFit <- train(Direction ~ Lag1 + Lag2, data = train,
  method = "knn",
  trControl = ctrl,
  preProcess = c("center","scale"),
  tuneGrid = data.frame(k = seq(1,500,by=10))
)

knnFit$bestTune

##      k
## 50 491
```



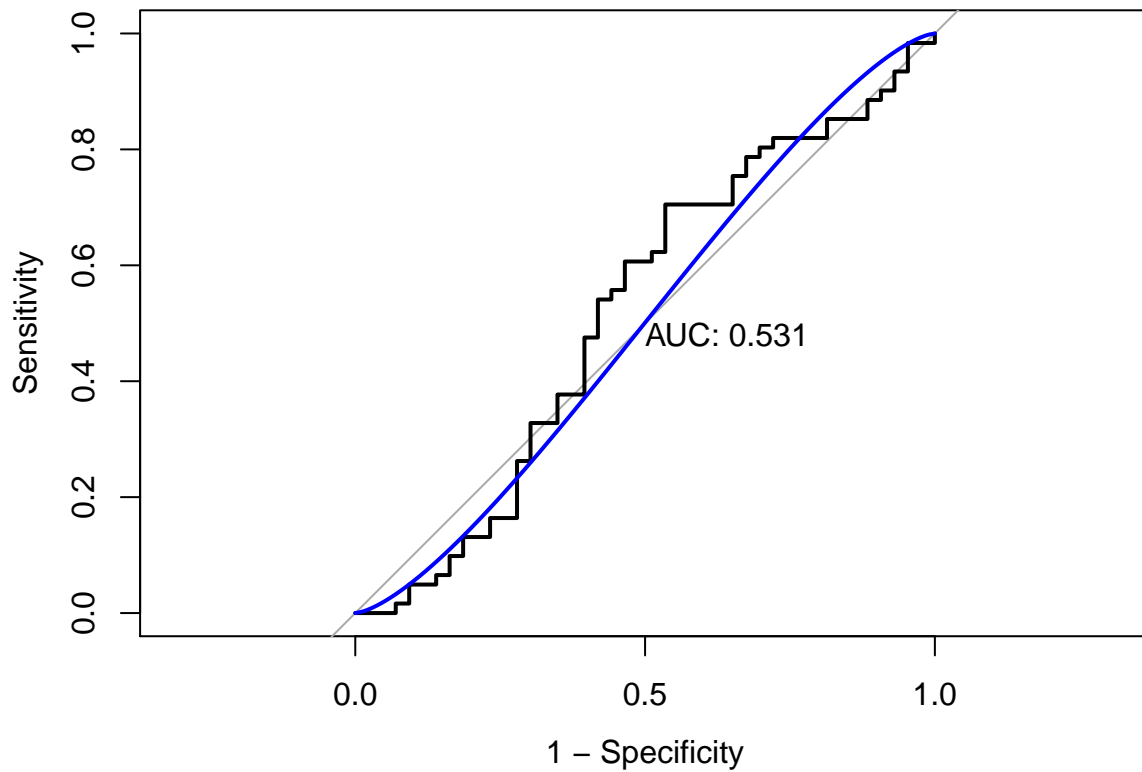
```
plot(knnFit)
```



```
knnPredict <- predict.train(knnFit, newdata = test , type="prob")
knnROC <- roc(test$Direction, knnPredict[, "Up"], levels = c("Down", "Up"))
knnROC

##
## Call:
## roc.default(response = test$Direction, predictor = knnPredict[,      "Up"], levels = c("Down", "Up"))
##
## Data: knnPredict[, "Up"] in 43 controls (test$Direction Down) < 61 cases (test$Direction Up).
## Area under the curve: 0.5309

plot(knnROC, type="S", legacy.axes = TRUE, print.auc=T)
plot(smooth(knnROC), col = 4, add = TRUE)
```



The AUC is 0.531. However, when doing the KNN, we choose  $k$  through cv and the range of  $k$  to choose should be from 1 to the size of training data (say, 985), since KNN identifies  $k$  records in the training data that are the “nearest” in similarity. However, it seems that the range of  $k$  for cv cannot be too large otherwise R will give error, so we choose 1-500 to find the best  $k$ .

**Result:** Higher the AUC, better the model is at successfully predicting classification. So judging from AUC, LDA model does the best prediction and QDA is the worst. However, the AUC's of all 4 models are close to each other and they are all slightly greater than 0.5, meaning all 4 models cannot predict categorical response Direction well.