P8106 HW 3

Minjie Bao

Contents

```r
library(ISLR)
library(tidyverse)
library(caret)
library(AppliedPredictiveModeling)
library(glmnet)
library(e1071)
library(pROC)
library(MASS)
library(mlbench)
library(class)
library(klaR)
```

Data preparation

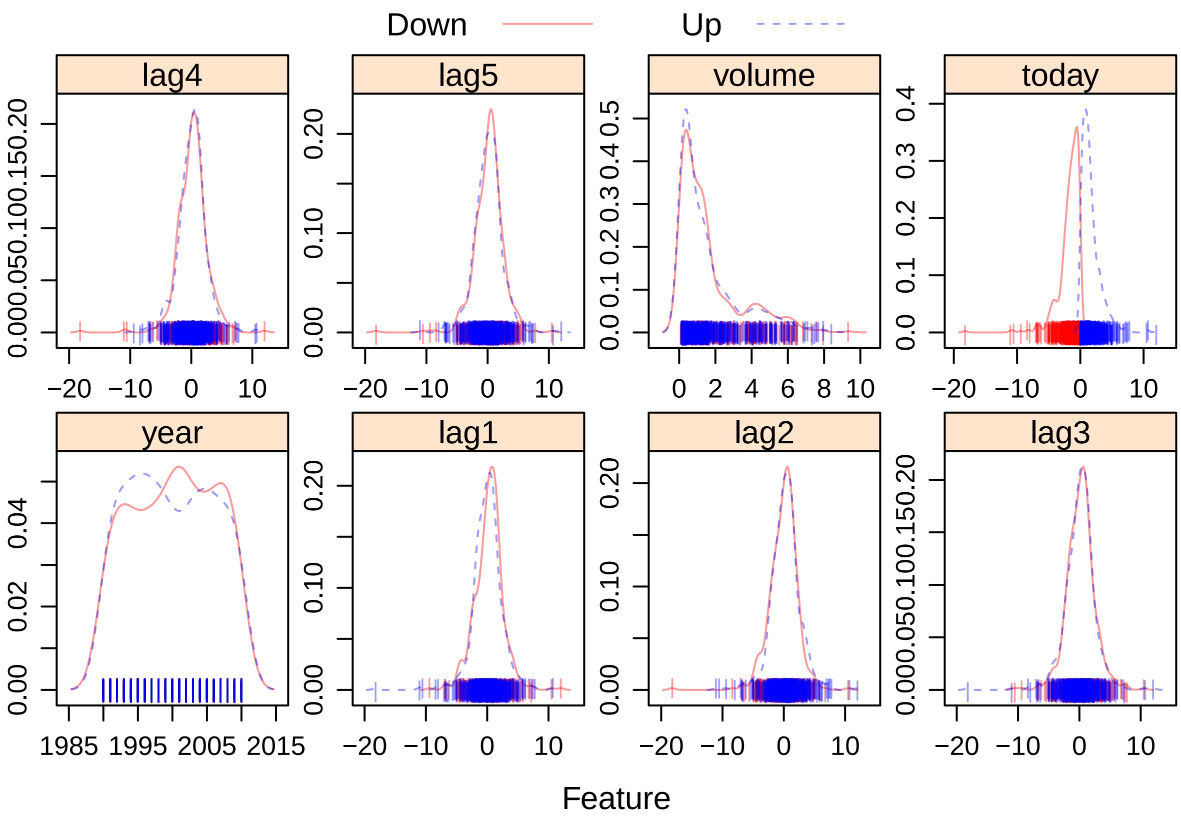
```r
data("Weekly")
weekly_df = Weekly %>%
  janitor::clean_names()
#head(weekly_df)
#skimr::skim(weekly_df)
summary(weekly_df)
```

```
##       year           lag1               lag2               lag3         
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       lag4               lag5              volume            today         
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540  
##  Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410  
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499  
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050  
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260  
##  direction 
##  Down:484  
##  Up  :605  
##            
##            
##            
##            
```

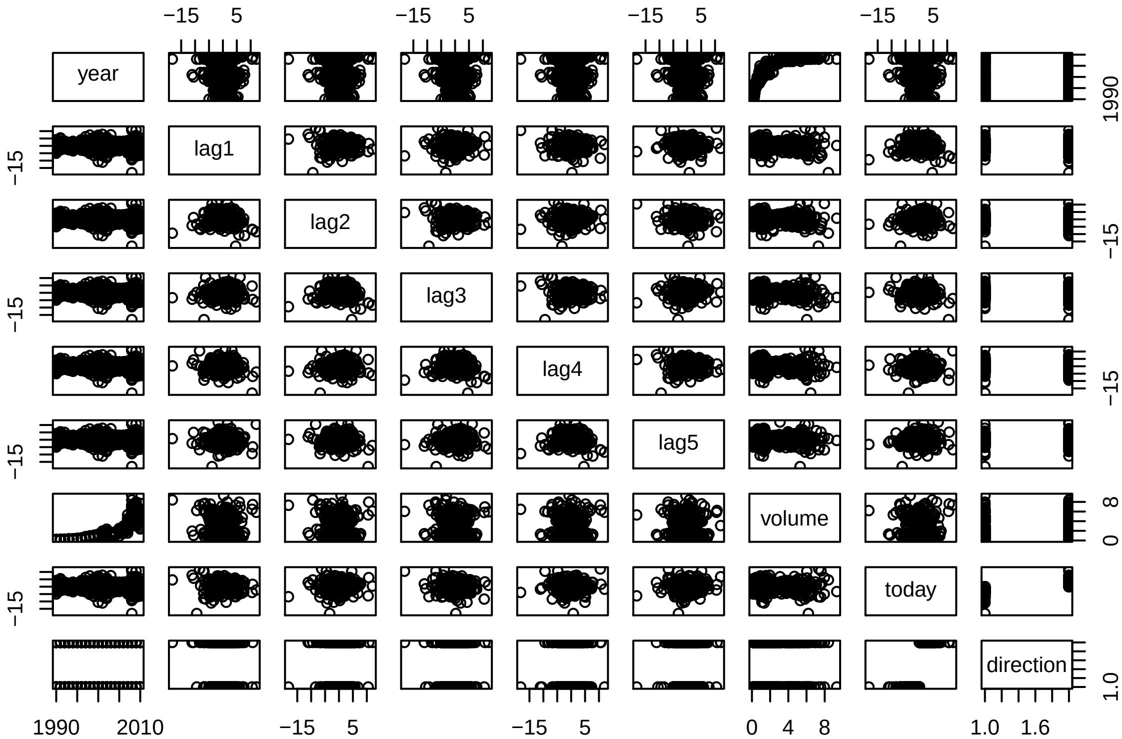
(a) Produce some graphical summaries of the Weekly data

```r
# density plot
transparentTheme(trans = .4)
```

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




```
# pairs scatterplot
pairs(weekly_df)
```



(b) logistic regression and confusion matrix

Use the data from 1990 to 2008 as the training data and the held-out data as the test data. 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? Compute the confusion matrix and overall fraction of correct predictions using the test data. Briefly explain what the confusion matrix is telling you.

```
# divide data into train and test
row_train = weekly_df$year<=2008
row_test = weekly_df[!row_train,]

# logistic regression
glm.fit = glm(direction~lag1+lag2+lag3+lag4+lag5+volume,
                data = weekly_df,
                subset = row_train,
                family = binomial(link = 'logit'))
summary(glm.fit)
```

```
##
## Call:
## glm(formula = direction ~ lag1 + lag2 + lag3 + lag4 + lag5 +
##     volume, family = binomial(link = "logit"), data = weekly_df,
##     subset = row_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -1.7186  -1.2498   0.9823   1.0841   1.4911
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.33258    0.09421   3.530 0.000415 ***
## lag1        -0.06231    0.02935  -2.123 0.033762 *
## lag2         0.04468    0.02982   1.499 0.134002
## lag3        -0.01546    0.02948  -0.524 0.599933
## lag4        -0.03111    0.02924  -1.064 0.287241
## lag5        -0.03775    0.02924  -1.291 0.196774
## volume      -0.08972    0.05410  -1.658 0.097240 .
## ---
## 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: 1342.3  on 978  degrees of freedom
## AIC: 1356.3
##
## Number of Fisher Scoring iterations: 4
```

```r
contrasts(weekly_df$direction)
```

```
##      Up
## Down  0
## Up    1
```

```r
# confusion matrix
test_pred_prob <- predict(glm.fit, newdata = weekly_df[-row_train,],
                           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 = weekly_df$direction[-row_train],
                positive = "Up")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down  Up
##       Down  111 114
##       Up    372 491
##
##                Accuracy : 0.5533
##                  95% CI : (0.5232, 0.5831)
##     No Information Rate : 0.5561
##     P-Value [Acc > NIR] : 0.585
##
##                   Kappa : 0.0437
##
```

```
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8116
##             Specificity : 0.2298
##          Pos Pred Value : 0.5689
##          Neg Pred Value : 0.4933
##              Prevalence : 0.5561
##          Detection Rate : 0.4513
##    Detection Prevalence : 0.7932
##       Balanced Accuracy : 0.5207
##
##        'Positive' Class : Up
##
```

From the logistic regression summary output, we can see that only lag1 is significant with p-value = 0.0338 < 0.05.

From the confusion matrix:

The accuracy is 0.5533, which means the overall fraction of correct prediction is 0.5533 with 95% CI between 0.5232 and 0.5831.

The NIR (No Information Rate) is 0.5865, which means the fraction of "Up" class in both predicted and trained dataset is 0.5865.

The p-value is 0.585 > 0.05, which means we failed to reject the null hypothesis and conclude that accuracy is equal to no information rate.

The kappa value is 0.0437, which means the agreement between the predictive value and the true value is 0.0437. A kappa value of 1 represents perfect agreement, while a value of 0 represents no agreement.

The sensitivity is 0.8116, measures the proportion of actual positives that are correctly identified TP/(TP+FN).

The specificity is 0.2298, measures the proportion of actual negative that are correctly identified TN/(FP+TN).

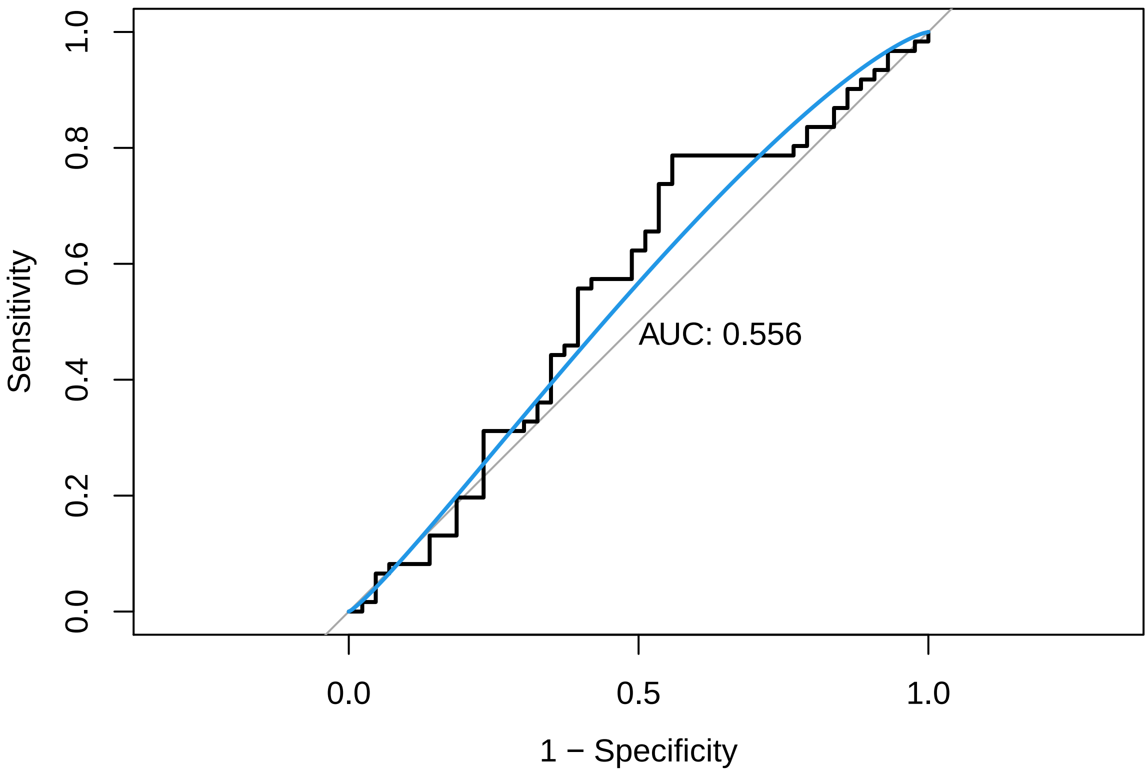
(c) logistic regression, ROC curve and AUC

Now fit the logistic regression model using the training data period from 1990 to 2008, with Lag1 and Lag2 as the predictors. Plot the ROC curve using the test data and report the AUC.

```r
# fit regression using training data
glm.fit_train = glm(direction ~ lag1 + lag2,
                    data = weekly_df,
                    subset = row_train,
                    family = binomial)

# predict using test data
test.pred.prob = predict(glm.fit_train, newdata = row_test, type = "response")

# plot ROC curve and report AUC
roc.glm <- roc(row_test$direction, test.pred.prob)
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```

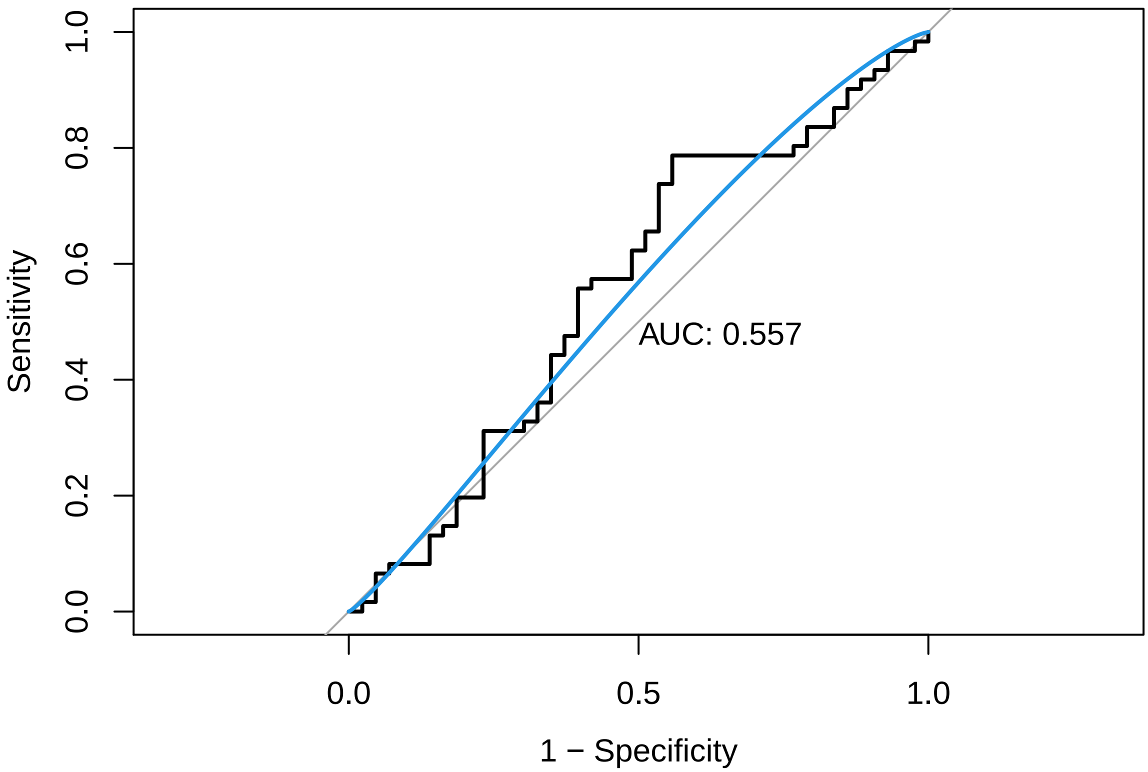


AUC for GLM is 0.556.

(d) Repeat (c) using LDA and QDA.

LDA

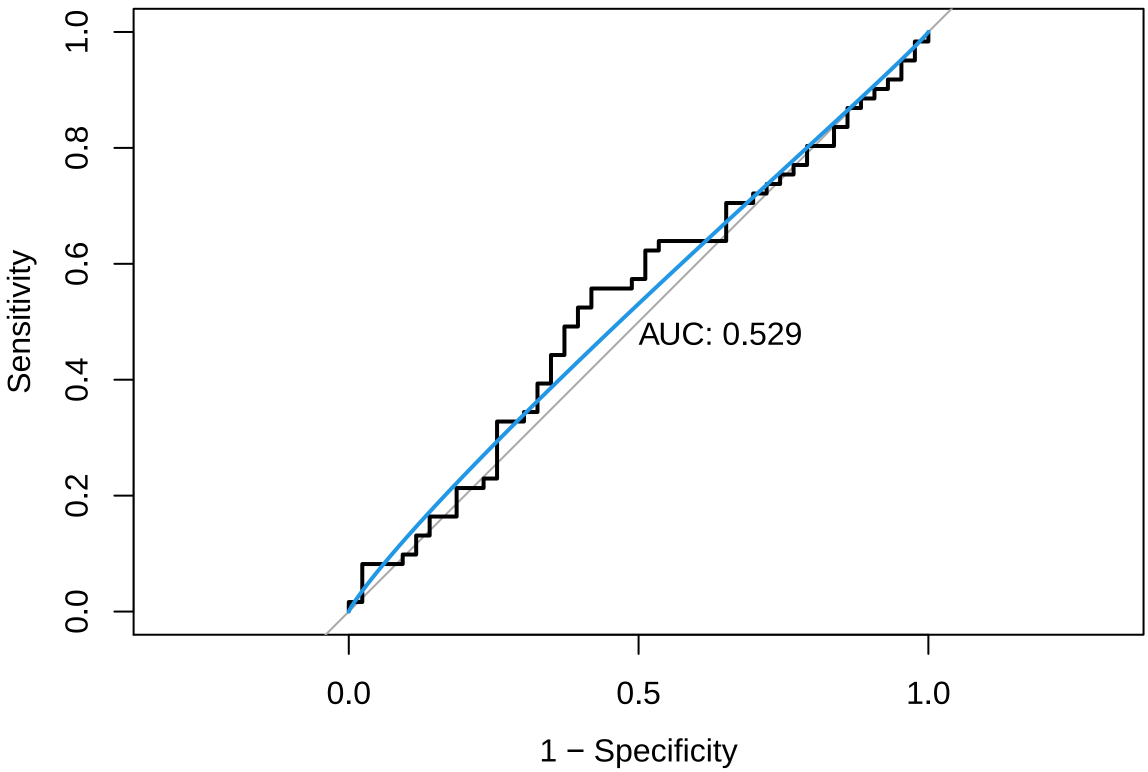
```
# fit model on training and predict on test
lda.fit = lda(direction ~ lag1 + lag2,
              data = weekly_df,
              subset = row_train)
lda.pred = predict(lda.fit,
                   newdata = row_test)
# plot ROC curve
roc.lda = roc(row_test$direction, lda.pred$posterior[,2],
           levels = c("Down", "Up"))
plot(roc.lda, legacy.axes = T, print.auc = T)
plot(smooth(roc.lda),col = 4, add = TRUE)
```



AUC for LDA is 0.557.

QDA

```
# fit model on trainning and predict on test
qda.fit = qda(direction ~ lag1 + lag2,
              data = weekly_df,
              subset = row_train)
qda.pred = predict(qda.fit,
                   newdata = row_test)
# plot ROC curve
roc.qda = roc(row_test$direction, qda.pred$posterior[,2],
           levels = c("Down", "Up"))
plot(roc.qda, legacy.axes = T, print.auc = T)
plot(smooth(roc.qda), col = 4, add = TRUE)
```



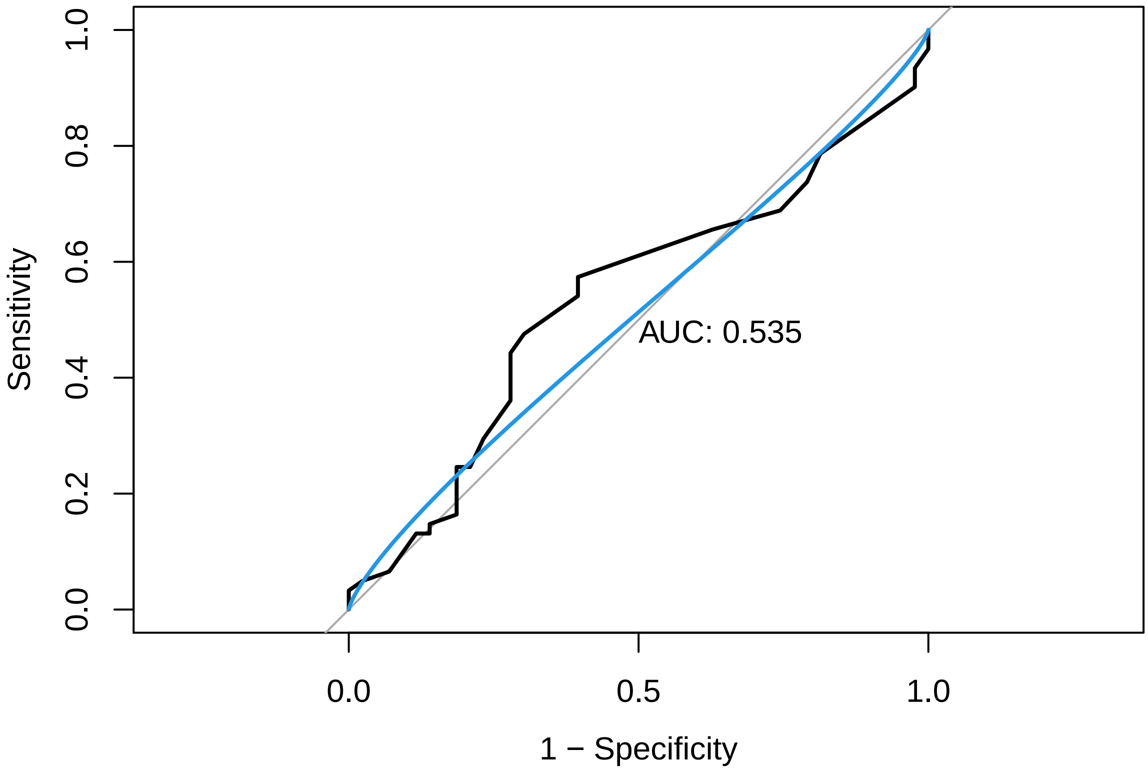
AUC for QDA is 0.529.

(e) Repeat (c) using KNN. Briefly discuss your results in (c) to (e).

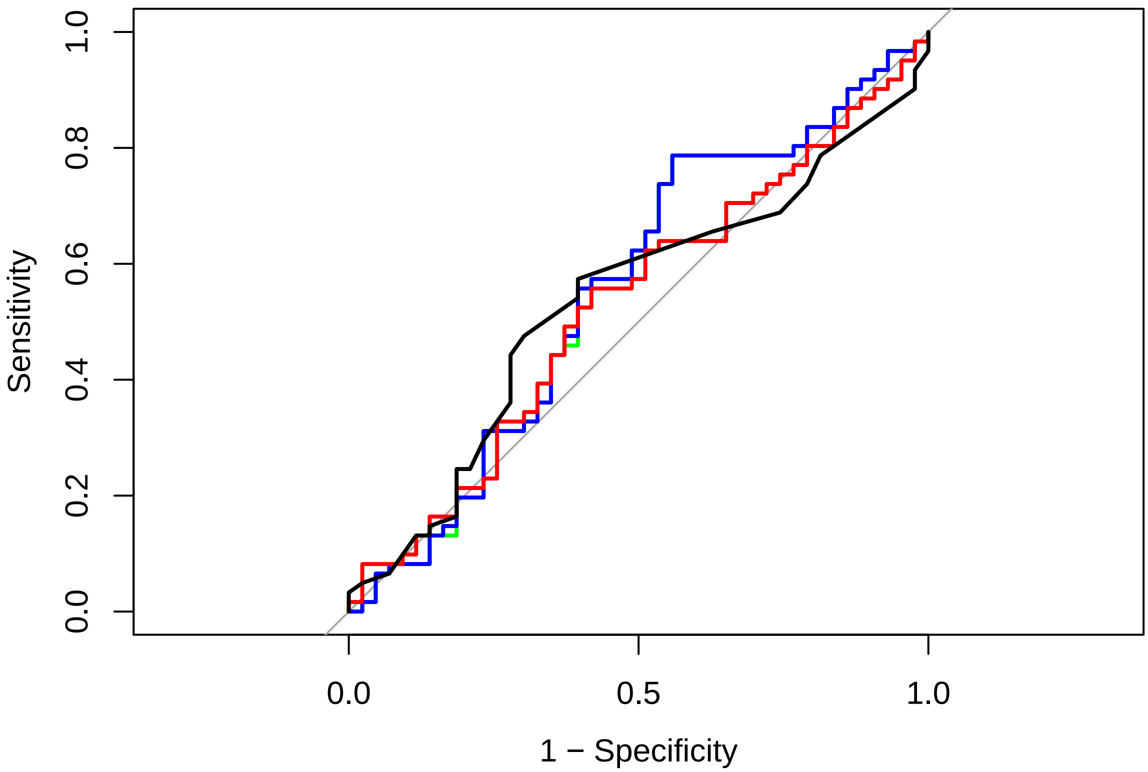
```
set.seed(2)
ctrl <- trainControl(method = "repeatedcv",
                     repeats = 5,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)
knn.fit <- train(x = weekly_df[row_train, 2:3],
                   y = weekly_df$direction[row_train],
                   method = "knn",
                   preProcess = c("center","scale"),
                   tuneGrid = data.frame(k = seq(1,200,by=5)),
                   trControl = ctrl)
summary(knn.fit)
```

```
##             Length Class      Mode
## learn       2      -none-     list
## k           1      -none-     numeric
## theDots     0      -none-     list
## xNames      2      -none-     character
## problemType 1      -none-     character
## tuneValue   1      data.frame list
## obsLevels   2      -none-     character
## param       0      -none-     list
```

```r
# predict on test data
knn.pred = predict(knn.fit, newdata = row_test, type = "prob")
# plot ROC curve
roc.knn = roc(row_test$direction, knn.pred$Up, levels = c("Down", "Up"))
plot(roc.knn, legacy.axes = T, print.auc = T)
plot(smooth(roc.knn),col = 4, add = TRUE)
```




```r
# model comparison
plot(roc.glm, col = "green", legacy.axes = TRUE) #GLM
plot(roc.lda, col = "blue", add = TRUE) #LDA
plot(roc.qda, col = "red", add = TRUE) #QDA
plot(roc.knn, col = "black", add = TRUE) #KNN
```



AUC for KNN is 0.535.

After comparing the AUC and ROC curves among LGM, LDA, QDA and KNN, we can see that LDA has the largest AUC = 0.557. This means the LDA has a better performance at distinguishing between the positive and negative classes than other models. All these models' AUC are close to 0.5, and an AUC of 0.5 suggests no discrimination. From the ROC curves, we can also see that LDA (blue ROC curve) performs better than other models since the closer an ROC curve is to the upper left corner, the more efficient is the test.