Machine Learning Hw3

Ekta Chaudhary

10/04/2020

Weekly S&P Stock Market Data

Description Weekly percentage returns for the S&P 500 stock index between 1990 and 2010.

Format A data frame with 1089 observations on the following 9 variables. Year: The year that the observation was recorded Lag1: Percentage return for previous week Lag2: Percentage return for 2 weeks previous Lag3: Percentage return for 3 weeks previous Lag4: Percentage return for 4 weeks previous Lag5: Percentage return for 5 weeks previous Volume: Volume of shares traded (average number of daily shares traded in billions) Today: Percentage return for this week Direction: A factor with levels Down and Up indicating whether the market had a positive or negative return on a given week

```
library(ISLR)
library(MASS)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

```
library(MASS)
library(e1071)
library(mlbench)
library(pROC)
```

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

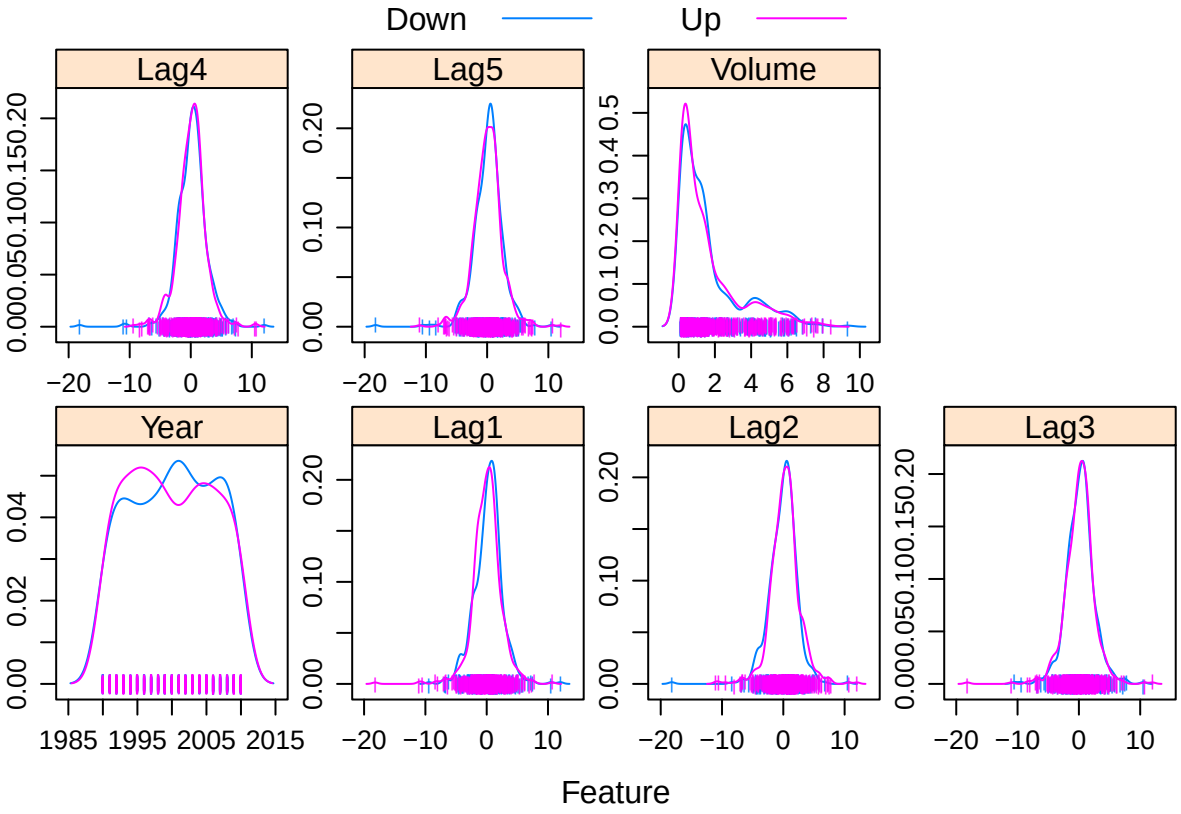
```
##
## Attaching package: 'pROC'
```

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

```
library(AppliedPredictiveModeling)
```

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

```
data(Weekly)

Weekly = Weekly[,-8]

featurePlot(x = Weekly[, 1:7],
            y = Weekly$Direction,
            scales = list(x = list(relation = "free"),
                          y = list(relation = "free")),
            plot = "density", pch = "|",
            auto.key = list(columns = 2))
```



(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?

```
Weekly_dat = Weekly[,-1]
```

```
glm.fit <- glm(Direction~.,
               data = Weekly_dat,
               family = binomial)
```

```r
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ ., family = binomial, data = Weekly_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
```

Lookign at the p-values, we can say that at 5% level of significance, Lag2 is statistically significant.

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

```r
test.pred.prob  <- predict(glm.fit, newdata = Weekly_dat,
                           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_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
##
```

- The Sensitivity is 92.07 % i.e., 92.07 % of True positives were predicted correctly.
- The Specificity is 11.16 % i.e., 11.16 % of True Negatives were predicted correctly.
- The PPV is 56.43 % i.e., the precision is 56.43 %.
- The Kappa value is low at 0.035.

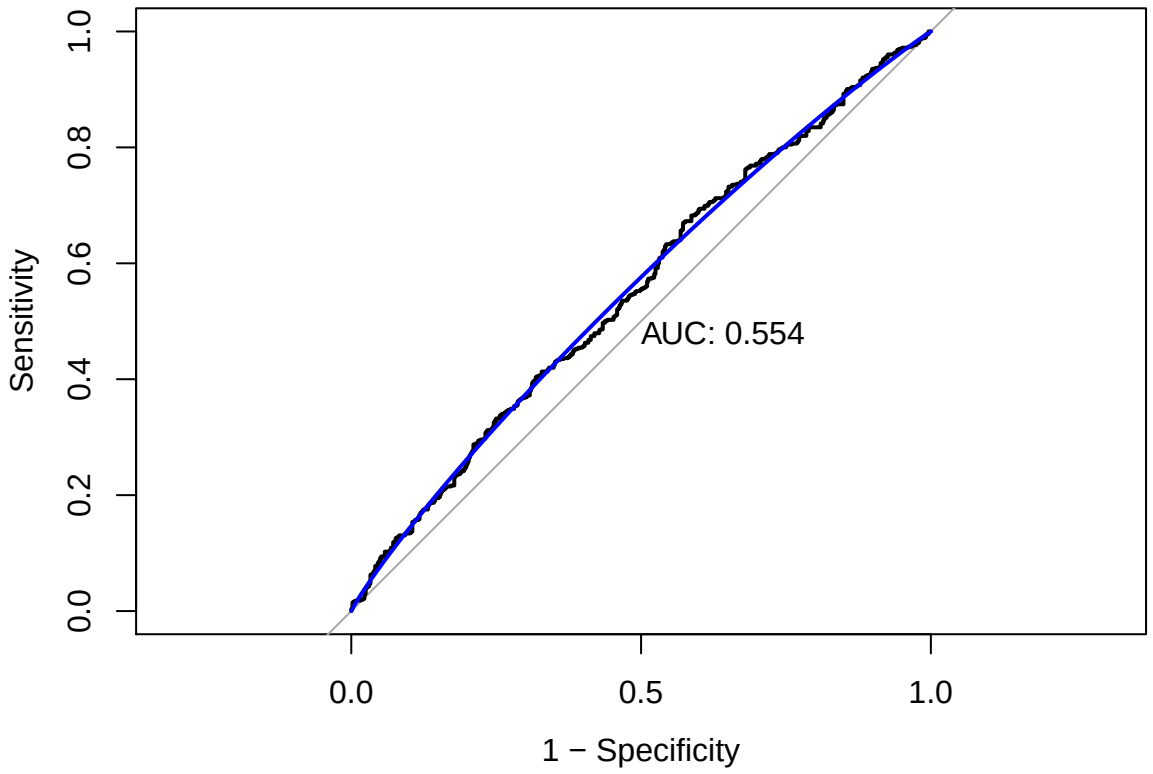
(d) Plot the ROC curve using the predicted probability from logistic regression and report the AUC.

```
roc.glm <- roc(Weekly_dat$Direction, test.pred.prob)
```

```
## Setting levels: control = Down, case = Up
```

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

```
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.

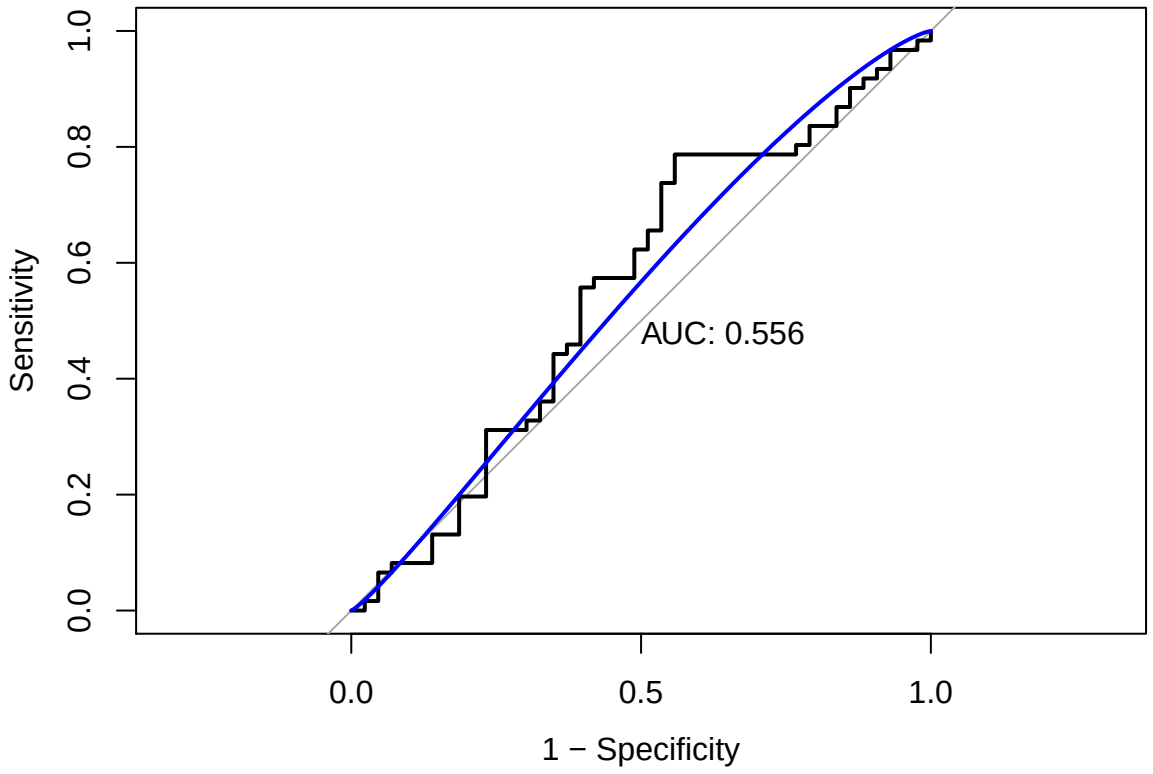
```
train_dat = (Weekly$Year < 2009)
Weekly_dat2 = Weekly_dat[!train_dat, 1:2]
Direction_new = Weekly_dat$Direction[!train_dat]
```

```
glm.fit <- glm(Direction~ Lag1+Lag2,
               data = Weekly_dat,
               family = binomial,
               subset = train_dat)

glm.probs = predict(glm.fit, Weekly_dat2, type = "response")

test.pred <- rep("Down", length(glm.probs))
test.pred[glm.probs > 0.5] <- "Up"
```

```
roc.glm <- roc(Direction_new, glm.probs)
```

```
## Setting levels: control = Down, case = Up

## Setting direction: controls < cases
```

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

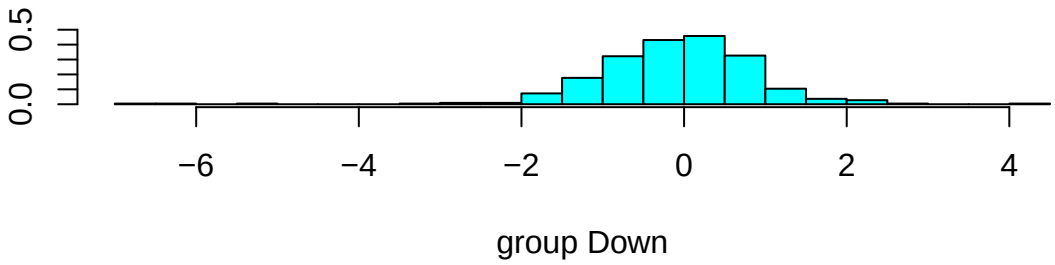


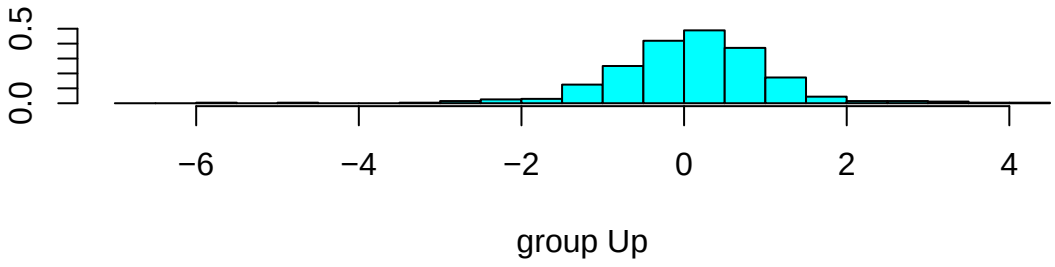

```
# The AUC is 0.556
```

(f) Repeat (e) using LDA and QDA

Using LDA

```
lda.fit <- lda(Direction ~ Lag1+Lag2, data = Weekly_dat,
               subset = train_dat)
plot(lda.fit)
```



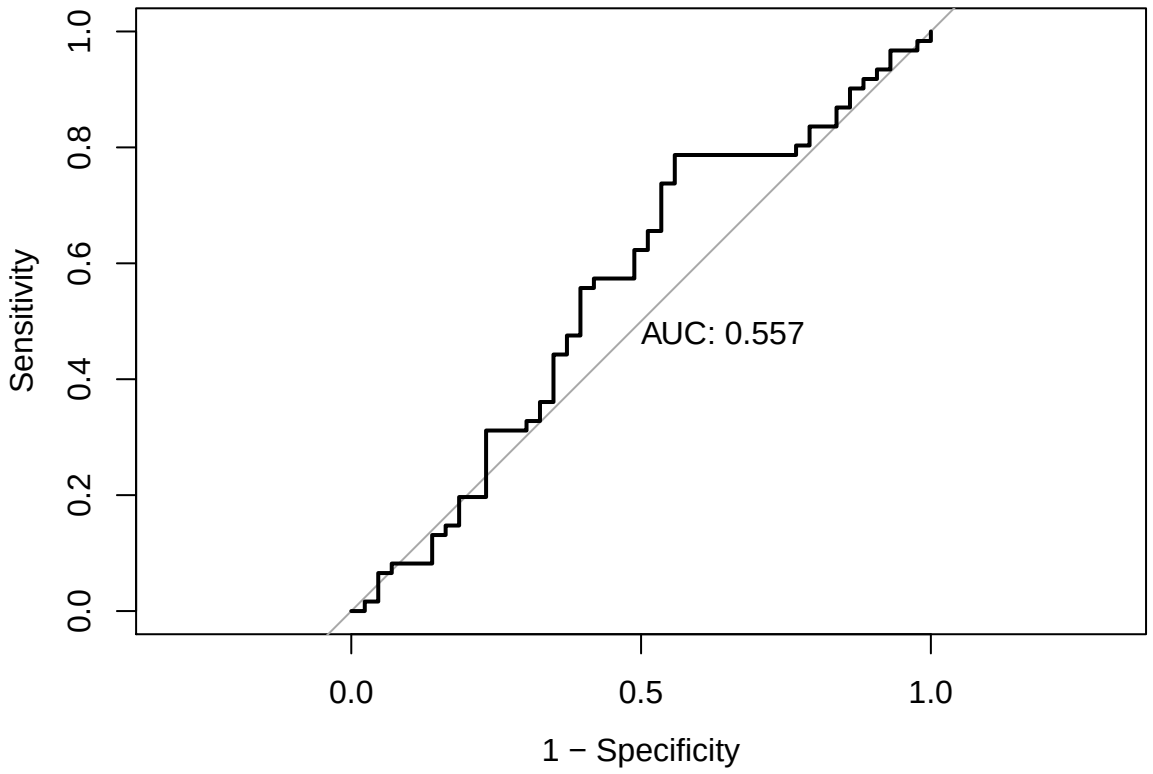


Evaluating the test set performance using ROC.

```
lda.pred <- predict(lda.fit, newdata = Weekly_dat2)

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

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

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



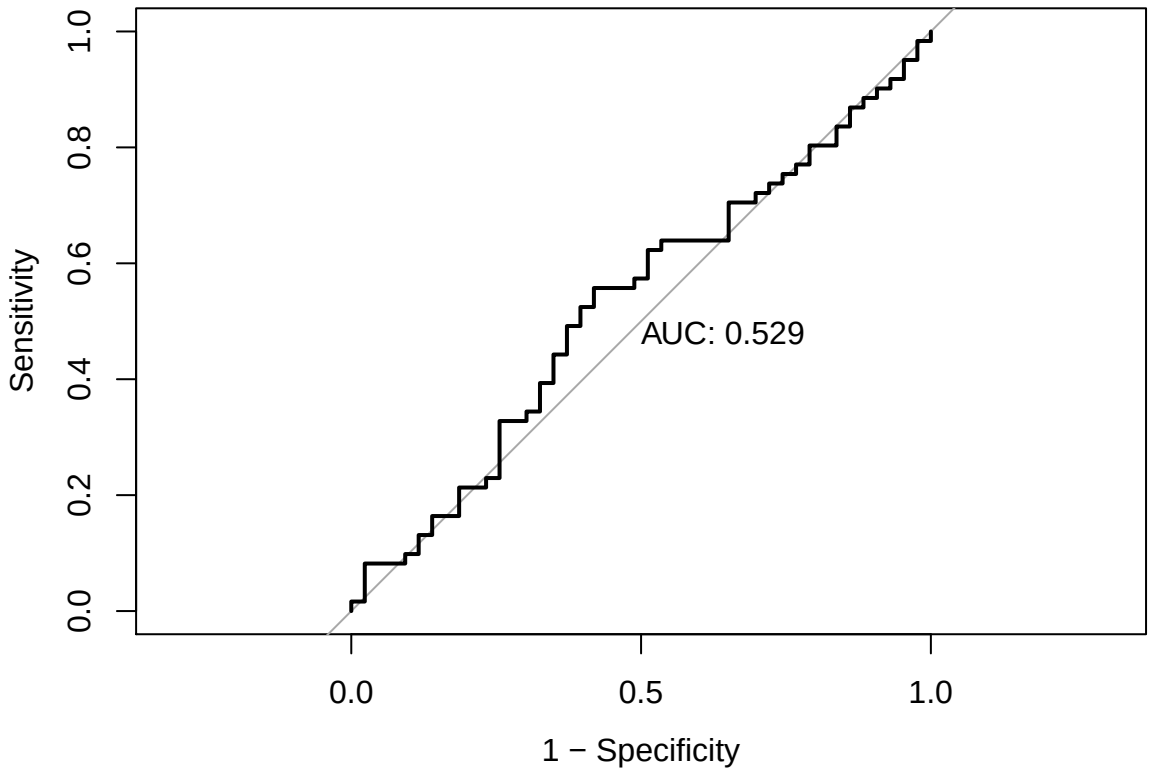
The AUC for LDA is 0.557

Using QDA

```
# usING qda() in MASS
qda.fit <- qda(Direction~ Lag1 + Lag2, data = Weekly_dat,
               subset = train_dat)

qda.pred <- predict(qda.fit, newdata = Weekly_dat2)

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

```
## Setting direction: controls > cases
```

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

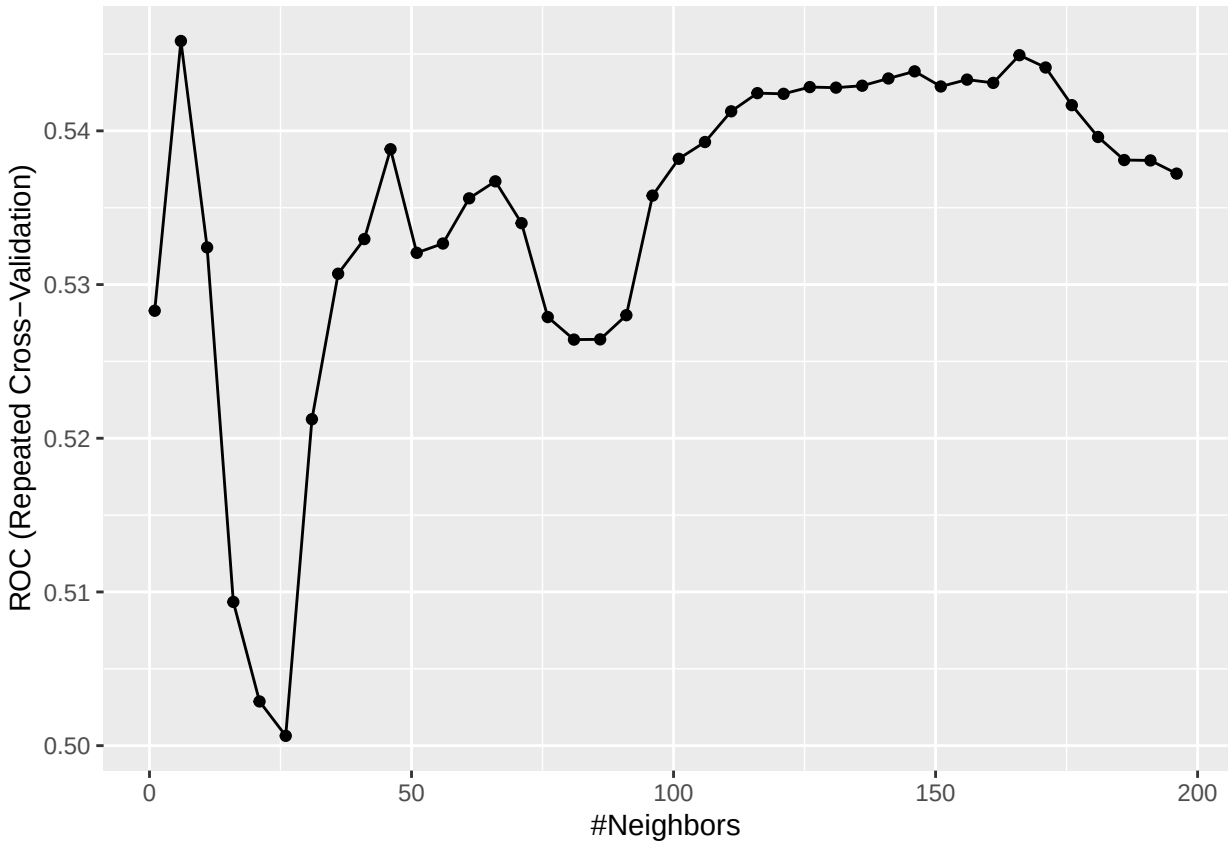


The AUC for QDA is 0.529.

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

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

```
set.seed(1)
model.knn <- train(x = Weekly_dat[train_dat,1:2],
                   y = Weekly_dat$Direction[train_dat],
                   method = "knn",
                   preProcess = c("center","scale"),
                   tuneGrid = data.frame(k = seq(1,200, by = 5)),
                   trControl = ctrl)

ggplot(model.knn)
```

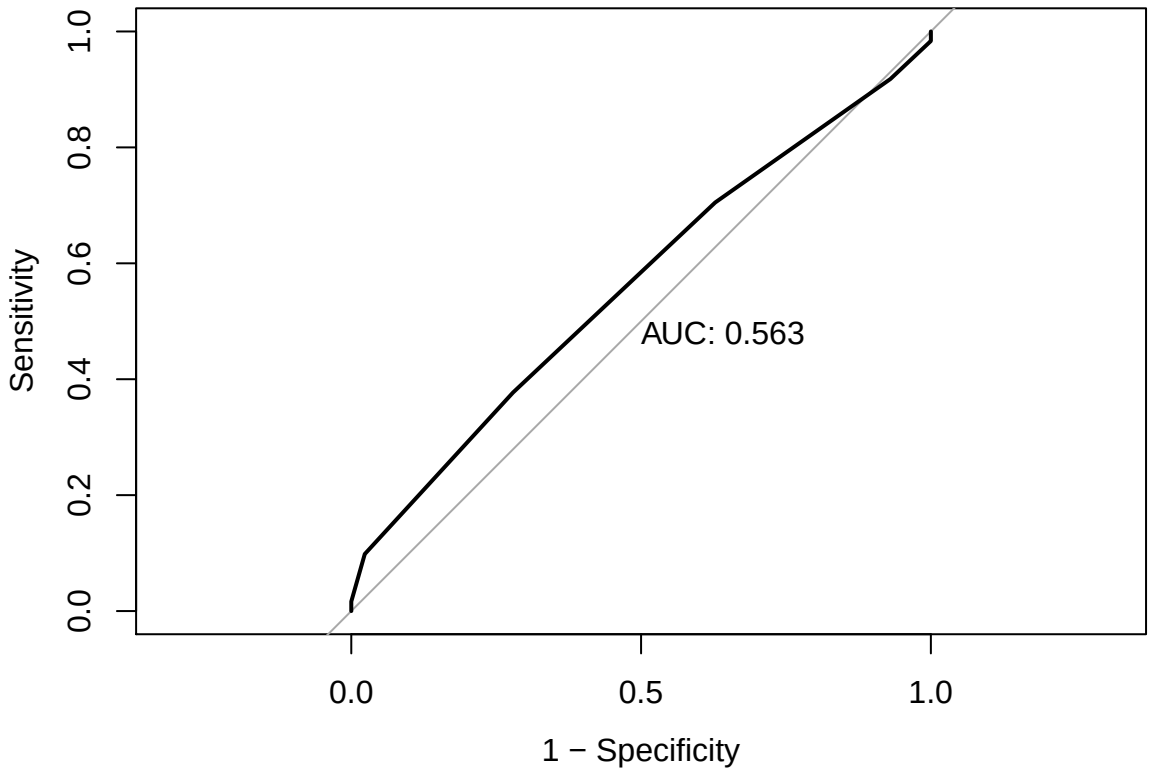



```
knn.pred <- predict(model.knn,newdata = Weekly_dat2, type = "prob")[,2]
roc.knn <- roc(Direction_new, knn.pred)
```

```
## Setting levels: control = Down, case = Up
```

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

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

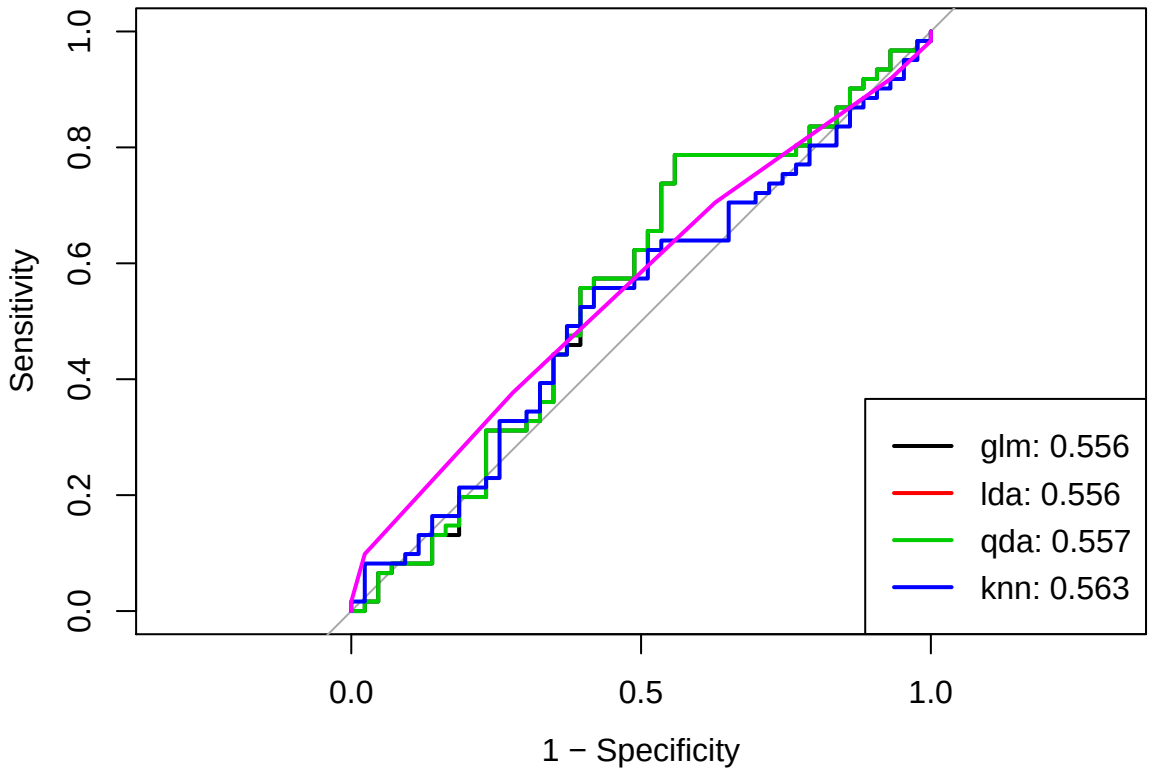



```
# The
```

AUC is 0.563

Comparing the results:

```
auc <- c(roc.glm$auc[1], roc.glm$auc[1], roc.lda$auc[1], roc.knn$auc[1])

plot(roc.glm, legacy.axes = TRUE)
plot(roc.lda, col = 3, add = TRUE)
plot(roc.qda, col = 4, add = TRUE)
plot(roc.knn, col = 6, add = TRUE)
modelNames <- c("glm","lda","qda","knn")
legend("bottomright", legend = paste0(modelNames, ": ", round(auc,3)),
col = 1:6, lwd = 2)
```



The KNN appears to be the best model since the AUC is highest for KNN.