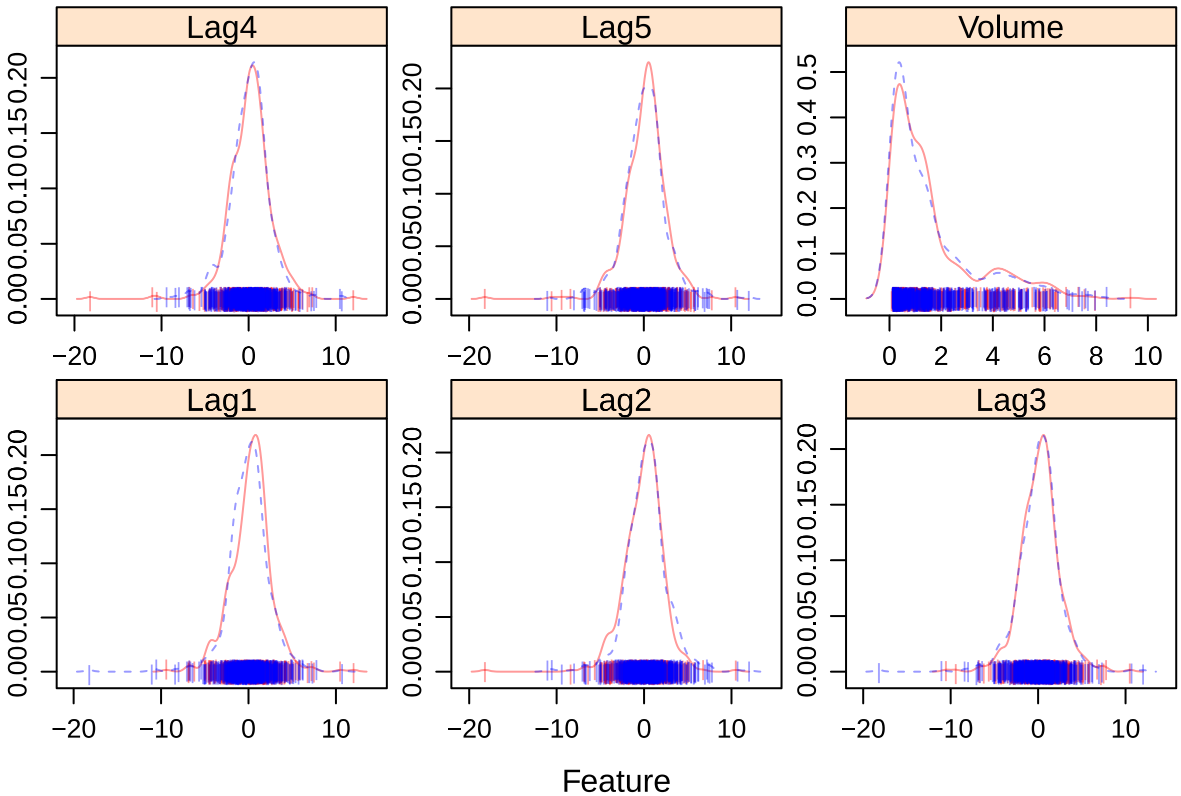
DS2_HW3

Siyan Chen

4/6/2019

(a) Produce some graphical summaries of theWeeklydata.

```
transparentTheme(trans = .4)
featurePlot(x = df[, 2:7],
            y = df$Direction,
            scales = list(x=list(relation = "free"),
                          y=list(relation = "free")),
            plot = "density", pch = "|")
```



(b) Use the full data set to perform a logistic regression withDirectionas the responseand the five Lagv ariables plus Volumeas predictors. Do any of the predictors appearto be statistically significant? If so, which ones?

```
set.seed(1)
rowTrain = createDataPartition(y = df$Direction,
                               p = 0.75,
                               list = FALSE)
glm_fit = glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,
              data = df,
```

```r
               subset = rowTrain,
               family = binomial)
contrasts(df$Direction)
```

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

```r
summary(glm_fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = df, subset = rowTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8407  -1.2503   0.9628   1.0737   1.6492
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.31400    0.10037   3.129  0.00176 **
## Lag1        -0.06315    0.03027  -2.086  0.03694 *
## Lag2         0.07588    0.03136   2.420  0.01553 *
## Lag3         0.00262    0.03144   0.083  0.93358
## Lag4        -0.02396    0.03023  -0.793  0.42807
## Lag5        -0.02942    0.03184  -0.924  0.35547
## Volume      -0.05148    0.04150  -1.241  0.21478
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1122.4  on 816  degrees of freedom
## Residual deviance: 1108.2  on 810  degrees of freedom
## AIC: 1122.2
##
## Number of Fisher Scoring iterations: 4
```

Yes, predictors Lag1, Lag2 appear to be statistically significant.

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

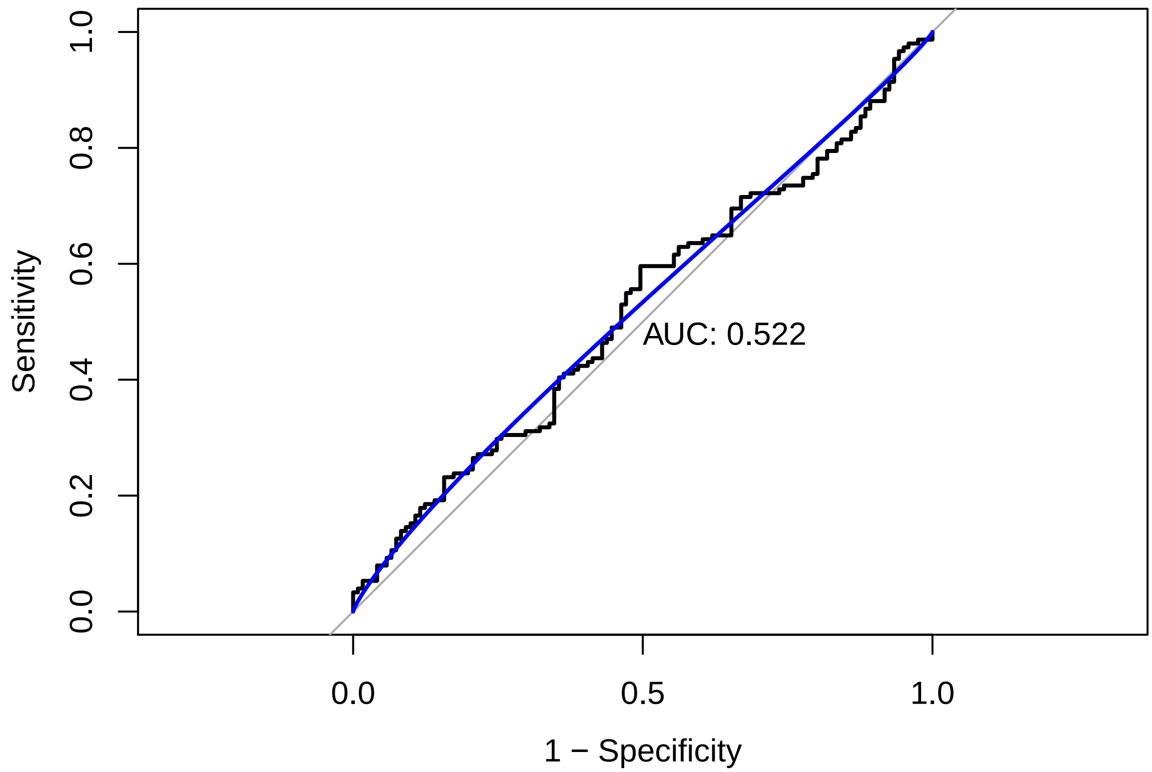
```r
# Bayes classigier(cutoff 0.5)
test.pred.prob = predict(glm_fit, newdata = df[-rowTrain,],
                         type = "response")
test.pred = rep("Down", length(test.pred.prob))
test.pred[test.pred.prob>0.5] = "Up"

# confustionMatrix
confusionMatrix(data = as.factor(test.pred),
                reference = as.factor(df$Direction[-rowTrain]),
                positive = "Up")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down  Up
##       Down   16  28
##       Up    105 123
##
##                Accuracy : 0.511
##                  95% CI : (0.4499, 0.5719)
##     No Information Rate : 0.5551
##     P-Value [Acc > NIR] : 0.9361
##
##                   Kappa : -0.0568
##  Mcnemar's Test P-Value : 4.397e-11
##
##             Sensitivity : 0.8146
##             Specificity : 0.1322
##          Pos Pred Value : 0.5395
##          Neg Pred Value : 0.3636
##              Prevalence : 0.5551
##          Detection Rate : 0.4522
##    Detection Prevalence : 0.8382
##       Balanced Accuracy : 0.4734
##
##        'Positive' Class : Up
##
```

From the confusion matrix, sensiticity is 0.8146 (the probability of true "Up" is 0.8146 when the direction is predicted to be Up) and the sensitivity is 0.1322 (the probability of not "Up" is 0.1322 when the direction is predicted to be down) Accuracy tells the overall probaility of correct classifier is 0.511. Kappa is negative, which means that there is less agrrement than would be expected by chance give the marginal distribution.

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

```r
roc_glm = roc(df$Direction[-rowTrain], test.pred.prob)
plot(roc_glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc_glm), col = 4, add = TRUE)
```



Based on the plot, the AUC us 0.522, which suggests that the capability of logistic models distinguishing between classes is not good.

(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_subset = df %>%
  filter(1990<=Year& Year<=2008)
test_subset = anti_join(df, train_subset)
```

```
## Joining, by = c("Year", "Lag1", "Lag2", "Lag3", "Lag4", "Lag5", "Volume", "Today", "Direction")
```

```
rowtrain = train_subset$Direction
rowtest = test_subset$Direction

glm_fit1 = glm(Direction~ Lag1 + Lag2,
               data = train_subset,
               family = binomial)
summary(glm_fit1)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = train_subset)
##
## 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
```

```
contrasts(train_subset$Direction)
```

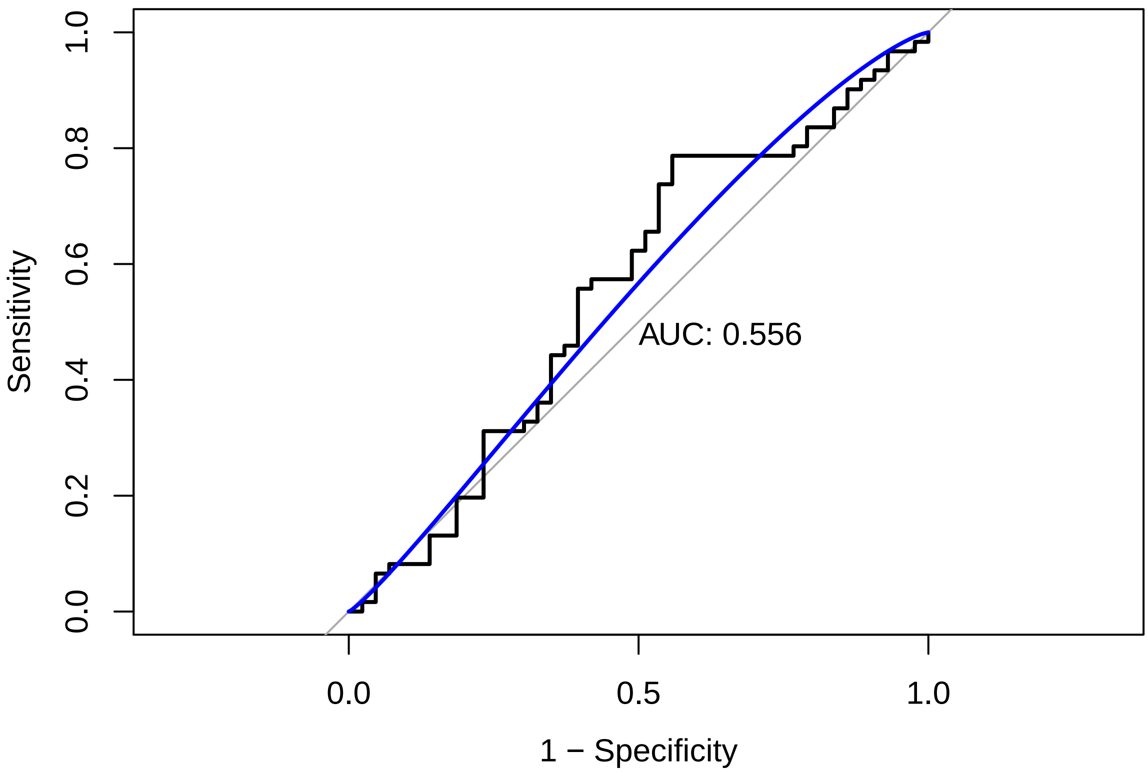
```
##      Up
## Down  0
## Up    1
```

```
pred.test.value = predict(glm_fit1,
                          newdata = test_subset,
                          type = "response")
# Bayes Method Cutoff
pred.test = rep("Down", length(pred.test.value))
pred.test[pred.test.value>0.5] = "Up"

# Confusion Matrix
confusionMatrix(data = as.factor(pred.test),
                reference = as.factor(rowtest),
                positive = "Up")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    7  8
##       Up     36 53
##
##                Accuracy : 0.5769
##                  95% CI : (0.4761, 0.6732)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.6193
##
##                   Kappa : 0.035
##  Mcnemar's Test P-Value : 4.693e-05
##
##             Sensitivity : 0.8689
##             Specificity : 0.1628
##          Pos Pred Value : 0.5955
##          Neg Pred Value : 0.4667
##              Prevalence : 0.5865
##          Detection Rate : 0.5096
```

```
##    Detection Prevalence : 0.8558
##       Balanced Accuracy : 0.5158
##
##        'Positive' Class : Up
##
```

```
# ROC
roc1 = roc(test_subset$Direction, pred.test.value)
plot(roc1, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc1), col = 4, add =TRUE)
```



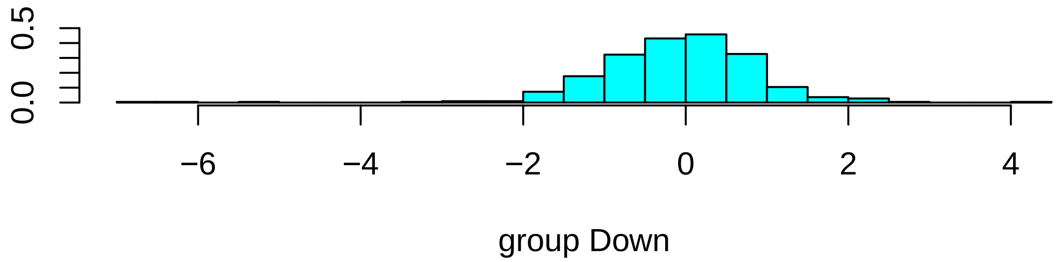
AUC is 0.556. The model does better job to classify compared to model in c.

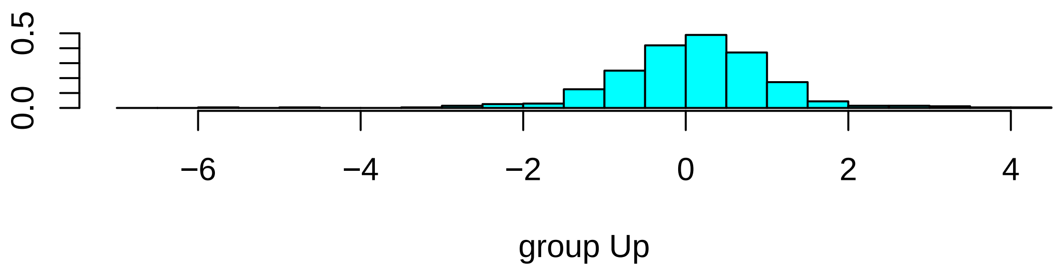
(f) Repeat (e) using LDA and QDA.

```
#Discriminant analysis
## LDA
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
lda.fit = lda(Direction ~ Lag1 + Lag2,
              data = train_subset)
plot(lda.fit)
```

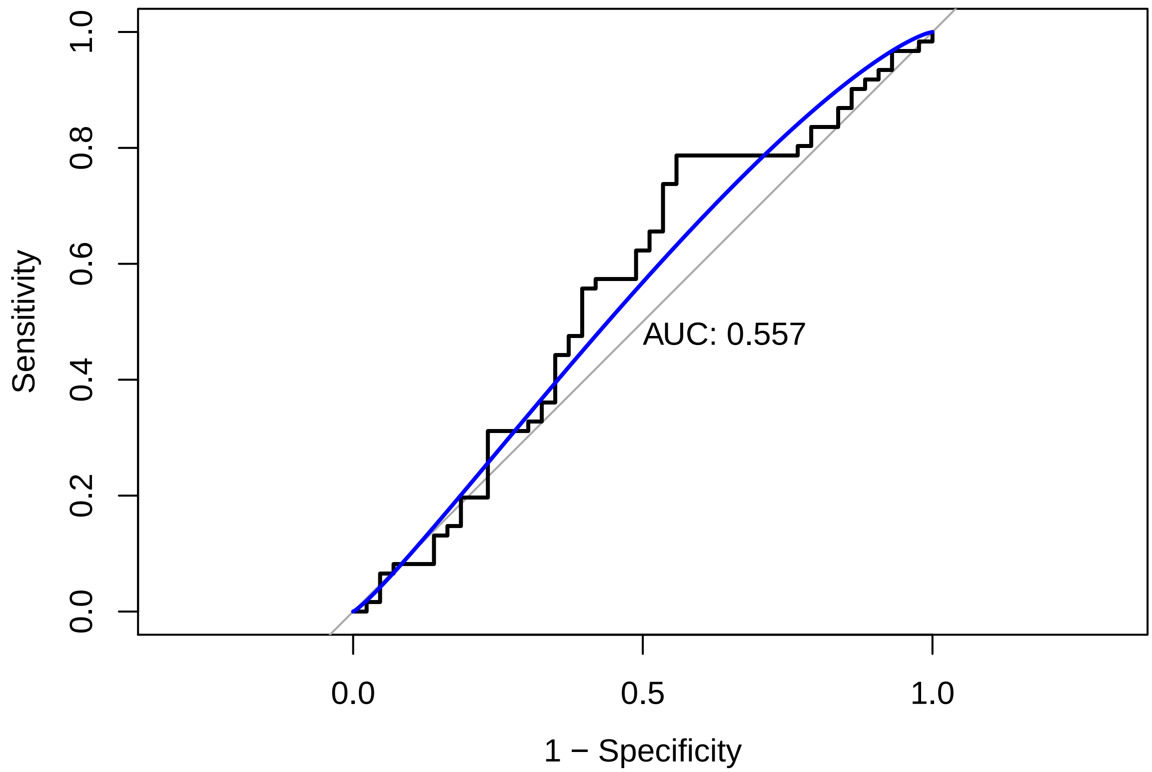





```
# evaluate the test set performance using roc
lda.pred = predict(lda.fit, newdata = test_subset)
head(lda.pred$posterior)
```

```
##        Down        Up
## 1 0.5602039 0.4397961
## 2 0.3079163 0.6920837
## 3 0.4458032 0.5541968
## 4 0.4785107 0.5214893
## 5 0.4657943 0.5342057
## 6 0.5262907 0.4737093
```

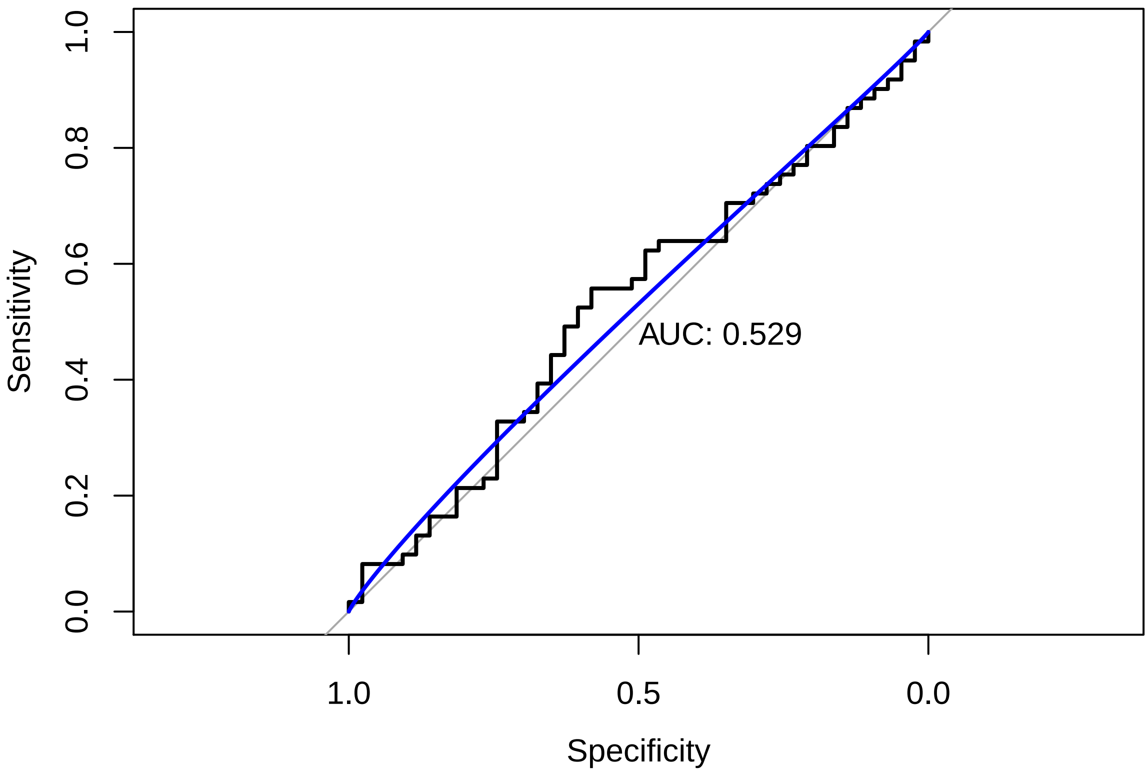
```
roc.lda = roc(test_subset$Direction,lda.pred$posterior[,2],
              levels = c("Down","Up"))
plot(roc.lda, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.lda), col = 4, add =TRUE)
```




```
## QDA
qda.fit = qda(Direction ~ Lag1 + Lag2,
              data = train_subset)
qda.pred = predict(qda.fit, newdata = test_subset)
head(qda.pred$posterior)
```

```
##        Down        Up
## 1 0.5436205 0.4563795
## 2 0.3528814 0.6471186
## 3 0.2227273 0.7772727
## 4 0.3483016 0.6516984
## 5 0.4598550 0.5401450
## 6 0.5119613 0.4880387
```

```
roc.qda = roc(test_subset$Direction, qda.pred$posterior[,2],
              levels = c("Down","Up"))
plot(roc.qda, legacy.axix = TRUE, print.auc = TRUE)
plot(smooth(roc.qda), col = 4, add = TRUE)
```



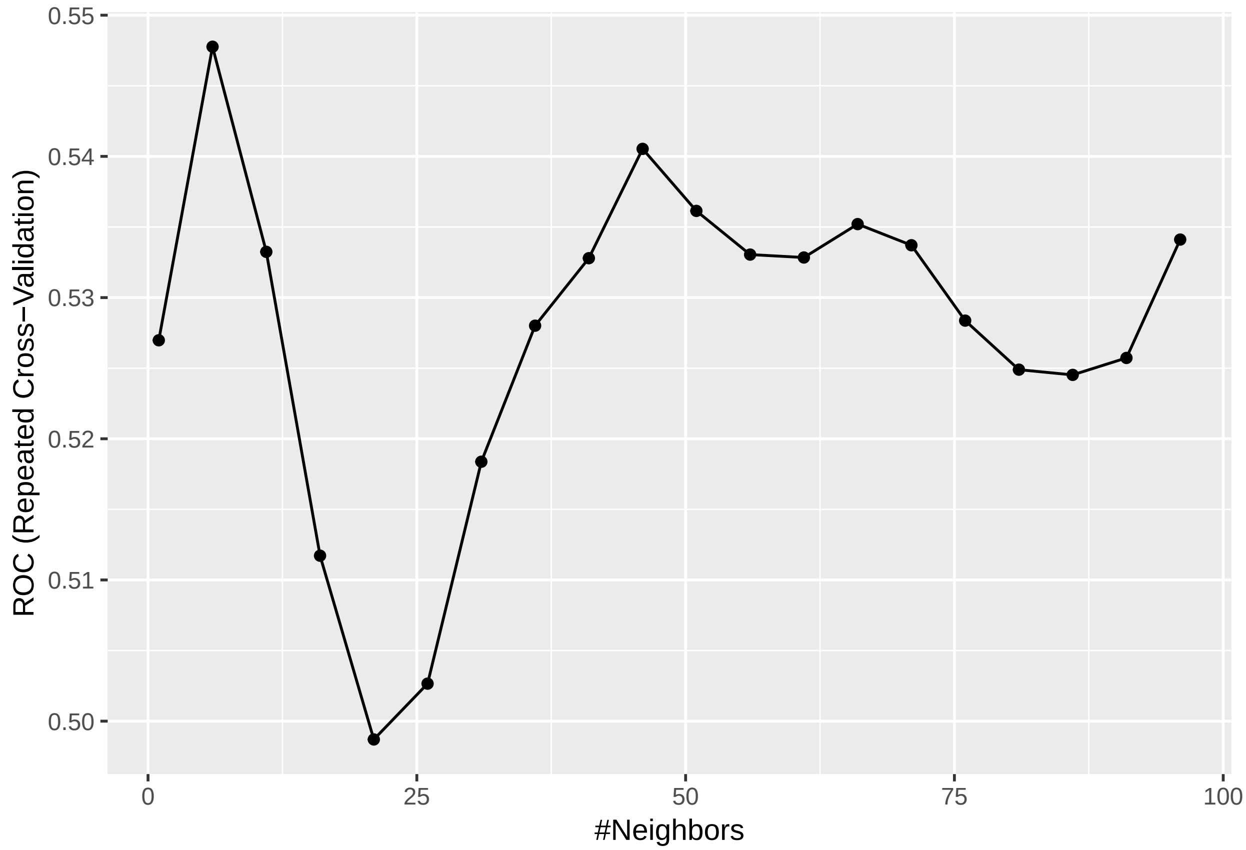
AUC for LDA is 0.557 and AUC for QDA is 0.529

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

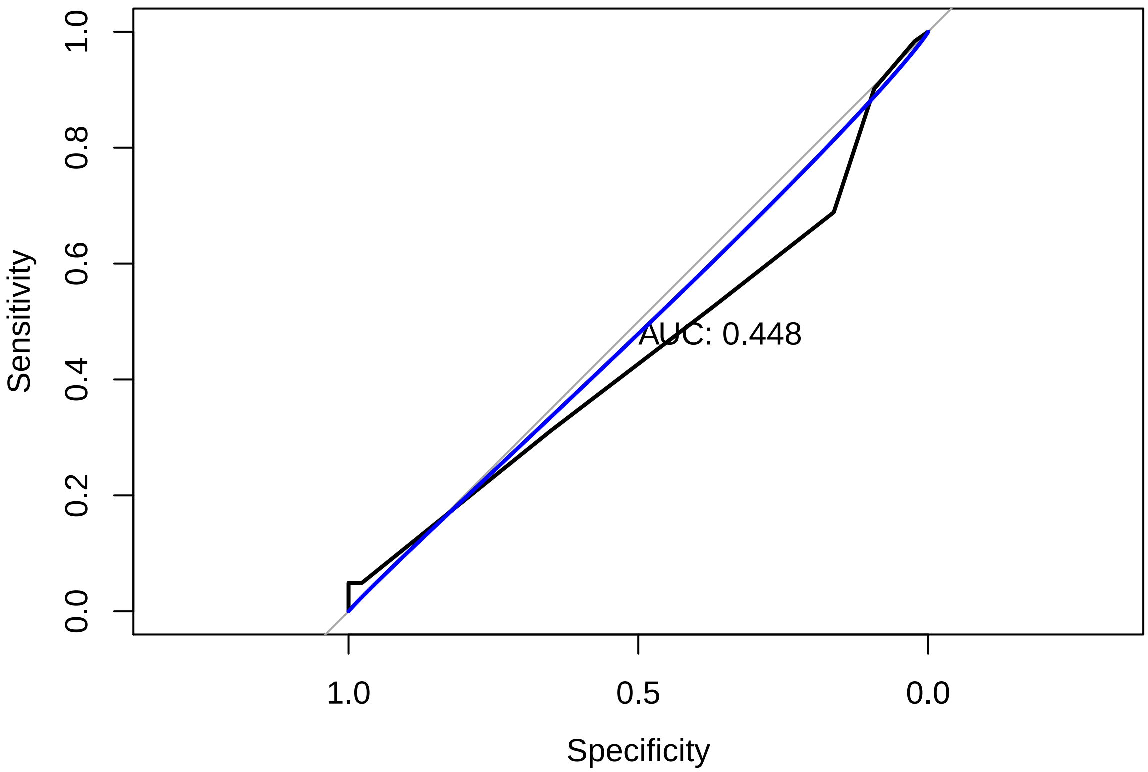
```
# neighbor for classfication
# Using caret
ctrl <-trainControl(method = "repeatedcv",repeats = 5,summaryFunction = twoClassSummary,classProbs = TRU

model.knn = train(x = train_subset[, 2:3],
                  y = train_subset$Direction,
                  method = "knn",
                  preProcess = c("center", "scale"),
                  tuneGrid = data.frame(k = seq(1,100, by = 5)),
                  trControl = ctrl)
```

```
## Warning in train.default(x = train_subset[, 2:3], y =
## train_subset$Direction, : The metric "Accuracy" was not in the result set.
## ROC will be used instead.
```

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




```
# predict.train DO CENTRAL AND SCALE AUTOMATICALLY
# # how to central and scale here ?
# predic_knn = predict.train(model.knn, testX = test_subset[, 2:3],
#              testY = test_subset$Direction)


predic_knn1 = predict(model.knn$finalModel, newdata = test_subset[, 2:3])

roc_knn = roc(test_subset$Direction, predic_knn1[,2],
              levels = c("Down","Up"))
plot(roc_knn, legacy.axis = TRUE, print.auc = TRUE)
plot(smooth(roc_knn), col = 4, add = TRUE)
```



AUC for KNN is 0.566.