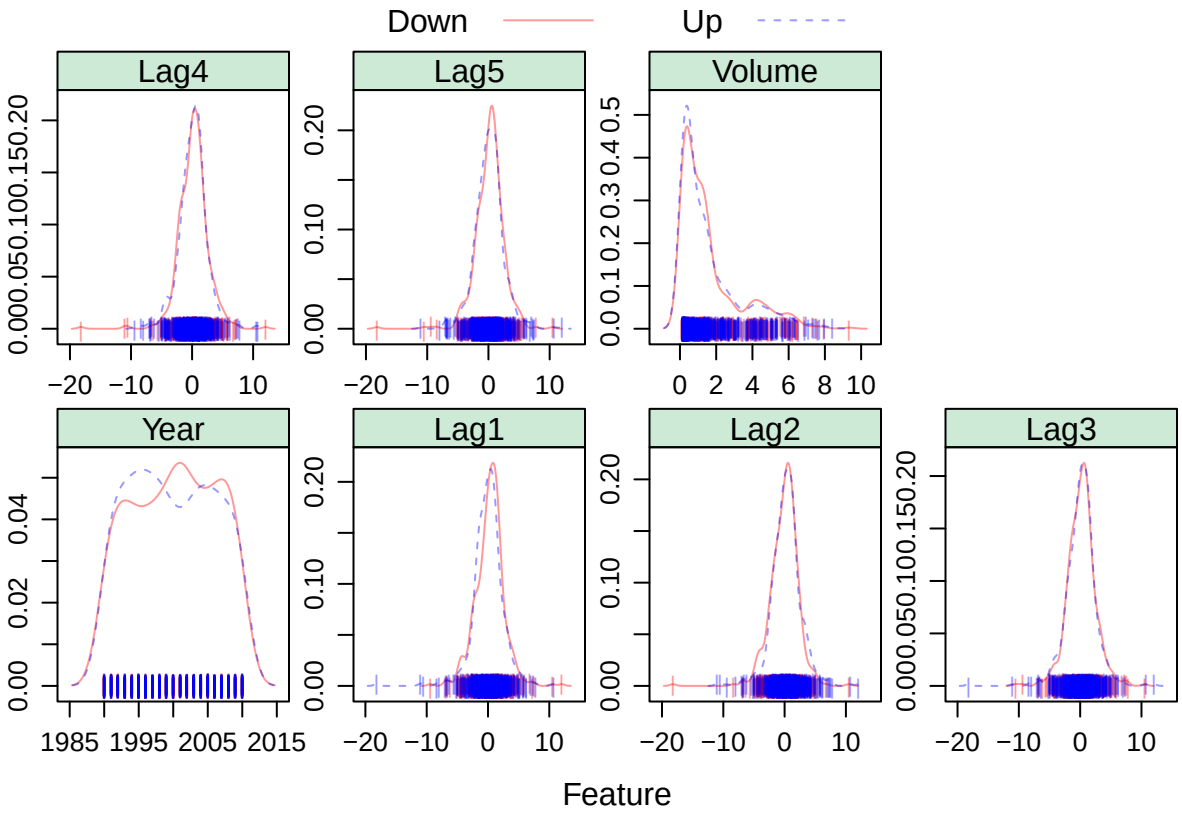
p8106_hw3_dd2948

David DeStephano

April 12, 2020

```
## Loading required package: lattice

## Loading required package: ggplot2

## Loading required package: Matrix

## Loaded glmnet 3.0-2

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

##
## Attaching package: 'pROC'

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

## -- Attaching packages ------------------------- tidyverse 1.2.1 --

## v tibble  2.1.3     v purrr   0.3.3
## v tidyr   1.0.0     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v tibble  2.1.3     v forcats 0.4.0

## -- Conflicts ---------------------------- tidyverse_conflicts() --
## x tidyr::expand() masks Matrix::expand()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## x tidyr::pack()   masks Matrix::pack()
## x dplyr::select() masks MASS::select()
## x tidyr::unpack() masks Matrix::unpack()
```

```
data(Weekly)
Weekly<-Weekly %>% select(-Today)
dat <- Weekly
theme1 <- transparentTheme(trans = .4)
theme1$strip.background$col <- rgb(.0, .6, .2, .2)
trellis.par.set(theme1)

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



Logistic regression

```
set.seed(1)

dat<-Weekly %>% select(-Year)

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

summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ ., 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
```

```
contrasts(dat$Direction)
```

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

Lag2 is the only significant variable

```
set.seed(1)
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
##
```

The confusion matrix tells us that the sensitivity is 92% and specificity is very low at 11%. Only 56 percent of predictions are correctly classified

Sensitivity measures the proportions of true positives that were predicted correctly

Specificity is the proportion of true negatives that were predicted correctly as negative

Kappa is only .035, which is very far from 1. A value of one would indicate good model performance.

ROC curve

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

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

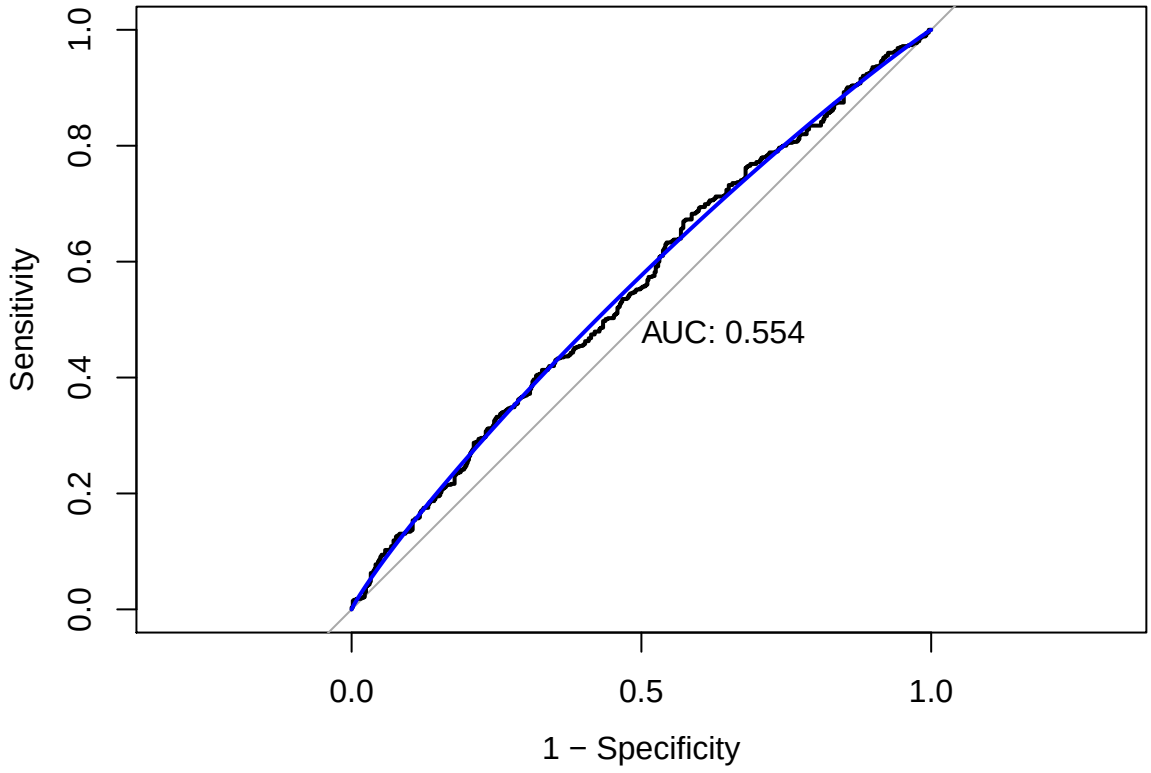
```
## Setting direction: controls < cases
```

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

```
roc_glm = roc(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 only 0.554, not much better than a flip of a coin

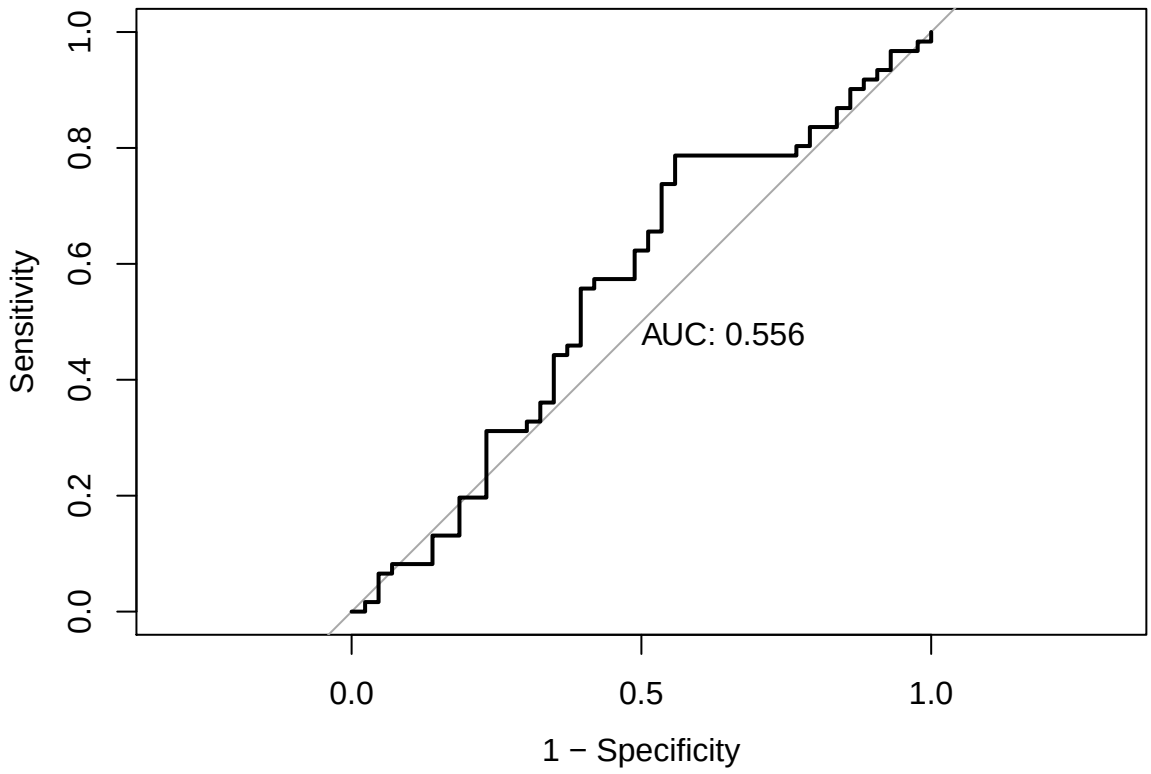
Redo analysis with 1990-2008 as training data

```
dat <- Weekly %>% select(-Lag3, -Lag4, -Lag5, -Volume)

train<- dat %>% filter(Year<=2008)
test<- dat %>% filter(Year>=2009)

#Regression model on 1990-2008
glm.fit <- glm(Direction~Lag1 + Lag2,
               data = train,
               family = binomial)


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

roc.glm2 <-roc(test$Direction,
               test.pred.prob)
```

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

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

```
plot(roc.glm2, legacy.axes = TRUE, print.auc =TRUE)
```



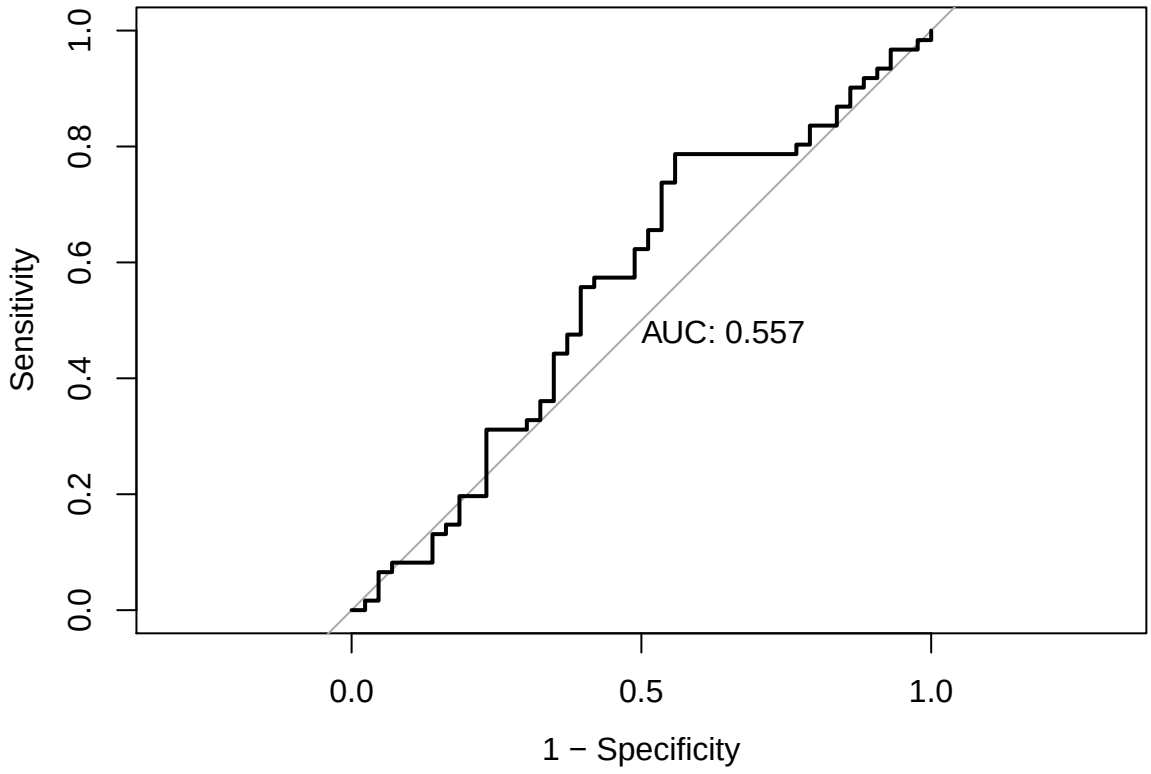
The AUC is still low at 0.556

LDA

```
lda.fit <- lda(Direction~Lag1 + Lag2,
               data = train)

test.pred.prob <- predict(lda.fit, newdata = test,
                          type = "response")


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

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

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

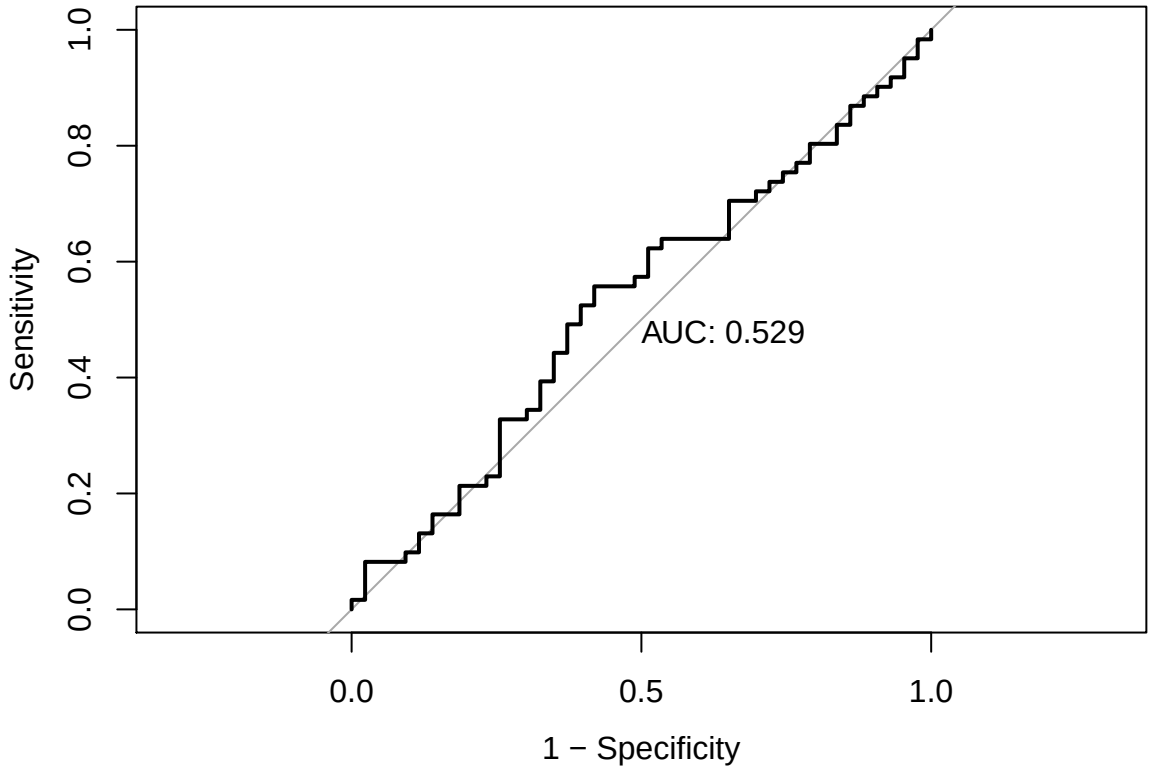


QDA

```r
set.seed(1)
qda.fit <- qda(Direction~Lag1 + Lag2,
               data = train)

test.pred.prob <- predict(qda.fit, newdata = test,
                          type = "response")


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

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

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

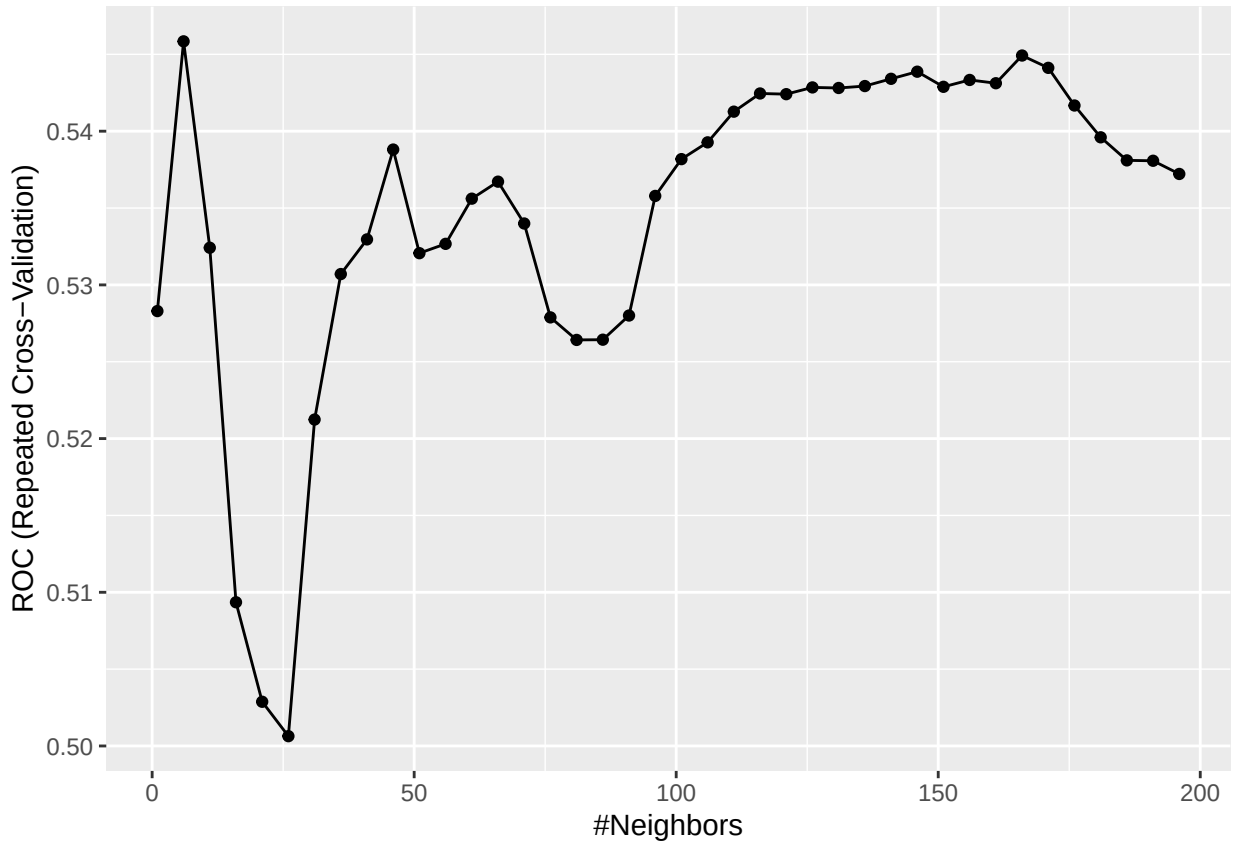


Knn

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


set.seed(1)
model.knn <- train(x = train[2:3],
                   y = train$Direction,
                   method = "knn",
                   preProcess = c("center","scale"),
                   tuneGrid = data.frame(k =seq(1,200,by=5)),
                   trControl = ctrl)
```

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

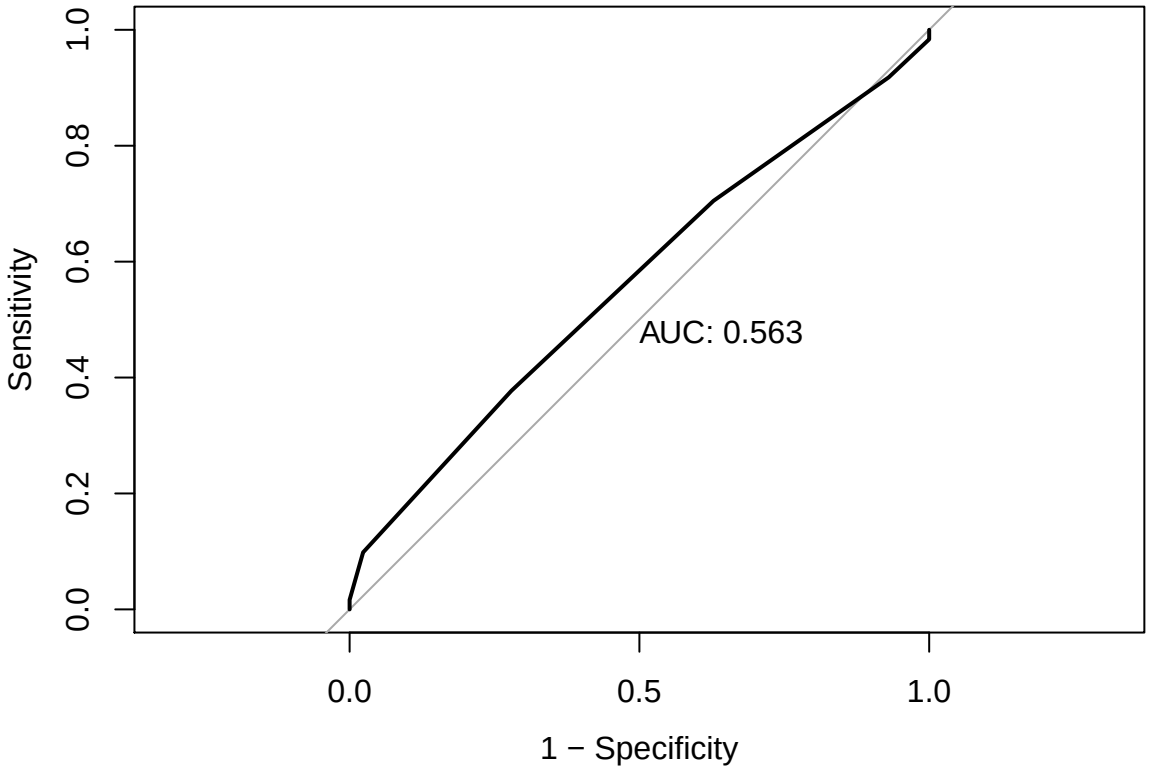
```
ggplot(model.knn)
```




```
test.pred.prob <- predict(model.knn, newdata = test,
                          type = "prob")

roc.knn <-roc(test$Direction,
              test.pred.prob$Up,
              levels = c("Down", "Up"))
```

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

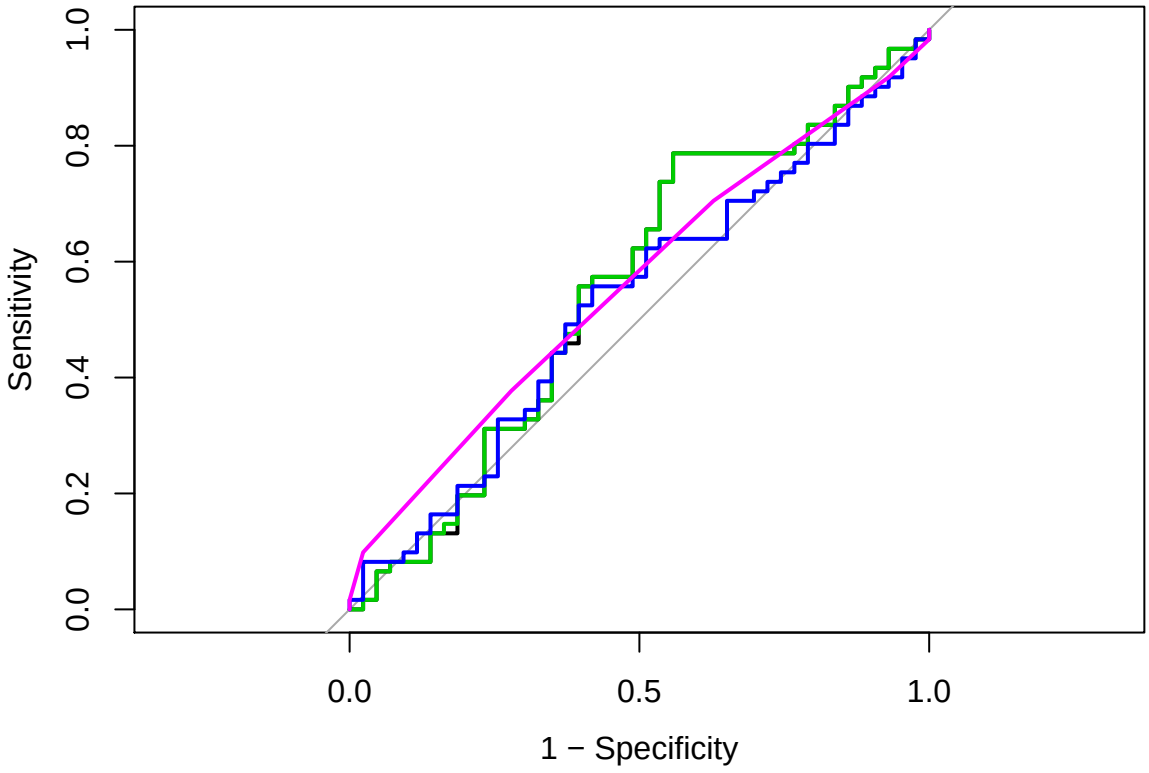
```
plot(roc.knn, legacy.axes = TRUE, print.auc =TRUE)
```



The AUC for knn when predicting 2009 onwards is 0.56 for "Up". LDA, GLM, and KNN modles perform equally at 0.56, but no model predicts the data well, as they are all close to 0.5.

Plot the comparisons

```
plot(roc.glm2, 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")
```