p8106_hw3_jsg2145

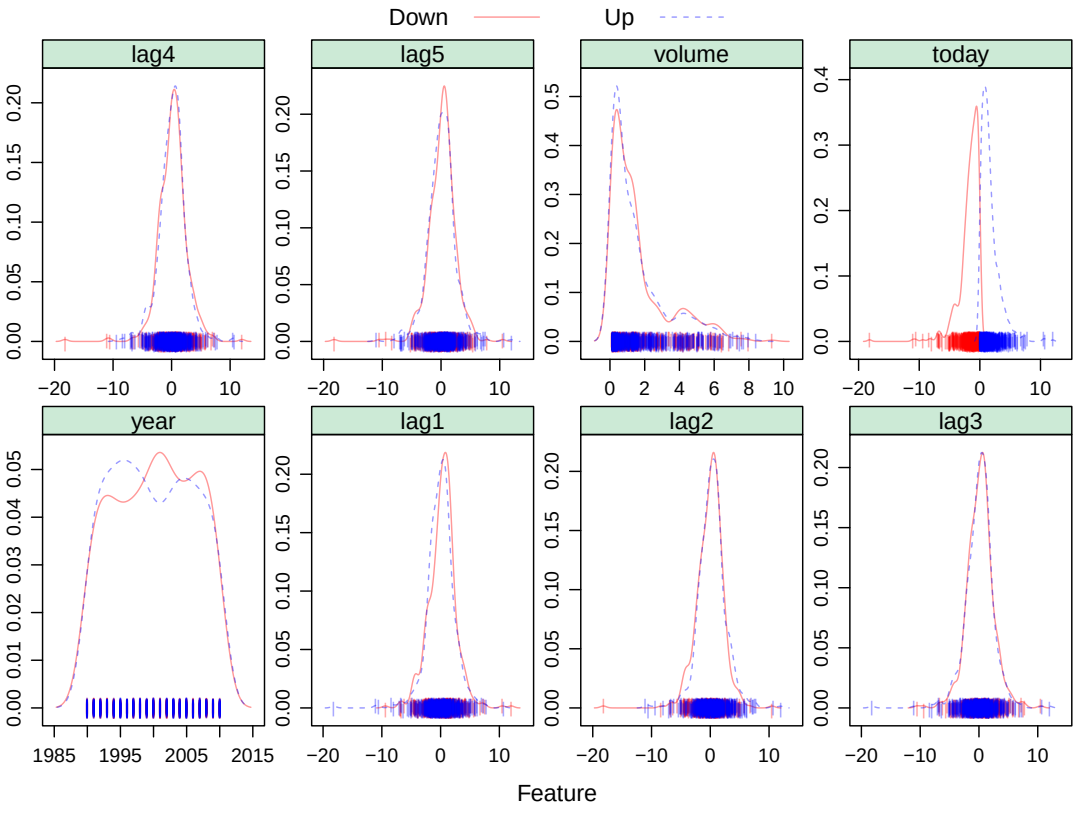
Jared Garfinkel

4/14/2020

```
data("Weekly")

df = Weekly %>%
  janitor::clean_names()
```

Part a

```
theme1 <- transparentTheme(trans = .4)
theme1$strip.background$col <- rgb(.0, .6, .2, .2)
trellis.par.set(theme1)

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



Part b

```
df_glm <- glm(direction ~ lag1 + lag2 + lag3 + lag4 + lag5 + volume,
              data=df,
              family="binomial")
summary(df_glm)
```

```
##
## Call:
## glm(formula = direction ~ lag1 + lag2 + lag3 + lag4 + lag5 +
##     volume, family = "binomial", data = df)
##
## 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
```

Only the lag2 predictor appears significant at $p = 0.0296 < 0.05$.

Part c

```
df_pred = predict(df_glm, type = "response")
df_test_pred = rep("Down", length(df_pred))
df_test_pred[df_pred > 0.5] = "Up"
df_confusion = confusionMatrix(data = as.factor(df_test_pred),
                               reference = df$direction,
                               positive = "Up")
```

The sensitivity is 0.9207, indicating a high degree of true positives, while the specificity is 0.1116, indicating that when the market goes down, there are less than 12% of true negatives. Kappa is 0.035, a measure of the agreement between the predictive value and true value.

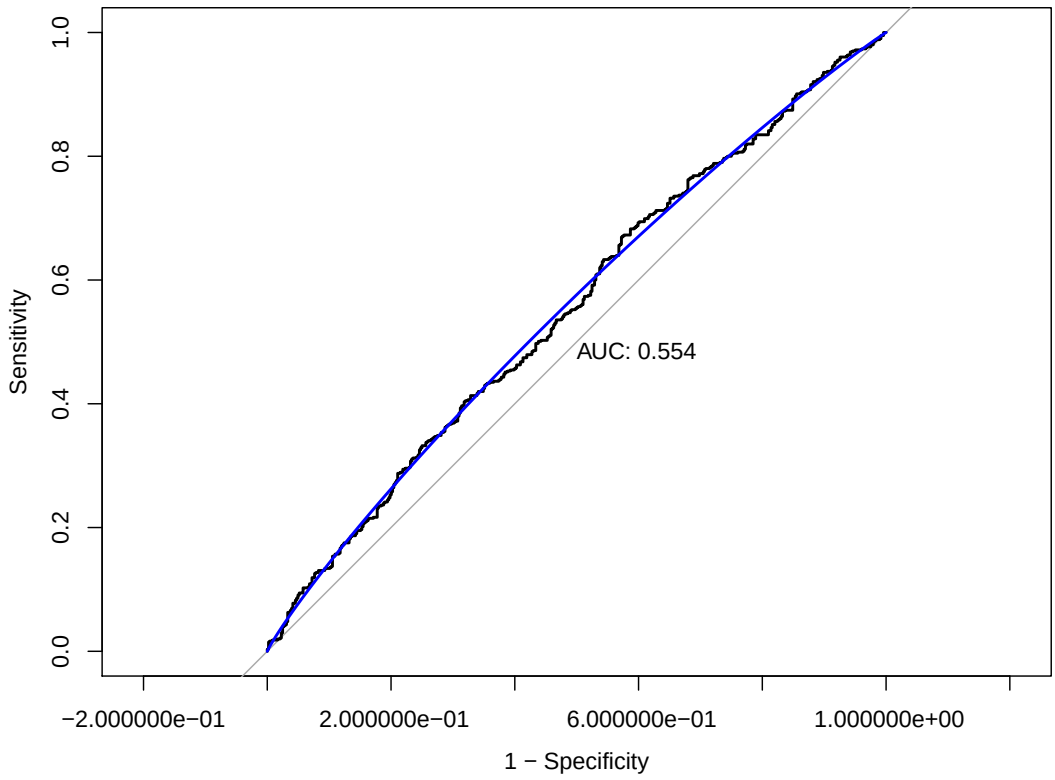
Part d

```
df_roc <- roc(df$direction, df_pred)
```

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

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

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



The AUC is 0.5537.

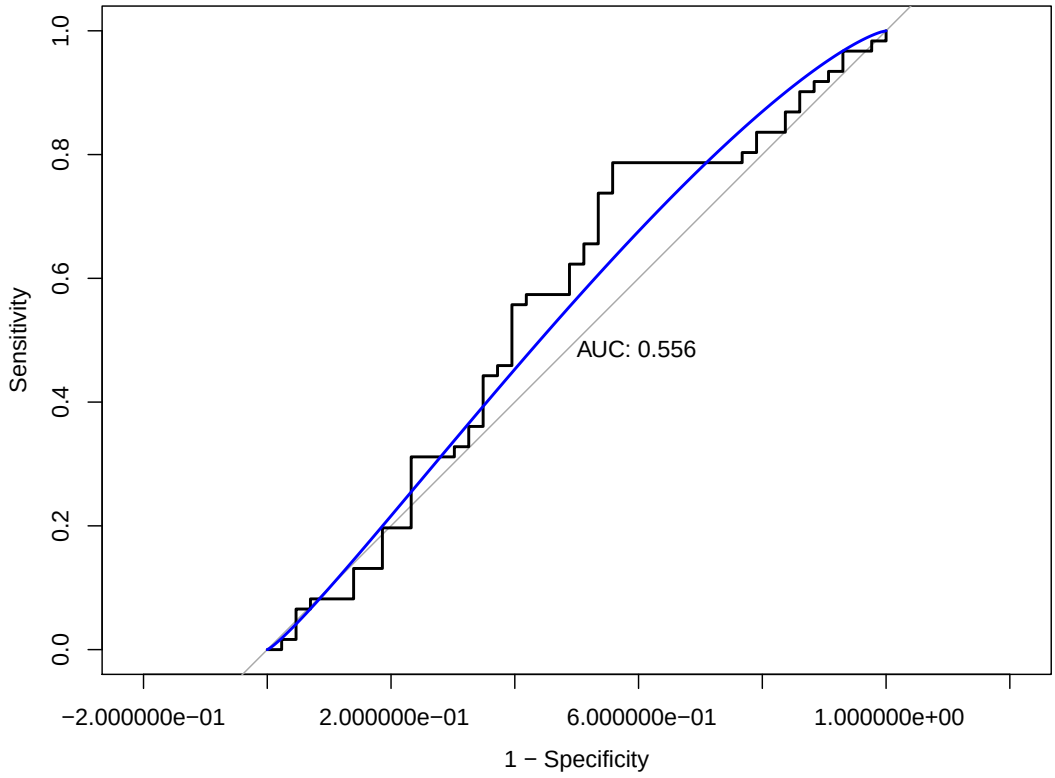
```
df_train = df %>%
  filter(year < 2009)

df_test = df %>%
  filter(year > 2008)
```

```
df_glm_train <- glm(direction ~ lag1 + lag2, data = df_train, family = "binomial")
df_glm_test <- predict(df_glm_train, type = "response", newdata = df_test)
df_roc_split <- roc(df_test$direction, df_glm_test)
```

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

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

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



The AUC is 0.5559.

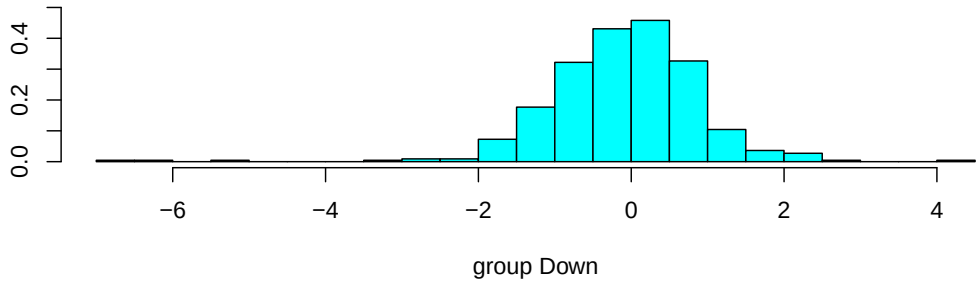
Part f

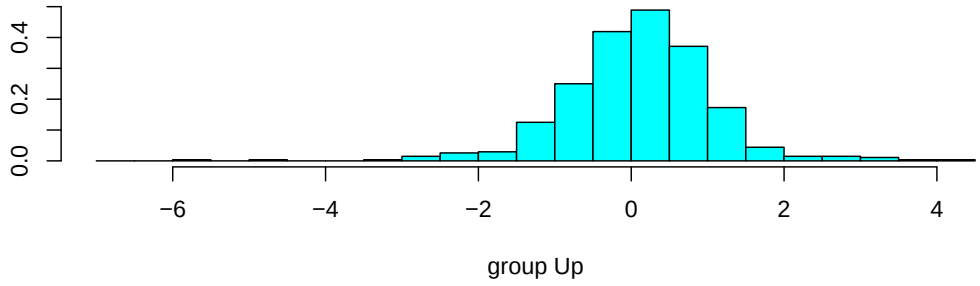
```
df_lda = lda(direction ~ lag1 + lag2, data = df_train)
df_lda
```

```
## Call:
## lda(direction ~ lag1 + lag2, data = df_train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##               lag1        lag2
## Down  0.289444444 -0.03568254
## Up   -0.009213235  0.26036581
##
```

```
## Coefficients of linear discriminants:
##             LD1
## lag1 -0.3013148
## lag2  0.2982579
```

```
plot(df_lda)
```



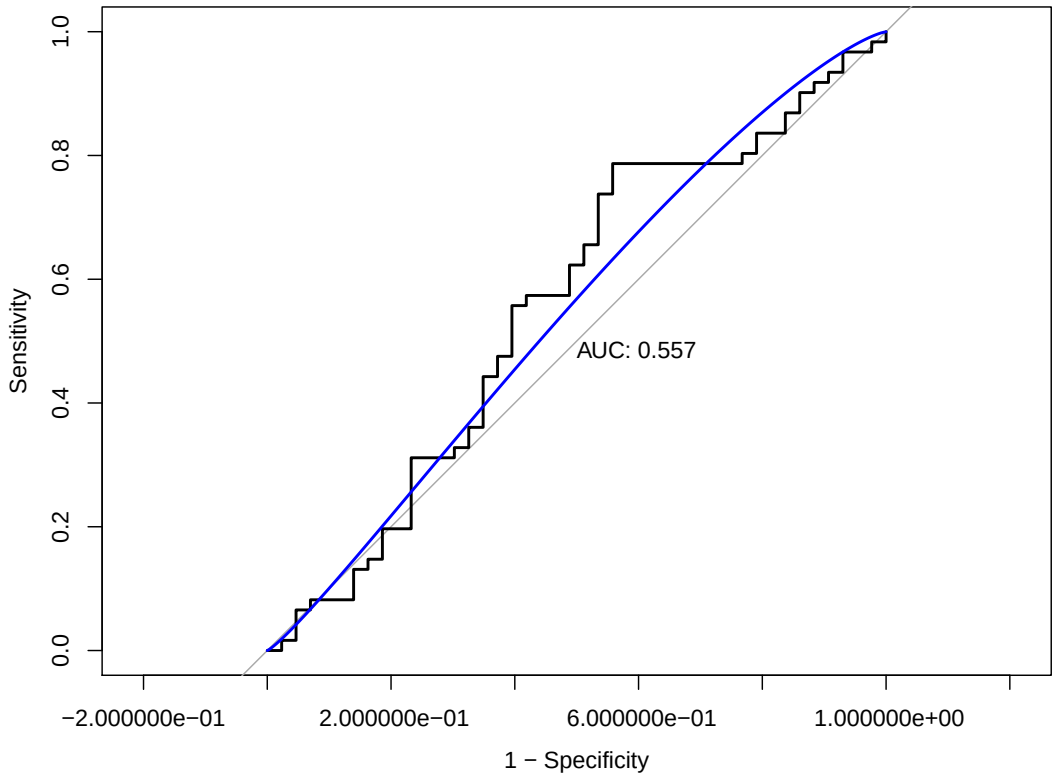



```
df_lda_pred <- predict(df_lda, newdata = df_test)

df_roc_lda <- roc(df_test$direction, df_lda_pred$posterior[,2])
```

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

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

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

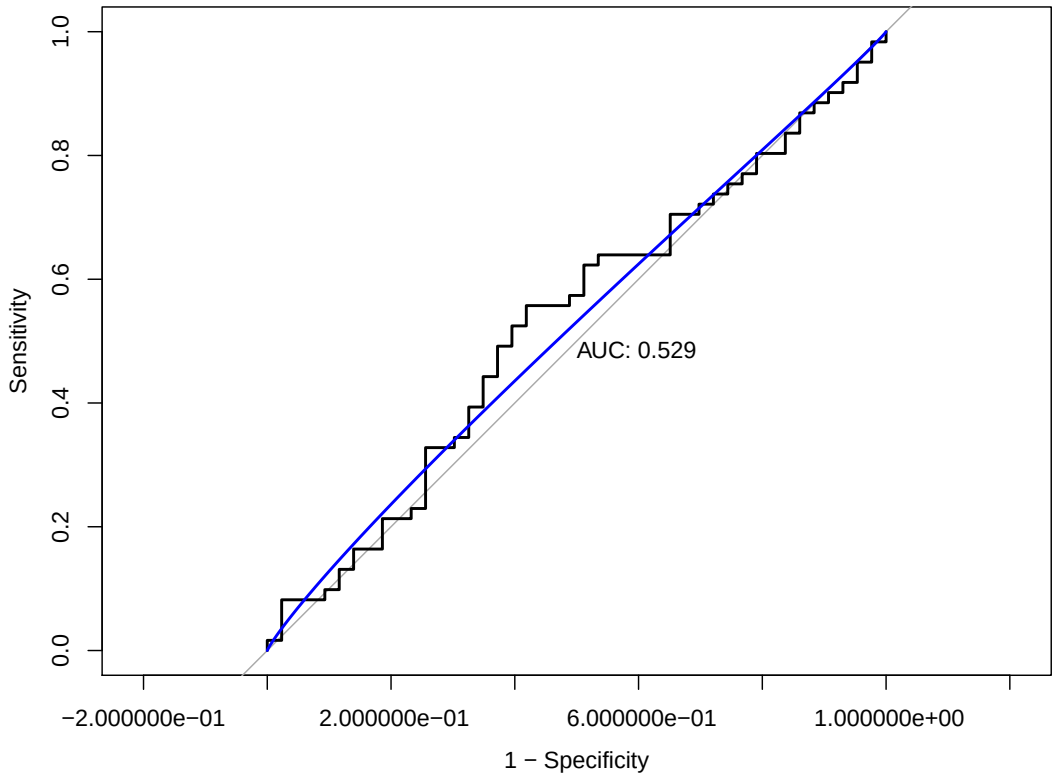


The AUC is 0.5566.

```
df_qda <- qda(direction ~ lag1 + lag2, data = df_train)
df_qda_pred <- predict(df_qda, newdata = df_test)

df_roc_qda <- roc(df_test$direction, df_qda_pred$posterior[,2],
                  levels = c("Down", "Up"))
```

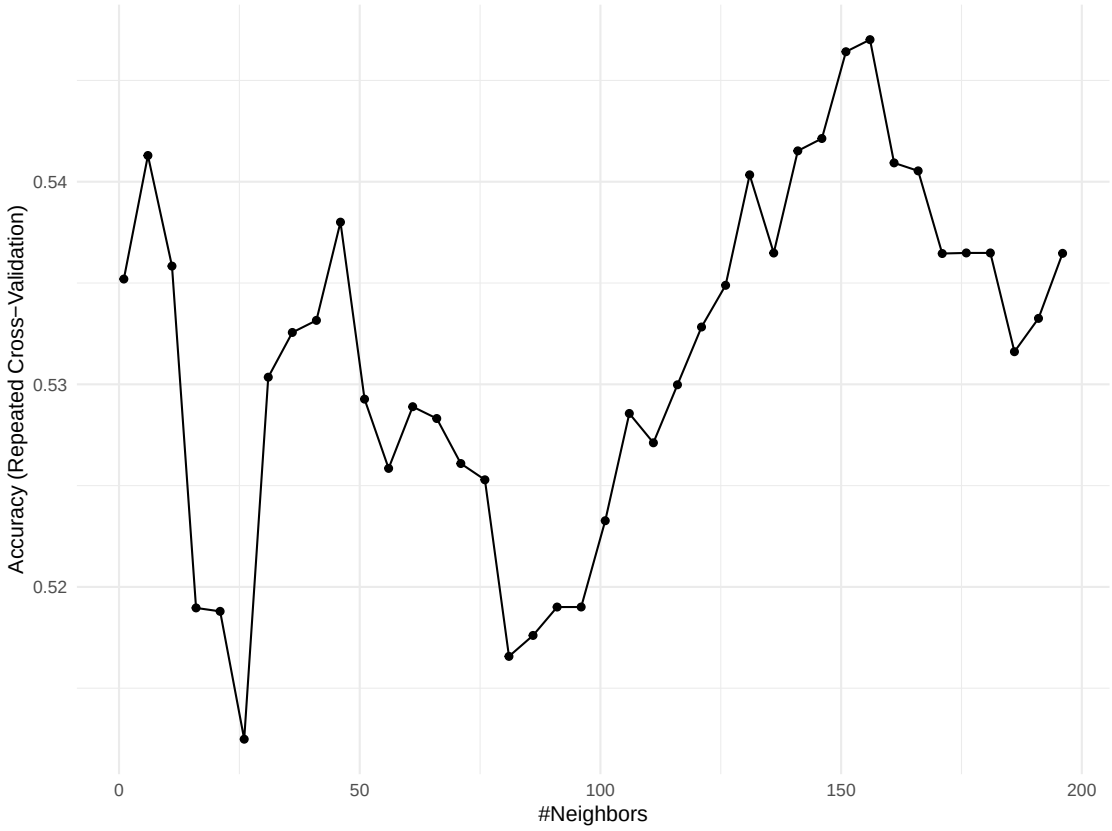
```
## Setting direction: controls > cases
```

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



The AUC is 0.5288.

```
ctrl1 = trainControl(method = "repeatedcv", number = 10, repeats = 5)

df_knn = train(x = df_train[2:3],
               y = df_train$direction,
               method = "knn",
               preProcess = c("center","scale"),
               tuneGrid = data.frame(k = seq(1, 200, by = 5)),
               trControl = ctrl1)

ggplot(df_knn)
```

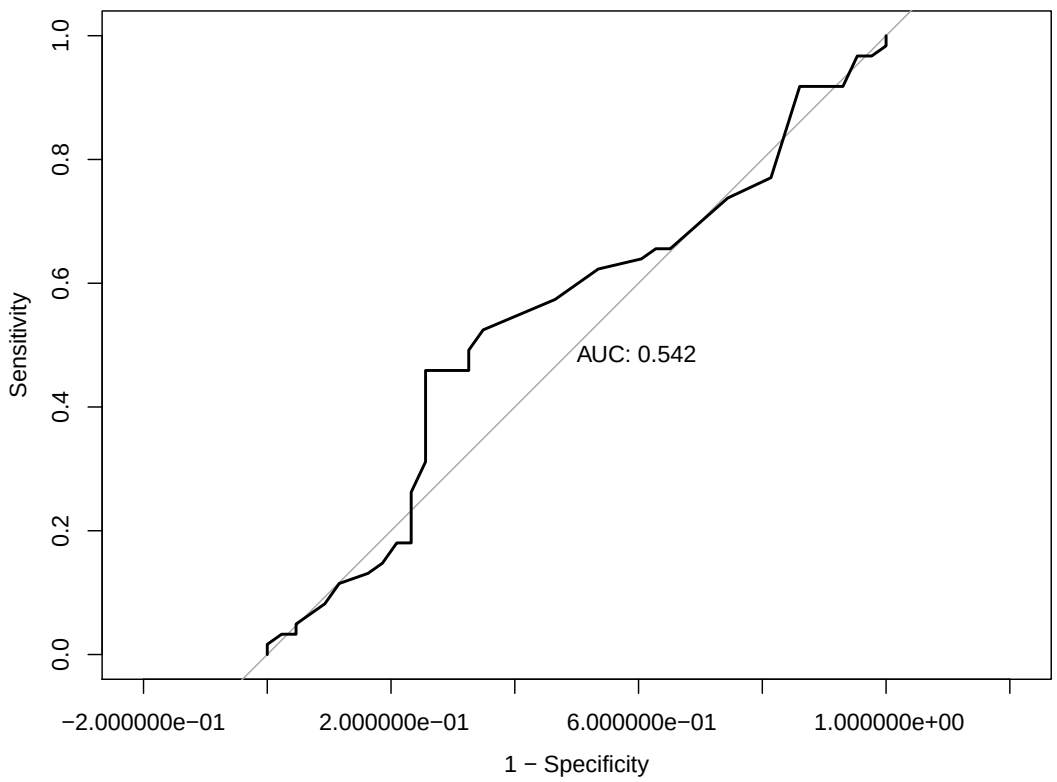



```
df_knn_pred = predict(df_knn, newdata = df_test, type = "prob")[,2]
df_roc_knn = roc(df_test$direction, df_knn_pred)
```

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

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

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



The AUC for KNN is 0.5419. This is relatively higher than the other AUC's.