p8106_hw3_yg2625

Yue Gu

April 9, 2019

This questions will be answered using the Weekly data set, which is part of the ISLR package. This data is similar in nature to the Smarket data on the textbook except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010. A description of the data can be found by typing ?Weekly in the Console. (Note that the column Today is not a predictor.)

Load Data

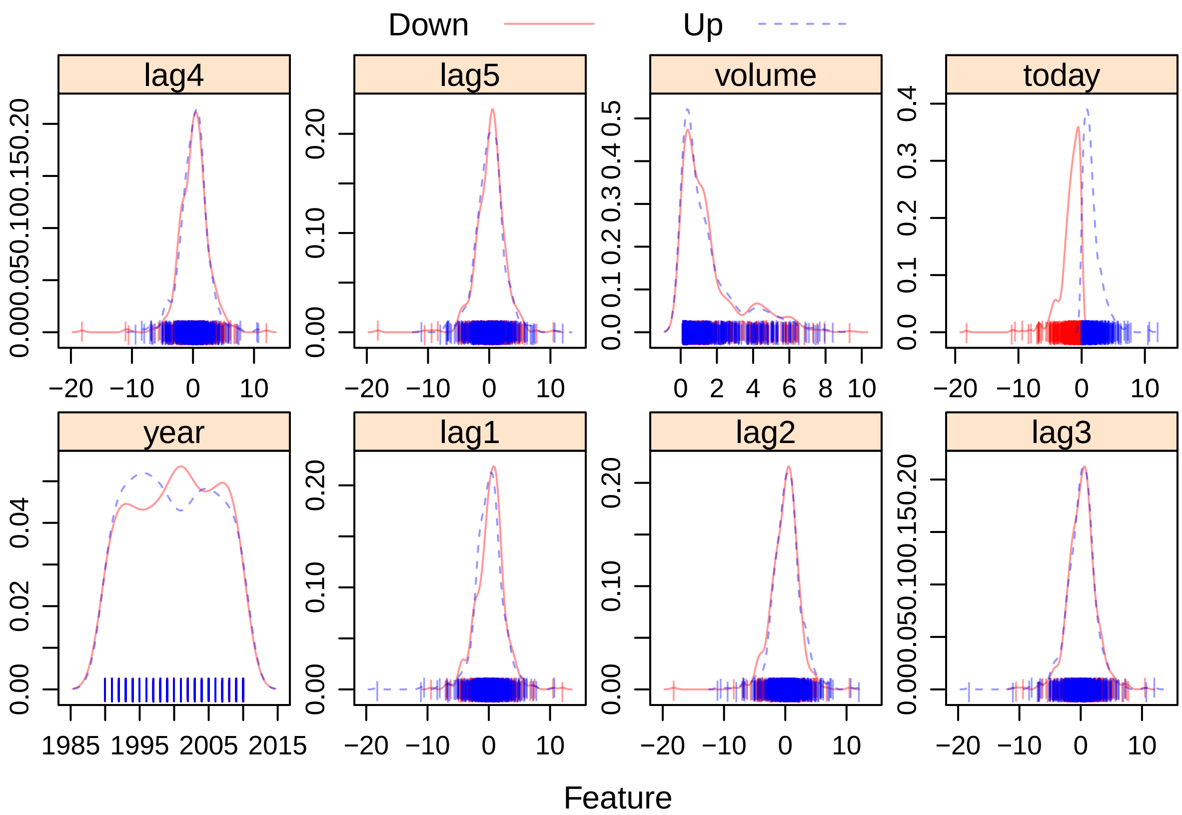
```
data("Weekly")
weekly = Weekly %>%
  janitor::clean_names()
head(weekly)
```

```
##   year   lag1   lag2   lag3   lag4   lag5    volume  today direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

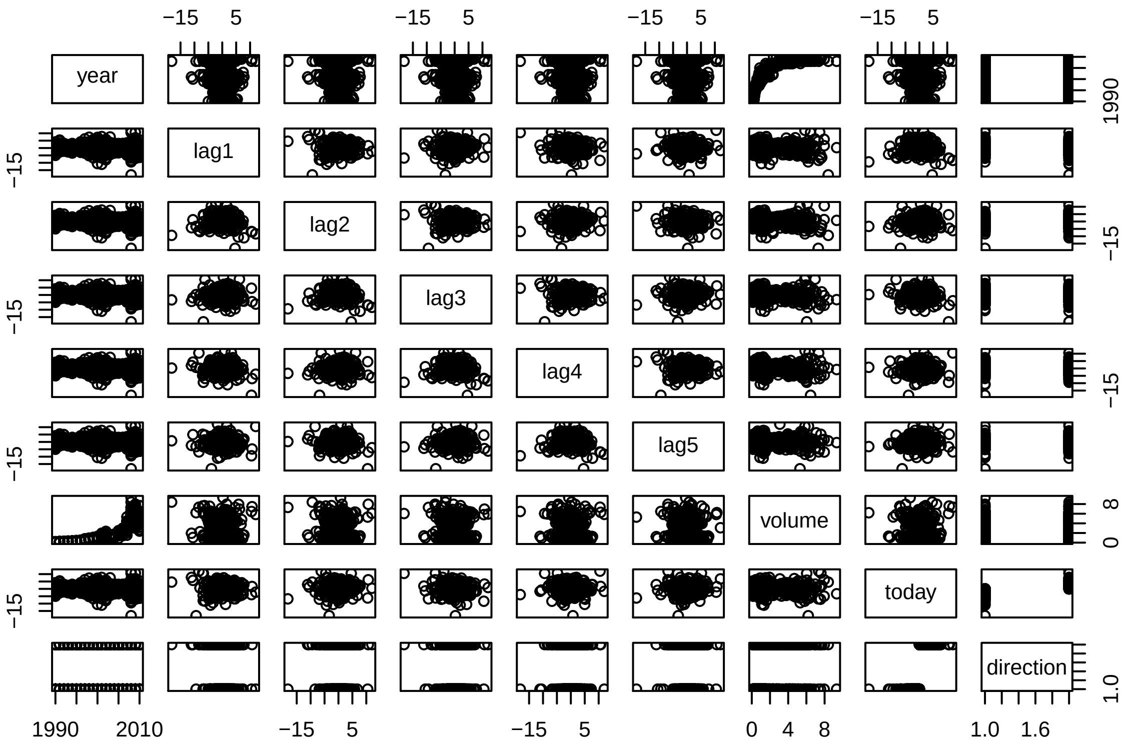
(a) Produce some graphical summaries of the Weekly data.

```
# start from some simple visualization of weekly

# density plot
transparentTheme(trans = .4)
featurePlot(x = weekly[, 1:8],
            y = weekly$direction,
            scales = list(x=list(relation="free"),
                          y=list(relation="free")),
            plot = "density", pch = "|",
            auto.key = list(columns = 2))
```




```r
# pairs scatterplot
pairs(weekly)
```



(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?

```
glm.fit <- glm(direction ~ lag1 + lag2 + lag3 + lag4 + lag5 + volume,
               data = weekly,
               family = binomial)
summary(glm.fit)
```

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

Based on the outputs, lag2 is significant with p-value = $0.0296 < 0.05$.

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

```
# first consider the Bayes classifier(cutoff = 0.5) and evaluate its performance on the data

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 = weekly$direction,
                positive = "Up")
```

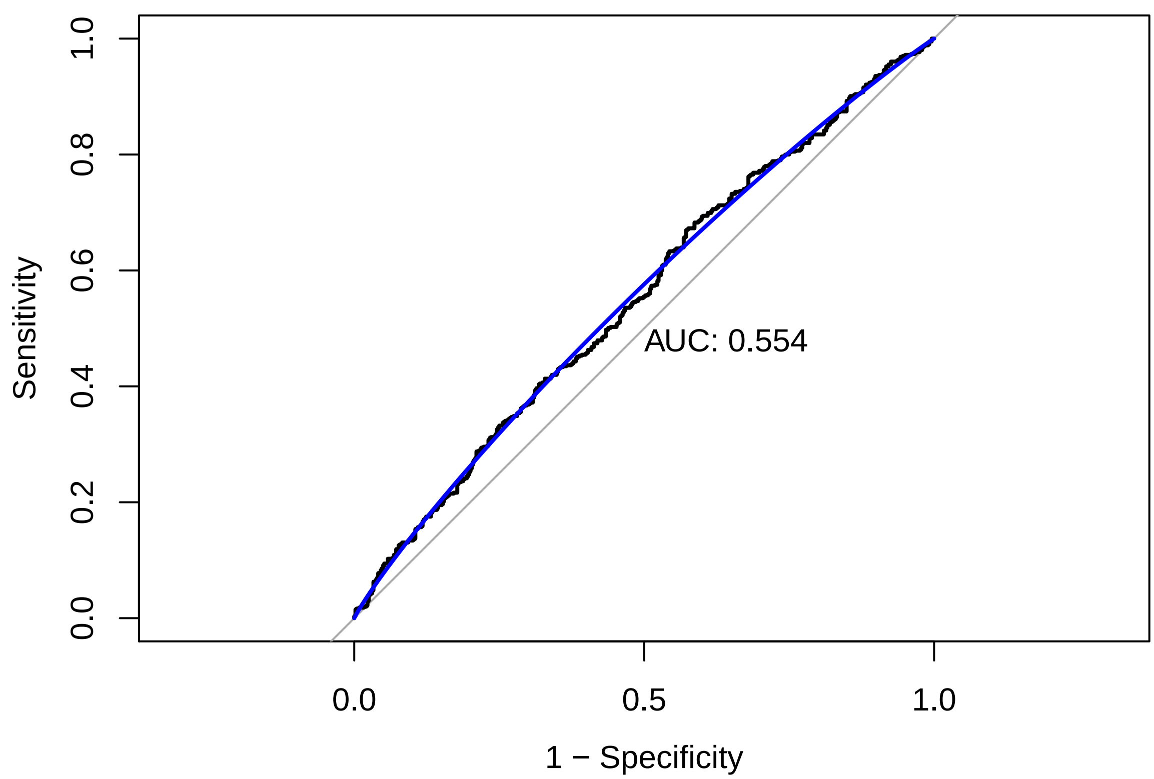
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down  Up
##       Down   54  48
##       Up    430 557
##
##                Accuracy : 0.5611
##                  95% CI : (0.531, 0.5908)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : 0.369
##
##                   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
##
```

Based on the output, the confusion matrix provides us with following results:
1. **Accuracy = 0.5611:** provides the probability of the correct classifer, which is the overall fraction of correct predictions. ((TP+TN)/n = (54+557)/1089)
2. **NIR = 0.5556:** provides the larger proportion of total positive observation vs. the proportion of total negative observations. (max((TP+FP)/n, (FN+TN)/n))
3. **Kappa = 0.035:** measures the agreement between classification and truth values. A kappa value closed to 1 meaning a good performance of the model.
4. **Sensitivity = 0.9207:** measures the proportion of actual positives that are correctly identified. (TP/(TP+FN))
5. **Specificity = 0.1116:** measures the proportion of actual negatives that are correctly identifed. (TN/(FP+TN))

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

```
roc.glm <- roc(weekly$direction, test.pred.prob)
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```



AUC = 0.554.

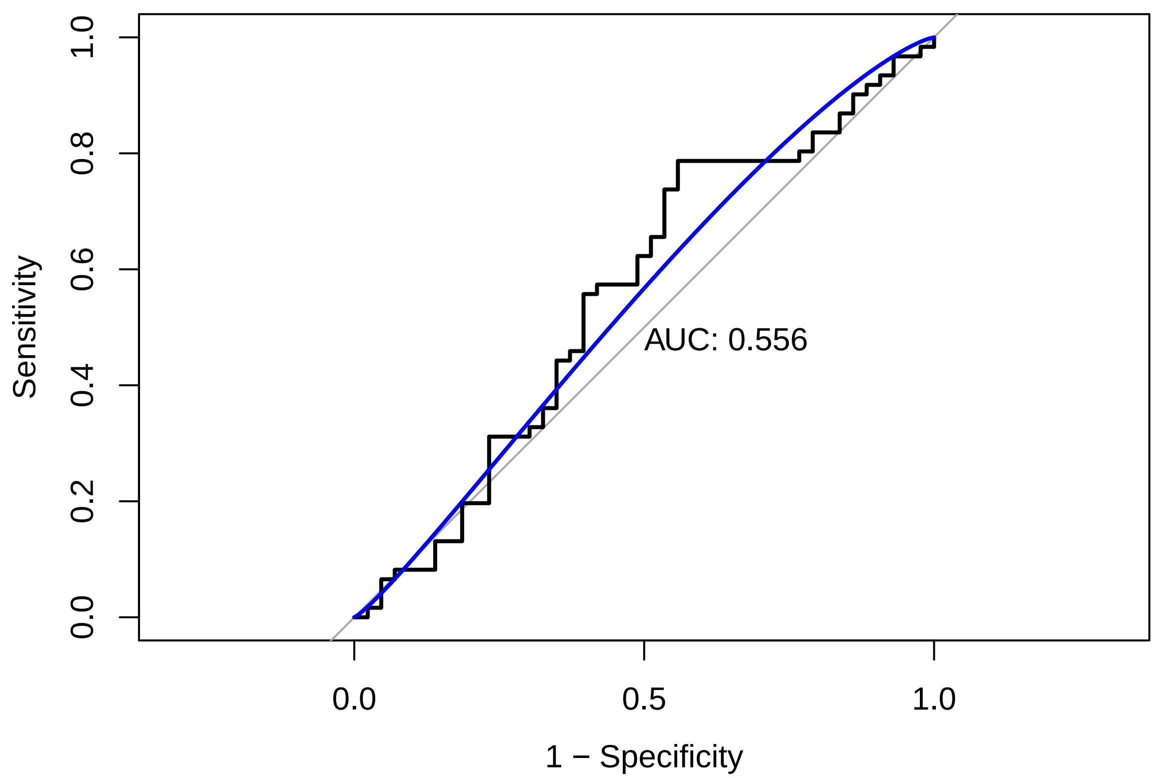
(e) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag1 and Lag2 as the predictors. Plot the ROC curve using the held out data (that is, the data from 2009 and 2010) and report the AUC.

```
# divide data into train and test
train_data = subset(weekly, year <= 2008)
test_data = subset(weekly, year>=2009)

# fit regression using training data
glm.fit_tr = glm(direction ~ lag1 + lag2,
               data = train_data,
               family = binomial)

# predict using test data
test.pred.prob2 = predict(glm.fit_tr,
                          newdata = test_data,
                          type = "response")
test.pred2 = rep("Down", length(test.pred.prob2))
test.pred2[test.pred.prob2 > 0.5] = "Up"

# plot ROC curve and report AUC
roc.glm2 <- roc(test_data$direction, test.pred.prob2)
plot(roc.glm2, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm2), col = 4, add = TRUE)
```



AUC = 0.556.

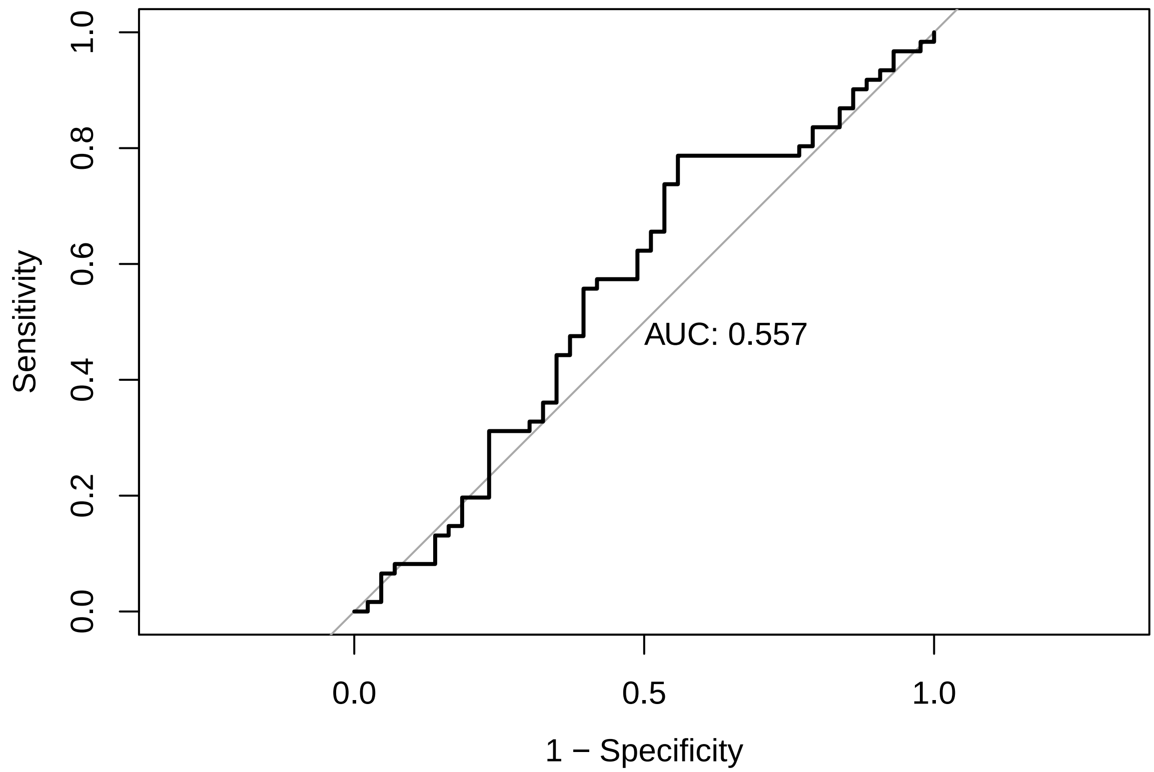
(f) Repeat (e) using LDA and QDA.

LDA

```
# fit model on trainning and predict on test
lda.fit = lda(direction ~ lag1 + lag2,
              data = train_data)
lda.pred = predict(lda.fit,
                   newdata = test_data)

# plot ROC curve
roc.lda = roc(test_data$direction, lda.pred$posterior[,2],
           levels = c("Down", "Up"))

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



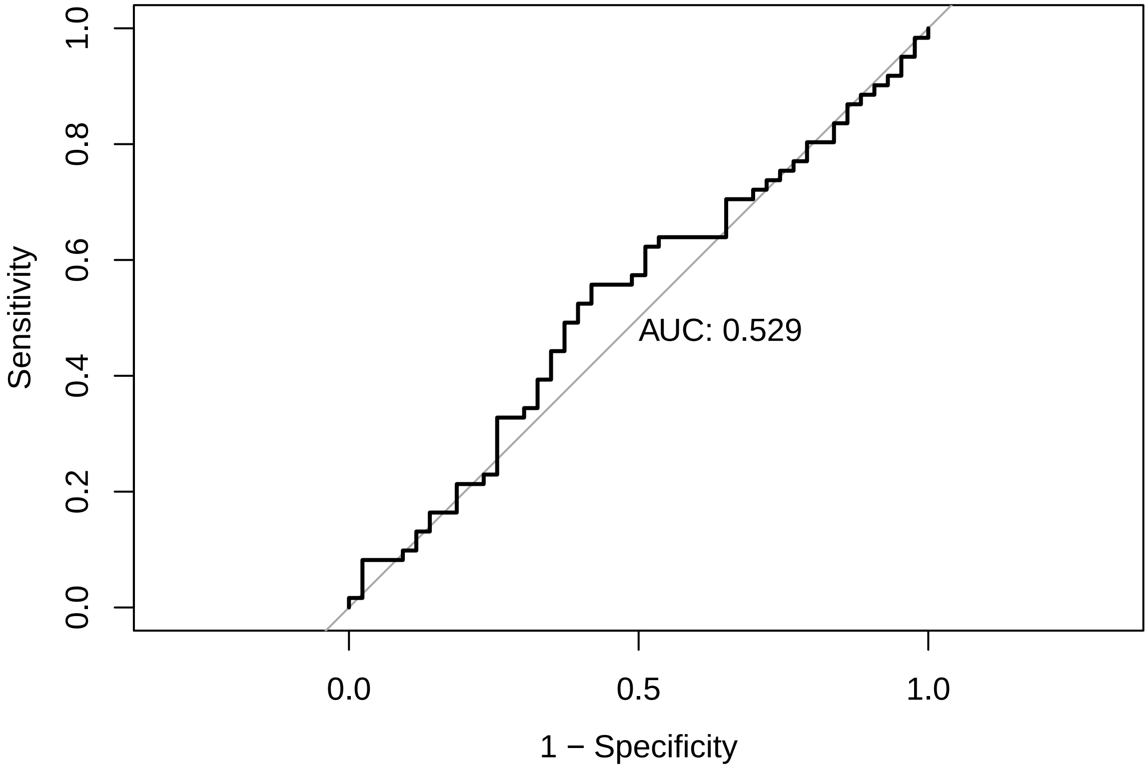
AUC = 0.557.

QDA

```
# fit model on trainning and predict on test
qda.fit = qda(direction ~ lag1 + lag2,
              data = train_data)
qda.pred = predict(qda.fit,
                   newdata = test_data)

# plot ROC curve
roc.qda = roc(test_data$direction, qda.pred$posterior[,2],
           levels = c("Down", "Up"))

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



AUC = 0.529.

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