Chapter 4

Book Work

Simple Logistic Regression

```r
data.default <- data.table(ISLR::Default)[, dflt := ifelse(default == "Yes", 1, 0)]

summary(model1 <- glm(dflt ~ balance, data = data.default, family = "binomial"))
```

```

Call:
glm(formula = dflt ~ balance, family = "binomial", data = data.default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2697  -0.1465  -0.0589  -0.0221   3.7589

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.065e+01  3.612e-01  -29.49   <2e-16 ***
balance      5.499e-03  2.204e-04   24.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1596.5  on 9998  degrees of freedom
AIC: 1600.5

Number of Fisher Scoring iterations: 8
```

```r
predict(model1, newdata = data.frame( balance = c(1000, 2000) ), type = "response")
```

```
          1           2
0.005752145 0.585769370
```

```r
data.default[, is_student := ifelse(student == "Yes", 1, 0)]

summary(model2 <- glm(dflt ~ is_student, data = data.default, family = "binomial"))
```

```

Call:
glm(formula = dflt ~ is_student, family = "binomial", data = data.default)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.2970  -0.2970  -0.2434  -0.2434   2.6585

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.50413    0.07071  -49.55  < 2e-16 ***
is_student   0.40489    0.11502    3.52 0.000431 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 2908.7  on 9998  degrees of freedom
AIC: 2912.7

Number of Fisher Scoring iterations: 6
```

```r
predict(model2, newdata = data.frame( is_student = c(1, 0) ), type = "response")
```

```
         1          2
0.04313859 0.02919501
```

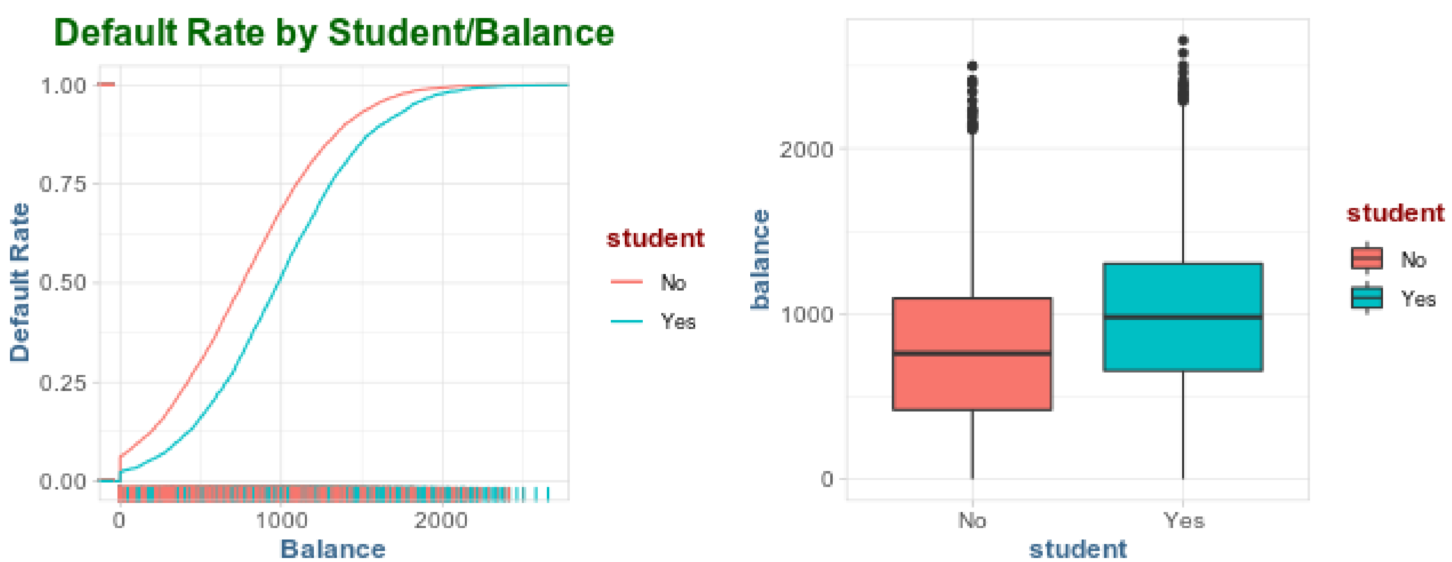
Multiple Logistic Regression

```r
summary(model3 <- glm(dflt ~ balance + is_student, data = data.default, family = "binomial"))
```

```
Call:
glm(formula = dflt ~ balance + is_student, family = "binomial",
    data = data.default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4578  -0.1422  -0.0559  -0.0203   3.7435

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.075e+01  3.692e-01 -29.116  < 2e-16 ***
balance      5.738e-03  2.318e-04  24.750  < 2e-16 ***
is_student  -7.149e-01  1.475e-01  -4.846 1.26e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.7  on 9997  degrees of freedom
AIC: 1577.7

Number of Fisher Scoring iterations: 8
```

```
p1 <- ggplot(data.default, aes(balance, dflt, color = student)) +
   stat_ecdf() +
   geom_rug(aes(balance, dflt)) +
   labs(x = "Balance", y = "Default Rate", title = "Default Rate by Student/Balance")

p2 <- ggplot(data.default, aes(student, balance, fill = student)) +
   geom_boxplot()

grid.arrange(p1, p2, nrow = 1)
```




```
predict(model3, newdata =
          data.frame( balance = c(1500, 1500),
                      is_student = c(1, 0) ),
        type = "response")
```

```
         1          2
0.05430945 0.10504923
```

R Lab

```r
Smarket <- as.data.table(ISLR::Smarket)

names(Smarket)
```

```
[1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
[7] "Volume"    "Today"     "Direction"
```

```r
dim(Smarket)
```

```
[1] 1250    9
```

```r
summary(Smarket)
```

```
      Year           Lag1                Lag2                Lag3
 Min.   :2001   Min.   :-4.922000   Min.   :-4.922000   Min.   :-4.922000
 1st Qu.:2002   1st Qu.:-0.639500   1st Qu.:-0.639500   1st Qu.:-0.640000
 Median :2003   Median : 0.039000   Median : 0.039000   Median : 0.038500
 Mean   :2003   Mean   : 0.003834   Mean   : 0.003919   Mean   : 0.001716
 3rd Qu.:2004   3rd Qu.: 0.596750   3rd Qu.: 0.596750   3rd Qu.: 0.596750
 Max.   :2005   Max.   : 5.733000   Max.   : 5.733000   Max.   : 5.733000
      Lag4                Lag5              Volume           Today
 Min.   :-4.922000   Min.   :-4.92200   Min.   :0.3561   Min.   :-4.922000
 1st Qu.:-0.640000   1st Qu.:-0.64000   1st Qu.:1.2574   1st Qu.:-0.639500
 Median : 0.038500   Median : 0.03850   Median :1.4229   Median : 0.038500
 Mean   : 0.001636   Mean   : 0.00561   Mean   :1.4783   Mean   : 0.003138
 3rd Qu.: 0.596750   3rd Qu.: 0.59700   3rd Qu.:1.6417   3rd Qu.: 0.596750
 Max.   : 5.733000   Max.   : 5.73300   Max.   :3.1525   Max.   : 5.733000
 Direction
 Down:602
 Up  :648
```

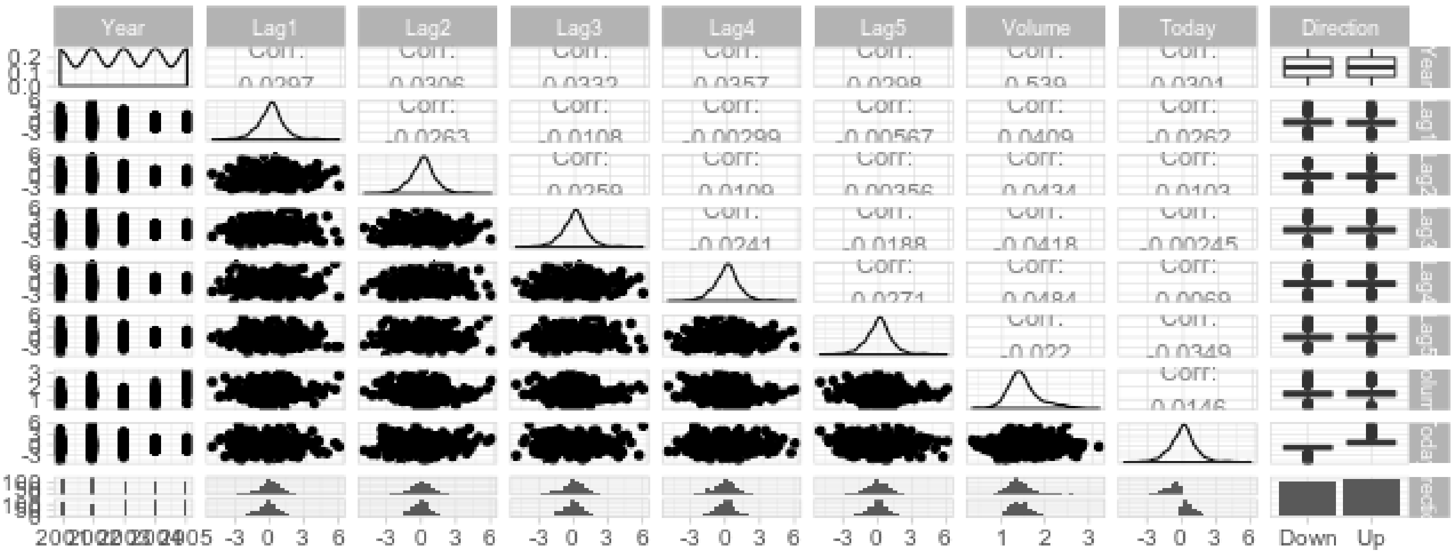
Pairs

```r
ggpairs(Smarket) %>%
   print(progress = F)
```

```
Warning in ggmatrix_gtable(x, ...): Please use the 'progress' parameter in
your ggmatrix-like function call. See ?ggmatrix_progress for a few examples.
ggmatrix_gtable 'progress' and 'progress_format' will soon be deprecated.TRUE
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

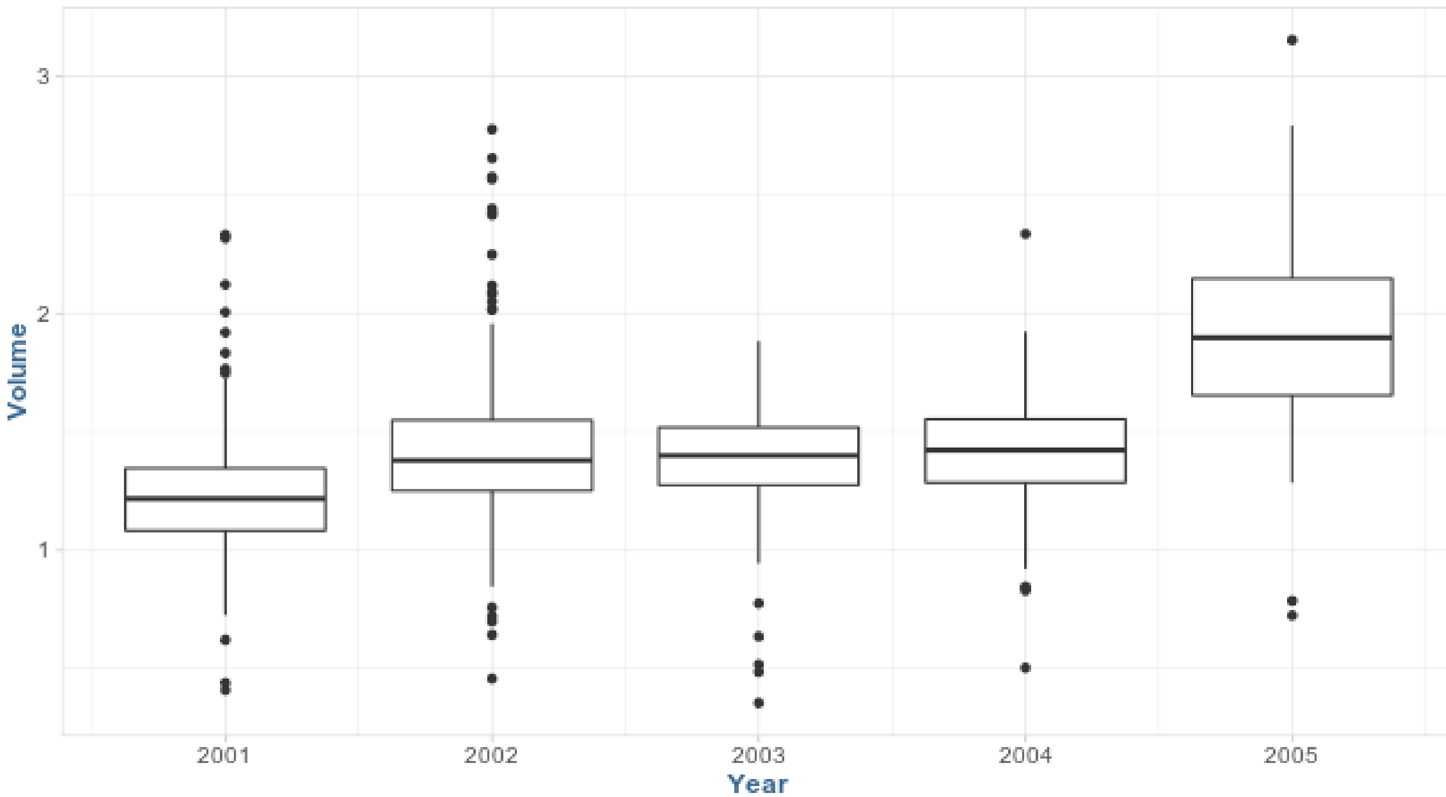
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
cor(Smarket %>% select(-Direction))
```

```
             Year         Lag1         Lag2         Lag3         Lag4
Year   1.00000000  0.029699649  0.030596422  0.033194581  0.035688718
Lag1   0.02969965  1.000000000 -0.026294328 -0.010803402 -0.002985911
Lag2   0.03059642 -0.026294328  1.000000000 -0.025896670 -0.010853533
Lag3   0.03319458 -0.010803402 -0.025896670  1.000000000 -0.024051036
Lag4   0.03568872 -0.002985911 -0.010853533 -0.024051036  1.000000000
Lag5   0.02978799 -0.005674606 -0.003557949 -0.018808338 -0.027083641
Volume 0.53900647  0.040909908 -0.043383215 -0.041823686 -0.048414246
Today  0.03009523 -0.026155045 -0.010250033 -0.002447647 -0.006899527
               Lag5      Volume        Today
Year    0.029787995  0.53900647  0.030095229
Lag1   -0.005674606  0.04090991 -0.026155045
Lag2   -0.003557949 -0.04338321 -0.010250033
Lag3   -0.018808338 -0.04182369 -0.002447647
Lag4   -0.027083641 -0.04841425 -0.006899527
Lag5    1.000000000 -0.02200231 -0.034860083
Volume -0.022002315  1.00000000  0.014591823
Today  -0.034860083  0.01459182  1.000000000
```

```r
ggplot(Smarket) +
   geom_boxplot(aes(Year, Volume, group = Year))
```



Logistic Regression

```r
summary(glm.fits <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
                 data = Smarket, family = binomial))
```

```

Call:
glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
    Volume, family = binomial, data = Smarket)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.446  -1.203   1.065   1.145   1.326

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.126000   0.240736  -0.523    0.601
Lag1        -0.073074   0.050167  -1.457    0.145
Lag2        -0.042301   0.050086  -0.845    0.398
Lag3         0.011085   0.049939   0.222    0.824
Lag4         0.009359   0.049974   0.187    0.851
```

```
Lag5          0.010313   0.049511   0.208    0.835
Volume        0.135441   0.158360   0.855    0.392

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1731.2  on 1249  degrees of freedom
Residual deviance: 1727.6  on 1243  degrees of freedom
AIC: 1741.6

Number of Fisher Scoring iterations: 3
```

```
coef(glm.fits)
```

```
 (Intercept)         Lag1         Lag2         Lag3         Lag4         Lag5
-0.126000257 -0.073073746 -0.042301344  0.011085108  0.009358938  0.010313068
      Volume
 0.135440659
```

Probabilites of going up (first 10 trading days)

```
glm.probs <- predict(glm.fits, type = "response")
head(glm.probs, 10)
```

```
        1         2         3         4         5         6         7         8
0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.5092292
        9        10
0.5176135 0.4888378
```

```
contrasts(Smarket$Direction)
```

```
     Up
Down  0
Up    1
```

Predictions

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

```
table(glm.pred, Smarket$Direction)
```

```

glm.pred Down  Up
    Down  145 141
    Up    457 507
```

```
mean(glm.pred == Smarket$Direction)
```

```
[1] 0.5216
```

Validation

Get the holdout set.

```
train <- (Smarket$Year < 2005)
Smarket.2005 <- Smarket[!train]
dim(Smarket.2005)
```

```
[1] 252   9
```

```
Direction.2005 <- Smarket$Direction[!train]
```

Train the logistic regression model.

```
glm.fits <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
                data = Smarket, family = binomial, subset = train)

glm.probs <- predict(glm.fits, Smarket.2005, type = "response")
```

Test

```
glm.pred <- rep("Down", 252)
glm.pred[glm.probs > 0.5] <- "Up"

table(glm.pred, Direction.2005)
```

```
        Direction.2005
glm.pred Down Up
    Down   77 97
    Up     34 44
```

```
mean(glm.pred == Direction.2005)
```

```
[1] 0.4801587
```

Model 2

```
summary(glm.fits <- glm(Direction ~ Lag1 + Lag2, data = Smarket, family = binomial, subset = tr
```

```

Call:
glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Smarket,
    subset = train)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.345  -1.188   1.074   1.164   1.326

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)  0.03222    0.06338   0.508    0.611
Lag1        -0.05562    0.05171  -1.076    0.282
Lag2        -0.04449    0.05166  -0.861    0.389

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1383.3  on 997  degrees of freedom
Residual deviance: 1381.4  on 995  degrees of freedom
AIC: 1387.4

Number of Fisher Scoring iterations: 3
```

```
glm.probs <- predict(glm.fits, Smarket.2005, type = "response")
glm.pred <- rep("Down", nrow(Smarket.2005))

glm.pred[glm.probs >= 0.5] <- "Up"

table(glm.pred, Direction.2005)
```

```
        Direction.2005
glm.pred Down  Up
    Down   35  35
    Up     76 106
```

```
mean(glm.pred == Direction.2005)
```

```
[1] 0.5595238
```

```
predict(glm.fits, newdata = data.table(Lag1 = c(1.2, 1.5),
                                       Lag2 = c(1.1, -0.8)),
        type = "response")
```

```
        1         2
0.4791462 0.4960939
```

Linear Discriminant Analysis

LDA is from **MASS** package.

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

```
        Length Class  Mode
prior   2      -none- numeric
counts  2      -none- numeric
means   4      -none- numeric
scaling 2      -none- numeric
lev     2      -none- character
```

```
svd     1      -none- numeric
N       1      -none- numeric
call    4      -none- call
terms   3      terms  call
xlevels 0      -none- list
```

```r
lda.fit
```

```
Call:
lda(Direction ~ Lag1 + Lag2, data = Smarket, subset = train)

Prior probabilities of groups:
    Down       Up
0.491984 0.508016

Group means:
            Lag1        Lag2
Down  0.04279022  0.03389409
Up   -0.03954635 -0.03132544

Coefficients of linear discriminants:
            LD1
Lag1 -0.6420190
Lag2 -0.5135293
```

```r
lda.pred <- predict(lda.fit, Smarket.2005)

names(lda.pred)
```

```
[1] "class"     "posterior" "x"
```

Predictions:

```r
lda.class <- lda.pred$class
table(lda.class, Direction.2005)
```

```
         Direction.2005
lda.class Down  Up
     Down   35  35
     Up     76 106
```

Note: almost identical to logistic regression.

```r
mean(lda.class == Direction.2005)
```

```
[1] 0.5595238
```

```r
sum(lda.pred$posterior[, 1] >= 0.5)
```

```
[1] 70
```

```r
sum(lda.pred$posterior[, 1] < 0.5)
```

```
[1] 182
```

The posterior probabilites output by the model corresponds to the probability that the market will decrease.

```r
lda.pred$posterior[1:20, 1]
```

```
        1         2         3         4         5         6         7         8 
0.4901792 0.4792185 0.4668185 0.4740011 0.4927877 0.4938562 0.4951016 0.4872861 
        9        10        11        12        13        14        15        16 
0.4907013 0.4844026 0.4906963 0.5119988 0.4895152 0.4706761 0.4744593 0.4799583 
       17        18        19        20 
0.4935775 0.5030894 0.4978806 0.4886331 
```

```r
lda.class[1:20]
```

```
 [1] Up   Up   Up   Up   Up   Up   Up   Up   Up   Up   Up   Down Up   Up   Up  
[16] Up   Up   Down Up   Up  
Levels: Down Up
```

Apply a threshold of 90% to predictions:

```r
sum(lda.pred$posterior[, 1] > .9)
```

```
[1] 0
```

Quadratic Discriminant Analysis

```r
summary(qda.fit <- qda(Direction ~ Lag1 + Lag2, data = Smarket, subset = train))
```

```
        Length Class  Mode     
prior   2      -none- numeric  
counts  2      -none- numeric  
means   4      -none- numeric  
scaling 8      -none- numeric  
ldet    2      -none- numeric  
lev     2      -none- character
N       1      -none- numeric  
call    4      -none- call     
terms   3      terms  call     
xlevels 0      -none- list     
```

```r
qda.fit
```

```
Call:
qda(Direction ~ Lag1 + Lag2, data = Smarket, subset = train)
```

```
Prior probabilities of groups:
    Down       Up 
0.491984 0.508016 

Group means:
            Lag1        Lag2
Down  0.04279022  0.03389409
Up   -0.03954635 -0.03132544
```

Predictions

```
qda.class <- predict(qda.fit, Smarket.2005)$class
table(qda.class, Direction.2005)
```

```
         Direction.2005
qda.class Down  Up
     Down   30  20
     Up     81 121
```

```
mean(qda.class == Direction.2005)
```

```
[1] 0.5992063
```

K-Nearest Neighbors

Data Setup

```
train.X <- with(Smarket, cbind(Lag1, Lag2))[train, ]
test.X <- with(Smarket, cbind(Lag1, Lag2))[!train, ]
train.Direction <- Smarket$Direction[train]
```

KNN

```
set.seed(1)

knn.pred <- knn(train.X, test.X, train.Direction, k = 1)
table(knn.pred, Direction.2005)
```

```
        Direction.2005
knn.pred Down Up
    Down   43 58
    Up     68 83
```

```
set.seed(1)

knn.pred <- knn(train.X, test.X, train.Direction, k = 3)
table(knn.pred, Direction.2005)
```

```
        Direction.2005
knn.pred Down Up
    Down   48 55
    Up     63 86
```

Caravan Insurance Data

```r
caravan <- Caravan

dim(caravan)
```

```
[1] 5822   86
```

```r
summary(caravan$Purchase)
```

```
  No  Yes
5474  348
```

```r
table(caravan$Purchase) %>% prop.table()
```

```

        No        Yes
0.94022673 0.05977327
```

```r
standardized.X <- scale(caravan[, -86])

var(caravan[, 1])
```

```
[1] 165.0378
```

```r
var(caravan[, 2])
```

```
[1] 0.1647078
```

```r
var(standardized.X[, 1])
```

```
[1] 1
```

```r
var(standardized.X[, 2])
```

```
[1] 1
```

- K=1

```r
test <- 1:1000
train.X <- standardized.X[-test,]
test.X <- standardized.X[test,]

train.Y <- caravan$Purchase[-test]
test.Y <- caravan$Purchase[test]
```

```r
set.seed(1)

knn.pred <- knn(train.X, test.X, train.Y, k = 1)
mean(test.Y != knn.pred)
```

```
[1] 0.118
```

```r
mean(test.Y != "No")
```

```
[1] 0.059
```

```r
result <- table(knn.pred, test.Y)
result
```

```
        test.Y
knn.pred  No Yes
     No  873  50
     Yes  68   9
```

```r
result %>% prop.table()
```

```
        test.Y
knn.pred    No   Yes
     No  0.873 0.050
     Yes 0.068 0.009
```

- K=3

```r
set.seed(1)

knn.pred <- knn(train.X, test.X, train.Y, k = 3)
mean(test.Y != knn.pred)
```

```
[1] 0.074
```

```r
mean(test.Y != "No")
```

```
[1] 0.059
```

```r
result <- table(knn.pred, test.Y)
result
```

```
        test.Y
knn.pred  No Yes
     No  921  54
     Yes  20   5
```

```r
result %>% prop.table()
```

```
        test.Y
```

```
knn.pred    No   Yes
     No  0.921 0.054
     Yes 0.020 0.005
```

- K=5

```
set.seed(1)

knn.pred <- knn(train.X, test.X, train.Y, k = 5)
mean(test.Y != knn.pred)
```

```
[1] 0.066
```

```
mean(test.Y != "No")
```

```
[1] 0.059
```

```
result <- table(knn.pred, test.Y)
result
```

```
        test.Y
knn.pred  No Yes
     No  930  55
     Yes  11   4
```

```
result %>% prop.table()
```

```
        test.Y
knn.pred    No   Yes
     No  0.930 0.055
     Yes 0.011 0.004
```

Logistic Regression Alternative

```
glm.fits <- glm(Purchase ~ ., data = caravan, family = binomial,
                subset = -test)
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
glm.probs <- predict(glm.fits, caravan[test,], type = "response")

# .5 cut-off
glm.pred <- rep("No", 1000)
glm.pred[glm.probs > .5] <- "Yes"

results <- table(glm.pred, test.Y)
results %>% prop.table()
```

```
        test.Y
glm.pred    No   Yes
     No  0.934 0.059
```

```
     Yes 0.007 0.000
```

```
# .25 cut-off
glm.pred <- rep("No", 1000)
glm.pred[glm.probs > .25] <- "Yes"

results <- table(glm.pred, test.Y)
results %>% prop.table()
```

```
        test.Y
glm.pred    No   Yes
     No  0.919 0.048
     Yes 0.022 0.011
```

```
# Quiz

bal <- 1936.75

exp(-10.6513 + 0.0055 * bal) / ( 1 + exp(-10.6513 + 0.0055*bal))
```

```
[1] 0.5002062
```

```
b0 <- -6; b1 <- 0.05; b2 <- 1
x1 <- 50; x2 <- 3.5
exp(b0 + b1 * x1 + b2 * x2) / ( 1 + exp(b0 + b1 * x1 + b2 * x2))
```

```
[1] 0.5
```

Conceptual

1.)

Using a little bit of algebra, prove that (4.2) is equivalent to (4.3). In other words, the logistic function representaion and logit representation for the logistic regression models are equivalent.

4.2) $p(x) = \frac{e^{(\beta_0 + \beta_1 X)}}{1 + e^{\beta_0 + \beta_1 X}}$

4.3) $\frac{p(x)}{1 - p(x)} = e^{\beta_0 + \beta_1 X}$

Solution

$1 - p(x) = 1 - \frac{e^{(\beta_0 + \beta_1 X)}}{1 + e^{\beta_0 + \beta_1 X}} = \frac{1}{1 + e^{\beta_0 + \beta_1 X}}$

$\frac{1}{1 - p(x)} = 1 + e^{\beta_0 + \beta_1 X}$

2.)

It was stated in the text that classifying an observation to the class for which (4.13) is largest. Prove that this is the case. In other words, under the assumption that the observations in the *k*th class are drawn from a $N \sim (\mu, \sigma^2)$ distribution, the Bayes' clssifier assigns an observation to the class for which the discriminant function is maximized.

4.12:

$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}(x-\mu_k)^2}}{\sum_{l=1}^{K} \pi_l e^{-\frac{1}{2\sigma^2}(x-\mu_k)^2}}$

Solution

$f''_x = ln\pi_k + ln(\frac{1}{\sqrt{2\pi\sigma}}) + ln\, e^{-\frac{1}{2\sigma^2}(x-\mu_k)^2}$

$f'''_x = ln\, \pi_k - \frac{1}{2\sigma^2}(x-\mu_k)^2$

$f'''_x = ln\, \pi_k + \frac{x\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2}$

$\delta_k(x) = \frac{\mu_k}{\sigma^2}x - \frac{\mu_k^2}{2\sigma^2} + log(\pi_k)$

3.)

This problem relates to the QDA model, in which the observations within each class are drawn from a normal distribution with a class-specific mean vector and a class specific covariance matrix. We consider the simple case where p=1; i.e. there is only one feature. Suppose that we have K classes, and if an observation belongs to the kth class then X comes from a one-dimensional normal distribution, $X \sim N(\mu_k, \sigma_k^2)$. Recall that the density function for the one-dimensional normal distribution is given in (4.11). Prove that in this case, the Bayes' classifier is not linear. Argue that it is in fact quadratic.

Solution

From the previous answer, we can expand the last term which is not linear in x.

4.)

When the number of features p is large, there tends to be a deterioration in the performance of KNN and other local approaches that perform prediction using only observations that are near the test observation for which a prediction must be made. This phenomenon is known as the curse of dimensionality, and it ties into the fact that non-parametric approaches often perform poorly when p is large. We will now investigate this curse.

a.) Suppose that we have a set of observations, each with measurements on p=1 feature, X. We assume that X is uniformly (evenly) distributed on [0,1]. Associated with each observation is a response value. Suppose that we wish to predict a test observation's response using only observations that are within 10% of the range of X closest to that test observation. For instance, in order to predict the response for a test observation with X=0.6, we will use observations in the range [0.55,0.65]. On average, what fraction of the available observations will we use to make the prediction ?

Applied

10.)

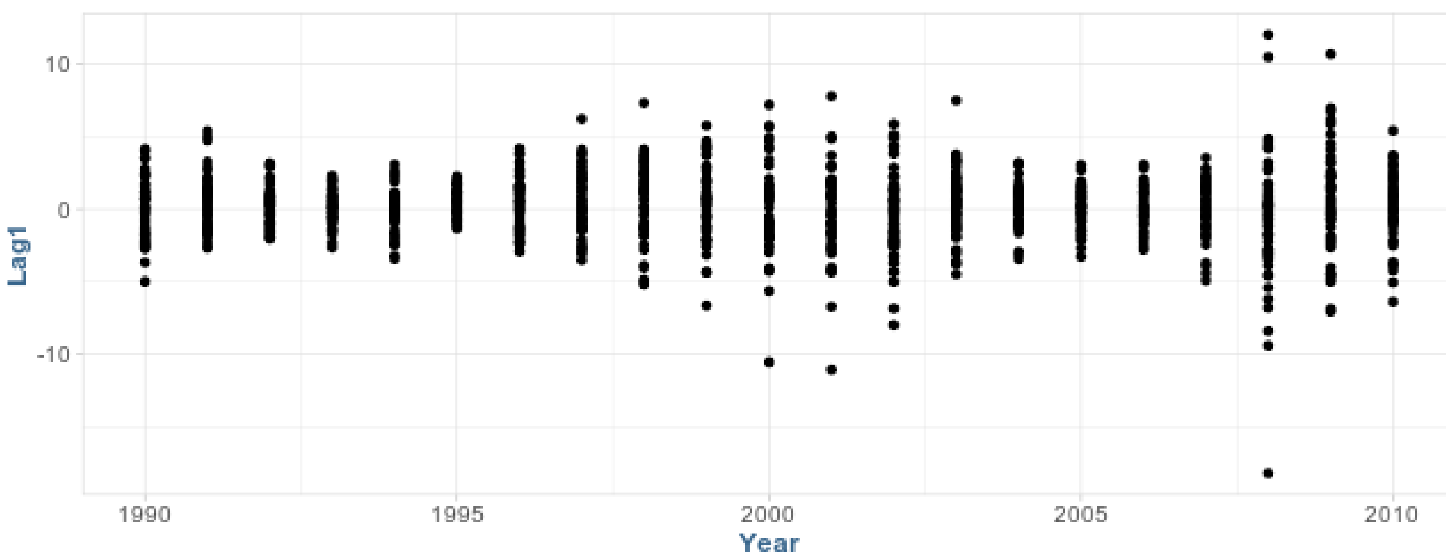
This question should be answered using the "Weekly" data set, which is part of the "ISLR" package. This data is similar in nature to the "Smarket" data from this chapter's lab, except that it contains 1089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

```
weekly <- as.data.table(ISLR::Weekly)

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
```

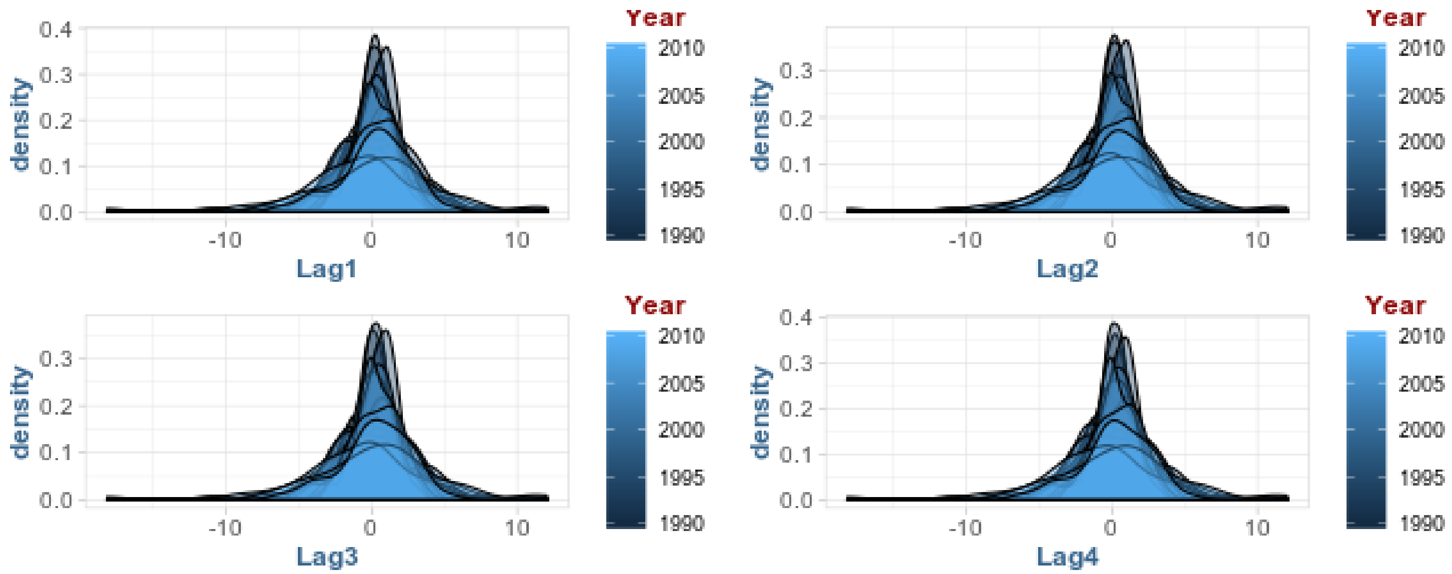
a.) Produce some numerical and graphical summaries of the "Weekly" data. Do there appear to be any patterns ?

```
weekly %>% ggplot() +
  geom_point(aes(Year, Lag1))
```




```
p1 <- weekly %>% ggplot() +
  geom_density(aes(Lag1, group = Year, fill = Year), alpha = .4)

p2 <- weekly %>% ggplot() +
  geom_density(aes(Lag2, group = Year, fill = Year), alpha = .4)
```

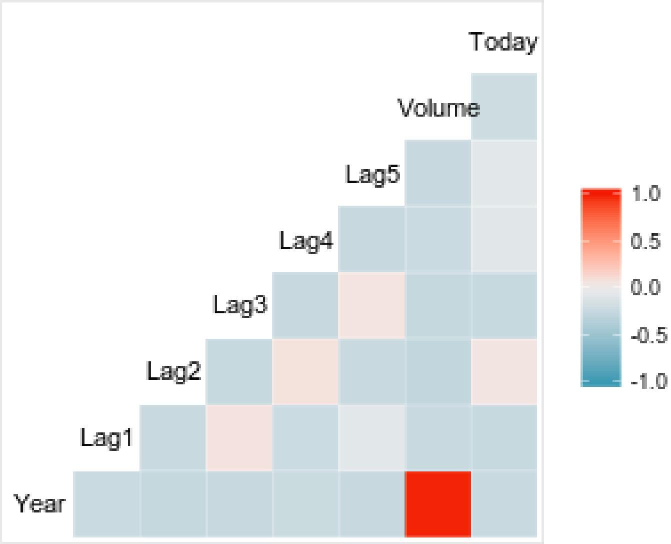
```
p3 <- weekly %>% ggplot() +
   geom_density(aes(Lag3, group = Year, fill = Year), alpha = .4)

p4 <- weekly %>% ggplot() +
   geom_density(aes(Lag4, group = Year, fill = Year), alpha = .4)

grid.arrange(p1, p2, p3, p4, nrow = 2)
```




```
cor <- cor(weekly %>% select(-Direction))
cor
```

```
              Year         Lag1        Lag2        Lag3         Lag4
Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
               Lag5      Volume        Today
Year   -0.030519101  0.84194162 -0.032459894
Lag1   -0.008183096 -0.06495131 -0.075031842
Lag2   -0.072499482 -0.08551314  0.059166717
Lag3    0.060657175 -0.06928771 -0.071243639
Lag4   -0.075675027 -0.06107462 -0.007825873
Lag5    1.000000000 -0.05851741  0.011012698
Volume -0.058517414  1.00000000 -0.033077783
Today   0.011012698 -0.03307778  1.000000000
```

```r
ggcorr(cor)
```



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. Use the summary function to print the results. Do any of the predictors appear to be statistically significant ? If so, which ones ?

```r
summary(glm.fit <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = weekly, fa
```

```
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
```

```r
glm.fit
```

```

Call:  glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
    Volume, family = binomial, data = weekly)

Coefficients:
(Intercept)         Lag1         Lag2         Lag3         Lag4         Lag5
    0.26686     -0.04127      0.05844     -0.01606     -0.02779     -0.01447
     Volume
   -0.02274

Degrees of Freedom: 1088 Total (i.e. Null);  1082 Residual
Null Deviance:      1496
Residual Deviance: 1486     AIC: 1500
```

It appears only **Lag2** is statistically significant.

c.) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```r
glm.pred <- ifelse(predict(glm.fit, type = "response") > .5, "Up", "Down")

table(glm.pred, weekly$Direction) %>% prop.table()
```

```

glm.pred       Down         Up
    Down 0.04958678 0.04407713
    Up   0.39485767 0.51147842
```

We may conclude that the percentage of correct predictions on the training data is (54+557)/1089 wich is equal to 56.1065197%. In other words 43.8934803% is the training error rate, which is often overly optimistic. We could also say that for weeks when the market goes up, the model is right 92.0661157% of the time (557/(48+557)). For weeks when the market goes down, the model is right only 11.1570248% of the time (54/(54+430)).

d.) Now fit the logistic regression model using a training data period from 1990 to 2008, with "Lag2" as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 to 2010).

```r
train <- weekly[Year < 2009,]
test <- weekly[Year > 2009, .(Direction, Lag2)]
```

```r
summary(glm.fit <- glm(Direction ~ Lag2, data = train, family = binomial))
```

```

Call:
glm(formula = Direction ~ Lag2, family = binomial, data = train)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.536  -1.264   1.021   1.091   1.368

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.20326    0.06428   3.162  0.00157 **
Lag2         0.05810    0.02870   2.024  0.04298 *
---
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: 1350.5  on 983  degrees of freedom
AIC: 1354.5

Number of Fisher Scoring iterations: 4
```

```r
test$pred <- ifelse(predict(glm.fit, newdata = test[, .(Lag2)], type = "response") > .5, "Up",

table(test$Direction, test$pred) %>% prop.table() * 100
```

```

            Down        Up
  Down  9.615385 28.846154
  Up    1.923077 59.615385
```

```r
mean(test$Direction == test$pred)
```

```
[1] 0.6923077
```

e.) Repeat (d) using LDA.

```r
summary(lda.fit <- lda(Direction ~ Lag2, data = train))
```

```
        Length Class  Mode
prior   2      -none- numeric
counts  2      -none- numeric
means   2      -none- numeric
scaling 1      -none- numeric
```

```
lev      2      -none- character
svd      1      -none- numeric
N        1      -none- numeric
call     3      -none- call
terms    3      terms  call
xlevels  0      -none- list
```

```r
lda.fit
```

```
Call:
lda(Direction ~ Lag2, data = train)

Prior probabilities of groups:
     Down        Up
0.4477157 0.5522843

Group means:
            Lag2
Down -0.03568254
Up    0.26036581

Coefficients of linear discriminants:
           LD1
Lag2 0.4414162
```

```r
test$pred <- predict(lda.fit, newdata = test[, .(Lag2)], type = "response")$class

mean(test$Direction == test$pred)
```

```
[1] 0.6923077
```

```r
table(test$Direction, test$pred) %>% prop.table() * 100
```

```


            Down        Up
  Down  9.615385 28.846154
  Up    1.923077 59.615385
```

f.) Repeat (d) using QDA.

```r
summary(qda.fit <- qda(Direction ~ Lag2, data = train))
```

```
        Length Class  Mode
prior   2      -none- numeric
counts  2      -none- numeric
means   2      -none- numeric
scaling 2      -none- numeric
ldet    2      -none- numeric
```

```
lev     2      -none- character
N       1      -none- numeric
call    3      -none- call
terms   3      terms  call
xlevels 0      -none- list
```

```r
qda.fit
```

```
Call:
qda(Direction ~ Lag2, data = train)

Prior probabilities of groups:
     Down        Up
0.4477157 0.5522843

Group means:
            Lag2
Down -0.03568254
Up    0.26036581
```

```r
lda.test <- test[, .(Direction, Lag2)]

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

lda.test[, pred := lda.pred$class]

with(lda.test, mean(Direction == pred))
```

```
[1] 0.6153846
```

```r
with(lda.test, table(Direction, pred)) %>% prop.table() * 100
```

```
         pred
Direction     Down       Up
     Down  0.00000 38.46154
     Up    0.00000 61.53846
```

g.) Repeat (d) using KNN with k = 1.

```r
train <- weekly[Year < 2009,]
test <- weekly[Year > 2009, .(Direction, Lag2)]

train.X <- train[, .(Lag2)]
test.X <- test[, .(Lag2)]
train.Direction <- train$Direction

knn.pred <- knn(train.X, test.X, train.Direction, k = 1)
```

```r
table(test$Direction, knn.pred) %>% prop.table()
```

```
      knn.pred
             Down        Up
  Down 0.1538462 0.2307692
  Up   0.2884615 0.3269231
```

```r
mean(test$Direction == knn.pred)
```

```
[1] 0.4807692
```

h.) Which of these methods appear to provide the best results on this data?

LDA and Logistic Regression appear to have the best performance on this particular set of data.

11.)

In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the *Auto* data set.

```r
auto <- as.data.table(ISLR::Auto)

auto %>% glimpse()
```

```
Observations: 392
Variables: 9
$ mpg          <dbl> 18, 15, 18, 16, 17, 15, 14, 14, 14, 15, 15, 14, 15, 14, …
$ cylinders    <dbl> 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 4, 6, 6, 6, 4,…
$ displacement <dbl> 307, 350, 318, 304, 302, 429, 454, 440, 455, 390, 383, 3…
$ horsepower   <dbl> 130, 165, 150, 150, 140, 198, 220, 215, 225, 190, 170, 1…
$ weight       <dbl> 3504, 3693, 3436, 3433, 3449, 4341, 4354, 4312, 4425, 38…
$ acceleration <dbl> 12.0, 11.5, 11.0, 12.0, 10.5, 10.0, 9.0, 8.5, 10.0, 8.5,…
$ year         <dbl> 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, …
$ origin       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 3,…
$ name         <fct> chevrolet chevelle malibu, buick skylark 320, plymouth s…
```

a.) Create a binary variable, *mpg01*, that contains a 1 if mpg contains a value above its median. You can compute the median using the median() function. Note you may find it helpful to use the *data.frame()* function to create a single data set containing both *mpg01* and other Auto variables.

```r
cutpoint <- median(auto$mpg)

auto[, mpg01 := mpg > cutpoint]
```

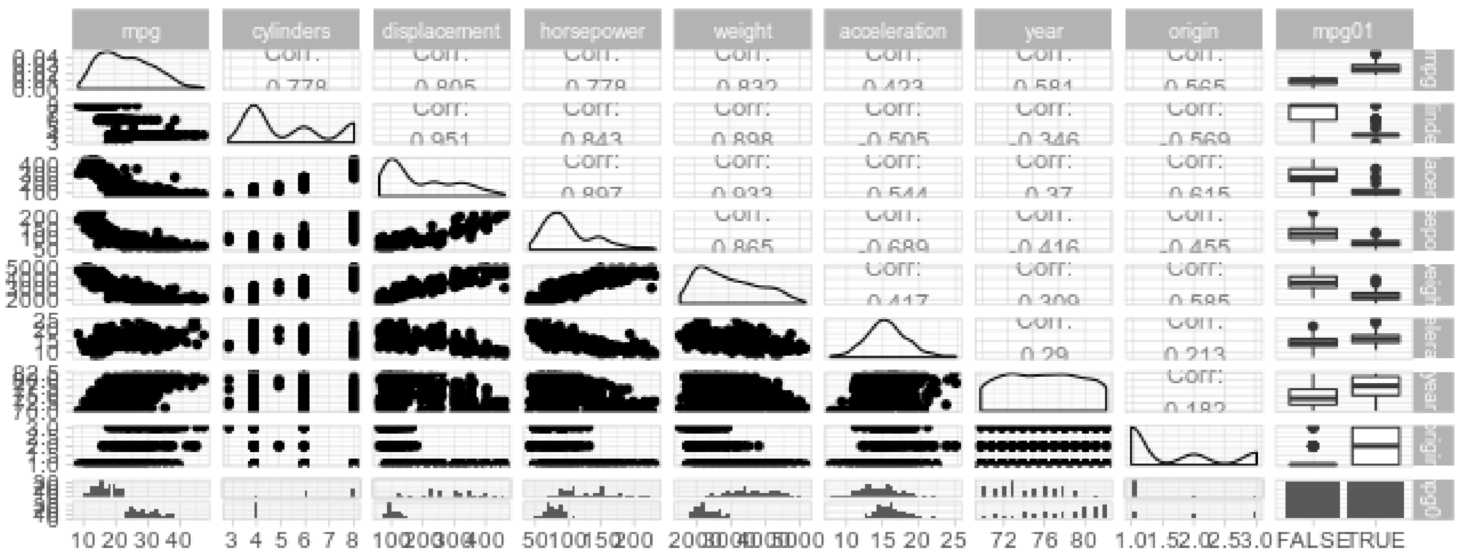
b.) Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question.

```r
summary(auto)
```

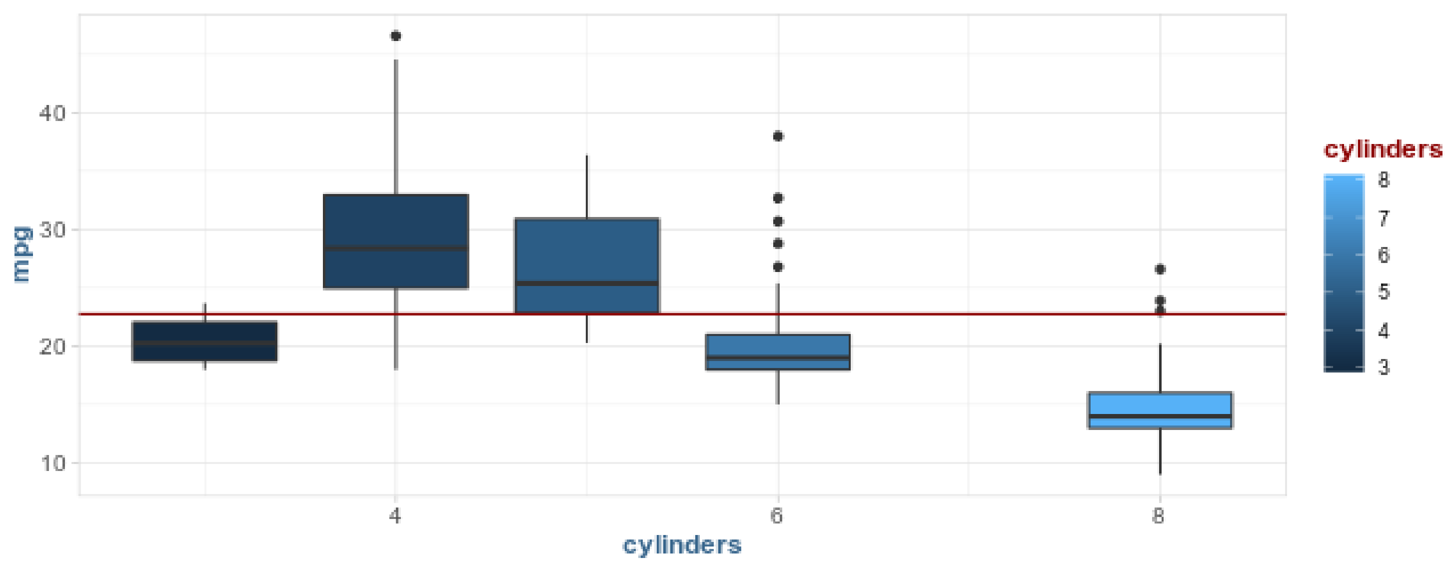
```
      mpg          cylinders      displacement     horsepower        weight    
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613  
 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225  
 Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804  
 Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978  
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615  
 Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140  
                                                                               
  acceleration        year           origin                      name    
 Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5  
 1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5  
 Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5  
 Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4  
 3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4  
 Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4  
                                                 (Other)           :365  
   mpg01        
 Mode :logical  
 FALSE:196      
 TRUE :196      
```

```r
ggpairs(auto %>% select(-name))
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

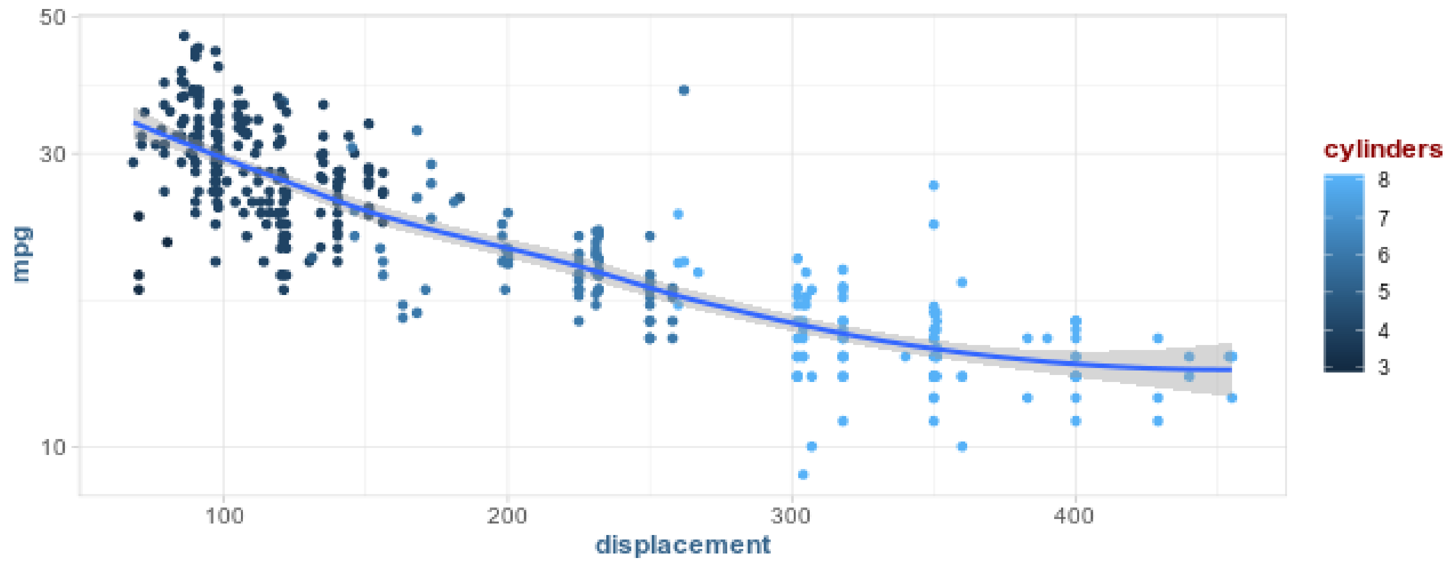



```r
ggplot(auto, aes(cylinders, mpg, fill = cylinders)) +
   geom_boxplot(aes(group = cylinders)) +
   geom_hline(yintercept = cutpoint, col = "darkred")
```



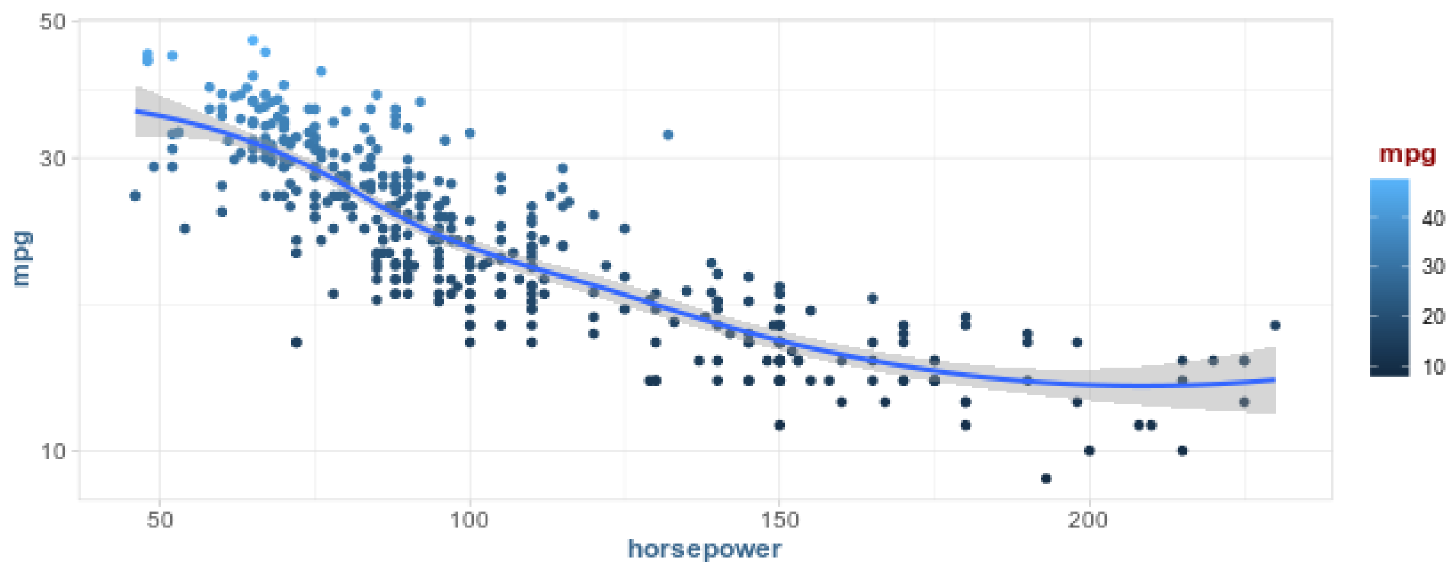

```r
ggplot(auto, aes(displacement, mpg)) +
   geom_point(aes(color = cylinders)) +
   geom_smooth(method = "auto") +
   scale_y_continuous(trans = "log10")
```

```
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```r
ggplot(auto, aes(horsepower, mpg)) +
   geom_point(aes(color = mpg)) +
   geom_smooth(method = "auto") +
   scale_y_continuous(trans = "log10")
```

```
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



c.) Split the data in to training / test sets.

```r
auto[, prob := ifelse(mpg01 == T, 1, 0)]

auto.split <- initial_split(auto, prop = 0.7, strata = "mpg01")

auto.train <- training(auto.split)
auto.test <- testing(auto.split)
```

d.) Perform Logistic Regression.

```
auto.train %>% glimpse()
```

```
Observations: 276
Variables: 11
$ mpg          <dbl> 18, 15, 18, 16, 17, 15, 14, 14, 14, 15, 15, 14, 22, 18, …
$ cylinders    <dbl> 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 6, 6, 6, 4, 4, 4, 4,…
$ displacement <dbl> 307, 350, 318, 304, 302, 429, 454, 440, 455, 390, 383, 3…
$ horsepower   <dbl> 130, 165, 150, 150, 140, 198, 220, 215, 225, 190, 170, 1…
$ weight       <dbl> 3504, 3693, 3436, 3433, 3449, 4341, 4354, 4312, 4425, 38…
$ acceleration <dbl> 12.0, 11.5, 11.0, 12.0, 10.5, 10.0, 9.0, 8.5, 10.0, 8.5,…
$ year         <dbl> 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, …
$ origin       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 2, 2, 2,…
$ name         <fct> chevrolet chevelle malibu, buick skylark 320, plymouth s…
$ mpg01        <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, …
$ prob         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,…
```

```
glm.fit1 <- glm(mpg01 ~ cylinders, data = auto.train, family = binomial)
glm.fit2 <- glm(mpg01 ~ horsepower, data = auto.train, family = binomial)

auto.train1 <- broom::augment(glm.fit1, auto.train) %>% mutate(.fitted = exp(.fitted))
auto.train2 <- broom::augment(glm.fit2, auto.train) %>% mutate(.fitted = exp(.fitted))

auto.train1 %>% summary()
```

```
  .rownames             mpg          cylinders      displacement
 Length:276         Min.   :10.00   Min.   :3.000   Min.   : 68.0
 Class :character   1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0
 Mode  :character   Median :22.75   Median :4.500   Median :151.0
                    Mean   :23.40   Mean   :5.504   Mean   :195.3
                    3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:272.0
                    Max.   :44.60   Max.   :8.000   Max.   :455.0

   horsepower        weight      acceleration        year           origin
 Min.   : 46.0   Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
 1st Qu.: 75.0   1st Qu.:2242   1st Qu.:13.57   1st Qu.:73.00   1st Qu.:1.000
 Median : 95.0   Median :2822   Median :15.40   Median :76.00   Median :1.000
 Mean   :105.0   Mean   :2982   Mean   :15.49   Mean   :75.85   Mean   :1.554
 3rd Qu.:122.8   3rd Qu.:3622   3rd Qu.:17.30   3rd Qu.:79.00   3rd Qu.:2.000
 Max.   :225.0   Max.   :4955   Max.   :24.60   Max.   :82.00   Max.   :3.000

                      name       mpg01              prob
 amc matador            :  5   Mode :logical   Min.   :0.0
 amc hornet             :  4   FALSE:138       1st Qu.:0.0
```

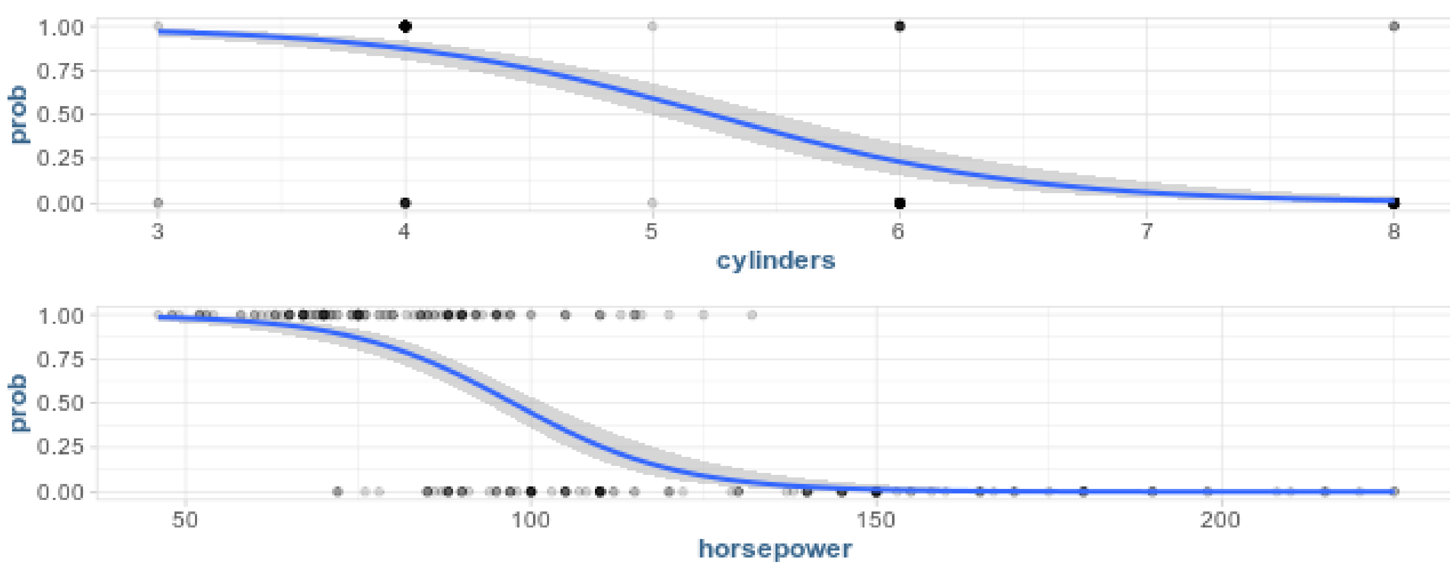
```
 chevrolet chevette        :  4   TRUE :138       Median :0.5  
 chevrolet caprice classic:  3                    Mean   :0.5  
 chevrolet impala          :  3                   3rd Qu.:1.0  
 chevrolet nova            :  3                   Max.   :1.0  
 (Other)                   :254                                
    .fitted            .se.fit            .resid              .hat         
 Min.   : 0.01341   Min.   :0.1896   Min.   :-2.65665   Min.   :0.003878  
 1st Qu.: 0.01341   1st Qu.:0.2508   1st Qu.:-0.30483   1st Qu.:0.003878  
 Median : 4.19606   Median :0.2568   Median : 0.04041   Median :0.006924  
 Mean   : 3.83774   Mean   :0.3299   Mean   : 0.03344   Mean   :0.007246  
 3rd Qu.: 6.93749   3rd Qu.:0.5448   3rd Qu.: 0.51895   3rd Qu.:0.006924  
 Max.   :33.08653   Max.   :0.5448   Max.   : 2.94113   Max.   :0.011814  
                                                                          
     .sigma           .cooksd            .std.resid      
 Min.   :0.8047   Min.   :2.620e-05   Min.   :-2.66222  
 1st Qu.:0.8231   1st Qu.:5.433e-05   1st Qu.:-0.30616  
 Median :0.8236   Median :5.060e-04   Median : 0.04050  
 Mean   :0.8227   Mean   :4.494e-03   Mean   : 0.03339  
 3rd Qu.:0.8241   3rd Qu.:1.845e-03   3rd Qu.: 0.52076  
 Max.   :0.8242   Max.   :1.457e-01   Max.   : 2.94685  
```

```r
auto.train2 %>% summary()
```

```
  .rownames              mpg          cylinders      displacement  
 Length:276         Min.   :10.00   Min.   :3.000   Min.   : 68.0  
 Class :character   1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0  
 Mode  :character   Median :22.75   Median :4.500   Median :151.0  
                    Mean   :23.40   Mean   :5.504   Mean   :195.3  
                    3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:272.0  
                    Max.   :44.60   Max.   :8.000   Max.   :455.0  
                                                                   
   horsepower        weight      acceleration        year           origin     
 Min.   : 46.0   Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000  
 1st Qu.: 75.0   1st Qu.:2242   1st Qu.:13.57   1st Qu.:73.00   1st Qu.:1.000  
 Median : 95.0   Median :2822   Median :15.40   Median :76.00   Median :1.000  
 Mean   :105.0   Mean   :2982   Mean   :15.49   Mean   :75.85   Mean   :1.554  
 3rd Qu.:122.8   3rd Qu.:3622   3rd Qu.:17.30   3rd Qu.:79.00   3rd Qu.:2.000  
 Max.   :225.0   Max.   :4955   Max.   :24.60   Max.   :82.00   Max.   :3.000  
                                                                               
                        name       mpg01              prob    
 amc matador              :  5   Mode :logical   Min.   :0.0  
 amc hornet               :  4   FALSE:138       1st Qu.:0.0  
 chevrolet chevette       :  4   TRUE :138       Median :0.5  
 chevrolet caprice classic:  3                   Mean   :0.5  
```

```
 chevrolet impala          :  3                   3rd Qu.:1.0
 chevrolet nova            :  3                   Max.   :1.0
 (Other)                   :254
    .fitted            .se.fit            .resid               .hat
 Min.   : 0.00002   Min.   :0.1720   Min.   :-2.12288   Min.   :4.534e-05
 1st Qu.: 0.11915   1st Qu.:0.1901   1st Qu.:-0.52763   1st Qu.:6.731e-03
 Median : 1.22712   Median :0.2829   Median : 0.07749   Median :7.406e-03
 Mean   : 5.40380   Mean   :0.3866   Mean   : 0.03572   Mean   :7.246e-03
 3rd Qu.: 6.61656   3rd Qu.:0.4850   3rd Qu.: 0.55177   3rd Qu.:8.006e-03
 Max.   :76.14852   Max.   :1.4519   Max.   : 2.43529   Max.   :1.300e-02

     .sigma          .cooksd            .std.resid
 Min.   :0.8624   Min.   :0.0000000   Min.   :-2.13136
 1st Qu.:0.8729   1st Qu.:0.0001070   1st Qu.:-0.53108
 Median :0.8744   Median :0.0007248   Median : 0.07764
 Mean   :0.8734   Mean   :0.0035949   Mean   : 0.03589
 3rd Qu.:0.8749   3rd Qu.:0.0028550   3rd Qu.: 0.55398
 Max.   :0.8750   Max.   :0.0956606   Max.   : 2.44779
```

```r
p1 <- ggplot(auto.train1, aes(cylinders, prob)) +
   geom_point(alpha = 0.15) +
   geom_smooth(method = "glm", method.args = list(family = "binomial"))

p2 <- ggplot(auto.train2, aes(horsepower, prob)) +
   geom_point(alpha = 0.15) +
   geom_smooth(method = "glm", method.args = list(family = "binomial"))

gridExtra::grid.arrange(p1, p2, nrow = 2)
```



```r
tidy(glm.fit1)
```

```
# A tibble: 2 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)     8.19     0.860      9.51 1.82e-21
2 cylinders      -1.56     0.169     -9.26 2.09e-20
```

```r
tidy(glm.fit2)
```

```
# A tibble: 2 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)   8.21      1.04        7.91 2.56e-15
2 horsepower   -0.0842    0.0110     -7.69 1.47e-14
```

```r
exp(coef(glm.fit1))
```

```
 (Intercept)    cylinders
3589.1967834    0.2096772
```

```r
exp(coef(glm.fit2))
```

```
 (Intercept)   horsepower
3670.0566719    0.9192059
```

```r
confint(glm.fit1)
```

```
Waiting for profiling to be done...
```

```
                2.5 %    97.5 %
(Intercept)  6.602396  9.984287
cylinders   -1.915732 -1.253153
```

```r
confint(glm.fit2)
```

```
Waiting for profiling to be done...
```

```
                 2.5 %      97.5 %
(Intercept)  6.3415589 10.42586597
horsepower  -0.1076986 -0.06460986
```

Cross-validated Logistic Regression

```r
cols.exclude <- c("name", "mpg", "prob")
auto.train3 <- auto.train[, -cols.exclude, with = F]

summary(glm.fit3 <- glm(mpg01 ~ ., data = auto.train3, family = binomial))
```

```
Call:
glm(formula = mpg01 ~ ., family = binomial, data = auto.train3)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.27576  -0.11505   0.01117   0.22541   2.95845

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)  -18.634238   6.543570  -2.848 0.004403 **
cylinders     -0.198333   0.474392  -0.418 0.675890
displacement   0.003795   0.013334   0.285 0.775967
horsepower    -0.006867   0.027470  -0.250 0.802592
weight        -0.005180   0.001422  -3.642 0.000271 ***
acceleration   0.072515   0.173210   0.419 0.675467
year           0.432114   0.084699   5.102 3.36e-07 ***
origin         0.190709   0.433854   0.440 0.660249
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 382.62  on 275  degrees of freedom
Residual deviance: 114.12  on 268  degrees of freedom
AIC: 130.12

Number of Fisher Scoring iterations: 7
```

```r
auto.train3 <- auto.train[, -cols.exclude, with = F][, mpg01 := as.factor(mpg01)]

cv.model.logit <- train(
   mpg01 ~ .,
   data = auto.train3,
   method = "glm",
   family = "binomial",
   trControl = trainControl(method = "cv", number = 10)
)

auto.train3$pred <- predict(cv.model.logit, auto.train3)

# in-sample performance
table(auto.train3$mpg01, auto.train3$pred) %>% prop.table()
```

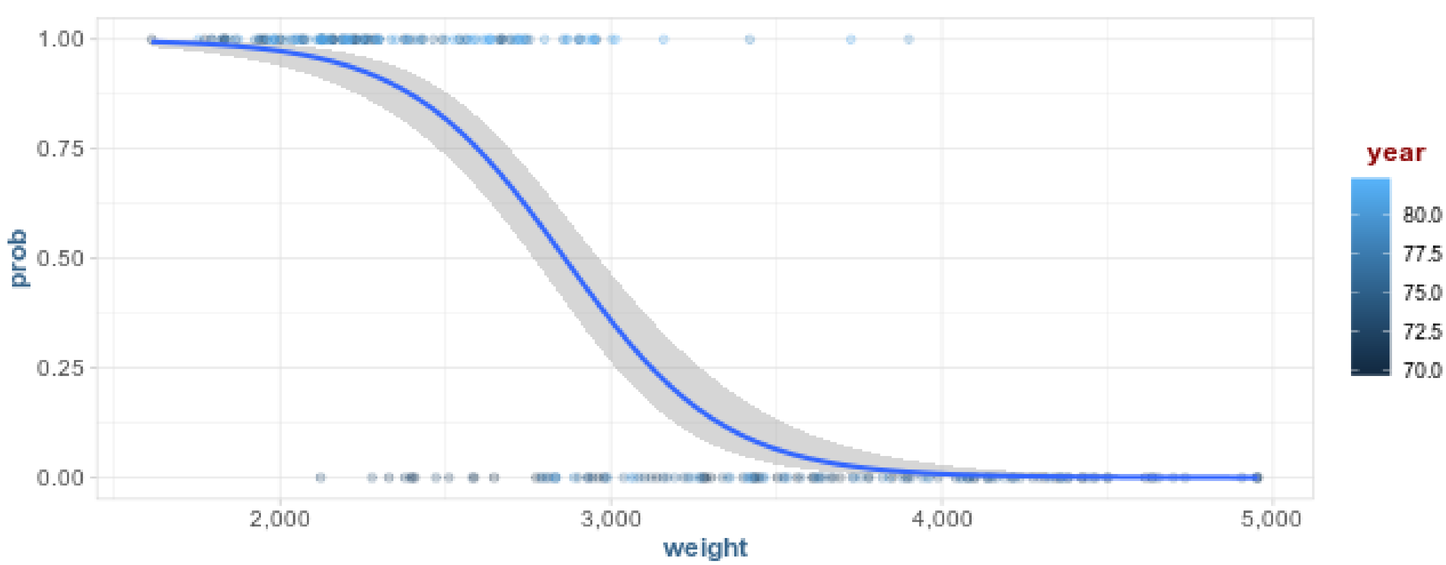
```
             FALSE       TRUE
```

```
  FALSE 0.44565217 0.05434783
  TRUE  0.03260870 0.46739130
```

```
# out of sample results
auto.test$pred <- predict(cv.model.logit, newdata = auto.test, type = "raw")

table(auto.test$mpg01, auto.test$pred) %>% prop.table() %>% round(digits = 3)
```

```
        FALSE  TRUE
  FALSE 0.440 0.060
  TRUE  0.026 0.474
```

```
ggplot(auto.train, aes(weight, prob, color = year)) +
   geom_point(alpha = 0.25) +
   geom_smooth(method = "glm", method.args = list(family = "binomial")) +
   scale_x_continuous(labels = scales::comma)
```



e.) Perform Linear Discriminant Analysis

Cross-validated Linear Discriminant Analysis

```
auto.train.lda <- auto.train[, -cols.exclude, with = F][, mpg01 := as.factor(mpg01)]

summary(lda.fit1 <- lda(mpg01 ~ ., data = auto.train.lda))
```

```
        Length Class  Mode
prior    2     -none- numeric
counts   2     -none- numeric
means   14     -none- numeric
scaling  7     -none- numeric
lev      2     -none- character
```

```
svd       1      -none- numeric
N         1      -none- numeric
call      3      -none- call
terms     3      terms  call
xlevels   0      -none- list
```

```
cv.model.lda <- train(
   mpg01 ~ .,
   auto.train.lda,
   method = "lda",
   family = "binomial",
   trControl = trainControl(method = "cv", number = 10)
)

# in-sample performance
auto.train.lda$pred <- predict(cv.model.lda)

with(auto.train.lda, mean(mpg01 == pred))
```

```
[1] 0.9057971
```

```
with(auto.train.lda, table(mpg01, pred)) %>% prop.table()
```

```
       pred
mpg01        FALSE       TRUE
  FALSE 0.42391304 0.07608696
  TRUE  0.01811594 0.48188406
```

```
# out of sample performance

auto.test.lda <- auto.test
auto.test.lda$pred <- predict(cv.model.lda, newdata = auto.test.lda)

with(auto.test.lda, mean(mpg01 == pred))
```

```
[1] 0.8965517
```

```
with(auto.test.lda, table(mpg01, pred)) %>% prop.table()
```

```
       pred
mpg01        FALSE       TRUE
  FALSE 0.41379310 0.08620690
  TRUE  0.01724138 0.48275862
```

Cross-validated KNN

```
auto.train.X <- auto.train[, .(mpg01, weight, year)][, mpg01 := as.factor(mpg01)]
auto.test.X <- auto.test[, .(mpg01, weight, year)]
```

```r
knn.fit <- train(
   mpg01 ~ weight + year,
   data = auto.train.X,
   method = "knn",
   trControl = trainControl(method = "cv", number = 10),
   tuneLength = 20
)

summary(knn.fit)
```

```
            Length Class      Mode
learn       2      -none-     list
k           1      -none-     numeric
theDots     0      -none-     list
xNames      2      -none-     character
problemType 1      -none-     character
tuneValue   1      data.frame list
obsLevels   2      -none-     character
param       0      -none-     list
```

```r
auto.knn <- auto.train
# in-sample performance
auto.knn$pred <- predict(knn.fit)

with(auto.knn, mean(mpg01 == pred))
```

```
[1] 0.9166667
```

```r
with(auto.knn, table(mpg01, pred)) %>% prop.table()
```

```
       pred
mpg01        FALSE       TRUE
  FALSE 0.45289855 0.04710145
  TRUE  0.03623188 0.46376812
```

```r
# out of sample performance
auto.test.knn <- auto.test

auto.test.knn$pred <- predict(knn.fit, newdata = auto.test.knn)

with(auto.test.knn, mean(mpg01 == pred))
```

```
[1] 0.862069
```

```r
with(auto.test.knn, table(mpg01, pred)) %>% prop.table()
```

```
       pred
mpg01        FALSE       TRUE
```

```
FALSE 0.41379310 0.08620690
TRUE  0.05172414 0.44827586
```

12.)

a.) Write a function, Power(), that prints out the result of rasing 2 to the 3rd power. In other words, your function should compute 2^3.

```
Power <- function(x) {
  print(2^3)
}
Power()
```

```
[1] 8
```

b.) Create a new function, Power2, that allows you to pass any two numbers, x and a, and prints out the value of x^a.

```
Power2 <- function(a, x) paste(a, "to the", x, "is", a ^ x)
Power2(3, 8)
```

```
[1] "3 to the 8 is 6561"
```

clean-up workspace

```
rm(list = ls())
```