Ch4_Lab

Adrien Osakwe

Stock Market Data

Working with the *Smarket* data set in the ISLR2 package. Looking to use data from 1250 days of stock indices to predict stock price direction.

```
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.1.2
```

```
names(Smarket)
```

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

```
dim(Smarket)
```

```
## [1] 1250    9
```

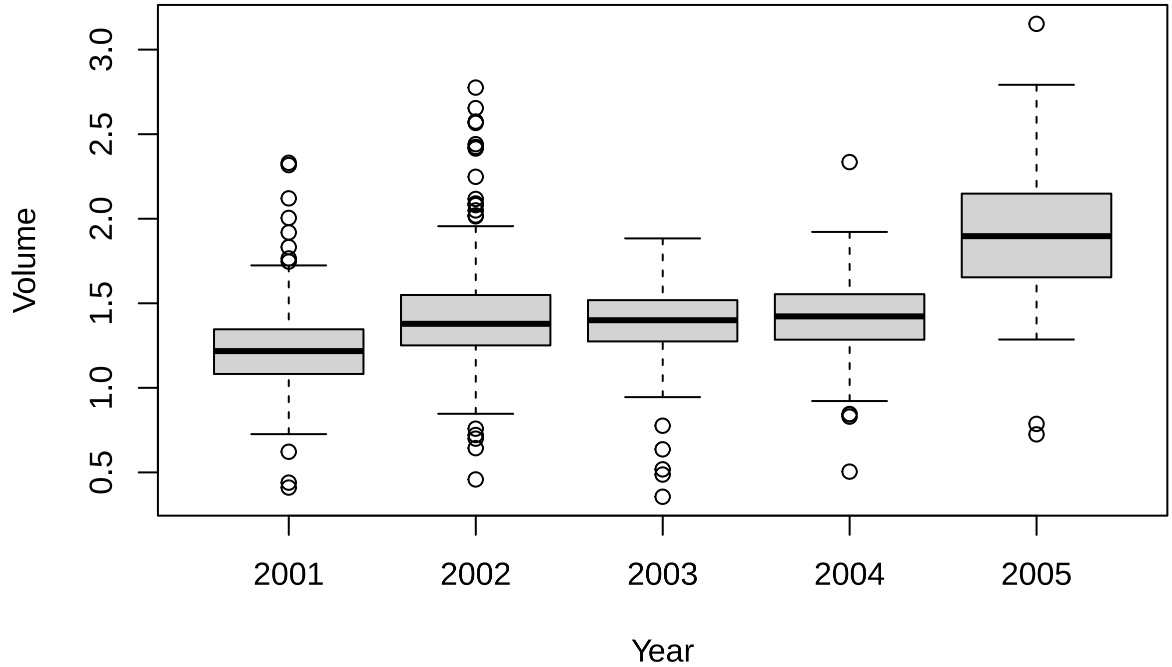
```
summary(Smarket)
```

```
##       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  
##            
##            
##            
##            
```

```r
## Generate correlation matrix using the quantitative variables
cor(Smarket[,-9])
```

```
##              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
# A quick scan of the matrix indicates a correlation between the year and the
#Volume
attach(Smarket)
boxplot(Volume ~ Year)
```



Logistic Regression

Using log regression to predict direction with lag[1:5] and volume

```r
#Use glm() with family = binomial to cal log-reg
glm.fits <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data =
                  Smarket, family = binomial)

summary(glm.fits)
```

```
##
## 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
```

```
#smallest p-value is for lag1. as b1 is negative, a positive return lag1 would
#predict and positive return today
summary(glm.fits)$coef
```

```
##                 Estimate Std. Error    z value  Pr(>|z|)
## (Intercept) -0.126000257 0.24073574 -0.5233966 0.6006983
## Lag1        -0.073073746 0.05016739 -1.4565986 0.1452272
## Lag2        -0.042301344 0.05008605 -0.8445733 0.3983491
## Lag3         0.011085108 0.04993854  0.2219750 0.8243333
## Lag4         0.009358938 0.04997413  0.1872757 0.8514445
## Lag5         0.010313068 0.04951146  0.2082966 0.8349974
## Volume       0.135440659 0.15835970  0.8552723 0.3924004
```

```
## Running predict() here requires us to set the type parameter to "response" to
# output a posterior probability
contrasts(Direction) #this informs us that the probability is for today being Up
```

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

```
glm.probs <- predict(glm.fits, type = "response")
glm.probs[1:10]
```

```
##         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
```

```
#Hm, it's all 50/50... de la poudre de perlinpinpin!

#Can then convert P(x|y) into labels based on their values
glm.pred <- rep("Down",1250)
glm.pred[glm.probs > 0.5] <- "Up"
#Plot predictions against training values
table(glm.pred,Direction)
```

```
##         Direction
```

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

```r
#Calculate Model Accuracy for training data
train_error <- (145 + 507)/1250
#Or do
mean(glm.pred == Direction)
```

```
## [1] 0.5216
```

```r
## ~50% accuracy on the training data... oh boy

##Creating a test data set
train <- ( Year < 2005)
Smarket.2005 <- Smarket[!train ,]
dim (Smarket.2005)
```

```
## [1] 252   9
```

```r
Direction.2005 <- Direction[!train]

#repeat log-red with data from before 2005
glm.fits1 <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data =
                   Smarket, family = binomial, subset = train)
glm.probs1 <- predict(glm.fits1, Smarket.2005, type = "response")
glm.pred1 <- rep("Down", 252)
glm.pred1[glm.probs1 > 0.5] <- "Up"
table(glm.pred1,Direction.2005)
```

```
##          Direction.2005
## glm.pred1 Down Up
##      Down   77 97
##      Up     34 44
```

```r
mean(glm.pred1 == Direction.2005)
```

```
## [1] 0.4801587
```

```r
mean(glm.pred1 != Direction.2005)
```

```
## [1] 0.5198413
```

```r
#Model is worse than a guess OoO


#Rebuild model using less parameters
#repeat log-red with data from before 2005
glm.fits2 <- glm(Direction ~ Lag1 + Lag2, data =
                   Smarket, family = binomial, subset = train)
```

```r
glm.probs2 <- predict(glm.fits2, Smarket.2005, type = "response")
glm.pred2 <- rep("Down", 252)
glm.pred2[glm.probs2 > 0.5] <- "Up"
table(glm.pred2,Direction.2005)
```

```
##          Direction.2005
## glm.pred2 Down  Up
##      Down   35  35
##      Up     76 106
```

```r
mean(glm.pred2 == Direction.2005)
```

```
## [1] 0.5595238
```

```r
# based on this confusion matrix one could develop a trading strategy
# (don't necessarily need a perfect model, just need one that improves your
#chances)
```

Linear Discriminant Analysis

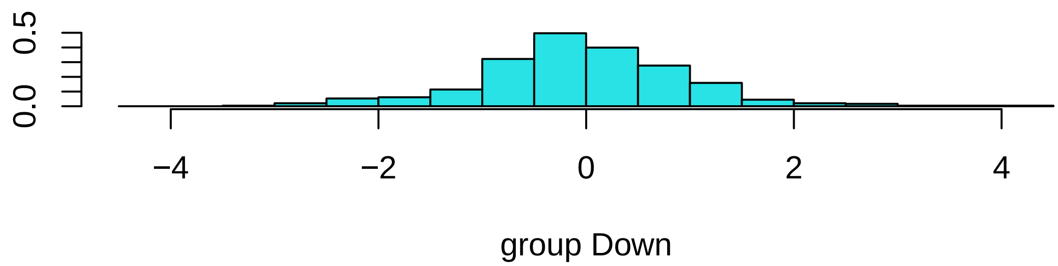
```r
##LDAs are fit with the lda function
library(MASS)
```

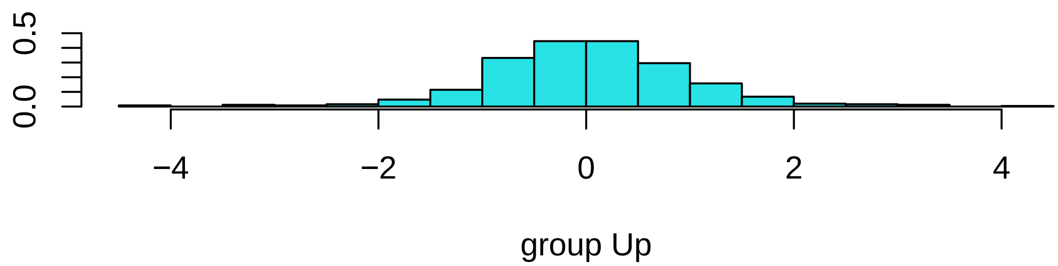
```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:ISLR2':
##
##     Boston
```

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

```
plot(lda.fit)
```






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

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

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

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

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

```
## [1] 0.5595238
```

```
#Can recreate the labels by applying a 50% threshold to the posterior
#probabilities
## May want to change posterior threshold depending on what bias we see in test
#results. Too many FP? Could try increasing the threshold
```

Quadratic Discriminant Analysis

```r
## Called with the qda() function

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

```r
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
```

```r
#At first glance we would say that the model is good at predicting Ups However, there is a very high FP
mean(qda.class == Direction.2005)
```

```
## [1] 0.5992063
```

```r
#Since the overall accuracy is 60% it's still much better than the previous models but is still a conce
```

Naive Bayes

```r
#Need to load the *e1071* library which contains the Naive Bayes method for R
library(e1071)

nb.fit <- naiveBayes(Direction ~ Lag1 + Lag2, data = Smarket, subset = train)

nb.fit
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
```

```
##
## A-priori probabilities:
## Y
##     Down       Up
## 0.491984 0.508016
##
## Conditional probabilities:
##       Lag1
## Y              [,1]     [,2]
##   Down  0.04279022 1.227446
##   Up   -0.03954635 1.231668
##
##       Lag2
## Y              [,1]     [,2]
##   Down  0.03389409 1.239191
##   Up   -0.03132544 1.220765
```

```r
nb.class <- predict(nb.fit, Smarket.2005)
table(nb.class, Direction.2005)
```

```
##         Direction.2005
## nb.class Down  Up
##     Down   28  20
##     Up     83 121
```

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

```
## [1] 0.5912698
```

```r
#can also generate the probabilities for each prediction
nb.pred <- predict(nb.fit, Smarket.2005, type = "raw")
nb.pred[1:5,]
```

```
##           Down        Up
## [1,] 0.4873164 0.5126836
## [2,] 0.4762492 0.5237508
## [3,] 0.4653377 0.5346623
## [4,] 0.4748652 0.5251348
## [5,] 0.4901890 0.5098110
```

K-Nearest Neighbours

```r
#Performed by using the knn() function

#Need a matrix composed of predictors from training data (train.x)
#A matrix with predictors from data we are trying to predict (test.xt)
#A vector with class labels for train.x (train.Direction)
#K, the number of NN to use in classifier
library(class)
library(ISLR2)
```

```r
#Setup test data
attach(Smarket)
```

```
## The following objects are masked from Smarket (pos = 6):
##
##     Direction, Lag1, Lag2, Lag3, Lag4, Lag5, Today, Volume, Year
```

```r
train <- ( Year < 2005)
train.x <- cbind(Lag1,Lag2)[train,]
test.x <- cbind(Lag1,Lag2)[!train,]
train.Direction <- Direction[train]
Direction.2005 <- Direction[!train]
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
```

```r
#50% accuracy with k = 1, not great
mean(knn.pred == Direction.2005)
```

```
## [1] 0.5
```

```r
#K = 3
knn.pred2 <- knn(train.x,test.x, train.Direction, k = 3)
table(knn.pred2,Direction.2005)
```

```
##          Direction.2005
## knn.pred2 Down Up
##      Down   48 54
##      Up     63 87
```

```r
mean(knn.pred2 == Direction.2005)
```

```
## [1] 0.5357143
```

```r
##Overall the QDA model had the highest accuracy on this data set
#could still change what parameters we use in the model later to see if this brings us above the 60%
```

KNN on the caravan data set

```r
dim(Caravan)
```

```
## [1] 5822   86
```

```r
attach(Caravan)
summary(Purchase)
```

```
##   No  Yes
## 5474  348
```

```r
#Scaling all the predictors to prevent any bias due to different magnitudes from affecting knn predicti
standardized.X <- scale(Caravan[,-86])

#Training subsets
test <- 1:1000
train.X1 <- standardized.X[-test,]
test.X1 <- standardized.X[test,]
train.Y1 <- Purchase[-test]
test.Y1 <- Purchase[test]

set.seed(1)
knn.pred3 <- knn(train.X1,test.X1,train.Y1, k = 1)
mean(test.Y1 != knn.pred3)
```

```
## [1] 0.118
```

```r
##The error rate looks good here, but because it is higher than the proportion of Yes in the training d
mean(test.Y1 != "No")
```

```
## [1] 0.059
```

```r
table(knn.pred3,test.Y1)
```

```
##          test.Y1
## knn.pred3  No Yes
##       No  873  50
##       Yes  68   9
```

```r
(9)/(66+9)
```

```
## [1] 0.12
```

```r
#12% success rate if you only sell insurance to people who are predicted to buy it
# double the odds of just asking everyone (6%)
knn.pred4 <- knn(train.X1,test.X1,train.Y1, k = 3)
table(knn.pred4,test.Y1)
```

```
##          test.Y1
## knn.pred4  No Yes
##       No  920  54
##       Yes  21   5
```

```r
5/24
```

```
## [1] 0.2083333
```

```r
#K = 3 improves this even more!
knn.pred5 <- knn(train.X1,test.X1,train.Y1, k = 5)
table(knn.pred5,test.Y1)
```

```
##          test.Y1
## knn.pred5  No Yes
##       No  930  55
##       Yes  11   4
```

```r
4/15
```

```
## [1] 0.2666667
```

```r
#Even higher! unfortunately this comes at the cost of only suggesting 15 candidate buyers...


##Trying with a logistic regression
glm.fits3 <- glm(Purchase ~ . , data = Caravan, subset = -test, family = binomial)
```

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

```r
glm.probs3 <- predict(glm.fits3,Caravan[test,],type = "response")
glm.pred3 <- rep("No",1000)
glm.pred3[glm.probs3 > 0.5] <- "Yes"
table(glm.pred3,test.Y1)
```

```
##          test.Y1
## glm.pred3  No Yes
##       No  934  59
##       Yes   7   0
```

```r
#All yes predictions are wrong!
glm.pred4 <- rep("No",1000)
glm.pred4[glm.probs3 > 0.25] <- "Yes"
table(glm.pred4,test.Y1)
```

```
##          test.Y1
## glm.pred4  No Yes
##       No  919  48
##       Yes  22  11
```

```r
#Now have a 33% success rate which is the best we have seen so far
```

Poisson Regression

```r
attach(Bikeshare)
dim(Bikeshare)
```

```
## [1] 8645   15
```

```r
names(Bikeshare)
```

```
##  [1] "season"     "mnth"       "day"        "hr"         "holiday"
##  [6] "weekday"    "workingday" "weathersit" "temp"       "atemp"
## [11] "hum"        "windspeed"  "casual"     "registered" "bikers"
```

```r
#Testing linear regression
mod.lm <- lm(bikers ~ mnth + hr + workingday + temp + weathersit, data = Bikeshare)
summary(mod.lm)
```

```
##
## Call:
## lm(formula = bikers ~ mnth + hr + workingday + temp + weathersit,
##     data = Bikeshare)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -299.00  -45.70   -6.23   41.08  425.29
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -68.632      5.307 -12.932  < 2e-16 ***
## mnthFeb                      6.845      4.287   1.597 0.110398
## mnthMarch                   16.551      4.301   3.848 0.000120 ***
## mnthApril                   41.425      4.972   8.331  < 2e-16 ***
## mnthMay                     72.557      5.641  12.862  < 2e-16 ***
## mnthJune                    67.819      6.544  10.364  < 2e-16 ***
## mnthJuly                    45.324      7.081   6.401 1.63e-10 ***
## mnthAug                     53.243      6.640   8.019 1.21e-15 ***
## mnthSept                    66.678      5.925  11.254  < 2e-16 ***
## mnthOct                     75.834      4.950  15.319  < 2e-16 ***
## mnthNov                     60.310      4.610  13.083  < 2e-16 ***
## mnthDec                     46.458      4.271  10.878  < 2e-16 ***
## hr1                        -14.579      5.699  -2.558 0.010536 *
## hr2                        -21.579      5.733  -3.764 0.000168 ***
## hr3                        -31.141      5.778  -5.389 7.26e-08 ***
## hr4                        -36.908      5.802  -6.361 2.11e-10 ***
## hr5                        -24.135      5.737  -4.207 2.61e-05 ***
## hr6                         20.600      5.704   3.612 0.000306 ***
## hr7                        120.093      5.693  21.095  < 2e-16 ***
## hr8                        223.662      5.690  39.310  < 2e-16 ***
## hr9                        120.582      5.693  21.182  < 2e-16 ***
## hr10                        83.801      5.705  14.689  < 2e-16 ***
## hr11                       105.423      5.722  18.424  < 2e-16 ***
## hr12                       137.284      5.740  23.916  < 2e-16 ***
## hr13                       136.036      5.760  23.617  < 2e-16 ***
```

```
## hr14                         126.636      5.776  21.923  < 2e-16 ***
## hr15                         132.087      5.780  22.852  < 2e-16 ***
## hr16                         178.521      5.772  30.927  < 2e-16 ***
## hr17                         296.267      5.749  51.537  < 2e-16 ***
## hr18                         269.441      5.736  46.976  < 2e-16 ***
## hr19                         186.256      5.714  32.596  < 2e-16 ***
## hr20                         125.549      5.704  22.012  < 2e-16 ***
## hr21                          87.554      5.693  15.378  < 2e-16 ***
## hr22                          59.123      5.689  10.392  < 2e-16 ***
## hr23                          26.838      5.688   4.719 2.41e-06 ***
## workingday                     1.270      1.784   0.711 0.476810
## temp                         157.209     10.261  15.321  < 2e-16 ***
## weathersitcloudy/misty       -12.890      1.964  -6.562 5.60e-11 ***
## weathersitlight rain/snow   -66.494      2.965 -22.425  < 2e-16 ***
## weathersitheavy rain/snow -109.745     76.667  -1.431 0.152341
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 76.5 on 8605 degrees of freedom
## Multiple R-squared:  0.6745, Adjusted R-squared:  0.6731
## F-statistic: 457.3 on 39 and 8605 DF,  p-value: < 2.2e-16
```

```r
#Changing data values for mnth and hr
contrasts(Bikeshare$hr) <- contr.sum(24)
contrasts(Bikeshare$mnth) <- contr.sum(12)
mod.lm2 <- lm(bikers ~ mnth + hr + workingday + temp + weathersit, data = Bikeshare)
summary(mod.lm2)
```

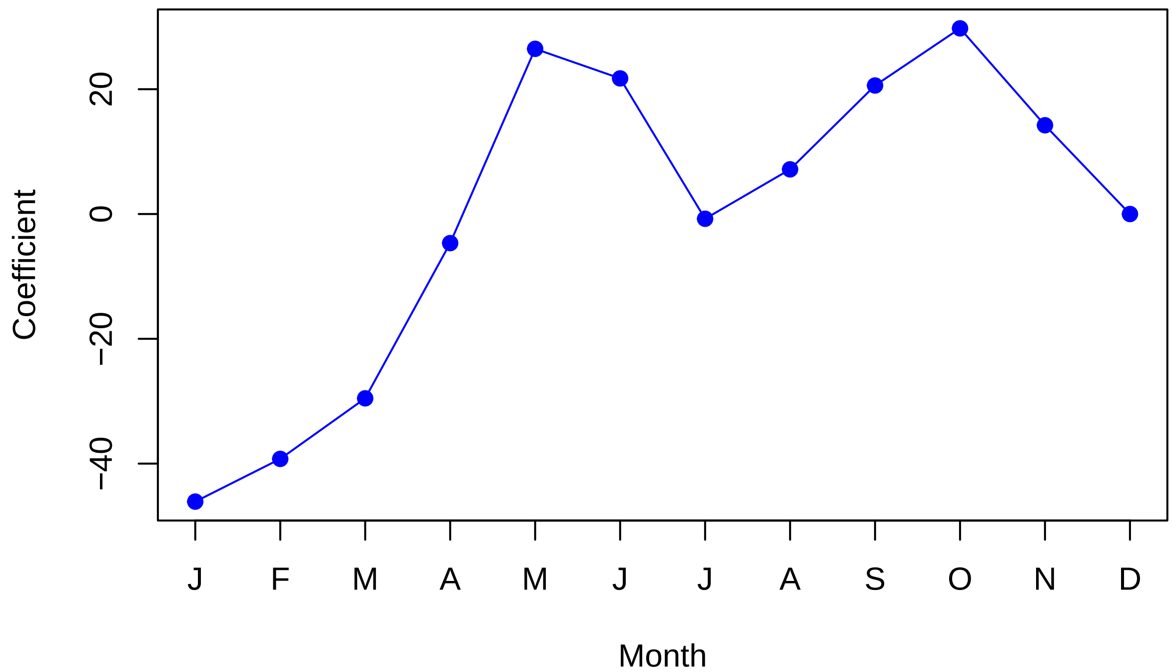
```
##
## Call:
## lm(formula = bikers ~ mnth + hr + workingday + temp + weathersit,
##     data = Bikeshare)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -299.00  -45.70   -6.23   41.08  425.29
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 73.5974     5.1322  14.340  < 2e-16 ***
## mnth1                      -46.0871     4.0855 -11.281  < 2e-16 ***
## mnth2                      -39.2419     3.5391 -11.088  < 2e-16 ***
## mnth3                      -29.5357     3.1552  -9.361  < 2e-16 ***
## mnth4                       -4.6622     2.7406  -1.701  0.08895 .
## mnth5                       26.4700     2.8508   9.285  < 2e-16 ***
## mnth6                       21.7317     3.4651   6.272 3.75e-10 ***
## mnth7                       -0.7626     3.9084  -0.195  0.84530
## mnth8                        7.1560     3.5347   2.024  0.04295 *
## mnth9                       20.5912     3.0456   6.761 1.46e-11 ***
## mnth10                      29.7472     2.6995  11.019  < 2e-16 ***
## mnth11                      14.2229     2.8604   4.972 6.74e-07 ***
## hr1                        -96.1420     3.9554 -24.307  < 2e-16 ***
## hr2                       -110.7213     3.9662 -27.916  < 2e-16 ***
## hr3                       -117.7212     4.0165 -29.310  < 2e-16 ***
```

```
## hr4                        -127.2828     4.0808 -31.191  < 2e-16 ***
## hr5                        -133.0495     4.1168 -32.319  < 2e-16 ***
## hr6                        -120.2775     4.0370 -29.794  < 2e-16 ***
## hr7                         -75.5424     3.9916 -18.925  < 2e-16 ***
## hr8                          23.9511     3.9686   6.035 1.65e-09 ***
## hr9                         127.5199     3.9500  32.284  < 2e-16 ***
## hr10                         24.4399     3.9360   6.209 5.57e-10 ***
## hr11                        -12.3407     3.9361  -3.135  0.00172 **
## hr12                          9.2814     3.9447   2.353  0.01865 *
## hr13                         41.1417     3.9571  10.397  < 2e-16 ***
## hr14                         39.8939     3.9750  10.036  < 2e-16 ***
## hr15                         30.4940     3.9910   7.641 2.39e-14 ***
## hr16                         35.9445     3.9949   8.998  < 2e-16 ***
## hr17                         82.3786     3.9883  20.655  < 2e-16 ***
## hr18                        200.1249     3.9638  50.488  < 2e-16 ***
## hr19                        173.2989     3.9561  43.806  < 2e-16 ***
## hr20                         90.1138     3.9400  22.872  < 2e-16 ***
## hr21                         29.4071     3.9362   7.471 8.74e-14 ***
## hr22                         -8.5883     3.9332  -2.184  0.02902 *
## hr23                        -37.0194     3.9344  -9.409  < 2e-16 ***
## workingday                    1.2696     1.7845   0.711  0.47681
## temp                        157.2094    10.2612  15.321  < 2e-16 ***
## weathersitcloudy/misty      -12.8903     1.9643  -6.562 5.60e-11 ***
## weathersitlight rain/snow  -66.4944     2.9652 -22.425  < 2e-16 ***
## weathersitheavy rain/snow -109.7446    76.6674  -1.431  0.15234
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 76.5 on 8605 degrees of freedom
## Multiple R-squared:  0.6745, Adjusted R-squared:  0.6731
## F-statistic: 457.3 on 39 and 8605 DF,  p-value: < 2.2e-16
```

```r
## Showing that both coding approaches do not change the model's predictions
all.equal(predict(mod.lm), predict(mod.lm2))
```

```
## [1] TRUE
```

```r
## Get month coefficients
coef.months <- c(coef(mod.lm2)[2:12], -sum(coef(mod.lm2[2:12])))
plot(coef.months, xlab = "Month", ylab = "Coefficient", xaxt = "n", col = "blue", pch = 19, type = "o")
axis(side = 1,at = 1:12, labels = c("J","F","M","A","M","J","J","A","S","O","N","D"))
```

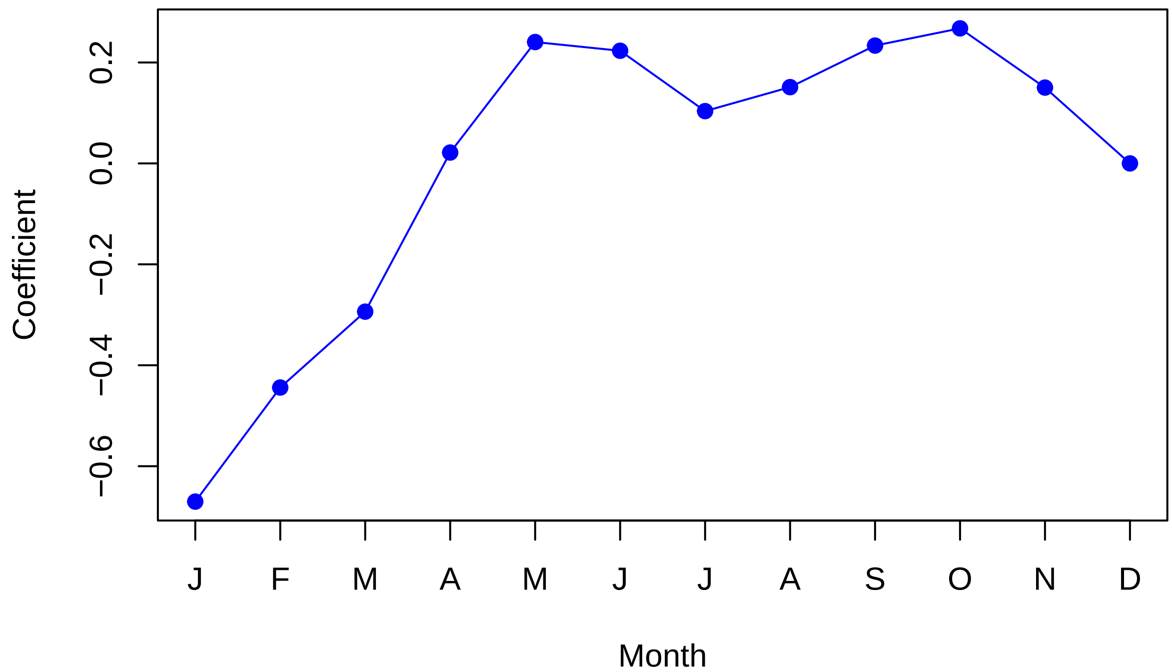



```
## now fitting a poisson regression on the data instead

mod.pois <- glm(bikers ~ mnth + hr + workingday + temp + weathersit, data = Bikeshare, family = poisson)
summary(mod.pois)
```

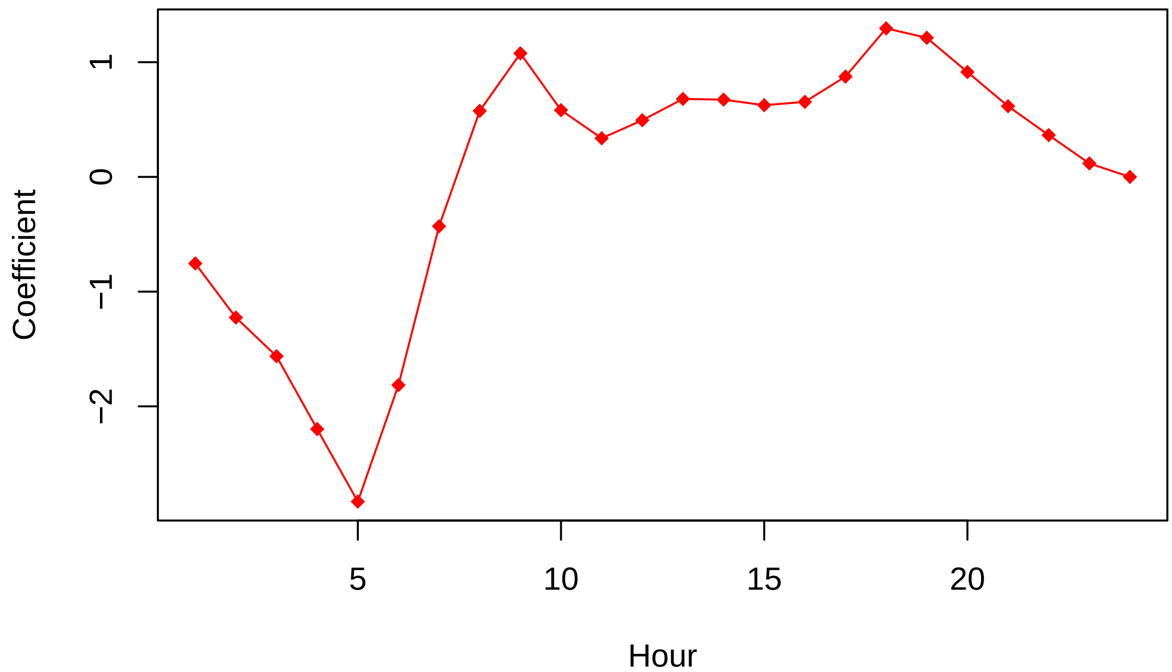
```
##
## Call:
## glm(formula = bikers ~ mnth + hr + workingday + temp + weathersit,
##     family = poisson, data = Bikeshare)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -20.7574   -3.3441   -0.6549    2.6999   21.9628
##
## Coefficients:
##                            Estimate Std. Error  z value Pr(>|z|)
## (Intercept)                4.118245   0.006021  683.964  < 2e-16 ***
## mnth1                     -0.670170   0.005907 -113.445  < 2e-16 ***
## mnth2                     -0.444124   0.004860  -91.379  < 2e-16 ***
## mnth3                     -0.293733   0.004144  -70.886  < 2e-16 ***
## mnth4                      0.021523   0.003125    6.888 5.66e-12 ***
## mnth5                      0.240471   0.002916   82.462  < 2e-16 ***
## mnth6                      0.223235   0.003554   62.818  < 2e-16 ***
## mnth7                      0.103617   0.004125   25.121  < 2e-16 ***
## mnth8                      0.151171   0.003662   41.281  < 2e-16 ***
## mnth9                      0.233493   0.003102   75.281  < 2e-16 ***
```

```
## mnth10                      0.267573   0.002785   96.091  < 2e-16 ***
## mnth11                      0.150264   0.003180   47.248  < 2e-16 ***
## hr1                        -0.754386   0.007879  -95.744  < 2e-16 ***
## hr2                        -1.225979   0.009953 -123.173  < 2e-16 ***
## hr3                        -1.563147   0.011869 -131.702  < 2e-16 ***
## hr4                        -2.198304   0.016424 -133.846  < 2e-16 ***
## hr5                        -2.830484   0.022538 -125.586  < 2e-16 ***
## hr6                        -1.814657   0.013464 -134.775  < 2e-16 ***
## hr7                        -0.429888   0.006896  -62.341  < 2e-16 ***
## hr8                         0.575181   0.004406  130.544  < 2e-16 ***
## hr9                         1.076927   0.003563  302.220  < 2e-16 ***
## hr10                        0.581769   0.004286  135.727  < 2e-16 ***
## hr11                        0.336852   0.004720   71.372  < 2e-16 ***
## hr12                        0.494121   0.004392  112.494  < 2e-16 ***
## hr13                        0.679642   0.004069  167.040  < 2e-16 ***
## hr14                        0.673565   0.004089  164.722  < 2e-16 ***
## hr15                        0.624910   0.004178  149.570  < 2e-16 ***
## hr16                        0.653763   0.004132  158.205  < 2e-16 ***
## hr17                        0.874301   0.003784  231.040  < 2e-16 ***
## hr18                        1.294635   0.003254  397.848  < 2e-16 ***
## hr19                        1.212281   0.003321  365.084  < 2e-16 ***
## hr20                        0.914022   0.003700  247.065  < 2e-16 ***
## hr21                        0.616201   0.004191  147.045  < 2e-16 ***
## hr22                        0.364181   0.004659   78.173  < 2e-16 ***
## hr23                        0.117493   0.005225   22.488  < 2e-16 ***
## workingday                  0.014665   0.001955    7.502 6.27e-14 ***
## temp                        0.785292   0.011475   68.434  < 2e-16 ***
## weathersitcloudy/misty     -0.075231   0.002179  -34.528  < 2e-16 ***
## weathersitlight rain/snow -0.575800   0.004058 -141.905  < 2e-16 ***
## weathersitheavy rain/snow -0.926287   0.166782   -5.554 2.79e-08 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1052921  on 8644  degrees of freedom
## Residual deviance:  228041  on 8605  degrees of freedom
## AIC: 281159
##
## Number of Fisher Scoring iterations: 5
```

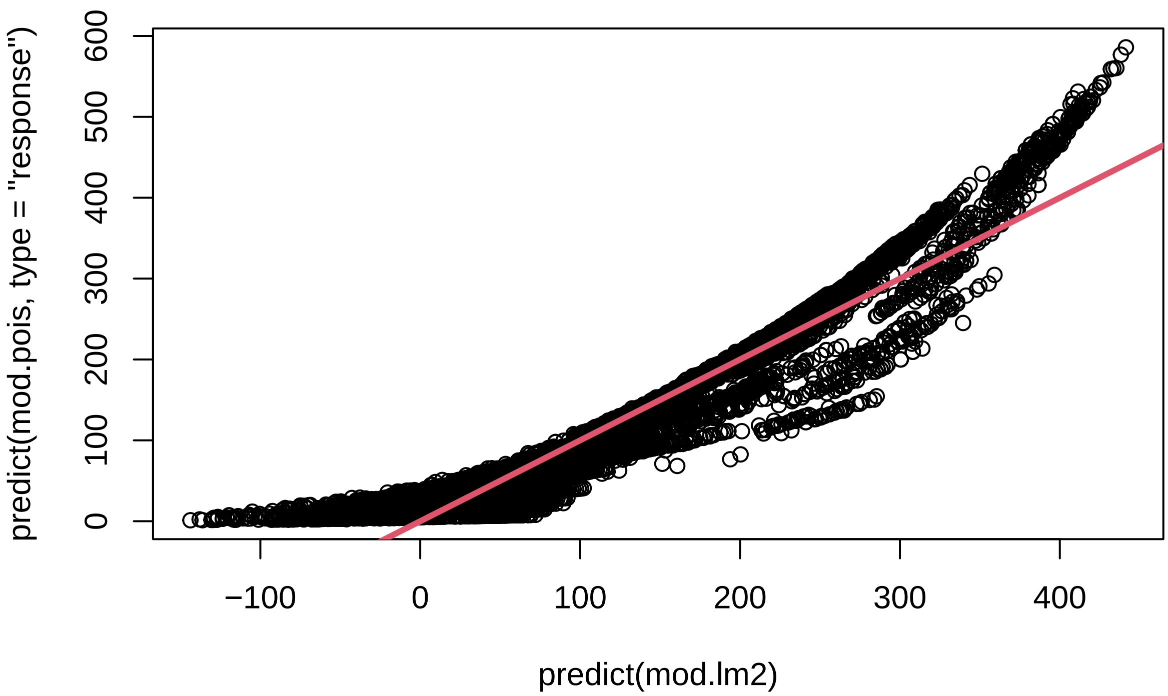
```
coef.months2 <- c(coef(mod.pois)[2:12], -sum(coef(mod.pois[2:12])))
plot(coef.months2, xlab = "Month", ylab = "Coefficient", xaxt = "n", col = "blue", pch = 19, type = "o"
axis(side = 1,at = 1:12, labels = c("J","F","M","A","M","J","J","A","S","O","N","D"))
```




```
coef.hr <- c(coef(mod.pois)[13:35], -sum(coef(mod.pois[13:35])))
plot(coef.hr, xlab = "Hour", ylab = "Coefficient", col = "red", pch = 18, type = "o")
```




```
plot(predict(mod.lm2),predict(mod.pois, type = "response"))
abline(0,1,col = 2, lwd = 3)
```



Lab is complete