DS\_HW4

RE6124019\_吳明軒

2023-11-23

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

## 3.6 Lab: Linear Regression

library(MASS)  
library(ISLR2)

## Warning: package 'ISLR2' was built under R version 4.3.2

##   
## Attaching package: 'ISLR2'

## The following object is masked from 'package:MASS':  
##   
## Boston

head (Boston)

## crim zn indus chas nox rm age dis rad tax ptratio lstat medv  
## 1 0.00632 18 2.31 0 0.538 6.575 65.2 4.0900 1 296 15.3 4.98 24.0  
## 2 0.02731 0 7.07 0 0.469 6.421 78.9 4.9671 2 242 17.8 9.14 21.6  
## 3 0.02729 0 7.07 0 0.469 7.185 61.1 4.9671 2 242 17.8 4.03 34.7  
## 4 0.03237 0 2.18 0 0.458 6.998 45.8 6.0622 3 222 18.7 2.94 33.4  
## 5 0.06905 0 2.18 0 0.458 7.147 54.2 6.0622 3 222 18.7 5.33 36.2  
## 6 0.02985 0 2.18 0 0.458 6.430 58.7 6.0622 3 222 18.7 5.21 28.7

lm.fit <- lm( medv ~ lstat , data = Boston )  
attach ( Boston )  
lm.fit <- lm(medv ~ lstat)  
  
lm.fit

##   
## Call:  
## lm(formula = medv ~ lstat)  
##   
## Coefficients:  
## (Intercept) lstat   
## 34.55 -0.95

summary(lm.fit)

##   
## Call:  
## lm(formula = medv ~ lstat)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.168 -3.990 -1.318 2.034 24.500   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 34.55384 0.56263 61.41 <2e-16 \*\*\*  
## lstat -0.95005 0.03873 -24.53 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.216 on 504 degrees of freedom  
## Multiple R-squared: 0.5441, Adjusted R-squared: 0.5432   
## F-statistic: 601.6 on 1 and 504 DF, p-value: < 2.2e-16

names(lm.fit)

## [1] "coefficients" "residuals" "effects" "rank"   
## [5] "fitted.values" "assign" "qr" "df.residual"   
## [9] "xlevels" "call" "terms" "model"

coef(lm.fit)

## (Intercept) lstat   
## 34.5538409 -0.9500494

confint(lm.fit)

## 2.5 % 97.5 %  
## (Intercept) 33.448457 35.6592247  
## lstat -1.026148 -0.8739505

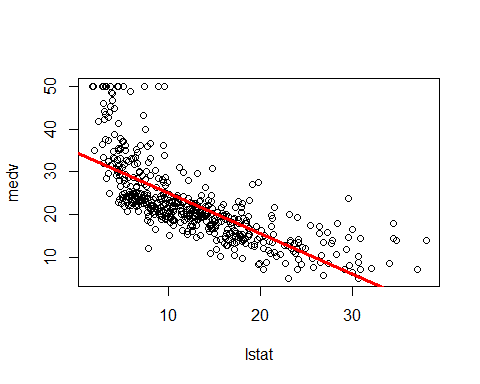
predict(lm.fit, data.frame(lstat = c(5, 10, 15)), interval = "confidence")

## fit lwr upr  
## 1 29.80359 29.00741 30.59978  
## 2 25.05335 24.47413 25.63256  
## 3 20.30310 19.73159 20.87461

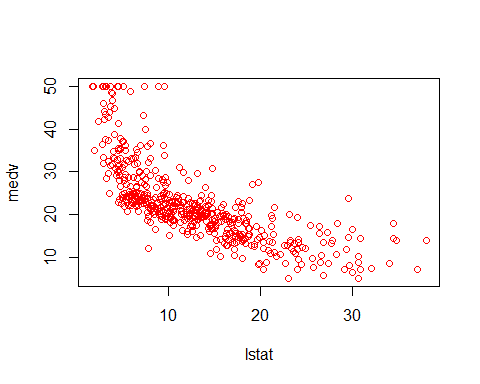
predict(lm.fit, data.frame(lstat = c(5, 10, 15)),interval = "prediction")

## fit lwr upr  
## 1 29.80359 17.565675 42.04151  
## 2 25.05335 12.827626 37.27907  
## 3 20.30310 8.077742 32.52846

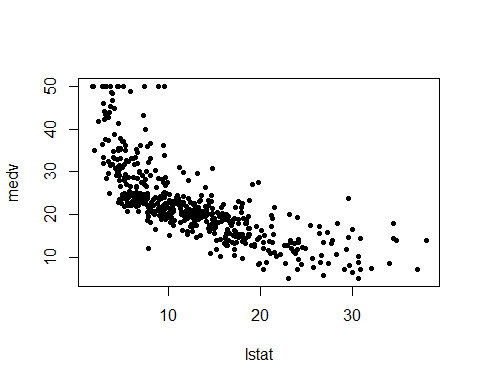
plot(lstat, medv)  
abline(lm.fit)  
  
abline(lm.fit, lwd = 3)  
abline(lm.fit, lwd = 3, col = "red")



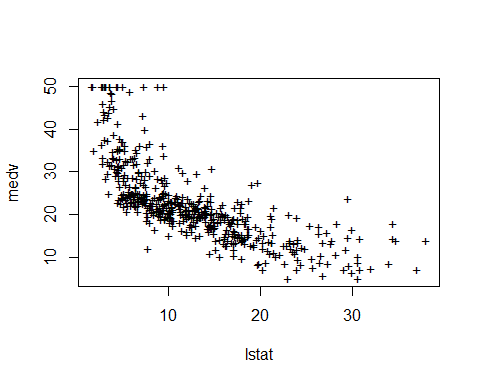
plot(lstat, medv, col = "red")



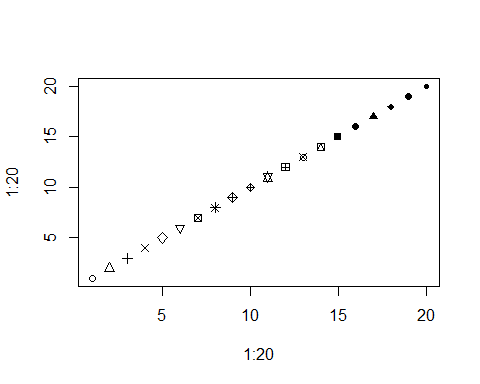
plot(lstat, medv, pch = 20)



plot(lstat, medv, pch = "+")



plot(1:20, 1:20, pch = 1:20)



par(mfrow = c(2, 2))  
#plot(lm.fit)  
  
plot(predict(lm.fit), residuals(lm.fit))  
plot(predict(lm.fit), rstudent(lm.fit))  
  
plot(hatvalues(lm.fit))  
which.max(hatvalues(lm.fit))

## 375   
## 375

lm.fit <- lm(medv ~ lstat + age, data = Boston)   
summary(lm.fit)

##   
## Call:  
## lm(formula = medv ~ lstat + age, data = Boston)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.981 -3.978 -1.283 1.968 23.158   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 33.22276 0.73085 45.458 < 2e-16 \*\*\*  
## lstat -1.03207 0.04819 -21.416 < 2e-16 \*\*\*  
## age 0.03454 0.01223 2.826 0.00491 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.173 on 503 degrees of freedom  
## Multiple R-squared: 0.5513, Adjusted R-squared: 0.5495   
## F-statistic: 309 on 2 and 503 DF, p-value: < 2.2e-16

lm.fit <- lm(medv ~ ., data = Boston)  
summary(lm.fit)

##   
## Call:  
## lm(formula = medv ~ ., data = Boston)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.1304 -2.7673 -0.5814 1.9414 26.2526   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 41.617270 4.936039 8.431 3.79e-16 \*\*\*  
## crim -0.121389 0.033000 -3.678 0.000261 \*\*\*  
## zn 0.046963 0.013879 3.384 0.000772 \*\*\*  
## indus 0.013468 0.062145 0.217 0.828520   
## chas 2.839993 0.870007 3.264 0.001173 \*\*   
## nox -18.758022 3.851355 -4.870 1.50e-06 \*\*\*  
## rm 3.658119 0.420246 8.705 < 2e-16 \*\*\*  
## age 0.003611 0.013329 0.271 0.786595   
## dis -1.490754 0.201623 -7.394 6.17e-13 \*\*\*  
## rad 0.289405 0.066908 4.325 1.84e-05 \*\*\*  
## tax -0.012682 0.003801 -3.337 0.000912 \*\*\*  
## ptratio -0.937533 0.132206 -7.091 4.63e-12 \*\*\*  
## lstat -0.552019 0.050659 -10.897 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.798 on 493 degrees of freedom  
## Multiple R-squared: 0.7343, Adjusted R-squared: 0.7278   
## F-statistic: 113.5 on 12 and 493 DF, p-value: < 2.2e-16

library(car)

## Warning: package 'car' was built under R version 4.3.2

## Loading required package: carData

## Warning: package 'carData' was built under R version 4.3.2

vif(lm.fit)

## crim zn indus chas nox rm age dis   
## 1.767486 2.298459 3.987181 1.071168 4.369093 1.912532 3.088232 3.954037   
## rad tax ptratio lstat   
## 7.445301 9.002158 1.797060 2.870777

lm.fit1 <- lm(medv ~ . - age, data = Boston)  
summary (lm.fit1)

##   
## Call:  
## lm(formula = medv ~ . - age, data = Boston)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.1851 -2.7330 -0.6116 1.8555 26.3838   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 41.525128 4.919684 8.441 3.52e-16 \*\*\*  
## crim -0.121426 0.032969 -3.683 0.000256 \*\*\*  
## zn 0.046512 0.013766 3.379 0.000785 \*\*\*  
## indus 0.013451 0.062086 0.217 0.828577   
## chas 2.852773 0.867912 3.287 0.001085 \*\*   
## nox -18.485070 3.713714 -4.978 8.91e-07 \*\*\*  
## rm 3.681070 0.411230 8.951 < 2e-16 \*\*\*  
## dis -1.506777 0.192570 -7.825 3.12e-14 \*\*\*  
## rad 0.287940 0.066627 4.322 1.87e-05 \*\*\*  
## tax -0.012653 0.003796 -3.333 0.000923 \*\*\*  
## ptratio -0.934649 0.131653 -7.099 4.39e-12 \*\*\*  
## lstat -0.547409 0.047669 -11.483 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.794 on 494 degrees of freedom  
## Multiple R-squared: 0.7343, Adjusted R-squared: 0.7284   
## F-statistic: 124.1 on 11 and 494 DF, p-value: < 2.2e-16

lm.fit1 <- update ( lm.fit , ~ . - age )  
  
summary(lm(medv ~ lstat \* age, data = Boston))

##   
## Call:  
## lm(formula = medv ~ lstat \* age, data = Boston)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.806 -4.045 -1.333 2.085 27.552   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 36.0885359 1.4698355 24.553 < 2e-16 \*\*\*  
## lstat -1.3921168 0.1674555 -8.313 8.78e-16 \*\*\*  
## age -0.0007209 0.0198792 -0.036 0.9711   
## lstat:age 0.0041560 0.0018518 2.244 0.0252 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.149 on 502 degrees of freedom  
## Multiple R-squared: 0.5557, Adjusted R-squared: 0.5531   
## F-statistic: 209.3 on 3 and 502 DF, p-value: < 2.2e-16

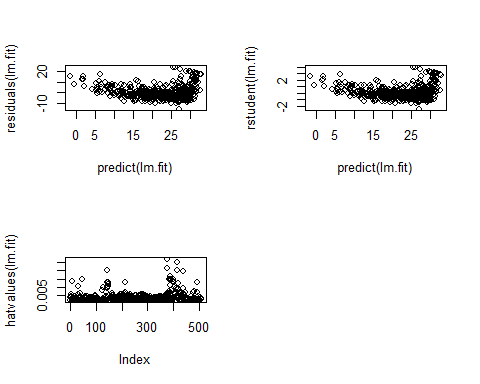
lm.fit2 <- lm(medv ~ lstat + I(lstat^2))   
summary(lm.fit2)

##   
## Call:  
## lm(formula = medv ~ lstat + I(lstat^2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.2834 -3.8313 -0.5295 2.3095 25.4148   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 42.862007 0.872084 49.15 <2e-16 \*\*\*  
## lstat -2.332821 0.123803 -18.84 <2e-16 \*\*\*  
## I(lstat^2) 0.043547 0.003745 11.63 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5.524 on 503 degrees of freedom  
## Multiple R-squared: 0.6407, Adjusted R-squared: 0.6393   
## F-statistic: 448.5 on 2 and 503 DF, p-value: < 2.2e-16

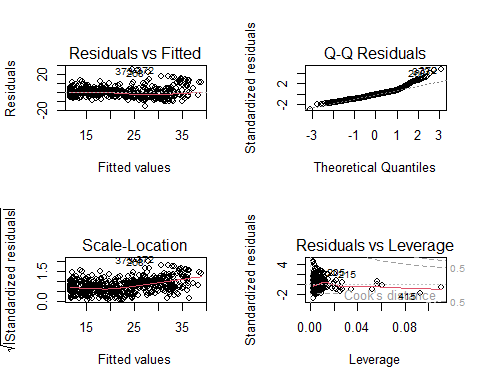
lm.fit <- lm(medv ~ lstat)  
anova(lm.fit, lm.fit2)

## Analysis of Variance Table  
##   
## Model 1: medv ~ lstat  
## Model 2: medv ~ lstat + I(lstat^2)  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 504 19472   
## 2 503 15347 1 4125.1 135.2 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

par(mfrow = c(2 , 2))



plot(lm.fit2)



lm.fit5 <- lm(medv ~ poly(lstat, 5))  
summary(lm.fit5)

##   
## Call:  
## lm(formula = medv ~ poly(lstat, 5))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -13.5433 -3.1039 -0.7052 2.0844 27.1153   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 22.5328 0.2318 97.197 < 2e-16 \*\*\*  
## poly(lstat, 5)1 -152.4595 5.2148 -29.236 < 2e-16 \*\*\*  
## poly(lstat, 5)2 64.2272 5.2148 12.316 < 2e-16 \*\*\*  
## poly(lstat, 5)3 -27.0511 5.2148 -5.187 3.10e-07 \*\*\*  
## poly(lstat, 5)4 25.4517 5.2148 4.881 1.42e-06 \*\*\*  
## poly(lstat, 5)5 -19.2524 5.2148 -3.692 0.000247 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5.215 on 500 degrees of freedom  
## Multiple R-squared: 0.6817, Adjusted R-squared: 0.6785   
## F-statistic: 214.2 on 5 and 500 DF, p-value: < 2.2e-16

summary(lm(medv ~ log(rm), data = Boston))

##   
## Call:  
## lm(formula = medv ~ log(rm), data = Boston)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -19.487 -2.875 -0.104 2.837 39.816   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -76.488 5.028 -15.21 <2e-16 \*\*\*  
## log(rm) 54.055 2.739 19.73 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.915 on 504 degrees of freedom  
## Multiple R-squared: 0.4358, Adjusted R-squared: 0.4347   
## F-statistic: 389.3 on 1 and 504 DF, p-value: < 2.2e-16

head(Carseats)

## Sales CompPrice Income Advertising Population Price ShelveLoc Age Education  
## 1 9.50 138 73 11 276 120 Bad 42 17  
## 2 11.22 111 48 16 260 83 Good 65 10  
## 3 10.06 113 35 10 269 80 Medium 59 12  
## 4 7.40 117 100 4 466 97 Medium 55 14  
## 5 4.15 141 64 3 340 128 Bad 38 13  
## 6 10.81 124 113 13 501 72 Bad 78 16  
## Urban US  
## 1 Yes Yes  
## 2 Yes Yes  
## 3 Yes Yes  
## 4 Yes Yes  
## 5 Yes No  
## 6 No Yes

lm.fit <- lm(Sales ~ . + Income:Advertising + Price:Age, data = Carseats)  
summary(lm.fit)

##   
## Call:  
## lm(formula = Sales ~ . + Income:Advertising + Price:Age, data = Carseats)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.9208 -0.7503 0.0177 0.6754 3.3413   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.5755654 1.0087470 6.519 2.22e-10 \*\*\*  
## CompPrice 0.0929371 0.0041183 22.567 < 2e-16 \*\*\*  
## Income 0.0108940 0.0026044 4.183 3.57e-05 \*\*\*  
## Advertising 0.0702462 0.0226091 3.107 0.002030 \*\*   
## Population 0.0001592 0.0003679 0.433 0.665330   
## Price -0.1008064 0.0074399 -13.549 < 2e-16 \*\*\*  
## ShelveLocGood 4.8486762 0.1528378 31.724 < 2e-16 \*\*\*  
## ShelveLocMedium 1.9532620 0.1257682 15.531 < 2e-16 \*\*\*  
## Age -0.0579466 0.0159506 -3.633 0.000318 \*\*\*  
## Education -0.0208525 0.0196131 -1.063 0.288361   
## UrbanYes 0.1401597 0.1124019 1.247 0.213171   
## USYes -0.1575571 0.1489234 -1.058 0.290729   
## Income:Advertising 0.0007510 0.0002784 2.698 0.007290 \*\*   
## Price:Age 0.0001068 0.0001333 0.801 0.423812   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.011 on 386 degrees of freedom  
## Multiple R-squared: 0.8761, Adjusted R-squared: 0.8719   
## F-statistic: 210 on 13 and 386 DF, p-value: < 2.2e-16

attach(Carseats)  
contrasts(ShelveLoc)

## Good Medium  
## Bad 0 0  
## Good 1 0  
## Medium 0 1

#LoadLibraries  
#LoadLibraries()  
  
LoadLibraries <- function() {  
 library(ISLR2)  
 library(MASS)   
 print("The libraries have been loaded.")  
}  
LoadLibraries

## function() {  
## library(ISLR2)  
## library(MASS)   
## print("The libraries have been loaded.")  
## }

LoadLibraries()

## [1] "The libraries have been loaded."

## 4.7 Lab: Classification Methods

library(ISLR2)  
names(Smarket)

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

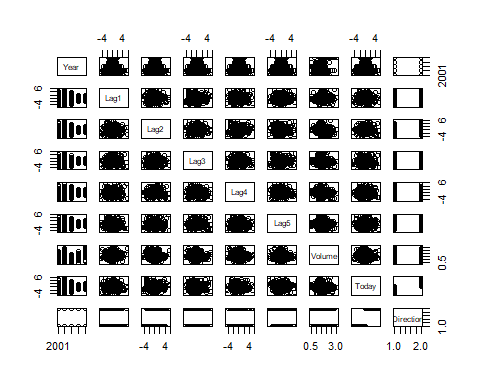
dim(Smarket)

## [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   
##   
##   
##   
##

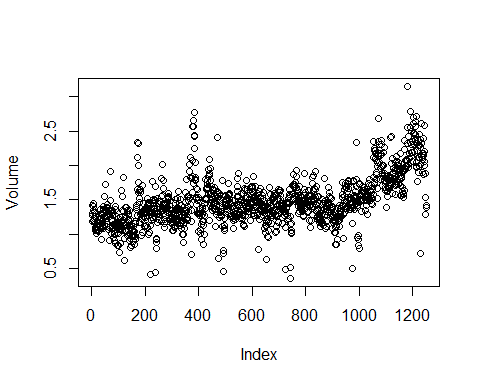
pairs(Smarket)



#cor(Smarket)  
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

attach(Smarket)  
plot(Volume)



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)  
##   
## 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

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

summary(glm.fits)$coef[, 4]

## (Intercept) Lag1 Lag2 Lag3 Lag4 Lag5   
## 0.6006983 0.1452272 0.3983491 0.8243333 0.8514445 0.8349974   
## Volume   
## 0.3924004

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

contrasts(Direction)

## Up  
## Down 0  
## Up 1

glm.pred <- rep("Down", 1250)  
glm.pred[glm.probs > .5] <- "Up"  
  
table(glm.pred, Direction)

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

mean(glm.pred == Direction)

## [1] 0.5216

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

## [1] 252 9

Direction.2005 <- Direction[!train]  
  
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")  
  
glm.pred <- rep("Down", 252)  
glm.pred[glm.probs > .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

mean(glm.pred != Direction.2005)

## [1] 0.5198413

glm.fits <- glm(Direction ~ Lag1 + Lag2 , data = Smarket,  
 family = binomial, subset = train)  
glm.probs <- predict(glm.fits, Smarket.2005, type = "response")  
glm.pred <- rep("Down", 252)  
glm.pred[glm.probs > .5] <- "Up"  
glm.pred[glm.probs > .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.frame(Lag1 = c(1.2 , 1.5) , Lag2 = c(1.1 ,-0.8)),  
 type = "response"  
)

## 1 2   
## 0.4791462 0.4960939

library(MASS)  
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

sum(lda.pred$posterior[, 1] >= .5)

## [1] 70

sum(lda.pred$posterior[, 1] < .5)

## [1] 182

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

## 999 1000 1001 1002 1003 1004 1005 1006   
## 0.4901792 0.4792185 0.4668185 0.4740011 0.4927877 0.4938562 0.4951016 0.4872861   
## 1007 1008 1009 1010 1011 1012 1013 1014   
## 0.4907013 0.4844026 0.4906963 0.5119988 0.4895152 0.4706761 0.4744593 0.4799583   
## 1015 1016 1017 1018   
## 0.4935775 0.5030894 0.4978806 0.4886331

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

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

## [1] 0

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

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

library(e1071)  
nb.fit <- naiveBayes(Direction ~ Lag1 + Lag2, data = Smarket, subset = train)  
nb.class <- predict(nb.fit, Smarket.2005)   
table(nb.class, Direction.2005)

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

mean(nb.class == Direction.2005)

## [1] 0.5912698

mean(Lag1[train][Direction[train] == "Down"])

## [1] 0.04279022

sd(Lag1[train][Direction[train] == "Down"])

## [1] 1.227446

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

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

mean(nb.class == Direction.2005)

## [1] 0.5912698

nb.preds <- predict(nb.fit ,Smarket.2005 , type = "raw")  
nb.preds[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

library(class)  
train.X <- cbind(Lag1, Lag2)[train,]  
test.X <- cbind(Lag1, Lag2)[!train,]  
train.Direction <- 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

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

## Direction.2005  
## knn.pred Down Up  
## Down 48 54  
## Up 63 87

mean(knn.pred == Direction.2005)

## [1] 0.5357143

dim(Caravan)

## [1] 5822 86

attach(Caravan)  
summary(Purchase)

## No Yes   
## 5474 348

standardized.X <- scale(Caravan[, -86])  
var(Caravan[, 1])

## [1] 165.0378

var(Caravan[, 2])

## [1] 0.1647078

var(standardized.X[, 1])

## [1] 1

var(standardized.X[, 2])

## [1] 1

test <- 1:1000  
train.X <- standardized.X[-test,]   
test.X <- standardized.X[test,]  
train.Y <- Purchase[-test]  
  
test.Y <- Purchase[test]  
set.seed (1)  
knn.pred <- knn (train.X, test.X, train.Y, k = 1)  
mean(test.Y != knn.pred)

## [1] 0.118

mean(test.Y != "No")

## [1] 0.059

table(knn.pred ,test.Y)

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

9 / (68 + 9)

## [1] 0.1168831

knn.pred <- knn(train.X ,test.X ,train.Y ,k = 3)  
table(knn.pred ,test.Y)

## test.Y  
## knn.pred No Yes  
## No 920 54  
## Yes 21 5

5 / 26

## [1] 0.1923077

knn.pred <- knn(train.X ,test.X ,train.Y ,k = 5)  
table(knn.pred ,test.Y)

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

4 / 15

## [1] 0.2666667

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")  
  
glm.pred <- rep("No", 1000)  
glm.pred[glm.probs > 0.5] <- "Yes"  
table(glm.pred ,test.Y)

## test.Y  
## glm.pred No Yes  
## No 934 59  
## Yes 7 0

glm.pred <- rep("No", 1000)  
glm.pred[glm.probs > 0.25] <- "Yes"  
table(glm.pred ,test.Y)

## test.Y  
## glm.pred No Yes  
## No 919 48  
## Yes 22 11

11 / (22 + 11)

## [1] 0.3333333

attach(Bikeshare)  
dim(Bikeshare)

## [1] 8645 15

names(Bikeshare)

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

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

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

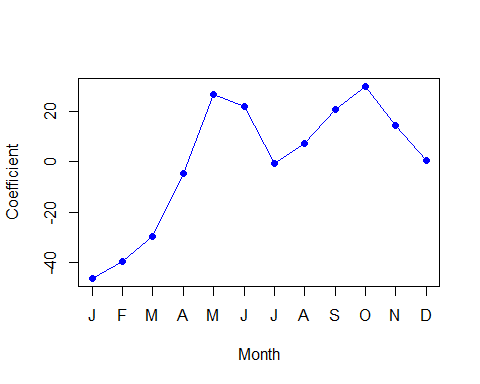
sum((predict(mod.lm) - predict(mod.lm2))^2)

## [1] 1.586608e-18

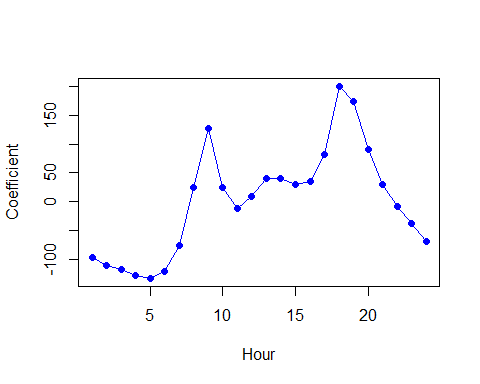
all.equal(predict(mod.lm), predict(mod.lm2))

## [1] TRUE

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"))



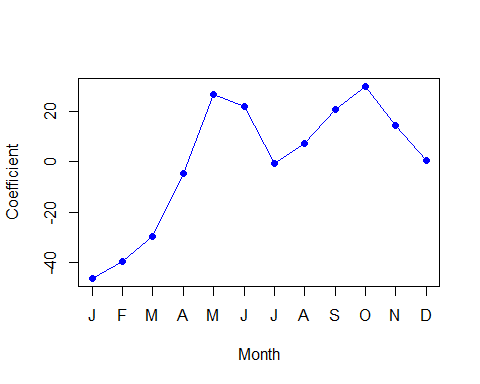
coef.hours <- c( coef ( mod.lm2 ) [13:35] ,  
 -sum ( coef ( mod.lm2 )[13:35]) )  
plot ( coef.hours , xlab = " Hour ", ylab = " Coefficient ",  
 col = " blue ", pch = 19 , type = "o")



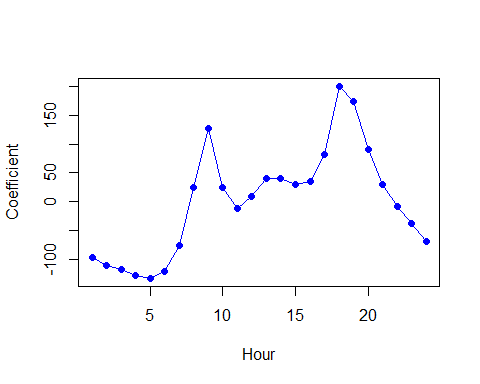
mod.pois <- glm (  
 bikers ~ mnth + hr + workingday + temp + weathersit ,  
 data = Bikeshare , family = poisson  
)  
summary ( mod.pois )

##   
## Call:  
## glm(formula = bikers ~ mnth + hr + workingday + temp + weathersit,   
## family = poisson, data = Bikeshare)  
##   
## 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

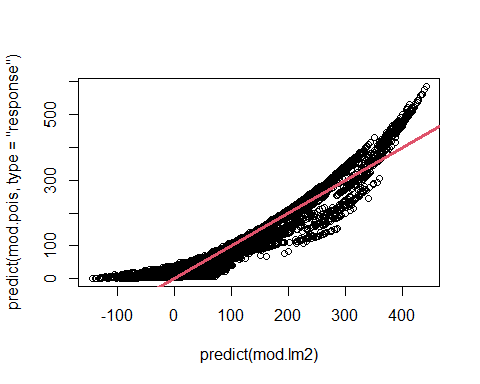
coef.months <- c(coef(mod.lm2)[2:12], -sum(coef(mod.lm2)[2:12]))  
plot(coef.months, xlab="Month", ylab="Coefficient",   
 xaxt="n", col="blue", pch=19, type="o")   
axis(side=1, at=1:12, labels=c("J", "F", "M", "A", "M", "J", "J", "A", "S", "O", "N", "D"))



coef.hours <- c(coef(mod.lm2)[13:35], -sum(coef(mod.lm2)[13:35]))   
plot(coef.hours, xlab = "Hour", ylab = "Coefficient",   
 col = "blue", pch = 19, type = "o")



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



## 4.7 Lab: Classification Methods

library(ISLR2)  
set.seed (1)  
train <- sample(392 , 196)  
  
lm.fit <- lm( mpg ~ horsepower , data = Auto , subset = train )  
attach ( Auto )  
mean (( mpg - predict ( lm.fit , Auto))[- train ]^2)

## [1] 23.26601

lm.fit2 <- lm( mpg ~ poly ( horsepower , 2) , data = Auto ,  
 subset = train )  
mean (( mpg - predict ( lm.fit2 , Auto ))[ - train ]^2)

## [1] 18.71646

lm.fit3 <- lm( mpg ~ poly ( horsepower , 3) , data = Auto ,  
 subset = train )  
mean (( mpg - predict ( lm.fit3 , Auto ))[ - train ]^2)

## [1] 18.79401

set.seed (2)  
train <- sample (392 , 196)  
lm.fit <- lm( mpg ~ horsepower , subset = train )  
mean (( mpg - predict ( lm.fit , Auto ))[- train ]^2)

## [1] 25.72651

lm.fit2 <- lm( mpg ~ poly ( horsepower , 2) , data = Auto ,  
 subset = train )  
mean (( mpg - predict ( lm.fit2 , Auto ))[ - train ]^2)

## [1] 20.43036

lm.fit3 <- lm( mpg ~ poly ( horsepower , 3) , data = Auto ,  
 subset = train )  
mean (( mpg - predict ( lm.fit3 , Auto ))[ - train ]^2)

## [1] 20.38533

glm.fit <- glm ( mpg ~ horsepower , data = Auto )  
coef ( glm.fit )

## (Intercept) horsepower   
## 39.9358610 -0.1578447

lm.fit <- lm( mpg ~ horsepower , data = Auto )  
coef ( lm.fit )

## (Intercept) horsepower   
## 39.9358610 -0.1578447

library ( boot )

##   
## Attaching package: 'boot'

## The following object is masked from 'package:car':  
##   
## logit

glm.fit <- glm ( mpg ~ horsepower , data = Auto )  
cv.err <- cv.glm ( Auto , glm.fit )  
cv.err$delta

## [1] 24.23151 24.23114

cv.error <- rep (0 , 10)  
for (i in 1:10) {  
 glm.fit <- glm ( mpg ~ poly ( horsepower , i) , data = Auto )  
 cv.error [ i] <- cv.glm ( Auto , glm.fit )$ delta [1]  
}  
cv.error

## [1] 24.23151 19.24821 19.33498 19.42443 19.03321 18.97864 18.83305 18.96115  
## [9] 19.06863 19.49093

set.seed (17)  
cv.error.10 <- rep (0 , 10)  
for (i in 1:10) {  
 glm.fit <- glm ( mpg ~ poly ( horsepower , i) , data = Auto )  
 cv.error.10[ i ] <- cv.glm ( Auto , glm.fit , K = 10) $ delta [1]  
}  
cv.error.10

## [1] 24.27207 19.26909 19.34805 19.29496 19.03198 18.89781 19.12061 19.14666  
## [9] 18.87013 20.95520

alpha.fn <- function ( data , index ) {  
 X <- data $X [ index ]  
 Y <- data $Y [ index ]  
 ( var (Y) - cov (X , Y) ) / ( var (X) + var (Y) - 2 \* cov (X , Y))  
}  
  
alpha.fn( Portfolio , 1:100)

## [1] 0.5758321

set.seed (7)  
alpha.fn( Portfolio , sample (100 , 100 , replace = T))

## [1] 0.5385326

boot ( Portfolio , alpha.fn , R = 1000)

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = Portfolio, statistic = alpha.fn, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 0.5758321 0.0007959475 0.08969074

boot.fn <- function ( data , index )  
 coef (lm( mpg ~ horsepower , data = data , subset = index ))  
boot.fn( Auto , 1:392)

## (Intercept) horsepower   
## 39.9358610 -0.1578447

set.seed (1)  
boot.fn( Auto , sample (392 , 392 , replace = T ))

## (Intercept) horsepower   
## 40.3404517 -0.1634868

boot ( Auto , boot.fn , 1000)

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = Auto, statistic = boot.fn, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 39.9358610 0.0549915227 0.841925746  
## t2\* -0.1578447 -0.0006210818 0.007348956

summary (lm( mpg ~ horsepower , data = Auto ) )$ coef

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 39.9358610 0.717498656 55.65984 1.220362e-187  
## horsepower -0.1578447 0.006445501 -24.48914 7.031989e-81

boot.fn <- function ( data , index )  
 coef (  
 lm( mpg ~ horsepower + I( horsepower ^2) ,  
 data = data , subset = index )  
 )  
  
set.seed (1)  
boot ( Auto , boot.fn , 1000)

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = Auto, statistic = boot.fn, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 56.900099702 3.511640e-02 2.0300222526  
## t2\* -0.466189630 -7.080834e-04 0.0324241984  
## t3\* 0.001230536 2.840324e-06 0.0001172164

summary (  
 lm( mpg ~ horsepower + I( horsepower ^2) , data = Auto )  
)$ coef

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 56.900099702 1.8004268063 31.60367 1.740911e-109  
## horsepower -0.466189630 0.0311246171 -14.97816 2.289429e-40  
## I(horsepower^2) 0.001230536 0.0001220759 10.08009 2.196340e-21