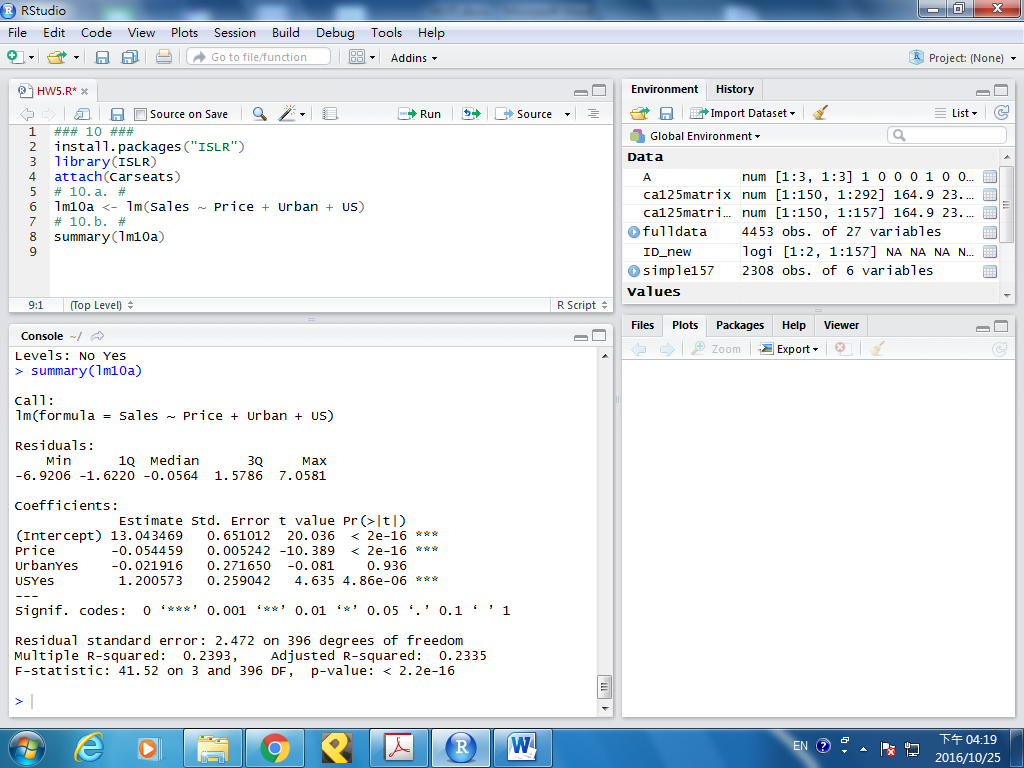
Statistical Learning and Data mining

Homework 5

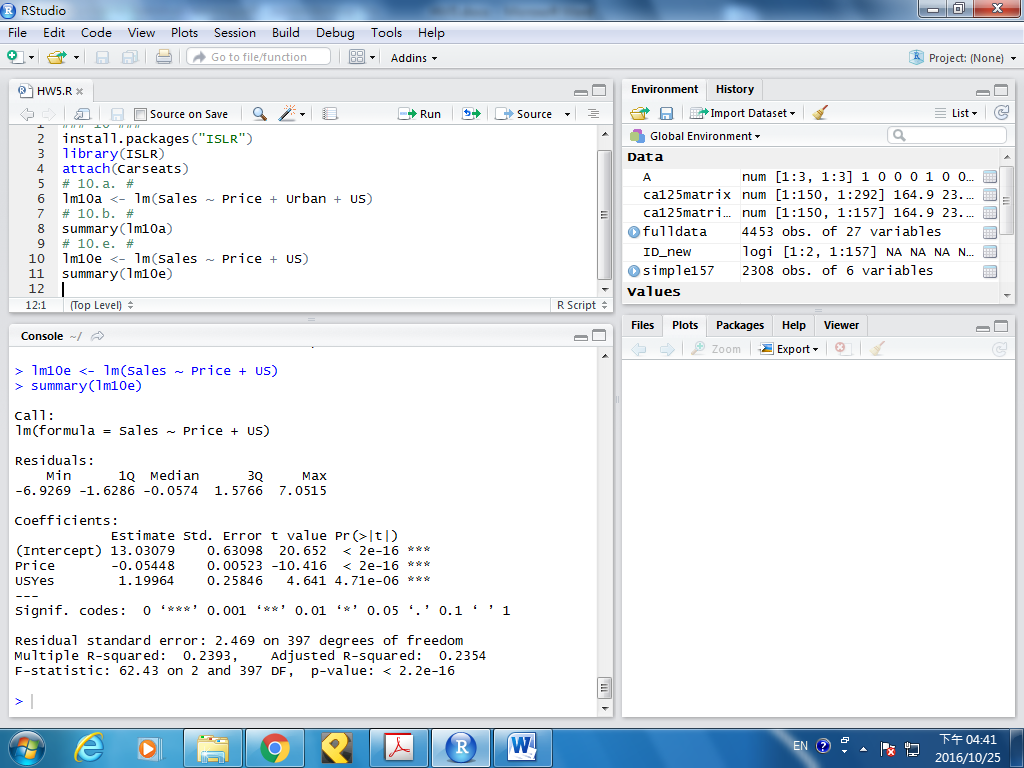
M052040003 鍾冠毅

1. KNN Classifier 主要處理質化資料，經由找到最靠近的K個樣本點中的眾數，也就是K個中出現最多次的類別，作為分類依據。KNN Regression則處理量化資料，當反應變數屬於數值類，則取欲預測的點附近K個樣本點，並將其值取平均，即為預測值。依照反應變數的類型，前者處理質化資料，後者處理量化資料。

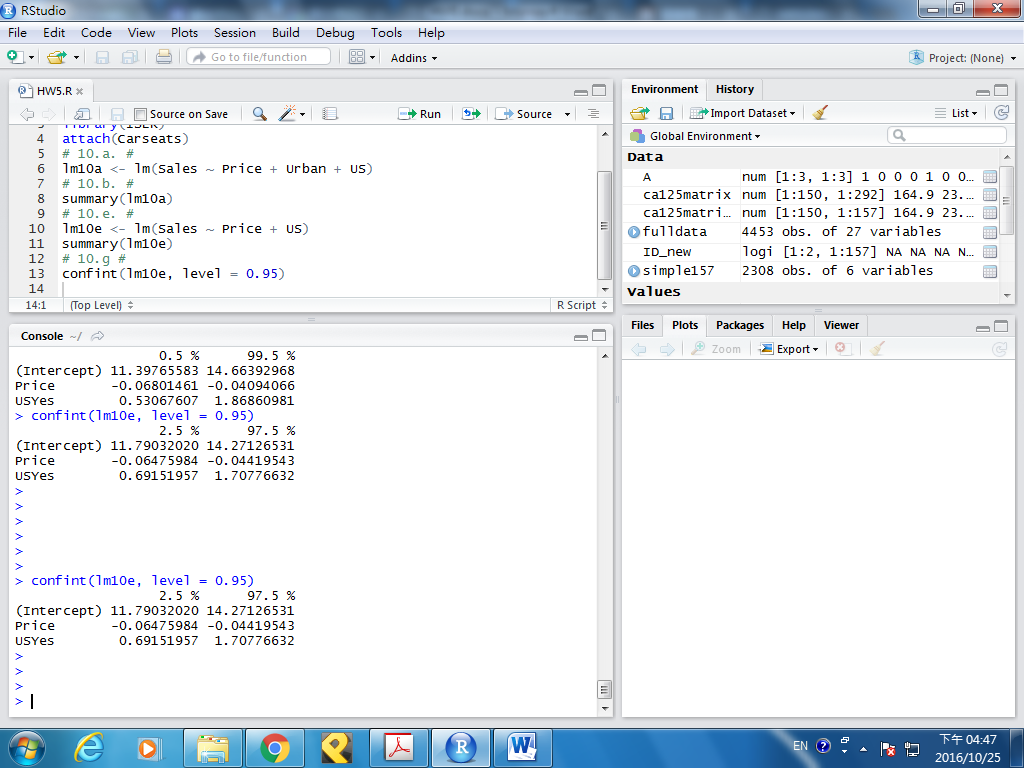


* + 1. Price 的p-value極小，代表該變數對Sales有顯著影響。又估計值為負，則Price對Sales有一定程度的副影響，也就是說，價格越高效售量越低。
    2. UrbanYes的p-value太大，沒有證據顯示該變數對模型的影響力。
    3. USYes的p-value足夠小，使得有證據拒絕係數為0之假設，故對模型有一定程度的影響力。又估計值為正，代表若該變數若為「是」則對模型有正影響。

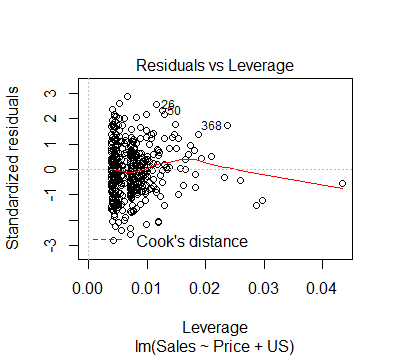
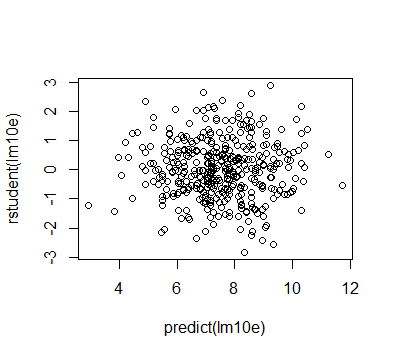
  2. Price、USYes的value小於0.05，則拒絕係數為0之虛無假設。



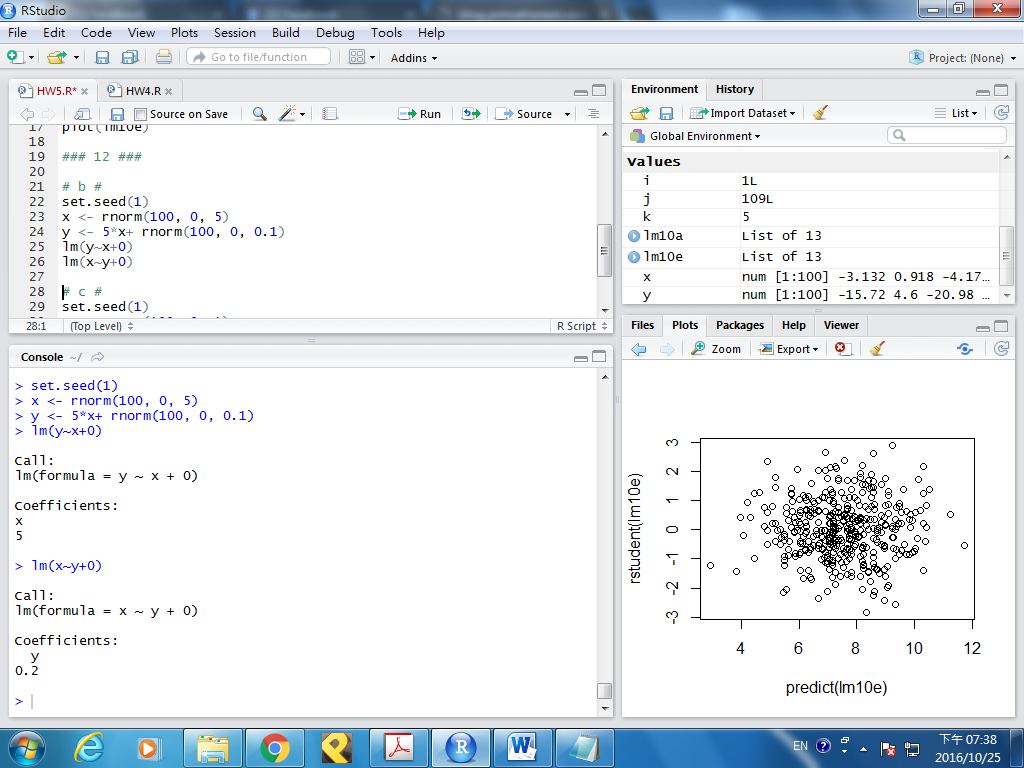
* 1. 兩者同為0.2393，但是有些微的上升。



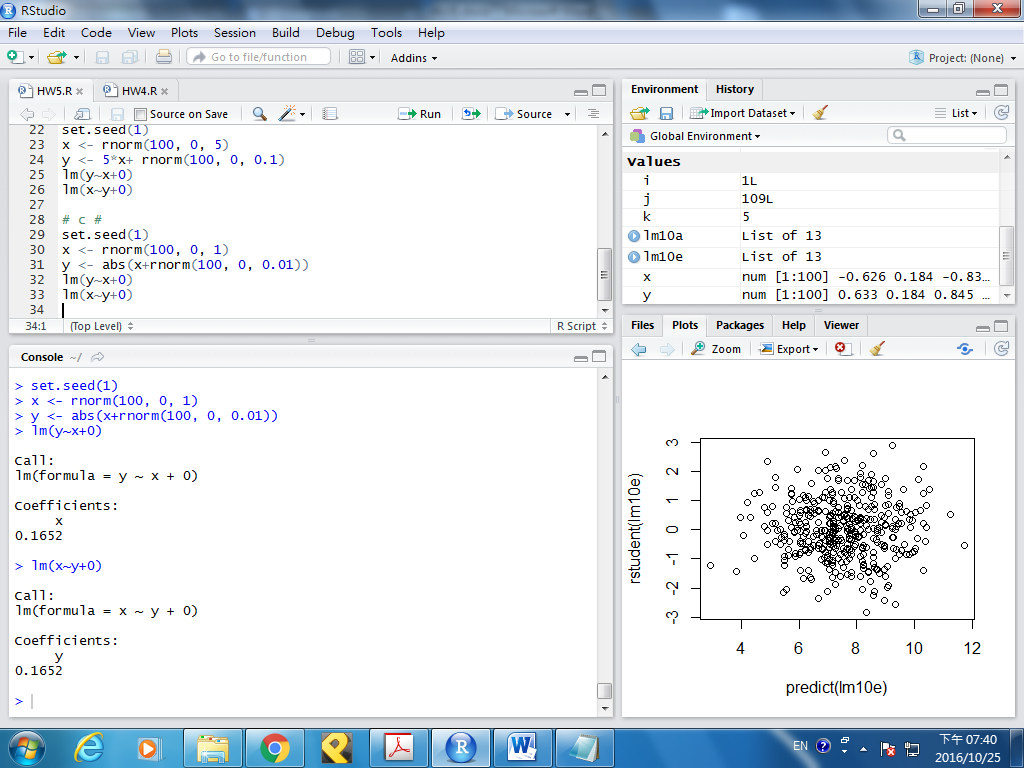
* 1. 左圖沒有點大於3或小於-3，則代表無離群值。右圖可以發現有槓桿點。



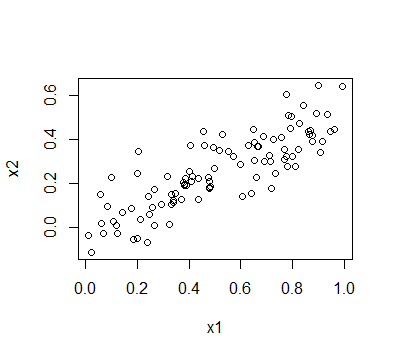
* 1. 當。



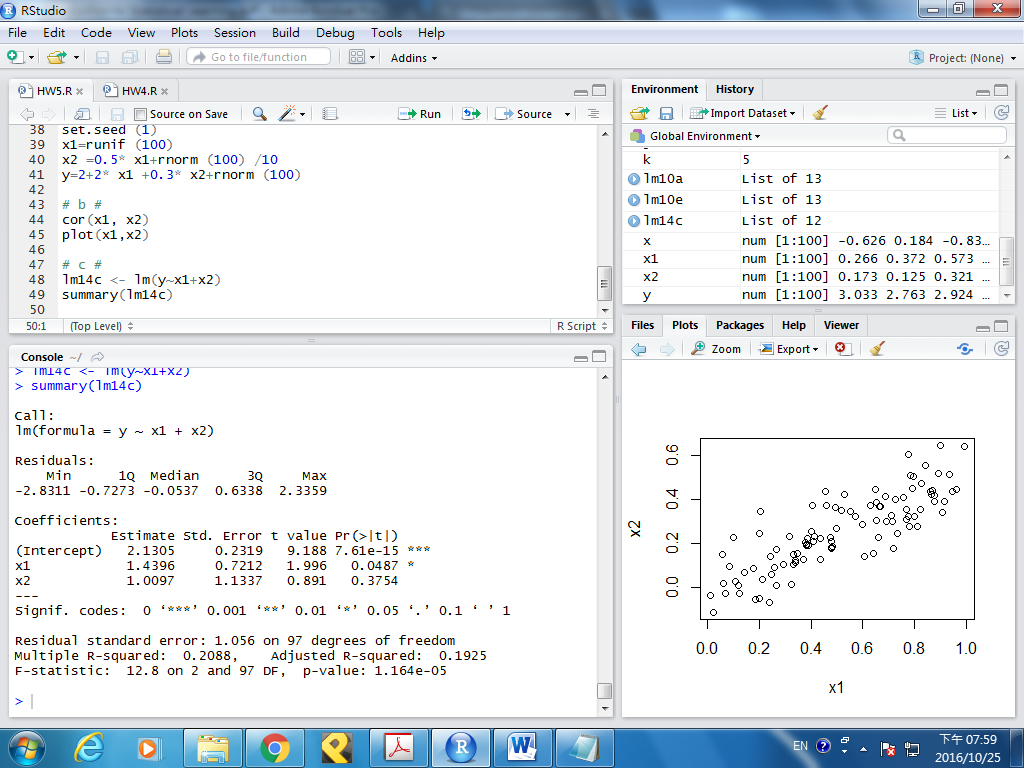




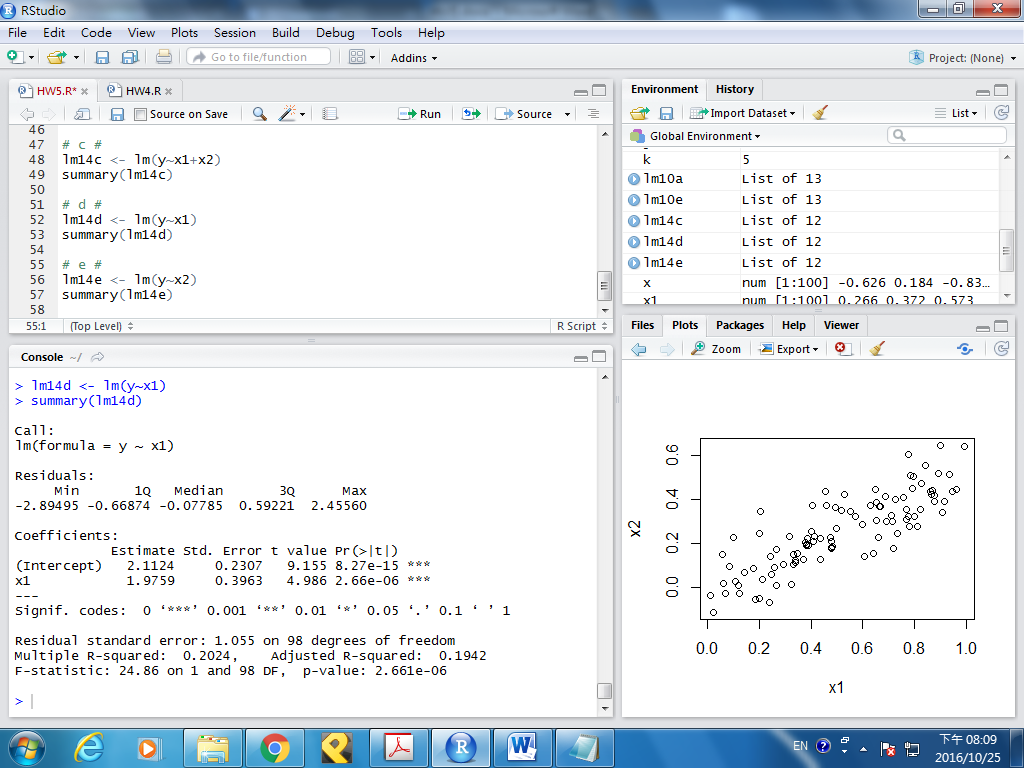
1. 1. , , ,
   2. ，有高度正相關。



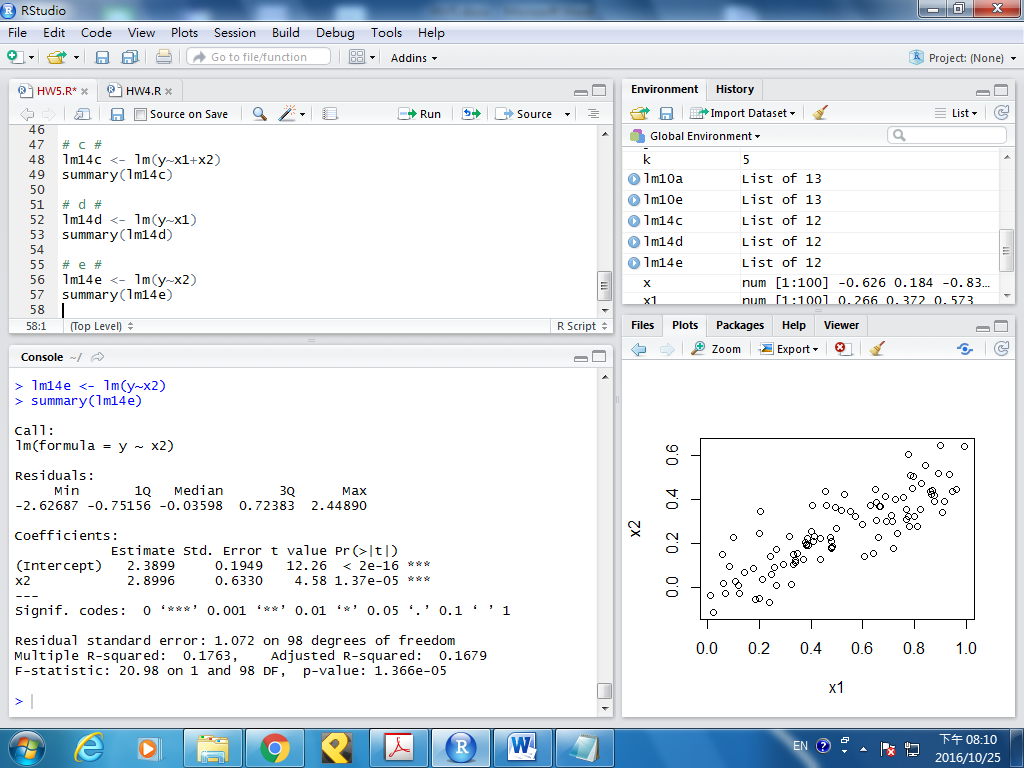
* 1. , , ，與真實的係數有差距。x2的係數p-value大於0.05，故無法拒絕係數為0之虛無假設；反之x1拒絕了係數為0之虛無假設。



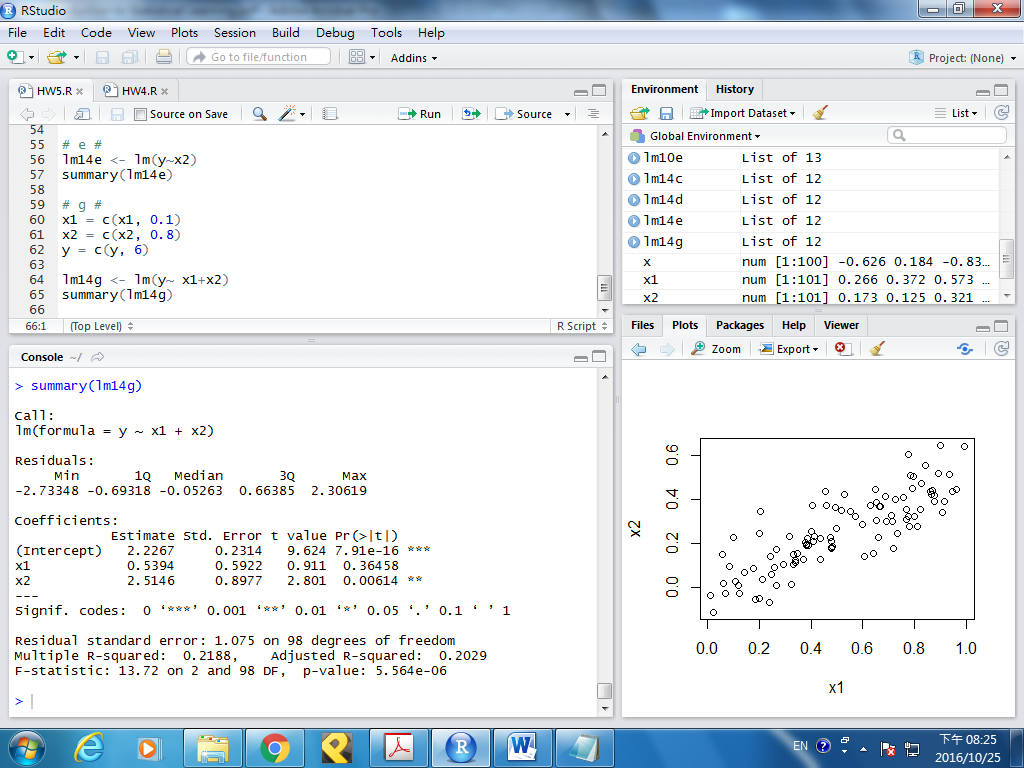
* 1. x1 的 p-value足夠小，使得有證據拒絕係數為0之假設。

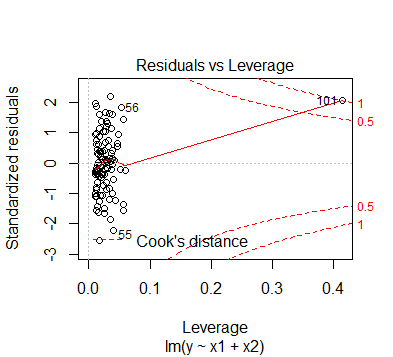
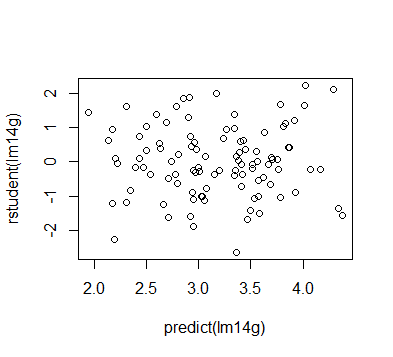


* 1. x2的 p-value足夠小，使得有證據拒絕係數為0之假設。

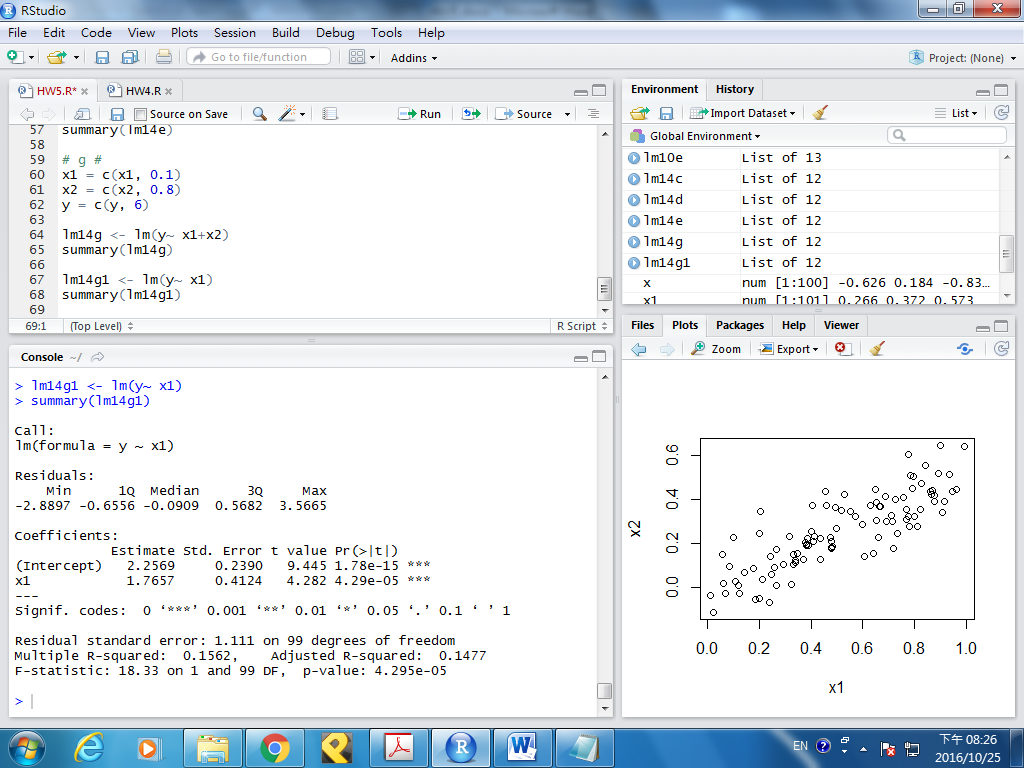


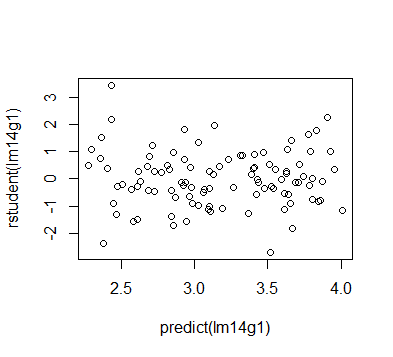
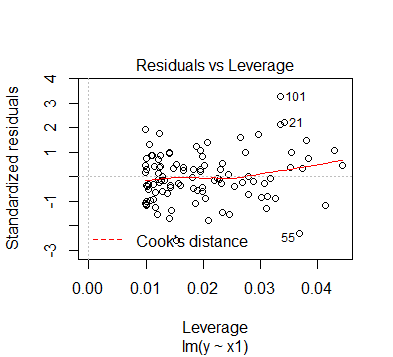
* 1. 若將x1與x2分別做為y的解釋變數，則皆為顯著，但是若同時使用x1與x2，則兩者顯著性將會降低。其原因係x1與x2有共線性的問題存在，導致模型無法分辨其變異是由哪個變數提供，故p-value同時提高。
     1. x1變得不顯著，而x2變得顯著。沒有任何樣本點的絕對值大於3，所以沒有離群值。第101個樣本點屬於槓桿點。



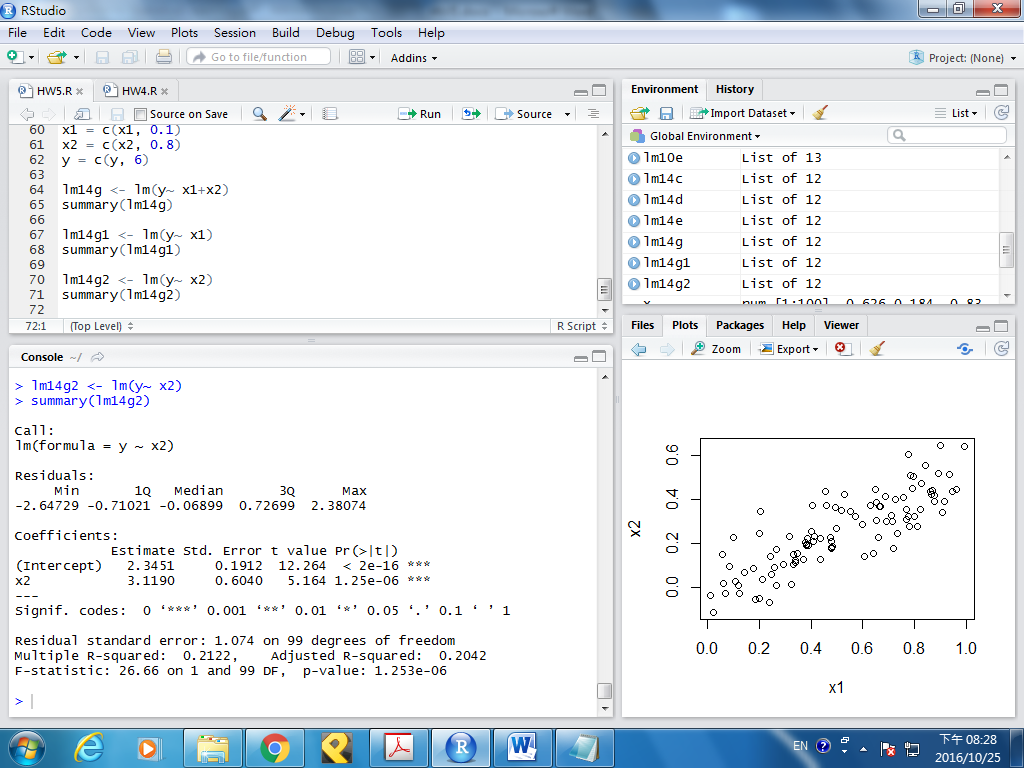


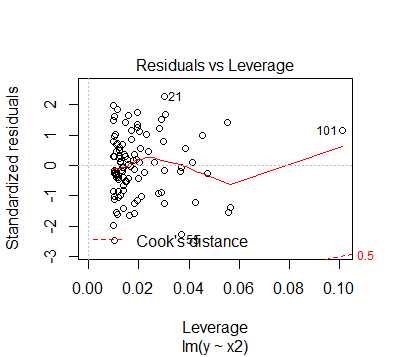
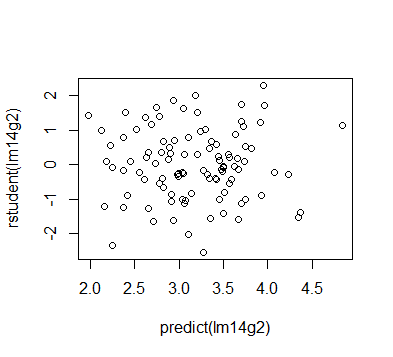
* + 1. 沒有離群值，也無槓桿點。



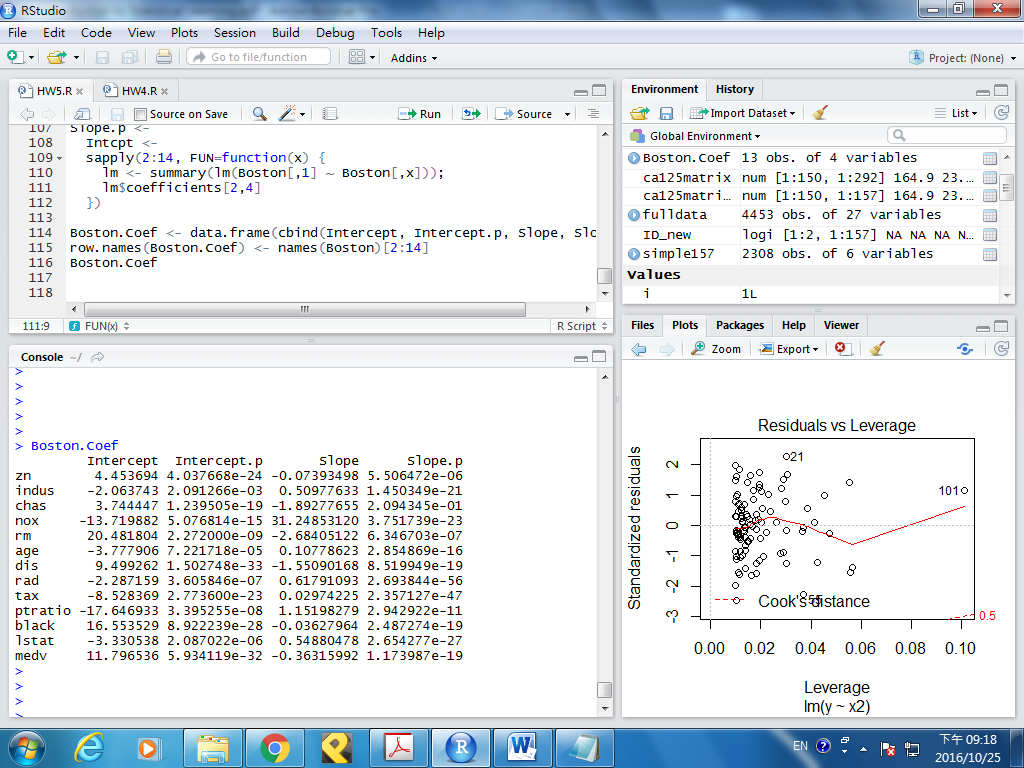
 

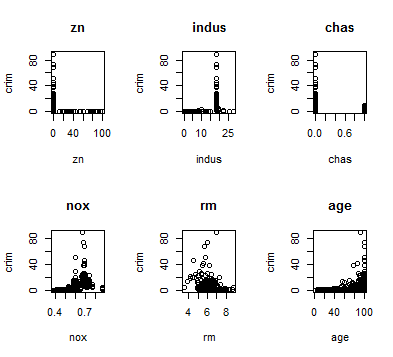
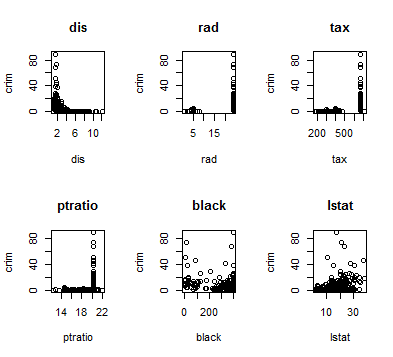
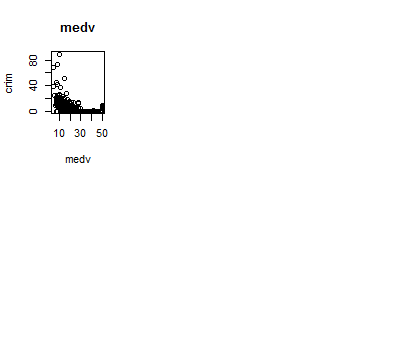
* + 1. 沒有離群值，但是有槓桿點（101）。



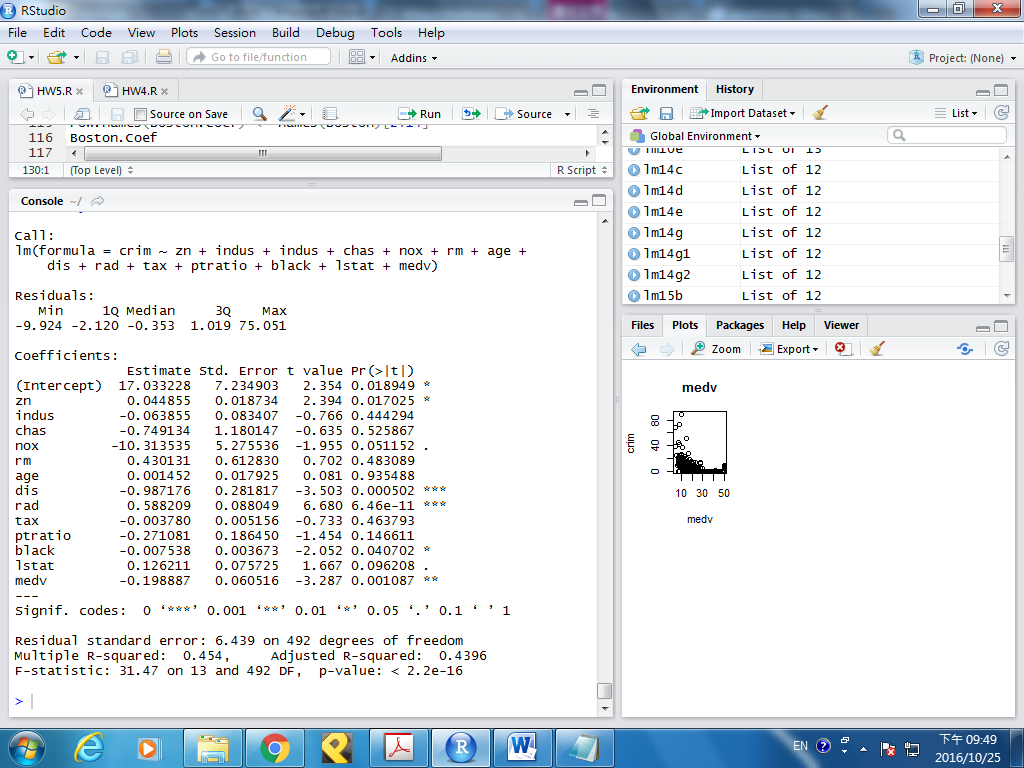


1. 1. 除了chas不顯著以外，其他都顯著。

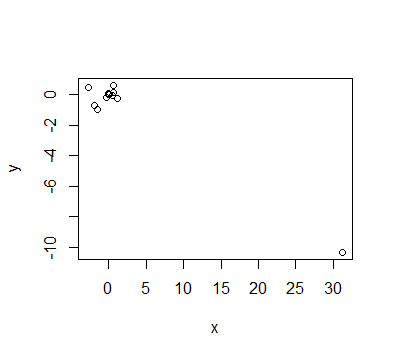


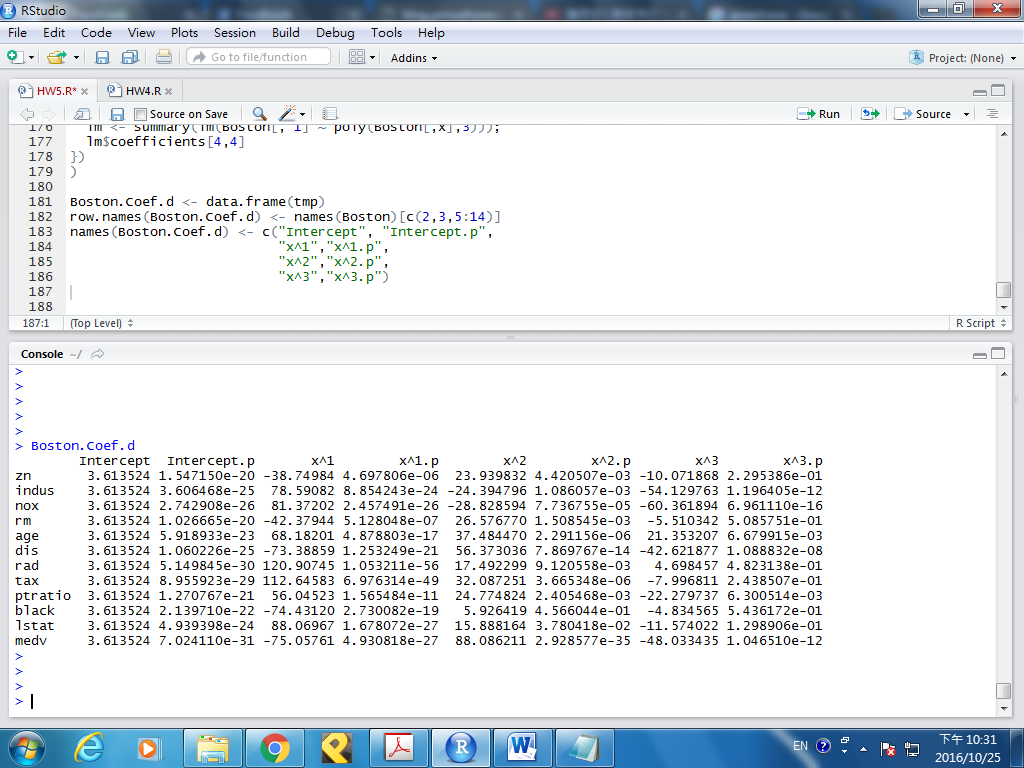
* 1. zn、 dis、 rad、 black、 medv的 p-value足夠小，使得有證據拒絕係數為0之假設。







* 1. 除了black只有一次顯著以外，其他變數的顯著都達二次或三次。另外chas變數為二元變數，故無法以三次多項式討論。



Appendix

### 10 ###

install.packages("ISLR")

library(ISLR)

attach(Carseats)

# 10.a. #

lm10a <- lm(Sales ~ Price + Urban + US)

# 10.b. #

summary(lm10a)

# 10.e. #

lm10e <- lm(Sales ~ Price + US)

summary(lm10e)

# 10.g #

confint(lm10e, level = 0.95)

# 10.h #

par(mfrow = c(1,1))

plot(predict(lm10e), rstudent(lm10e))

plot(lm10e)

### 12 ###

# b #

set.seed(1)

x <- rnorm(100, 0, 5)

y <- 5\*x+ rnorm(100, 0, 0.1)

lm(y~x+0)

lm(x~y+0)

# c #

set.seed(1)

x <- rnorm(100, 0, 1)

y <- abs(x+rnorm(100, 0, 0.01))

lm(y~x+0)

lm(x~y+0)

### 14 ###

# a #

set.seed (1)

x1=runif (100)

x2 =0.5\* x1+rnorm (100) /10

y=2+2\* x1 +0.3\* x2+rnorm (100)

# b #

cor(x1, x2)

plot(x1,x2)

# c #

lm14c <- lm(y~x1+x2)

summary(lm14c)

# d #

lm14d <- lm(y~x1)

summary(lm14d)

# e #

lm14e <- lm(y~x2)

summary(lm14e)

# g #

x1 = c(x1, 0.1)

x2 = c(x2, 0.8)

y = c(y, 6)

lm14g <- lm(y~ x1+x2)

summary(lm14g)

plot(predict(lm14g), rstudent(lm14g))

plot(lm14g)

lm14g1 <- lm(y~ x1)

summary(lm14g1)

plot(predict(lm14g1), rstudent(lm14g1))

plot(lm14g1)

lm14g2 <- lm(y~ x2)

summary(lm14g2)

plot(predict(lm14g2), rstudent(lm14g2))

plot(lm14g2)

### 15 ###

install.packages("MASS")

library(MASS)

lm <- summary(lm(Boston[, 1]~Boston[, 4]))

lm$coefficients

# a #

Intercept <- sapply(2:14, FUN=function(x) {

lm <- summary(lm(Boston[,1] ~ Boston[,x]));

lm$coefficients[1,1]})

Intercept.p <- sapply(2:14, FUN=function(x) {

lm <- summary(lm(Boston[,1] ~ Boston[,x]));

lm$coefficients[1,4]})

Slope <-

sapply(2:14, FUN=function(x) {

lm <- summary(lm(Boston[,1] ~ Boston[,x]));

lm$coefficients[2,1] })

cbind(Intercept, Slope)

Slope.p <-

Intcpt <-

sapply(2:14, FUN=function(x) {

lm <- summary(lm(Boston[,1] ~ Boston[,x]));

lm$coefficients[2,4] })

Boston.Coef <- data.frame(cbind(Intercept, Intercept.p, Slope, Slope.p))

row.names(Boston.Coef) <- names(Boston)[2:14]

Boston.Coef

par(mfrow = c(2,3))

for (k in 2:14) {

plot(Boston[,1] ~ Boston[,k],

main = names(Boston)[k],

xlab = names(Boston)[k],

ylab = names(Boston)[1])}

par(mfrow = c(1,1))

# b #

attach(Boston)

lm15b <- lm(crim ~ zn+indus+indus+chas+nox+rm+age+dis+rad+tax+ptratio+black+lstat+medv)

summary(lm15b)

# c #

plot(lm15b$coefficients[2:14] ~ Boston.Coef[,3], xlab="x", ylab = "y")

# d #

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,2],3)))

tmp <- cbind(

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[1,1]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[1,4]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[2,1]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[2,4]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[3,1]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[3,4]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[4,1]}),

sapply(c(2,3,5:14), function(x){

lm <- summary(lm(Boston[, 1] ~ poly(Boston[,x],3)));

lm$coefficients[4,4]}))

Boston.Coef.d <- data.frame(tmp)

row.names(Boston.Coef.d) <- names(Boston)[c(2,3,5:14)]

names(Boston.Coef.d) <- c("Intercept", "Intercept.p", "x^1","x^1.p","x^2","x^2.p","x^3","x^3.p")