PSU\_MLR

Sam Colon

10/13/2017

## 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:

data <- read.csv('brain.csv')  
head(data)

## Gender FSIQ VIQ PIQ Weight Height MRICount  
## 1 Female 133 132 124 118 64.5 816932  
## 2 Male 140 150 124 \* 72.5 1001121  
## 3 Male 139 123 150 143 73.3 1038437  
## 4 Male 133 129 128 172 68.8 965353  
## 5 Female 137 132 134 147 65.0 951545  
## 6 Female 99 90 110 146 69.0 928799

library(dplyr)

## Warning: package 'dplyr' was built under R version 3.4.1

##   
## Attaching package: 'dplyr'

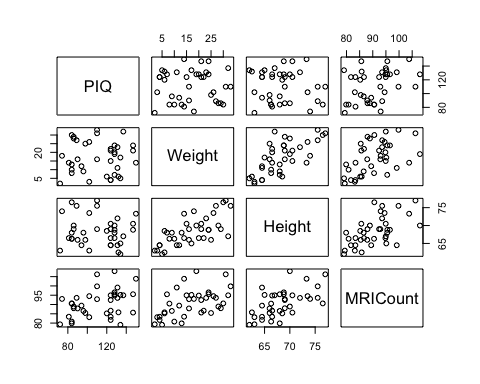
## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

data\_subset <- subset(data, select=c('PIQ', 'Weight','Height', 'MRICount'))

#remove case with missing data  
data\_subset <- data\_subset[-c(2,21),]  
data\_subset$MRICount <- data\_subset$MRICount/10000 #divide var by 10,000  
data\_subset$Weight <- as.numeric(data\_subset$Weight)  
#convert weight from factor to numeric

#make scatterplot matrix  
pairs(data\_subset)



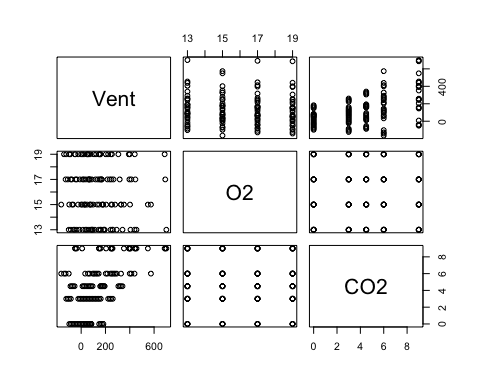
### It appears there's collinearity in the height and weight, although PIQ doesn't look linearly related to the other variables.

##build multiple reg model for this dataset with PIQ as response variable,  
model <- lm(PIQ ~ Weight+Height+MRICount, data=data\_subset)  
summary(model)

##   
## Call:  
## lm(formula = PIQ ~ Weight + Height + MRICount, data = data\_subset)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -32.840 -12.121 -3.848 14.221 51.730   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 109.70504 74.68680 1.469 0.151061   
## Weight -0.01635 0.50549 -0.032 0.974379   
## Height -2.70728 1.22553 -2.209 0.034009 \*   
## MRICount 2.06395 0.56443 3.657 0.000856 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 19.79 on 34 degrees of freedom  
## Multiple R-squared: 0.2949, Adjusted R-squared: 0.2327   
## F-statistic: 4.741 on 3 and 34 DF, p-value: 0.007218

### Baby Bird Dataset

bird\_df <- read.table('babybirds.txt', header=TRUE)  
pairs(bird\_df)



bird\_model <- lm(Vent ~ O2 + CO2, data=bird\_df)  
summary(bird\_model)

##   
## Call:  
## lm(formula = Vent ~ O2 + CO2, data = bird\_df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -356.57 -96.50 8.72 84.68 422.44   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 85.901 106.006 0.810 0.419   
## O2 -5.330 6.425 -0.830 0.408   
## CO2 31.103 4.789 6.495 2.1e-09 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 157.4 on 117 degrees of freedom  
## Multiple R-squared: 0.2682, Adjusted R-squared: 0.2557   
## F-statistic: 21.44 on 2 and 117 DF, p-value: 1.169e-08

### Soapsuds Example

##soapsuds example, computing regression coefficients  
soap\_df <- read.table('soapsuds.txt', header=TRUE)  
X <- as.matrix(soap\_df['soap'])  
X <- cbind(c(1,1,1,1,1,1,1),X) #add columns of 1s  
XtX <- t(X) %\*% X  
Y <- soap\_df$suds  
XtY <- t(X) %\*%Y  
X\_inv <- solve(XtX)  
b <- X\_inv%\*%XtY  
print(b)

## [,1]  
## -2.678571  
## soap 9.500000

### So we see the reg equation is suds = -2.68 + 9.500 \* soap

### What if we purposely make columns linearly dependent

X2 <- data.frame(X)  
X2$soap2 <- (X2$soap)\*2  
X2 <- as.matrix(X2)  
X2tX2 <- t(X2) %\*% X2  
X2\_inv <- solve(X2tX2) #we get error telling us matrix is singular, so inverse not computible.

## Error in solve.default(X2tX2): Lapack routine dgesv: system is exactly singular: U[3,3] = 0

### Pastry Sweetness Dataset: An experiment to assess how moisture content and sweetness of a pastry affects a taster's ratings of the pastry

pastry\_df <- read.csv('pastry.csv', header=TRUE)  
pastry\_df$Sweetness<- as.factor(pastry\_df$Sweetness)

### Check correlation of predictor variables

cor(pastry\_df$Moisture, as.numeric(pastry\_df$Sweetness), method="pearson")

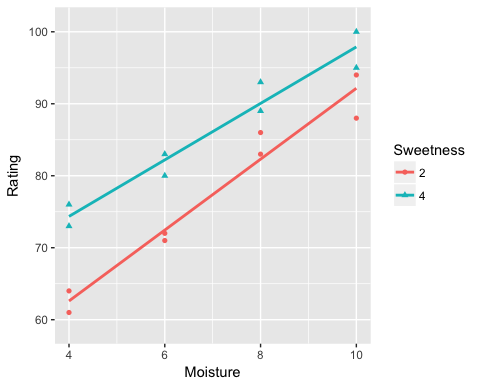
## [1] 0

cor.test(pastry\_df$Moisture, as.numeric(pastry\_df$Sweetness))

##   
## Pearson's product-moment correlation  
##   
## data: pastry\_df$Moisture and as.numeric(pastry\_df$Sweetness)  
## t = 0, df = 14, p-value = 1  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.4957053 0.4957053  
## sample estimates:  
## cor   
## 0

### We see 0 correlation.

library(ggplot2)  
#Plot of Rating vs Moisure w/markers for both levels of sweetness.   
ggplot(pastry\_df, aes(y = Rating, x= Moisture, colour= Sweetness, shape = Sweetness)) + geom\_point() +geom\_smooth(method='lm', fill=NA)



### Perform Simple and Multiple Regressions on Pastry Data

moisture\_model <- lm(Rating~ Moisture, data=pastry\_df)  
summary(moisture\_model)

##   
## Call:  
## lm(formula = Rating ~ Moisture, data = pastry\_df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.475 -4.688 -0.100 4.638 7.525   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 50.775 4.395 11.554 1.52e-08 \*\*\*  
## Moisture 4.425 0.598 7.399 3.36e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5.349 on 14 degrees of freedom  
## Multiple R-squared: 0.7964, Adjusted R-squared: 0.7818   
## F-statistic: 54.75 on 1 and 14 DF, p-value: 3.356e-06

anova(moisture\_model)

## Analysis of Variance Table  
##   
## Response: Rating  
## Df Sum Sq Mean Sq F value Pr(>F)   
## Moisture 1 1566.45 1566.45 54.751 3.356e-06 \*\*\*  
## Residuals 14 400.55 28.61   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

pastry\_df <- read.csv('pastry.csv', header=TRUE)  
sweetness\_model <- lm(Rating ~ Sweetness, data = pastry\_df)  
summary(sweetness\_model)

##   
## Call:  
## lm(formula = Rating ~ Sweetness, data = pastry\_df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.375 -7.312 -0.125 8.688 16.625   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 68.625 8.610 7.970 1.43e-06 \*\*\*  
## Sweetness 4.375 2.723 1.607 0.13   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 10.89 on 14 degrees of freedom  
## Multiple R-squared: 0.1557, Adjusted R-squared: 0.09539   
## F-statistic: 2.582 on 1 and 14 DF, p-value: 0.1304

anova(sweetness\_model)

## Analysis of Variance Table  
##   
## Response: Rating  
## Df Sum Sq Mean Sq F value Pr(>F)  
## Sweetness 1 306.25 306.25 2.5817 0.1304  
## Residuals 14 1660.75 118.62

pastry\_df <- read.csv('pastry.csv', header=TRUE)  
pastry\_mlr <- lm(Rating ~ Moisture + Sweetness, data=pastry\_df)  
summary(pastry\_mlr)

##   
## Call:  
## lm(formula = Rating ~ Moisture + Sweetness, data = pastry\_df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.400 -1.762 0.025 1.587 4.200   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 37.6500 2.9961 12.566 1.20e-08 \*\*\*  
## Moisture 4.4250 0.3011 14.695 1.78e-09 \*\*\*  
## Sweetness 4.3750 0.6733 6.498 2.01e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.693 on 13 degrees of freedom  
## Multiple R-squared: 0.9521, Adjusted R-squared: 0.9447   
## F-statistic: 129.1 on 2 and 13 DF, p-value: 2.658e-09

anova(pastry\_mlr)

## Analysis of Variance Table  
##   
## Response: Rating  
## Df Sum Sq Mean Sq F value Pr(>F)   
## Moisture 1 1566.45 1566.45 215.947 1.778e-09 \*\*\*  
## Sweetness 1 306.25 306.25 42.219 2.011e-05 \*\*\*  
## Residuals 13 94.30 7.25   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

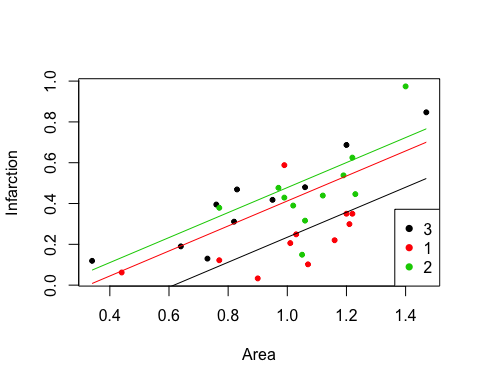
### Heart Rabbit Study

rabbit\_df <- read.csv('coolhearts.csv', header=TRUE)  
rabbit\_df$Group <- as.factor(rabbit\_df$Group)  
## Set Group 3 as reference level  
rabbit\_df <- within(rabbit\_df, Group <-relevel(Group, ref=3))  
  
#Build Reg Model  
rabbit\_lm <- lm(Infarction ~ factor(Group) + Area, data = rabbit\_df)  
summary(rabbit\_lm)

##   
## Call:  
## lm(formula = Infarction ~ factor(Group) + Area, data = rabbit\_df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.29410 -0.06511 -0.01329 0.07855 0.35949   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.13454 0.10402 -1.293 0.206459   
## factor(Group)1 -0.24348 0.06229 -3.909 0.000536 \*\*\*  
## factor(Group)2 -0.06566 0.06507 -1.009 0.321602   
## Area 0.61265 0.10705 5.723 3.87e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1395 on 28 degrees of freedom  
## Multiple R-squared: 0.6377, Adjusted R-squared: 0.5989   
## F-statistic: 16.43 on 3 and 28 DF, p-value: 2.363e-06

### So, the regression equation is -.135 + .613\*Area -.2435X2 - .0657X3

#Plot of Infarction vs Area, with separate regression lines for each group  
plot(Infarction~Area,rabbit\_df,col=rabbit\_df$Group,pch=20)  
curve(predict(rabbit\_lm,newdata=data.frame(Area=x,Group=1)),col=1,add=T)  
curve(predict(rabbit\_lm,newdata=data.frame(Area=x,Group=2)),col=2,add=T)  
curve(predict(rabbit\_lm,newdata=data.frame(Area=x,Group=3)),col=3,add=T)  
legend('bottomright', levels(factor(rabbit\_df$Group)), pch = 16, col = which(levels(rabbit\_df$Group) %in% levels(factor(rabbit\_df$Group))))



### So, we see that as risk Area increases, size of Infarction tends to increase.

### Calculating F-stat from regression output:

full\_anova <- anova(rabbit\_lm) #sum of squares from full model  
  
#reduced model  
rabbit\_lmr <- lm(Infarction ~ + Area, data = rabbit\_df)  
reduced\_anova <- anova(rabbit\_lmr)  
  
#The general F-stat is calculated as (SSE(R)-SSE(F)/df(SSE(R)-df(SSE(F))))/(SSE(F)/df(SSE(F))  
  
F\_stat <- (reduced\_anova$`Sum Sq`[2] -full\_anova$`Sum Sq`[3])/(reduced\_anova$Df[2]-full\_anova$Df[3])/(full\_anova$`Sum Sq`[3]/((full\_anova$Df[3])))  
F\_stat

## [1] 8.590176

### We can explicitly calculate the p-value of this F\_stat

pf(F\_stat, 2, 28, lower.tail=FALSE)

## [1] 0.001232872

### Therefore, we can conclude that there is a statistically significant relationship between the method of cooling treatment and infarction size.

### We can also compute a partial F-statistic for variables x2 and x3 using sequential sum of squares, where we add x2 and x3 in order, given that x1 is already in the model

rabbit\_partialF<- lm(Infarction ~ + Area + X2 + X3 , data = rabbit\_df)  
anova\_partial <- anova(rabbit\_partialF)  
rabbit\_partialF

##   
## Call:  
## lm(formula = Infarction ~ +Area + X2 + X3, data = rabbit\_df)  
##   
## Coefficients:  
## (Intercept) Area X2 X3   
## -0.13454 0.61265 -0.24348 -0.06566

anova\_partial

## Analysis of Variance Table  
##   
## Response: Infarction  
## Df Sum Sq Mean Sq F value Pr(>F)   
## Area 1 0.62492 0.62492 32.1115 4.504e-06 \*\*\*  
## X2 1 0.31453 0.31453 16.1622 0.000398 \*\*\*  
## X3 1 0.01981 0.01981 1.0181 0.321602   
## Residuals 28 0.54491 0.01946   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# The F stat becomes SSR(X2, X3|X1)/(2)/(SSE(X2,X3|28)) or MSR(X2, X3|X1)/MSE(X1,X2,X3)  
  
partial\_Fstat<- ((anova\_partial$`Sum Sq`[2]+ anova\_partial$`Sum Sq`[3])/2)/((anova\_partial$`Sum Sq`[4]/28))  
partial\_Fstat

## [1] 8.590176

### We see the resulting F-stat is the same

### To test if one slope parameter is 0, we can also use the t-test, as the square of the t-stat will equal the F-statistic

#The test for testing area is:  
rabbit\_lm <- lm(Infarction ~ + Area + X2 + X3 , data = rabbit\_df)  
full\_anova <-anova(rabbit\_lm)  
Area\_Fstat <- (full\_anova$`Sum Sq`[1]/full\_anova $Df[1])/(full\_anova$`Sum Sq`[4]/full\_anova$Df[4])  
Area\_Fstat

## [1] 32.11154

pf(Area\_Fstat, 1, 28, lower.tail=FALSE)

## [1] 4.503703e-06

summary <- summary(rabbit\_lm)  
#Get t-stat for Area and square it   
summary$coefficients[2,3] ^2

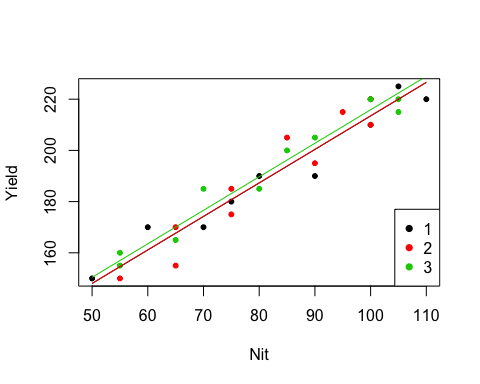
## [1] 32.7536

#We see it nearly equals F-stat above

### Sugar Beet Study

#plot(Infarction~Area,rabbit\_df,col=rabbit\_df$Group,pch=20)  
#curve(predict(rabbit\_lm,newdata=data.frame(Area=x,Group=1)),col=1,add=T)  
  
beet\_df <-read.csv('beets.csv')  
beet\_df$Treat <- as.factor(beet\_df$Treat)  
beet\_df <- within(beet\_df, Treat <-relevel(Treat, ref=1))

#Regression model  
beet\_lm <- lm(Yield ~ factor(Treat) + Nit, data = beet\_df)  
  
  
plot(Yield ~ Nit,beet\_df,col=beet\_df$Treat,pch=20)  
curve(predict(beet\_lm,newdata=data.frame(Nit=x,Treat=1)),col=1,add=T)  
curve(predict(beet\_lm,newdata=data.frame(Nit=x,Treat=2)),col=2,add=T)  
curve(predict(beet\_lm,newdata=data.frame(Nit=x,Treat=3)),col=3,add=T)  
legend('bottomright', levels(factor(beet\_df$Treat)), pch = 16, col = which(levels(beet\_df$Treat) %in% levels(factor(beet\_df$Treat))))



### So, we see a clear linear relationship between amount of nitrogen and yield of sugar beets