# Assignment 2

Derek Rouch

LLO 8200

Introduction to Data Science

September 1, 2019

## Setting up R

To begin, I cleared my Global Environment and get the necessary libraries: knitr, tidyverse, plotly, and Metrics.

## Clear environment  
rm(list=ls())  
  
## Get libraries  
library(knitr)  
knitr::opts\_chunk$set(echo = TRUE)  
library(tidyverse)

## ── Attaching packages ──────────────────────────────────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.2.1 ✔ purrr 0.3.2  
## ✔ tibble 2.1.3 ✔ dplyr 0.8.3  
## ✔ tidyr 0.8.3 ✔ stringr 1.4.0  
## ✔ readr 1.3.1 ✔ forcats 0.4.0

## ── Conflicts ─────────────────────────────────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(plotly)

##   
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':  
##   
## last\_plot

## The following object is masked from 'package:stats':  
##   
## filter

## The following object is masked from 'package:graphics':  
##   
## layout

library(Metrics)

Then, I loaded the pd.Rdata dataset, as well as the pd\_lab\_explain.Rdata codebook to decode the column headings.

## Load in the county-level data  
load("/Users/derekrouch/Documents/GitHub/data-science/pd.Rdata")  
  
## Load the variable descriptions  
load("/Users/derekrouch/Documents/GitHub/data-science/pd\_lab\_explain.Rdata")

## 1) Calculate the mean of the outcome.

With the data loaded, I was able to pipe it to the summarize function to view the unconditional average of the home ownership rate variable, homeown\_rate, which I found using the mean function.

## Unconditional average  
pd%>%summarize(mean\_homeown\_rate=mean(homeown\_rate,na.rm=TRUE))

## # A tibble: 1 x 1  
## mean\_homeown\_rate  
## <dbl>  
## 1 72.7

## 2) Use your mean as a prediction: Create a new variable that consists of the mean of the outcome.

To create a new variable, I piped the data to the mutate function, and added mean\_homeown\_rate as a new column in my original pd dataset.

##Unconditional average as a predictor  
pd <- pd%>%mutate(mean\_homeown\_rate=mean(homeown\_rate,na.rm=TRUE))

## 3) Calculate a summary measure of the errors for each observation—the difference between your prediction and the outcome.

Now that every observation in my dataset had the same value for mean\_homeown\_rate, I could calculate the error for each observation—that is, how far each county’s home ownership rate is from the mean of 72.7%.

To accomplish this, I subtracted the mean\_homeown\_rate from each individual county’s homeown\_rate. I named this difference error\_uncond. Counties with positive error values have home ownership rates higher than the national average, while counties with negative error values have home ownership rates below that of the national average.

## Calculating the error term  
pd <- pd%>%mutate(error\_uncond=homeown\_rate-mean\_homeown\_rate)

To see how far off this unconditional predication is, I calculated the root mean squared error, using the rmse function.

## Calculating the root mean squared error  
rmse\_uncond\_mean <- rmse(pd$homeown\_rate,pd$mean\_homeown\_rate)  
  
## Calling the RMSE  
rmse\_uncond\_mean

## [1] 7.653637

This root mean squared error tells me that the unconditional mean is off by 7.65 percent.

## 4) Calculate the mean of the outcome at levels of a predictor variable.

In hopes of making a better estimate, I wanted to determine a predictor variable. First, I examined the table. However, with 57 variables (the original 55 plus my 2 new ones), it felt like I would be shooting in the dark, so I wanted a way of enhancing my guesswork.

I knew this would require venturing to the far frontiers of my *R* knowledge, but I felt that there was no corner Google and StackExchange couldn’t get me out of, so I ventured on.

*I apologize in advance if the following process is an affront to real statistical analysis.*

My first decision was to create a correlation matrix.

To do this, I first needed to make sure that the data was in numeric form. I used sapply to determine the class of each variable.

## Determine each variable's class  
sapply(pd, class)

## fips pop2013 pop2010\_base   
## "character" "integer" "integer"   
## popchange\_pc pop2010 popu5   
## "numeric" "integer" "numeric"   
## popu18 pop65p female\_pc   
## "numeric" "numeric" "numeric"   
## white\_pc black\_pc am\_ind\_pc   
## "numeric" "numeric" "numeric"   
## asian\_pc hawaii\_pi\_pc twomore\_race\_pc   
## "numeric" "numeric" "numeric"   
## hispanic\_pc white\_non\_hispanic\_pc same\_house\_pc   
## "numeric" "numeric" "numeric"   
## foreign\_born\_pc other\_eng\_home\_pc hs\_grad\_pc   
## "numeric" "numeric" "numeric"   
## coll\_grad\_pc veterans travel\_time   
## "numeric" "integer" "numeric"   
## housing\_units homeown\_rate house\_unit\_multi   
## "integer" "numeric" "numeric"   
## median\_home\_val households person\_per\_hh   
## "integer" "integer" "numeric"   
## per\_capita\_inc median\_hh\_inc persons\_below\_poverty   
## "integer" "integer" "numeric"   
## pv\_nonfarm pv\_nonfarm\_employ pv\_nonfarm\_employ\_ch   
## "integer" "integer" "numeric"   
## nonemployer\_est firms firms\_black\_own\_pc   
## "integer" "integer" "numeric"   
## firms\_amind\_own\_pc firms\_asian\_own\_pc firms\_hawaii\_pi\_own\_pc   
## "numeric" "numeric" "numeric"   
## firms\_hispanic\_own\_pc firms\_female\_own\_pc manufacture\_ship   
## "numeric" "numeric" "numeric"   
## wholesale retail retail\_percap   
## "numeric" "numeric" "integer"   
## hospitality bldg\_permits land\_area   
## "integer" "integer" "numeric"   
## pop\_per\_square county percapinc.2010   
## "numeric" "character" "numeric"   
## percapinc.2012 mean\_homeown\_rate error\_uncond   
## "numeric" "numeric" "numeric"

This showed me that the dataset’s variables fips and county were stored as text, so I used subset to remove those columns and create a new, numeric-only dataset—which I named pd\_num\_only.

## Create a subset of the data, consisting of only numeric variables  
pd\_num\_only <- subset(pd, select = -c(fips, county))

With this new dataset, I was finally able to create my correlation matrix, which I named cor\_matrix. Since I am only concerned with homeown\_rate correlations, I made a subset of the matrix to include only that variable.

## Create a correlation matrix, rounded to two decimal places  
cor\_matrix <- cor(pd\_num\_only)

## Warning in cor(pd\_num\_only): the standard deviation is zero

round(cor\_matrix, 2)  
  
## Remove every column except `homeown\_rate` in the matrix  
cor\_homeown\_rate <- subset(cor\_matrix, select = c(homeown\_rate))

By viewing and sorting the new subset, I found two variables that showed some promise for predictive value. house\_unit\_multi (the percent of housing units in multi-unit structures) had a correlation coefficient of -0.68, and same\_house\_pc (the percent living in the same house for one or more years) had a correlation coefficient of 0.57.

Although negatively correlated, house\_unit\_multi had the strongest magnitude and was therefore my choice for predictor variable.

I then broke up house\_unit\_multi into four levels using the ntile function, and added it as a variable using mutate.

## Create a variable for quartiles of % housing units in multi-unit structures  
pd<-pd%>%mutate(house\_unit\_multi\_level=ntile(house\_unit\_multi,4))  
  
## Check for even distribution across levels  
table(pd$house\_unit\_multi\_level)

##   
## 1 2 3 4   
## 772 772 772 772

## View house\_unit\_multi\_level for each county  
pd%>%select(county,house\_unit\_multi,house\_unit\_multi\_level)%>%View()

From here, I grouped the data by house\_unit\_multi\_level, calculated the pred\_homeown\_rate for each level, and ranked them in descending order (i.e., higher predicted home ownership rates on top)

##Group by predictor level  
 pd<-pd%>%group\_by(house\_unit\_multi\_level)%>%  
 ##Calculate mean at each level of predictor  
 mutate(pred\_homeown\_rate=mean(homeown\_rate))%>%  
 ## Ungroup  
 ungroup()%>%   
 #Rank by prediction, with ties sorted randomly  
 mutate(pred\_homeown\_rate\_rank=rank(-pred\_homeown\_rate,ties.method="random"))

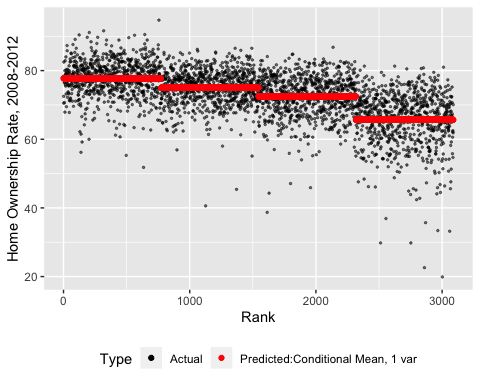
## 5) Use these conditional means as a prediction: for every county, use the conditional mean to provide a ‘’best guess” as to that county’s level of the outcome.

Finally, I could view the counties and sort them by pred\_homeown\_rate\_rank.

## View new pred\_homeown\_rate by county  
pd%>%select(county,house\_unit\_multi,house\_unit\_multi\_level,pred\_homeown\_rate, pred\_homeown\_rate\_rank)%>%View()

To visualize this as a plot, I used the ggplot function.

## Plotting   
gg<-ggplot(data=pd,aes(x=pred\_homeown\_rate\_rank,y=homeown\_rate,color="Actual"))  
  
## Stylizing the Plot  
gg<-gg+geom\_point(alpha=.5,size=.5)  
gg<-gg+geom\_point(aes(x=pred\_homeown\_rate\_rank,y=pred\_homeown\_rate,color="Predicted:Conditional Mean, 1 var"))  
gg<-gg+ scale\_color\_manual("Type",values=c("Predicted:Conditional Mean, 1 var"="red","Actual"="black"))  
gg<-gg+theme(legend.position="bottom")  
gg<-gg+xlab("Rank")+ylab("Home Ownership Rate, 2008-2012")  
  
## Calling the plot  
gg



## 6) Calculate a summary measure of the error in your predictions.

With my new pred\_homeown\_rate variable, I could now calculate the conditional error for each observation—that is, how far each county’s home ownership rate is from the mean of 72.7%, when predicted by house\_unit\_multi\_level.

To accomplish this, I subtracted the mean\_homeown\_rate from each individual county’s pred\_homeown\_rate. I named this difference error\_cond. Once again, counties with positive error values have home ownership rates higher than the national average, while counties with negative error values have home ownership rates below that of the national average.

## Calculating the error term  
pd <- pd%>%mutate(error\_cond=pred\_homeown\_rate-mean\_homeown\_rate)

As I did earlier, I also calculated the root mean squared error to see how far these values were off.

## Calculating the root mean squared error based on house\_unit\_multi\_level  
rmse\_cond\_mean <- rmse(pd$pred\_homeown\_rate,pd$mean\_homeown\_rate)  
  
## Calling the RMSE  
rmse\_cond\_mean

## [1] 4.47162

This new root mean squared error tells me that the conditional mean is off by 4.47 percent.

To see what how much adding the house\_unit\_multi\_level predictor variable improved my RMSE, I subtracted the two values.

## Calculating the difference between rmse\_uncond and rmse\_cond  
rmse\_difference <- rmse\_uncond\_mean-rmse\_cond\_mean  
  
## Calling the RMSE difference, rounded to two decimal places  
round(rmse\_difference, 2)

## [1] 3.18

To view this as a *percent improvement*, I divided my new rmse\_difference by the original rmse\_conditional\_mean.

## Calculating the percentage improvement  
rmse\_percent\_improved <- (rmse\_difference/rmse\_uncond\_mean)\*100  
  
## Calling the RMSE prcent improvement, rounded to two decimal places  
round(rmse\_percent\_improved, 2)

## [1] 41.58

My RMSE improved by roughly 41.6 percent when I applied my predictor variable.