hw\_week6

Ben Wilson

June 24, 2019

# Week 6 Homework

*Probability-Based Models, Missing Data, and Optimization*

## Setup

The packages included to run this notebook are:

* tidyverse
* mice

For exercise 13.2, I used SimPy, NumPy, and Itertools to complete the simulation in python.

## 13.2

*In this problem you can simulate a simplified airport security system at a busy airport. Passengers arrive according to a Poisson distribution with lambda = 5 per minute (i.e. mean interarrival rate mu1 = 0.2 minutes) to the boarding-pass check queue, where there are several servers who each have exponential services time with mean rate mu2 = 0.75 minutes.*

**HINT: Model the boarding-pass check as one block that has more than one resource**

*After that, the passengers are assigned to the shortest of the several personal-check queues, where they go through the personal scanner (time is uniformly distributed between 0.5 minutes and 1 minute).*

*Use Python with Simpy (my choice) to build a simulation of the system, and then vary the number of boarding-pass checkers and personal-check queues to determine how many are needed to keep average wait times below 15 minutes.*

In my simulation I found that 4 ID agents and 5 Personal Check agents worked best. They were able to get through 2,931 passengers for 600 minutes of simulation (10 hour workday).

Reference the airline-security-queue-sim.py for more details.

## 14.1

*The breast cancer data set provided has missing values*

*Tasks:*

1. Use the mean/mode imputation method to impute values for the missing data.
2. Use regression to impute values for the missing data.
3. Use regression with pertubation to impute values for the missing data.
4. (Optional) Compare the results and quality of classification models from the different methodologies of cleaning the data.

bc <- read.csv("data/breast-cancer-wisconsin.data.txt", header = FALSE)  
  
# Description pulled from http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29  
bc\_names <- c("sample\_number", "clump\_thickness", "unif\_csize", "unif\_cshape", "adhesion", "epith\_csize", "bare\_nuclei", "bland\_chromatin", "normal\_nucleoli", "mitoses", "class")  
  
names(bc) <- bc\_names

any(is.na(bc))

## [1] FALSE

There is no blank data.

Scanning the data, looks like bare\_nuclei is the only feature that needs to be imputed. A “?” has been inserted where there is not available data.

Let’s set that to NA just to make it a little easier to manipulate.

bc <- mutate(bc, bare\_nuclei = as.numeric(ifelse(bare\_nuclei == '?', NA, bare\_nuclei)))

Now we are ready to impute our data.

# Mean & Mode Imputation  
  
bn\_mean <- mean(bc$bare\_nuclei, na.rm = TRUE)  
  
# That's a very messy statement, but there is not `mode()` built into R surprisingly!  
bn\_mode <- as.integer(names(table(bc$bare\_nuclei))[which(table(bc$bare\_nuclei) == max(table(bc$bare\_nuclei)))])  
  
bc <- bc %>%   
 mutate(  
 bare\_nuclei\_mean = ifelse( is.na(bare\_nuclei), bn\_mean, bare\_nuclei ),  
 bare\_nuclei\_mode = ifelse( is.na(bare\_nuclei), bn\_mode, bare\_nuclei )  
 )

For the regression based imputation, we can leverage the work of the R community by making use of the mice package.

The package calls regression right on the predicted line “Deterministic Regression Imputation”.  
Regression with pertubations is called “Stochastic Regression Imputation”.

Let’s give these methods a try.

# WIthout Pertubations  
imp\_deterministic <- bc %>% select(  
 -sample\_number, # do not want the identifier  
 -bare\_nuclei\_mean, - bare\_nuclei\_mode, # do not want the other imputed columns  
 -class # do not want the target column  
 ) %>%   
 mice(m = 1, method = 'norm.predict')

##   
## iter imp variable  
## 1 1 bare\_nuclei  
## 2 1 bare\_nuclei  
## 3 1 bare\_nuclei  
## 4 1 bare\_nuclei  
## 5 1 bare\_nuclei

bn\_deterministic <- complete(imp\_deterministic) %>% select(bare\_nuclei\_deterministic = bare\_nuclei)  
  
# With Pertubations  
imp\_stochastic <- bc %>% select(  
 -sample\_number, # do not want the identifier  
 -bare\_nuclei\_mean, - bare\_nuclei\_mode, # do not want the other imputed columns  
 -class # do not want the target column  
 ) %>%   
 mice(m = 1, method = 'norm.nob')

##   
## iter imp variable  
## 1 1 bare\_nuclei  
## 2 1 bare\_nuclei  
## 3 1 bare\_nuclei  
## 4 1 bare\_nuclei  
## 5 1 bare\_nuclei

bn\_stochastic <- complete(imp\_stochastic) %>% select(bare\_nuclei\_stochastic = bare\_nuclei)  
  
bc <- bind\_cols(bc, bn\_deterministic, bn\_stochastic)

## 15.1

*Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?*

Annually our company sets a revenue target and supporting budget. I developed a Linear Programming model (a methodology in the Optimization family) to ensure we hit our revenue target, but minimize sales commission. This provides us the ideal mix of product and time of year for the sales, which helps the company set a reps quota.

X is a vector of products, made up of x1 .. xn.  
M is a vector of months, made up of m1 .. m12.  
T is a vector of sales teams, made up of t1 .. tn.

**Variables**

Vectors X, M, and T combine to give our three dimensional matrix of variables.

[x1, m1, t1] would tell us the amount of sales for product 1 in month 1 (January) with team 1.

**Constraints**

* Sales cannot be negative
* Sales of product x1 get a 10% kicker to incentivize pushing this product.
* Month 3 of a quarter must be at least 40% of quarterly sales.

**Objective Function**

Total revenue must equal $###MM…  
while mimizing commission