SynthesisEvaluation

Sarah Boese

3/1/2020

```
library(ProbBayes)
library(dplyr)
library(ggplot2)
require(gridExtra)
library(reshape)
library(runjags)
library(coda)
library(tidyverse)
library(fastDummies)
```

Here I read in the original and synthetic ACS data.

```
ACSdata_org <- read.csv("ACSdata_org.csv")
ACSdata_syn <- read.csv("ACSdata_syn.csv")
```

Now I create a vector of c_i's. I couldn't figure out a vectorized function so I would love to see an example of how to do so.

```
N<- 10000
c_i<-vector(length=N)
for(j in 1:N){
  vecy<- as.numeric(as.vector(ACSdata_syn[j,c(1,2,3)]))
  c_i[j]=nrow(filter(ACSdata_syn, SEX==vecy[1] & RACE==vecy[2] & MAR==vecy[3]))
}
```

Here I create a dataframe containing values for c_i and T_i. Here I assume T_i is one since all the known variables are the same in both synthetic and orginal dataframes. I'm not sure if that interpretation is correct. I apply vectorized functions from dplyr to calculate K_i and F_i. From there I was able to calculate expected match rate, true match rate and false match rate.

```
ones <- integer(N) +1
analysis_data<-data.frame(c_i, ones)
names(analysis_data)<-c("c_i", "T_i")
s<-nrow(filter(analysis_data, c_i==1))
analysis_data_1 <- analysis_data %>%
  mutate(K_i=if_else(c_i*T_i==1, 1, 0)) %>%
  mutate(F_i=if_else(c_i*(1-T_i)==1, 1, 0)) %>%
  mutate(expect_i=T_i/c_i) %>%
  mutate(true_i=K_i/N) %>%
  mutate(false_i=F_i/s)

expected_match<-sum(analysis_data_1$expect_i)
expected_match
```

```
## [1] 57
```

```r
true_match<-sum(analysis_data_1$true_i)
true_match
```

```
## [1] 4e-04
```

```r
false_match<-sum(analysis_data_1$false_i)
false_match
```

```
## [1] 0
```