Final Project Meeting

Le Thi Thanh Tam _ Pham Thi Anh Thu_Hoang Thuy Duong

1/29/2021

1. Load Data

```
## loading OD_RX.RData
load("E:/Practicum 2/OD_RX.RData")
df.RX = df
## loading OD_Detailing.RData
load("E:/Practicum 2/OD_Detailing.RData")
df.DT = df
# Remove df from workspace
rm(df)
# Drop the time info from the following variables
df.RX$WEEK_END_DATE = as.Date(df.RX$WEEK_END_DATE)
df.DT$WEEK_END_DATE = as.Date(df.DT$WEEK_END_DATE)

# Sort DATA
# Divide Patient insurance type into "HMO/PPO/POS" = "Y" and not = "N"
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "HMO/PPO/POS"] = 1
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "CASH"] = 0
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "INDEMNITY"] = 0
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "MEDICAID"] = 0
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "MEDICARE W/ SECONDARY COVERAGE"]
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "MEDICARE"] = 0
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "NONE"] = 0
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "NOT KNOWN"] = 0
df.RX$PATIENT_INSURANCE_TYPE_DESC[df.RX$PATIENT_INSURANCE_TYPE_DESC == "VA/GOVERNMENT"] = 0

library(dplyr)
library(data.table)
library(mlogit)
## divide into 2 dummy variables "Mild" = 1 and "not " = 0
df.RX$SEVERITY[df.RX$SEVERITY == "MILD"] = 1
df.RX$SEVERITY[df.RX$SEVERITY == "MODERATE"] = 0
df.RX$SEVERITY[df.RX$SEVERITY == "SEVERE"] = 0

# Sort df.RX and df.DT by GEO_PHYSICIAN_KEY and then by WEEK_END_DATE
df.RX = df.RX %>% arrange(GEO_PHYSICIAN_KEY, WEEK_END_DATE)
df.DT = df.DT %>% arrange(GEO_PHYSICIAN_KEY, WEEK_END_DATE)
```

2. Select Drugs

- By comparing between two dataframe (Prescription and Detailing Data), we decided to choose 4 drugs "Actos","Lantus", "Glucophage/Metformin", "Avandia"

```
# sort drugs
setDT(df.RX)[,.(Prescr=100*.N/nrow(df.RX)), by=PRODUCT][order(-rank(Prescr))]
```

```
##                   PRODUCT     Prescr
##  1:  Glucophage/Metformin 26.3970342
##  2:                Actos 12.9017726
##  3:         Sulfonylureas 11.8520080
##  4:        Other Insulins 11.2287619
##  5:                Lantus 10.2426444
##  6:         Glucophage XR  5.9679532
##  7:                Amaryl  5.8687629
##  8:   Glucovance/Metaglip  4.4573667
##  9:               Avandia  4.0668047
## 10:             Avandamet  3.1087912
## 11: Pran/Star/Prec/Glyset  1.7536029
## 12:             Avandaryl  0.8505573
## 13:              Fortamet  0.6926793
## 14:                Apidra  0.4302382
## 15:                Symlin  0.1810224
```

```
setDT(df.DT)[,.(Prescr=100*.N/nrow(df.RX)), by=PRODUCT][order(-rank(Prescr))]
```

```
##                   PRODUCT      Prescr
##  1:        Other Insulins 5.194681744
##  2:               Avandia 5.081439417
##  3:                 Actos 4.864047215
##  4:                Lantus 4.739232680
##  5:             Avandamet 1.729631880
##  6:                Apidra 1.663918267
##  7:              Fortamet 0.719130101
##  8:                Symlin 0.548440025
##  9:             Avandaryl 0.481486533
## 10: Pran/Star/Prec/Glyset 0.334354167
## 11:                Amaryl 0.043809075
## 12:         Glucophage XR 0.024384297
## 13:         Sulfonylureas 0.020664658
## 14:   Glucovance/Metaglip 0.019424779
## 15:  Glucophage/Metformin 0.007439277
```

```
# Select the drugs
vec.DrugSelected = c("Actos","Lantus", "Glucophage/Metformin", "Avandia")
df.RX = df.RX %>% filter(PRODUCT %in% vec.DrugSelected)
df.DT = df.DT %>% filter(PRODUCT %in% vec.DrugSelected)

## Construct a vector containing drug names from df.RX$PRODUCT (df.RX$BRAND has more names)
vec.DrugName = sort(unique(df.RX$PRODUCT))
## Compute # of drugs
N_Drug = length(vec.DrugName)
```

3. Select Time Window

- Choose only the last 40 weeks and exclude the last week because it includes only one date

```
# Filter out the last week of the observations
## Construct a vector containing unique week end dates
vec.WEDate = sort(unique(df.RX$WEEK_END_DATE))
df.RX = df.RX %>%
  filter(WEEK_END_DATE %in% tail(vec.WEDate, n=41))
df.RX = df.RX[df.RX$WEEK_END_DATE!="2008-07-04",]
df.DT = df.DT %>%
  filter(WEEK_END_DATE %in% tail(vec.WEDate, n=41))
df.DT = df.DT[df.DT$WEEK_END_DATE!="2008-07-04",]
```

4. Select Physicians

- Select Physicians who has made at least one prescription every month and has switched brand (see whether the doctor has changed the prescription or not)

```
# Select physicians and Brand
## count the number of physicians by months
df.A = df.RX[, .N, by = .(GEO_PHYSICIAN_KEY, MONTH)]
## count the number of months by physician
df.B = df.A[,.N, by = GEO_PHYSICIAN_KEY]
## Now sort the physicians with the highest scores
df.C = df.B[N==max(df.B$N)]
## Create the new data frame for those physicians
df.D = filter(df.RX, GEO_PHYSICIAN_KEY %in% df.C$GEO_PHYSICIAN_KEY)
## selecting the physicians with changes in prescription
df.final = df.D[SWITCHED_BRAND != "UNKNOWN",.N,by=GEO_PHYSICIAN_KEY]

# Construct a vector containing unique physician idf from df.DT
vec.PhyID = sort(unique(df.D$GEO_PHYSICIAN_KEY))
# Filter out physicians from RX dataframe who did NOT receive any detailing form the selected drugs
df.RX = df.RX %>% filter(GEO_PHYSICIAN_KEY %in% vec.PhyID)
vec.PhyID = sort(unique(df.RX$GEO_PHYSICIAN_KEY)) # 379
df.DT = df.DT %>% filter(GEO_PHYSICIAN_KEY %in% vec.PhyID)
vec.PhyID = sort(unique(df.DT$GEO_PHYSICIAN_KEY)) # 379
df.RX = df.RX %>% filter(GEO_PHYSICIAN_KEY %in% vec.PhyID)
# Now both have the same group of 379 physicians
# Compute # of physicians
N_Phy = length(vec.PhyID) # 379

## Construct a vector containing unique week end dates
vec.WEDate = sort(unique(df.RX$WEEK_END_DATE))
## Compute # of weeks
N_Week = length(vec.WEDate)
```

```
# Construct a matrix (N_Phy*N_Week) by N_Drug to save GoodWill variable (= cumulative # of details)
mat.GoodWill = matrix(0L, N_Phy*N_Week, N_Drug)
# Construct a vector N_Phy by 1 to save the total number of detailings
```

```
vec.N_Detail = rep(0L, N_Phy)
bloc = 1
eloc = N_Week
df.DTiwj = df.DT %>% select(GEO_PHYSICIAN_KEY, WEEK_END_DATE, PRODUCT)
for(i in 1:N_Phy) {
  Det_i = df.DTiwj %>% filter(GEO_PHYSICIAN_KEY == vec.PhyID[i])
  Det_iw = matrix(0L, N_Week, N_Drug)
  N_Obsi = nrow(Det_i)
  vec.N_Detail[i] = N_Obsi
  for (n_i in 1:N_Obsi) {
    dt_x = which(vec.WEDate == Det_i$WEEK_END_DATE[n_i])
    dt_y = which(vec.DrugName == Det_i$PRODUCT[n_i])
    Det_iw[dt_x, dt_y] = Det_iw[dt_x, dt_y] + 1
  }
  GoodWill_iw = apply(Det_iw, 2, cumsum)
  mat.GoodWill[bloc:eloc,] = GoodWill_iw
  bloc = bloc + N_Week
  eloc = eloc + N_Week
}
rm(df.DTiwj,Det_i, Det_iw, N_Obsi, dt_x, dt_y, GoodWill_iw, i, n_i, bloc, eloc)
# Construct a Goodwill variable for each RX observation
N_ObsRX = nrow(df.RX)
mat.GoodWillRX = matrix(0L, N_ObsRX, N_Drug)

for (n in 1:N_ObsRX) {
  phy_id = which(vec.PhyID == df.RX$GEO_PHYSICIAN_KEY[n])
  rx_week = which(vec.WEDate == df.RX$WEEK_END_DATE[n])
  gw_loc = (phy_id - 1)*N_Week + rx_week
  mat.GoodWillRX[n, ] = mat.GoodWill[gw_loc, ]
}
rm(mat.GoodWill, n, phy_id, rx_week, gw_loc)

df.GoodWill = as.data.frame(mat.GoodWillRX)
colnames(df.GoodWill) = c("Goodwill_GL","Goodwill_AC", "Goodwill_LT", "Goodwill_AV")
df.GoodWill$GEO_PHYSICIAN_KEY = df.RX$GEO_PHYSICIAN_KEY

# Create a dataframe for mlogit in a "long" format
vec.ObsIDRX = rep(1:N_ObsRX, each = N_Drug)
vec.PhyIDRX = rep(0L, N_ObsRX*N_Drug)
vec.ChIDRX = rep(0L, N_ObsRX*N_Drug)
vec.NewDrugName = rep(c("GL","AC","LT","AV"), N_ObsRX)
vec.ChoiceRX = rep(0L, N_ObsRX*N_Drug)
vec.Goodwill = rep(0L, N_ObsRX*N_Drug)
vec.Severity = rep(0L, N_ObsRX*N_Drug)
vec.Age = rep(0L, N_ObsRX*N_Drug)
vec.Insurance = rep(0L, N_ObsRX*N_Drug)

bloc = 1
eloc = 0
for (i in 1:N_Phy) {
  Phy_i = vec.PhyID[i]
  RX_i = df.RX %>% filter(GEO_PHYSICIAN_KEY == Phy_i)
  GW_i = mat.GoodWillRX[df.RX$GEO_PHYSICIAN_KEY == Phy_i,]
```

```
  N_i = nrow(RX_i)
  for (j in 1:N_i){
    eloc = eloc + N_Drug
    x1 = rep(i, N_Drug) # Physician ID
    vec.PhyIDRX[bloc:eloc] = x1
    x2 = rep(j, N_Drug) # Choice occasion ID
    vec.ChIDRX[bloc:eloc] = x2
    x3 = rep(0L, N_Drug)  # Choice
    x3[which(vec.DrugName == RX_i$PRODUCT[j])] = 1
    vec.ChoiceRX[bloc:eloc] = x3
    if (N_i == 1) {
      x4 = as.vector(as.double(GW_i[1:N_Drug]))
    } else {
      x4 = as.vector(as.double(GW_i[j, 1:N_Drug]))
    }
    vec.Goodwill[bloc:eloc] = x4
    x5 = rep(RX_i$SEVERITY[j], N_Drug)
    vec.Severity[bloc:eloc] = x5
    x6 = rep(RX_i$PATIENT_AGE[j], N_Drug)
    vec.Age[bloc:eloc] = x6
    x7 = rep(RX_i$PATIENT_INSURANCE_TYPE_DESC[j], N_Drug)
    vec.Insurance[bloc:eloc] = x7
    bloc = bloc + N_Drug
  }
}
```

5. Results

```
# data frame with vectors
df.Choice_ODRX = data.frame(obs = vec.ObsIDRX, drug = vec.NewDrugName,
                            choice = vec.ChoiceRX, goodwill = vec.Goodwill, severity = vec.Severity,
                            age = vec.Age, insurance = vec.Insurance)
df.ODRX2 = mlogit.data(df.Choice_ODRX, choice="choice", shape="long", alt.var = "drug", chid.var = "obs
# Model:
out = mlogit(choice ~ goodwill|severity+age+insurance, data=df.ODRX2)
summary(out)
```

```
##
## Call:
## mlogit(formula = choice ~ goodwill | severity + age + insurance,
##     data = df.ODRX2, method = "nr")
##
## Frequencies of alternatives:choice
##       AC       AV       GL       LT
## 0.043522 0.193729 0.247738 0.515010
##
## nr method
## 6 iterations, 0h:0m:6s
## g'(-H)^-1g = 1.16E-05
## successive function values within tolerance limits
```

```
##
## Coefficients :
##                   Estimate Std. Error z-value  Pr(>|z|)
## (Intercept):AV  2.3751336  0.1846379 12.8637 < 2.2e-16 ***
## (Intercept):GL  1.4718341  0.1833871  8.0258 1.110e-15 ***
## (Intercept):LT  2.8011057  0.1756790 15.9445 < 2.2e-16 ***
## goodwill        0.0552069  0.0025298 21.8228 < 2.2e-16 ***
## severity1:AV   -0.9138617  0.0954898 -9.5703 < 2.2e-16 ***
## severity1:GL    0.2024248  0.0844670  2.3965  0.016553 *
## severity1:LT    0.5579519  0.0810386  6.8850 5.778e-12 ***
## age:AV         -0.0122190  0.0026692 -4.5777 4.701e-06 ***
## age:GL          0.0013483  0.0026371  0.5113  0.609159
## age:LT         -0.0073501  0.0025294 -2.9059  0.003662 **
## insurance1:AV  -0.1511549  0.0688515 -2.1954  0.028137 *
## insurance1:GL   0.1867231  0.0675505  2.7642  0.005706 **
## insurance1:LT   0.2644010  0.0649058  4.0736 4.629e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-Likelihood: -31754
## McFadden R^2:  0.024702
## Likelihood ratio test : chisq = 1608.5 (p.value = < 2.22e-16)
```