ETM538 HW 4

Jordan Hilton

February 3, 2019

Initial Setup

Here's a modified version of the provided data load:

```
mem_claims             <- read.csv("Claims_Y1.csv")
mem_days               <- read.csv("DayInHospital_Y2.csv")
mem_info               <- read.csv("Members_Y1.csv")
risk_model             <- read.csv("risk_model_1.csv")
```

Here's the processing code:

```
colnames(mem_days) <- c("MemberID", "Days")
mem_to_risk <- merge(mem_days, risk_model, by="Days")
claims_to_risk <- merge(mem_claims, mem_to_risk, by = "MemberID")
```

Here's the code to calculate the a priori probabilities:

```
n_claims <- length(claims_to_risk[,1])     # note that we have to pick a column.

risks <- as.data.frame(as.character(claims_to_risk$RiskLevel))

riskl <- as.list(risks)

risk_count <- aggregate(risks, riskl, FUN=length)

colnames(risk_count) <- c("RiskLevel", "RiskCount")

a_priori <- risk_count

a_priori$Total <- n_claims

a_priori$Prob <- a_priori$RiskCount / n_claims

colnames(a_priori) <- c("RiskLevel", "RiskCount","Total", "Prob") ##Modified this from original code

write.csv(a_priori, file = "out_a_priori.csv", row.names = FALSE)
```

The calculation for the condition on Charlson index:

```
on_charlson <- data.frame(as.character(claims_to_risk$RiskLevel),
                             as.character(claims_to_risk$CharlsonIndex))

colnames(on_charlson) <- c("RiskLevel", "Charlson")

df_char <- as.data.frame(as.character(on_charlson$Charlson))

colnames(df_char) <- c("Charlson")

l_char <- on_charlson$Charlson
```

```r
l_risk <- on_charlson$RiskLevel

count_char <- aggregate(df_char, by=list(l_char, l_risk), FUN=length)

colnames(count_char) <- c("Charlson", "RiskLevel", "Count")

# check the total to make sure everything is present and accounted for.

n_chars <- sum(count_char$Count)

n_missing <- n_claims - n_chars

print(paste("A Posteriori Charlson -- ", toString(n_missing), " are missing."))
```

```
## [1] "A Posteriori Charlson --  0  are missing."
```

```r
post_char <- merge(count_char, risk_count, by="RiskLevel")

post_char$Prob <- post_char$Count / post_char$RiskCount

post_char$Label <- paste(post_char$Charlson, post_char$RiskLevel, sep="|")

# reorder the columns

post_char <- post_char[c("Label", "Charlson", "RiskLevel", "Count", "RiskCount", "Prob")]
```

The calculation for the condition on length of (first) stay:

```r
# extract length of stay as a vector

new_stay <- as.character(claims_to_risk$LengthOfStay)

# assign default value to missing columns

new_stay[new_stay == ''] <- '0 days'

on_stay <- data.frame(as.character(claims_to_risk$RiskLevel),
                          as.character(new_stay))

colnames(on_stay) <- c("RiskLevel", "Stay")

df_stay <- as.data.frame(as.character(on_stay$Stay))

colnames(df_stay) <- c("Stay")

l_stay <- on_stay$Stay

l_risk <- on_stay$RiskLevel

count_stay <- aggregate(df_stay, by=list(l_stay, l_risk), FUN=length)

colnames(count_stay) <- c("Stay", "RiskLevel", "Count")

# check the total to make sure everything is present and accounted for.
```

```r
n_stays <- sum(count_stay$Count)

n_missing <- n_claims - n_stays

print(paste("A Posteriori stay -- ", toString(n_missing), " are missing."))
```

```
## [1] "A Posteriori stay --  0  are missing."
```

```r
post_stay <- merge(count_stay, risk_count, by="RiskLevel")

post_stay$Prob <- post_stay$Count / post_stay$RiskCount

post_stay$Label <- paste(post_stay$Stay, post_stay$RiskLevel, sep="|")

# reorder the columns

post_stay <- post_stay[c("Label", "Stay", "RiskLevel", "Count", "RiskCount", "Prob")]
```

The calculation for the condition on primary condition group:

```r
on_pcg <- data.frame(as.character(claims_to_risk$RiskLevel),
                      as.character(claims_to_risk$PrimaryConditionGroup))

colnames(on_pcg) <- c("RiskLevel", "pcg")

df_pcg <- as.data.frame(as.character(on_pcg$pcg))

colnames(df_pcg) <- c("pcg")

l_pcg <- on_pcg$pcg

l_risk <- on_pcg$RiskLevel

count_pcg <- aggregate(df_pcg, by=list(l_pcg, l_risk), FUN=length)

colnames(count_pcg) <- c("pcg", "RiskLevel", "Count")

# check the total to make sure everything is present and accounted for.

n_pcgs <- sum(count_pcg$Count)

n_missing <- n_claims - n_pcgs

print(paste("A Posteriori pcg -- ", toString(n_missing), " are missing."))
```

```
## [1] "A Posteriori pcg --  0  are missing."
```

```r
post_pcg <- merge(count_pcg, risk_count, by="RiskLevel")

post_pcg$Prob <- post_pcg$Count / post_pcg$RiskCount

post_pcg$Label <- paste(post_pcg$pcg, post_pcg$RiskLevel, sep="|")

# reorder the columns
```

```r
post_pcg <- post_pcg[c("Label", "pcg", "RiskLevel", "Count", "RiskCount", "Prob")]
```

Writing out to csvs:

```r
write.csv(post_char, file="out_charlson.csv")
write.csv(post_stay, file="out_stay.csv")
write.csv(post_pcg, file="out_pcg.csv")
```

Question responses:

1. Dataset explanation

Explain how the R data pipeline works, by describing the role of each of the following R data sets:

- mem_to_risk - Combines the “Days in Hospital Y2” and “Risk Level” tables to provide information about the risk level of each member ID.
- claims_to_risk - Combines the “Claims Y1” table with the mem-to-risk table, so that now we have a single table with information about the claims in Y1 and the risk level in Y2 for each member.
- risk_count - A simple count of how many rows there are in claims-to-risk for each different risk level.
- a_priori - Contains the a priori probability for each risk level calculated from claims-to-risk.
- on_charlson - A subset of the claims-to-risk table containing only the Charlson index and risk level fields
- count_char - Counts the number of claims for each possible combination of risk level and Charlson index in claims-to-risk.
- post_char - Gets the total number of risk counts for each risk level, then uses the count from count_char to calculate the a posteriori probability

2. Spreadsheet operation

Charlson index seems to be the most influential variable- when you hold other variables constant and change Charlson index, you get the biggest change in predicted outcome.

3. Label column

The label column is produced in the R code by this line, which pastes together the risk level and the independent variable we’re looking at:

```r
post_pcg$Label <- paste(post_pcg$pcg, post_pcg$RiskLevel, sep="|")
```

It’s used by the Excel sheet as the key for the vlookup function to reference the chosen combination of independent variable and risk level for the naive Bayes calculation.

4. Missing values for long hospital stays

From the lecture, the way we shoudl do this is to add 1 count for every possible combination of independent variable and risk level to our probably calculator.

5. Unlikely B

First, here are some example combinations that predict risk levels A, C, and D:

- A
- C
- D Charlson:3-4 Stay:4- 8 weeks PCG:SEPSIS predicts for D at 67.2%

Looking at the probability tables, what’s happening is just that because that the counts for risk level B are lower than for other categories- presumably because the odds of spending exactly 2 days in the hospital are lower than the odds of spending either 1 day or between 3 and 5 days.

6. Changed Risk buckets

7. Adding a variable