QBS181 HW2

Mark Taylor

Setup

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(odbc)
library(sqldf)
```

```
## Loading required package: gsubfn

## Loading required package: proto

## Loading required package: RSQLite
```

```
library(stringr)
```

1

```
# Left join Phonecall with PhoneCall_Encounter to get Encountercodes
# (aliases as EnrollmentGroup so we need not rename)
Phonecall_Encounter <- dbGetQuery(con, "SELECT CustomerId,
                                        EncounterCode as EnrollmentGroup
                                    FROM Phonecall_Encounter")

# Assign EnrollmentGroup based on the current EnrollmentGroup value
# (corresponding to the code)
Phonecall_Encounter$EnrollmentGroup <-
  ifelse(Phonecall_Encounter$EnrollmentGroup == 125060000, "Clinical Alert",
  ifelse(Phonecall_Encounter$EnrollmentGroup == 125060001, "Health Coaching",
  ifelse(Phonecall_Encounter$EnrollmentGroup == 125060002, "Technical Question",
  ifelse(Phonecall_Encounter$EnrollmentGroup == 125060003, "Administrative",
  ifelse(Phonecall_Encounter$EnrollmentGroup == 125060004, "Other",
  ifelse(Phonecall_Encounter$EnrollmentGroup == 125060004, "Lack of engagement",
                                                           "NULL")
  )))))

sample_n(Phonecall_Encounter, 10)
```

2

```
# Return # of records for each EnrollmentGroup
sqldf("SELECT EnrollmentGroup, COUNT(*) recordNumber
FROM Phonecall_Encounter
GROUP BY EnrollmentGroup")
```

3

```
# import CallDuration
CallDuration <- dbGetQuery(con, "SELECT * FROM CallDuration")

# Merge with PhoneCall_Encounter
EncounterXDuration <- sqldf("SELECT PCE.*, CD.CallType, CD.CallDuration, CD.CallOutcome
          FROM CallDuration as CD
          LEFT JOIN Phonecall_Encounter as PCE
          ON CD.tri_CustomerIDEntityReference = PCE.CustomerId")

sample_n(EncounterXDuration, 10)
```

4

```
# Replace numbers with call types
EncounterXDuration$CallType <-
  ifelse(EncounterXDuration$CallType == 1, "Inbound", "Outbound")
```

```r
# Report count by call type
sqldf("SELECT CallType, COUNT(*) recordNumber
FROM EncounterXDuration
GROUP BY CallType")

# Replace numbers with call outcomes
EncounterXDuration$CallOutcome <-
  ifelse(EncounterXDuration$CallOutcome == 1, "No response",
  ifelse(EncounterXDuration$CallOutcome == 2, "Left voice mail", "successful"))

# Report count by call outcome
sqldf("SELECT CallOutcome, COUNT(*) recordNumber
FROM EncounterXDuration
GROUP BY CallOutcome")

# Report sum of call duration by EnrollmentGroup
sqldf("SELECT EnrollmentGroup, SUM(CallDuration)
FROM EncounterXDuration
GROUP BY EnrollmentGroup")
```

5

```r
# merge Conditions and Demographics as we import
DemoXCond <- dbGetQuery(con, "SELECT D.*, C.tri_name
FROM Conditions as C
LEFT JOIN Demographics as D
ON C.tri_patientid = D.contactid")

# import Text
Text <- dbGetQuery(con, "SELECT * FROM Text")

# merge Conditions and Demographics (already merged) with Text
DemoXCondXText <- sqldf("SELECT DC.*, T.SenderName, T.TextSentDate
      FROM Text as T
      LEFT JOIN DemoXCond as DC
      ON T.tri_contactId = DC.contactid")

# add aggregated Week column
DemoXCondXText <- DemoXCondXText %>% group_by(Week=floor_date(TextSentDate, "7 days"))

# count of texts by sender type for each week
WeeklyTextsBySender <- DemoXCondXText %>% count(Week, SenderName) %>% ungroup()

sample_n(WeeklyTextsBySender, 10)
```

6

```r
# count of texts by condition for each week
WeeklyTextsByCondition <- DemoXCondXText %>% count(Week, tri_name) %>% ungroup()
```

```r
sample_n(WeeklyTextsByCondition, 10)
```