Grade Retention Algorithm

author: “Aaron D. Schroeder”
date: “10/23/2014”
description: Takes yearly student record data in long-format, transforms and casts the data to wide-format, then finds number of times each grade was attended (times attended > 1 = retention).

1. Add Libraries and Import Data

Sample data includes client id, grade year (e.g. KG, 1, 2), and entry date (first day of school)

```
library(reshape2)
retentionData <- read.csv("GradeRetentionData.csv")
```

```
##    clientid gradeyear entrydate
## 1   ClientA         1  8/1/2010
## 2   ClientA         1  8/1/2011
## 3   ClientA         2  8/1/2012
## 4   ClientC         4  8/1/2008
## 5   ClientC         5  8/1/2009
## 6   ClientC         5  8/1/2010
## 7   ClientC         6  8/1/2011
## 8   ClientB        PK  8/1/2003
## 9   ClientB        KG  8/1/2004
## 10  ClientB        KG  8/1/2004
## 11  ClientB         1  8/1/2005
```

2. Transform column data where necessary and build new data frame

Reduce date to just year

```
yearattend <- format(as.Date(retentionData$entrydate, "%m/%d/%Y"), "%Y")
```

Custom function gradeNum is added to convert text grade number (e.g. “KG”) to numeric (e.g. “0”) to help with sorting. The function is applied using mapply, a vectorized approach (as opposed to looping).

```
gradeNum <- function(x){ switch(x, "KG" = 0, "K"  = 0, "PK" = -1, "P"  = -1, x) }
gradeyear <- mapply(gradeNum, x = as.character(retentionData$gradeyear))
```

Construct new data frame (here we use same name to replace old data frame).

```
retentionData <- data.frame(clientid = retentionData$clientid, gradeyear, yearattend)
```

```
##    clientid gradeyear yearattend
## 1   ClientA         1       2010
## 2   ClientA         1       2011
## 3   ClientA         2       2012
## 4   ClientC         4       2008
## 5   ClientC         5       2009
## 6   ClientC         5       2010
## 7   ClientC         6       2011
## 8   ClientB        -1       2003
## 9   ClientB         0       2004
## 10  ClientB         0       2004
## 11  ClientB         1       2005
```

3. Eliminate duplicate records

Two records with identical clientid, gradeyear, and yearattend are duplicates, not a grade retention. Notice record 10 was eliminated as a duplicate.

```
retentionData <- unique(retentionData[,c("clientid","gradeyear","yearattend")])
```

```
##    clientid gradeyear yearattend
## 1   ClientA         1       2010
## 2   ClientA         1       2011
## 3   ClientA         2       2012
## 4   ClientC         4       2008
## 5   ClientC         5       2009
## 6   ClientC         5       2010
## 7   ClientC         6       2011
## 8   ClientB        -1       2003
## 9   ClientB         0       2004
## 11  ClientB         1       2005
```

4. Cast to wide data frame

Create a single row for each clientid + gradeyear combination. Order by clientid and gradeyear.

```
castDF <- dcast(retentionData, clientid + gradeyear ~ yearattend, value.var="yearattend", fun.aggregate=length)
castDF <- castDF[order(castDF$clientid, castDF$gradeyear),]
```

```
##   clientid gradeyear 2003 2004 2005 2008 2009 2010 2011 2012
## 1  ClientA         1    0    0    0    0    0    1    1    0
## 2  ClientA         2    0    0    0    0    0    0    0    1
## 3  ClientB        -1    1    0    0    0    0    0    0    0
## 4  ClientB         0    0    1    0    0    0    0    0    0
## 5  ClientB         1    0    0    1    0    0    0    0    0
## 6  ClientC         4    0    0    0    1    0    0    0    0
## 7  ClientC         5    0    0    0    0    1    1    0    0
## 8  ClientC         6    0    0    0    0    0    0    1    0
```

5. Create summary column

Create row sums column and add to new wide data frame

```
rsums <- rowSums(castDF[c(3:ncol(castDF))])
castDF$times_attended <- rsums
```

```
##   clientid gradeyear 2003 2004 2005 2008 2009 2010 2011 2012 times_attended
## 1  ClientA         1    0    0    0    0    0    1    1    0              2
## 2  ClientA         2    0    0    0    0    0    0    0    1              1
## 3  ClientB        -1    1    0    0    0    0    0    0    0              1
## 4  ClientB         0    0    1    0    0    0    0    0    0              1
## 5  ClientB         1    0    0    1    0    0    0    0    0              1
## 6  ClientC         4    0    0    0    1    0    0    0    0              1
## 7  ClientC         5    0    0    0    0    1    1    0    0              2
## 8  ClientC         6    0    0    0    0    0    0    1    0              1
```