Marketing Analytics Midterm Project

Wen Zhang, Troy Zhen, Levent Kayın

Table of Contents

[Introduction 2](#_Toc510081685)

[Brief Background on Pernaloga 2](#_Toc510081686)

[Project Description and Scope 2](#_Toc510081687)

[Data Available 2](#_Toc510081688)

[Insights and Data preparation 3](#_Toc510081689)

[Diaper brands and their share in sales 3](#_Toc510081690)

[Appendix 4](#_Toc510081691)

[Data Dictionary 4](#_Toc510081692)

[Pyspark Code (Modeling) 4](#_Toc510081693)

[R Code (Data Pre and Post Processing) 5](#_Toc510081694)

# Introduction

## Brief Background on Pernaloga

Our Project is based on “Pernaloga” a leading süpermarket chain in Brasil. They own over 400 locations, sell over 10 thousand products in over 400 categories. Pernaloga has partnerships with its suppliers to fund their promotions.

30% of Pernaloga’s sales come from promotions. Pernaloga mostly uses in store promotions. However, they want to start offering personalized promotions to their customers to increase sales from products that are promoted. They would like to use analytics and machine learning to target customers and increase sales. Even though there is a cost associated with the necessary analysis the benefits usually outweigh the costs.

## Project Description and Scope

Our project is to promote Huggies’ Diapers. Kimberly-Clark plans to promote Huggies to customers who are buying other brands. However, Pernaloga already has a private label diaper brand and wants to exclude customers buying their private label from being targeted.

This is a good example of a real world machine learning project where we need to discuss the business impact of our model and tailor our algorithms to specific business needs.

## Data Available

We have data available on transactions and products. The transactions data table lists transaction history from 2016 and 2017 for roughly 8,000 customers. The data includes customer id, price paid, item purchased, discount on items and location. For more information see appendix.

The products table expands upon the products sold. We have information on the category, brand, subcategory and weather the brand is a private label or not. For more information see Appendix.

# Insights and Data preparation

## Diaper brands and their share in sales

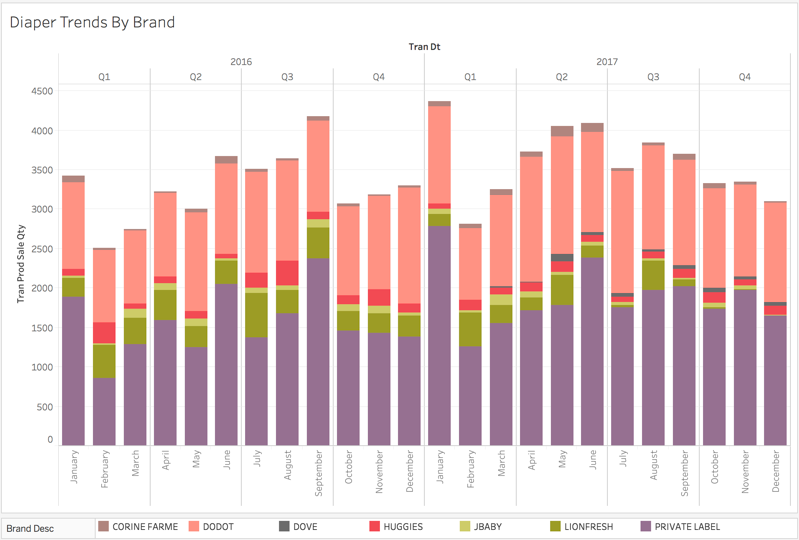
As seen in the chart the Pernaloga’s private label and Dodot dominate the market leaving Huggies with a relatively small market share (Pernaloga being the market). We also see that the market distribution of the Private Label and Dodot stays relatively stable and the other brands competing for the remainder.

Figure 1 Diaper sales by brand over time

It is also worth noting that Lionfresh started losing market share once Dove vas introduced in Q2 2017.

## Seasonality in Diaper Sales

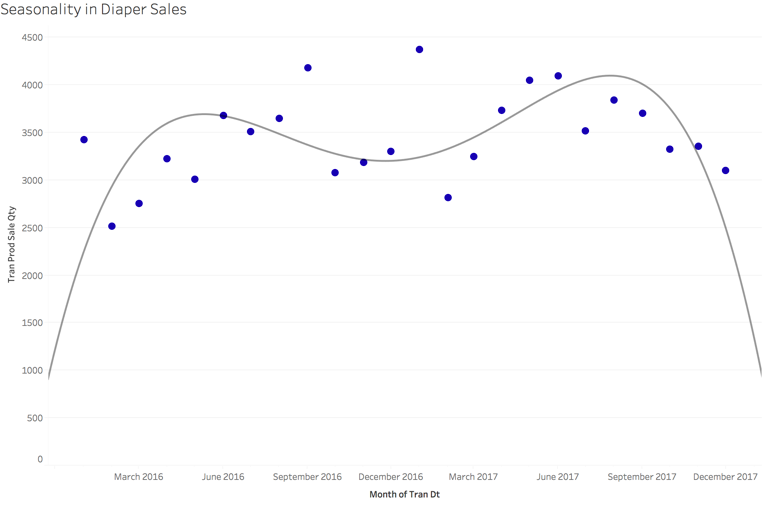
As seen in the chart diaper sales are seasonal an peak around april-june.

Figure 2 Seasonality in diaper sales

However, we believe that this is correlated with the human reproductive activities rather than customer behavior. Because if a person or family need diapers their behavior is not seasonal. They will buy diapers until their toddler is trained to use toilets. For this reason we decided not to consider seasonality in our analysis.

However this information might be

# Appendix

## Data Dictionary

1. transaction\_table.csv contains transaction history in 2016 and 2017 for close to 8,000 customers
   * cust\_id – Customer ID
   * tran\_id – Transaction ID
   * tran\_dt – Transaction Date
   * store\_id – Store ID
   * prod\_id – Product ID
   * prod\_unit – Product unit of measure: CT for count and KG for kilograms
   * prod\_unit\_price – Unit price of the product
   * tran\_prod\_sale\_qty – Quantity/units of the product in the transaction
   * tran\_prod\_sale\_amt – Sales amount for the product before discounts in the transaction
   * tran\_prod\_discount\_amt – Total amount of discounts applied to the product in the  transaction
   * tran\_prod\_offer\_cts – Total number of offers on the product resulting in the total  amount of discounts in the transaction
   * tran\_prod\_paid\_amt – Amount paid for the product after discounts are applied in the  transaction
2. product\_table.csv contains the product to subcategory and category mapping and descriptions  for about 11,000 products
   * prod\_id – Product ID
   * subcategory\_id – Subcategory ID
   * category\_id – Category ID
   * sub\_category\_desc – Subcategory name (in Portuguese)

* category\_desc – Category name (in Portuguese)
* category\_desc\_eng – Category name (in English)
* brand\_desc – Brand of the product, including NO LABEL and PRIVATE LABEL

## Pyspark Code (Modeling)

#reference: the bulk of this code is adopted from Prof. Panos' Machine Learning 2 course teaching materials

from pyspark.ml.recommendation import ALS

from pyspark.sql import Row

import pandas as pd

df = spark.read.load("data/final1.csv", format = 'csv', header = 'true')

df2 = df.withColumn("tot", df["tot"].cast("double")).withColumn("cust\_id", df["cust\_id"].cast("integer")).withColumn("prod\_id", df["prod\_id"].cast("integer"))

# Train an Alternating Least Squares model

als = ALS()\

.setMaxIter(5)\

.setRegParam(0.01)\

.setUserCol("cust\_id")\

.setItemCol("prod\_id")\

.setRatingCol("tot")

alsModel = als.fit(df2)

# Generate recommendations (users and then items) - output top 10 recommendations for each user or movie.

out = alsModel.recommendForAllItems(1000)\

.selectExpr("prod\_id", "explode(recommendations)")

#install pandas on AWS

#sudo pip install pandas

#write the recommendations into csv files and save it on AWS

out.toPandas().to\_csv('mycsv.csv')

#save the file on s3

#aws s3 cp mycsv.csv s3://mybucket

#download to local and proceed to R-studio for further manipulation

## R Code (Data Pre and Post Processing)

library(data.table)

library(dplyr)

library(rpart)

library(tibble)

library(tidyr)

library(lubridate)

setwd('C:/Users/qw004/Desktop/MKT 680/mid-term project/Pernalonga')

products <- fread('product\_table.csv')

trans <- fread('transaction\_table.csv')

#Function for standardization

standardize <- function(x) {

x <- (x-mean(x))/sd(x)

return(x)

}

#Find out the kinds of products that Huggies produces

huggies <- products %>% filter(brand\_desc == 'HUGGIES') %>% group\_by(subcategory\_id, category\_id) %>% summarise(num = n())

others <- products %>% filter((subcategory\_id == 93367 & category\_id == 95720) | (subcategory\_id == 93431 & category\_id == 95732), brand\_desc != 'HUGGIES')

all <- products %>% filter((subcategory\_id == 93367 & category\_id == 95720) | (subcategory\_id == 93431 & category\_id == 95732))

#Find all relevant transactions

trans\_new <- trans %>% filter(prod\_id %in% all$prod\_id)

#Get all relevant features for clustering

trans\_clust <- trans\_new %>% select(starts\_with('tran\_prod'))

clust <- kmeans(trans\_clust, centers = 6, iter.max = 20)

trans\_fin <- trans\_new %>% cbind(clust$cluster)

names(trans\_fin)[13] <- 'cluster'

#Customer metrics

cust\_val <- trans\_new %>% group\_by(cust\_id) %>% summarise(val = sum(tran\_prod\_paid\_amt))

cust\_val

cust\_amt <- trans\_new %>% group\_by(cust\_id) %>% summarise(amt = sum(tran\_prod\_sale\_qty))

cust\_tier <- merge(trans\_new, cust\_amt, by = 'cust\_id') %>%

merge(cust\_val, by = 'cust\_id') %>%

group\_by(cust\_id) %>%

mutate(tier = val/amt)

#cluster the customers based on their total value and average price of products they buy

cluster1\_cust <- cust\_tier %>% group\_by(cust\_id) %>%

summarise(amt = mean(amt), val = mean(val), tier = mean(tier))

cluster1\_cust <- apply(cluster1\_cust[,-1], 2, FUN = standardize) %>% cbind(cluster1\_cust[,1])

num <- 6

cluster1 <- cluster1\_cust %>% kmeans(centers = num, iter.max = 20)

##############################################################################

remove(trans\_new)

##############################################################################

#Merge the cluster number with the transaction data

trans\_fin1 <- cbind(cluster1\_cust, cluster1$cluster) %>% merge(trans\_fin, by = 'cust\_id')

names(trans\_fin1)[5] <- 'cust\_cluster'

#Compare all the clusters

this <- trans\_fin1 %>% summary()

this <- this[4,1:17]%>% as.data.frame()

for(i in 1:num){

that <- trans\_fin1 %>% filter(cust\_cluster == i) %>% summary()

that <- that[4,1:17] %>% as.data.frame()

this <- that[,1] %>% cbind(this)

names(this)[1] <- i

}

trans\_fin1$cust\_cluster <- trans\_fin1$cust\_cluster %>% as.factor()

trans\_fin1$store\_id <- trans\_fin1$store\_id %>% as.factor()

#Building logistic regression

attach(trans\_fin1) %>% suppressMessages()

LR <- lm(tran\_prod\_paid\_amt ~ cust\_cluster + amt + tier + val, data = trans\_fin1)

pred <- predict(LR, trans\_fin1)

LR %>% summary()

pred %>% summary()

for(i in unique(trans\_fin1$cust\_cluster)){

LR1 <- lm(tran\_prod\_paid\_amt ~ tran\_prod\_discount\_amt, data = trans\_fin1%>%filter(cust\_cluster == i))

cat('The cluster ', i, ' has a summary as follows:

')

print(LR1 %>% summary())

}

#From there we basically established that discount has everything to do with the customer's final paid amount.

####################################################################################

remove(LR)

remove(LR1)

####################################################################################

#Further manipulation on customer churn

churn <- function(x) {

y <- unique(x)

if(length(y)>1){

return(TRUE)

} else{

return(FALSE)

}

}

with\_discount <- trans\_fin1 %>% filter(tran\_prod\_discount\_amt < 0)

this <- trans\_fin1 %>% group\_by(cust\_id) %>% summarise(funs = churn(prod\_id))

trans\_fin2 <- trans\_fin1 %>% merge(this, by = 'cust\_id')

suppressMessages(attach(trans\_fin2))

lm(funs ~ tran\_prod\_discount\_amt, data = trans\_fin2) %>% summary()

#Label all the transactions for the churn

trans\_fin3 <- data.frame()

for(i in this$cust\_id){

m <- trans\_fin2 %>% filter(cust\_id == i) %>% arrange(tran\_dt)

m[1,18] = 0

if(nrow(m) > 1){

for(k in 1:(nrow(m)-1)){

if(m[k,9] != m[k+1,9]){

m[k+1,18] = 1

}else{

m[k+1,18] = 0

}

}

}else{

m[,18] = 0

}

trans\_fin3 <- trans\_fin3 %>% rbind(m)

}

glm(funs ~ tran\_prod\_discount\_amt, family = 'binomial', data = trans\_fin3) %>% summary()

#Now we incorporate the customers' buying habits into account

diaper\_cust\_buying <- trans %>% filter(cust\_id %in% this$cust\_id)

diaper\_cust\_buying\_sim <- diaper\_cust\_buying %>% select(cust\_id, prod\_id, starts\_with('tran')) %>% merge(products, by = 'prod\_id')

diaper\_cust\_buying\_sim <- diaper\_cust\_buying\_sim %>% select(-subcategory\_id,-sub\_category\_desc,-category\_desc,-brand\_desc,-category\_desc\_eng)

###########################################################################

remove(trans)

remove(diaper\_cust\_buying)

###########################################################################

with\_other\_prod <- diaper\_cust\_buying\_sim %>%

mutate(count = 1) %>%

group\_by(cust\_id,category\_id) %>%

summarise(count = n()) %>%

spread(category\_id, count)

with\_other\_prod[is.na(with\_other\_prod)] <- 0

#Now merge this info with the transaction data

trans\_fin4 <- trans\_fin3 %>% full\_join(with\_other\_prod, by = 'cust\_id')

trans\_fin4$tran\_dt <- trans\_fin4$tran\_dt %>% ymd()

trans\_fin4$cluster <- trans\_fin4$cluster %>% as.factor()

trans\_fin4$cust\_id <- trans\_fin4$cust\_id %>% as.factor()

trans\_fin4$tran\_id <- trans\_fin4$tran\_id %>% as.factor()

trans\_fin4$tran\_dt <- trans\_fin4$tran\_dt %>% as.factor()

trans\_fin4$store\_id <- trans\_fin4$store\_id %>% as.factor()

trans\_fin4$prod\_unit <- trans\_fin4$prod\_unit %>% as.factor()

trans\_fin4$prod\_id <- trans\_fin4$prod\_id %>% as.factor()

trans\_fin4$funs <- trans\_fin4$funs %>% as.factor()

trans\_fin4 <- trans\_fin4 %>% select(-prod\_unit)

#Split data into training set and test set

testind <- sample(seq(1,nrow(trans\_fin4)), size = 0.25\*nrow(trans\_fin4))

test <- trans\_fin4[testind,]

rest <- trans\_fin4[-testind,]

valind <- sample(seq(1,nrow(rest)), size = 0.33333333333\*nrow(rest))

val <- rest[valind,]

train <- rest[-valind,]

#The Naive Bayes model

library(e1071)

NB\_train <- train %>% select(-ends\_with('id'))

NB\_val <- val %>% select(-ends\_with('id'))

NB\_test <- test %>% select(-ends\_with('id'))

NB <- naiveBayes(funs ~ ., data = NB\_train)

pred\_NB <- predict(NB, NB\_val)

#Logistic Regression

LR <- glm(funs ~ ., family = binomial, data = train)

pred\_LR <- predict(LR, val)

#Decision tree

library(rpart)

DT <- rpart(funs ~ ., data = train)

pred\_DT <- predict(DT, val)

for(i in 1:nrow(pred\_DT)) {

if(pred\_DT[i,2] >= 0.5){

pred\_DT[i,2] = 1

}else{

pred\_DT[i,2] = 0

}

}

pred\_DT <- pred\_DT[,-1]

#Performance metrics

library(pROC)

library(caret)

confusionMatrix(pred\_DT,val$funs)

F\_meas(pred\_NB,val$funs)

####################################################################################

new <- trans\_fin4 %>%

select(cust\_id,tran\_prod\_sale\_amt,funs)

new1 <- new

new1$funs <- new1$funs %>% as.numeric()

new2 <- new1 %>% group\_by(cust\_id) %>%

summarise(prod\_sale = sum(tran\_prod\_sale\_amt), funs\_ = sum(funs)-n())

for(i in 1:nrow(new2)){

if(new2[i,3] > 0){

new2[i,3] = 1

}else{

new2[i,3] = 0

}

}

####################################################################################

#Manipulation for pyspark

total\_paid\_each <- trans %>% group\_by(cust\_id, prod\_id) %>% summarise(tot = sum(tran\_prod\_paid\_amt))

####################################################################################

setwd('C:/Users/qw004/Desktop/MKT 680/mid-term project')

write.csv(trans\_fin3, file = 'transactions.csv')

write.csv(trans\_fin4, file = 'transactions\_large.csv')

write.csv(val, file = 'validation.csv')

write.csv(train, file = 'training.csv')

write.csv(test, file = 'test.csv')

write.csv(total\_paid\_each, file = 'final1.csv')

####################################################################################

####################################################################################

#Post processing

final2 <- fread('mycsv.csv', header = TRUE)

final3 <- final2 %>% filter(prod\_id%in%all$prod\_id)

f <- final3 %>% separate(col, c('cust\_id','rating'), ', ')

f2 <- f %>% separate(cust\_id, c('null','cust\_id'), '=')

f3 <- f2 %>% separate(rating, c('null2','rating'), '=')

f4 <- gsub(')', replacement = '', x = f3$rating)

f4 <- data.frame(rating = f4)

f5 <- f3 %>% select(-rating) %>% bind\_cols(f4)

f6 <- f5 %>% select(-V1, -null, -null2)

#Isolate the Huggies products for targeted emailing

huggies\_prod <- products %>% filter(brand\_desc == 'HUGGIES')

final4 <- f6 %>% filter(prod\_id%in%huggies\_prod$prod\_id)

#Exclude customers who have bought private label in the past

private\_prod <- products %>% filter(brand\_desc == 'PRIVATE LABEL')

private\_cust <- trans\_fin4 %>% filter(prod\_id%in%private\_prod$prod\_id)

final4 <- final4 %>% filter(!cust\_id %in% private\_cust$cust\_id)

product\_ids <- final4$prod\_id %>% unique()

#Slice the data up by product id into 7 parts

out1 = final4 %>% filter(prod\_id == product\_ids[1])

out2 = final4 %>% filter(prod\_id == product\_ids[2])

out3 = final4 %>% filter(prod\_id == product\_ids[3])

out4 = final4 %>% filter(prod\_id == product\_ids[4])

out5 = final4 %>% filter(prod\_id == product\_ids[5])

out6 = final4 %>% filter(prod\_id == product\_ids[6])

out7 = final4 %>% filter(prod\_id == product\_ids[7])

#Write them into csv files

write.csv(out1, file = 'out1.csv')

write.csv(out2, file = 'out2.csv')

write.csv(out3, file = 'out3.csv')

write.csv(out4, file = 'out4.csv')

write.csv(out5, file = 'out5.csv')

write.csv(out6, file = 'out6.csv')

write.csv(out7, file = 'out7.csv')