Using statistical techniques for historical customer purchasing analysis

Authors:

Mohita Goplani, Ming Min, Alex Shoop, Yijiang (Chuck) Xu

Advisor:

Professor Fatemah Emdad

Academic Project for DS 502: Statistical Methods in Data Science

Table of Contents

Abstract

CVS is, first and foremost, a pharmacy company. They value their pharmacy retail chains, and they want to improve their customer engagement experience. CVS has a strong market share performance and keeps their customer purchasing records, via a customer membership program called the ExtraCare card program. These large datasets help CVS in engaging in more potentially interested customers who would receive targeted advertisements and promotions. Here in our project we aimed to build a model to help CVS track customers buying trend and target their potential customers better.

Introduction

CVS Pharmacy is a well-known and popular pharmacy drugstore brand in the USA, they have more than 9,600 locations 28,000 pharmacists and 51,000 pharmacy technicians. [1] In addition to the common household medicine and cosmetics, CVS stores (online and physical Front Shops) sell snacks, letters, and general convenience items. CVS's work-flow system contained clinical programs to help make sure patient care program is their key priority, it can feed their data into its clinical pipes to execute this type of patient intervention for their members. To make this system work, each CVS customer has an "ExtraCare" membership card which records all customer transactions and purchasing history. Furthermore, each CVS customer can create an online CVS.com profile and connect their ExtraCare member ID to their CVS.com online profile. It is through this data gathering and large datasets that lead to the CVS Chief financial officer to confidently say that CVS "improves [...] performance and the performance of those that [CVS] serves both from a clinical perspective and from a cost perspective." [1]

Most retailers in practice use feature advertising to increase store traffic and to communicate through the store. Managers have great interest in understanding how consumers react to such promotions as well as what types of promotions retailers should focus on in order to improve their store performance. [2] Therefore, in our project we want to help decide who would be more likely to shop in CVS store and who should receive a promotional coupon email in order to target potential customers and thus increase the store revenue.

The team received access to a dataset of 10,000+ customers with recorded information on customer visited dates, number of visits the customer has made, item quantity/money spent, and a coupon flag for each customer. The primary goal was to create a predictive model to estimate how much money (in USD) a CVS customer would spend on their next purchase.

Methodology

Data Extraction

The data used was extracted from the CVS database as secondary data. The relational database Teradata contained an enormous amount of data for this analysis. However, we as a team decided to examine only the last six months data for a sample of 10,000 customers to analyze their purchasing trends. In the beginning, there was limited data, and so we needed to test certain costumed parameters such as visit number of the customer and number of days from last visit. This data was not readily available in the database. However, we calculated and extracted these fields using SQL query language.

‘Primary_visit’ is the i-th time that the same customer has visited the store in the last 6 months. We used this data to check the frequency of the customers who visited the store.
In addition, we calculated the days between two visits of a customer to understand the average time interval between two visits for every customer.

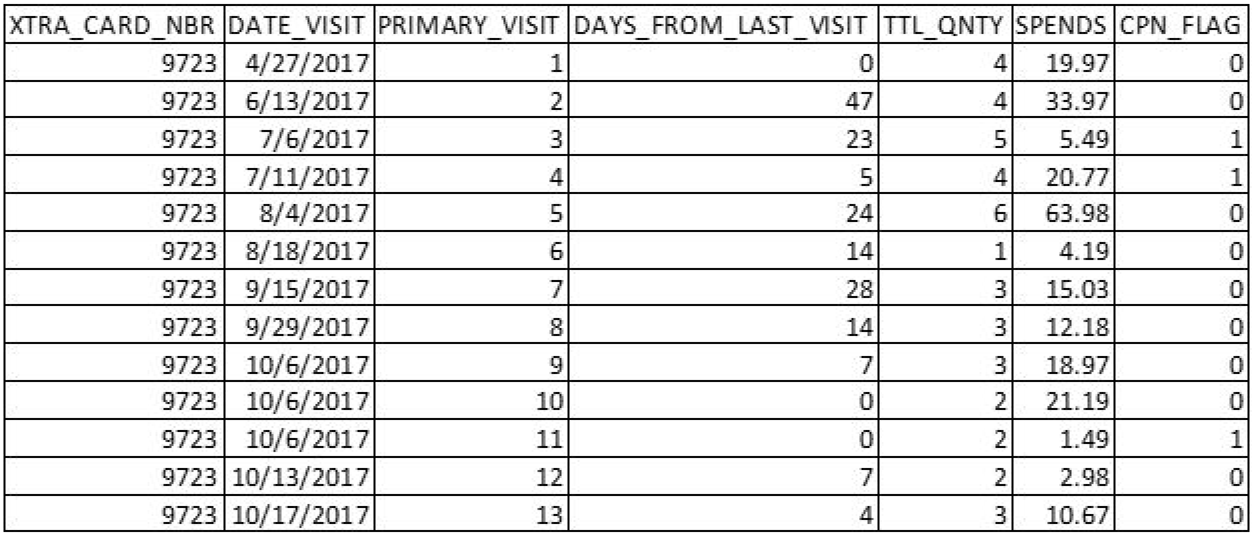
Furthermore, the team performed some data cleaning on our dataset. Certain irrelevant and unnecessary fields, such as entries containing only one recorded purchase, or incorrect USD spent due to a negative recorded purchase; one reason for this negative purchase field was due to the fact that a customer used a high-discount coupon on a low-priced item.

Finally, these calculated fields were then used as our independent variables to check the relationship between them and the spending probability of a customer. The following is a brief description of the dataset that we used for our analysis.

Dataset description:

The collected CVS customer dataset was historical data from the previous six months (between April 26, 2017 and October 24, 2017). The raw dataset was in an Excel spreadsheet, with seven columns and 418,670 rows (including headers). Below were the variables and their respective descriptions, followed by an example sampling of the dataset:

- XTRA_CARD_NBR (int): unique key for each CVS customer.
- DATE_VISIT (date): date when the CVS customer visited the store and purchased an item(s).
- PRIMARY_VISIT (int): the i-th time that the customer has visited the store in the past six months.
- DAYS_FROM_LAST_VISIT (int): days past since the CVS customer’s last visit.
- TTL_QNTY (int): total count of items that the customer purchased in their visit.
- SPENDS (float): total amount of money ($) that the customer spent in their visit.
- CPN_FLAG (bool): a binary flag indicating if the customer used a coupon in their visit.



XTRA_CARD_NBR	DATE_VISIT	PRIMARY_VISIT	DAYS_FROM_LAST_VISIT	TTL_QNTY	SPENDS	CPN_FLAG
9723	4/27/2017	1	0	4	19.97	0
9723	6/13/2017	2	47	4	33.97	0
9723	7/6/2017	3	23	5	5.49	1
9723	7/11/2017	4	5	4	20.77	1
9723	8/4/2017	5	24	6	63.98	0
9723	8/18/2017	6	14	1	4.19	0
9723	9/15/2017	7	28	3	15.03	0
9723	9/29/2017	8	14	3	12.18	0
9723	10/6/2017	9	7	3	18.97	0
9723	10/6/2017	10	0	2	21.19	0
9723	10/6/2017	11	0	2	1.49	1
9723	10/13/2017	12	7	2	2.98	0
9723	10/17/2017	13	4	3	10.67	0

Figure 1: Example sampling of the customer dataset.

Regression Analysis on Spending

For this part, we started with simple linear regression, by setting a new column named ‘SPENDPRV’ as one of the predictor which mainly concerned the influence of previous spending on the current spends. Then we took ‘SPENDS’ as our response variable, and took ‘PRIMARY_VISIT’, ‘DAYS_FROM_LAST_VISIT’, ‘CPN_FLAG’ as the other three independent variables. Furthermore, as what we’ve learned from Professor Fatemeh Emdad’s DS502/MA543 class [3], we did model fittings for polynomial regression and linear regression. Later we modified the response variable to ‘log(SPENDS)’ and compared these models by adjusted R^2. We randomly chose 50% of our data as the train data, and the other half as our test data; we believed this was appropriate because of the relatively large dataset that we were working with.

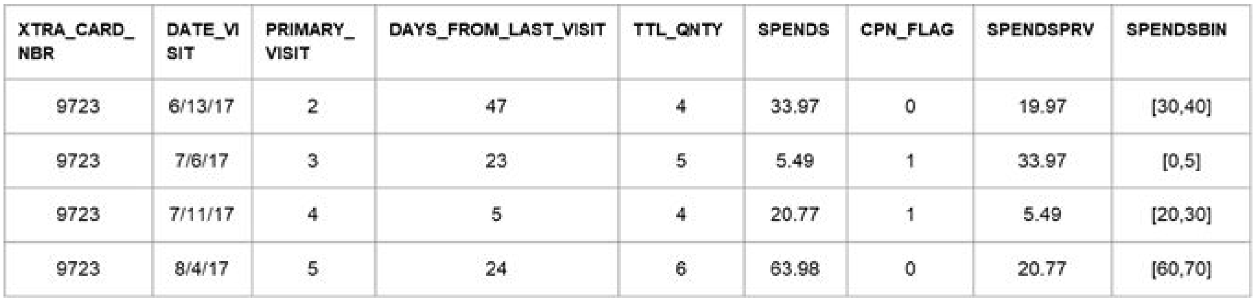
Regarding our reasoning behind modifying the response variable as ‘log(SPENDS),’ some customers may spend several dollars in one visit, but some may spend more than $1,000 dollars. Because of this large spread range of our spends, we transformed our ‘SPENDS’ to logarithm, and redid the linear regression for the changed ‘SPENDS’. Then we tried ‘Lasso’ and ‘Forward Subset Selection’ in order to also see if dimension reduction techniques would help with understanding our data and results.

Classification for Next Spending

We tried using all the regression method to fit an appropriate model to predict the next spends. However, we noticed that the accuracy rate of the models was quite low.
We then tried bucketing the spends in particular ranges to try to improve the accuracy of the model. Until now, we were trying to predict the exact value of the next spends. Using bucketing

we created buckets of particular spend ranges and predicted the next spend value by fitting it in the most appropriate bucket.

For example, if a particular customer had spent around $33 in this current visit, he would be a part of the bin that would range between 30 and 40. This new variable ‘SPENDSBIN’ was used as our new dependent variable. Once we received our new dependent variable, we used random foresting to classify our customers in their specific bins. Following is an example of the same:



XTRA_CARD_NBR	DATE_VISIT	PRIMARY_VISIT	DAYS_FROM_LAST_VISIT	TTL_QNTY	SPENDS	CPN_FLAG	SPENDSPRV	SPENDSBIN
9723	6/13/17	2	47	4	33.97	0	19.97	[30,40]
9723	7/6/17	3	23	5	5.49	1	33.97	[0,5]
9723	7/11/17	4	5	4	20.77	1	5.49	[20,30]
9723	8/4/17	5	24	6	63.98	0	20.77	[60,70]

Figure 2: Example sampling of our modified dataset set for classification purposes.

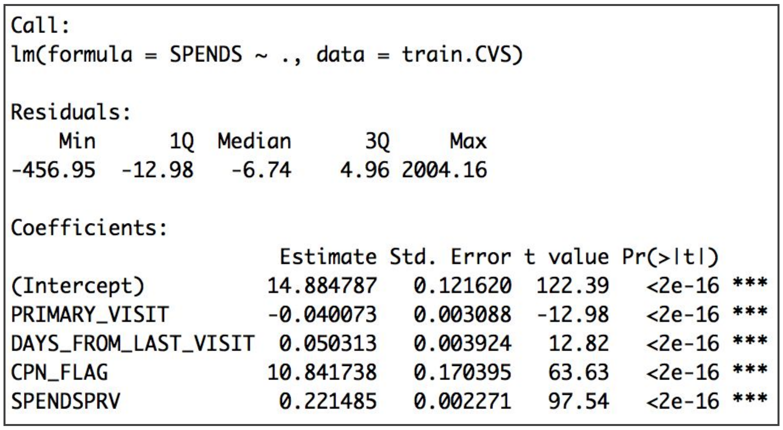
Random Forest Classification

Random foresting is a very efficient statistical method that builds on the idea of bagging but provides an improvement as it de-correlates the trees. It builds a number of decision trees on bootstrapped training dataset but each time a split in the tree is considered while building, a random sample of m predictors is selected out of the set of p predictors. This is done to reduce the variance in the model thus giving a more accurate prediction. We trained our data set using a random forest to classify the customer in the most appropriate bin for his next spends.

Results

Regression Results

The results of simple linear regression can be found in the following figure. From this, all p-values for each predictor is at a extremely low level, which means each predictor is significant at 99% level. The adjusted R^2 for simple linear model is 0.07995, the test MSE is 759.2, which is shown in the table.



```
Call:
lm(formula = SPENDS ~ ., data = train.CVS)

Residuals:
    Min      1Q  Median      3Q     Max 
-456.95  -12.98   -6.74    4.96 2004.16 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)          14.884787   0.121620  122.39   <2e-16 ***
PRIMARY_VISIT        -0.040073   0.003088  -12.98   <2e-16 ***
DAYS_FROM_LAST_VISIT  0.050313   0.003924   12.82   <2e-16 ***
CPN_FLAG             10.841738   0.170395   63.63   <2e-16 ***
SPENDSPRV             0.221485   0.002271   97.54   <2e-16 ***
```

Figure 3: Results of regular MSLR.

The adjusted R^2 for polynomial regression model is 0.08243, and the test MSE is 765.4. We have tried to fourth power of each predictor in this part. The adjusted R^2 changed a little bit.

When we modified the response variable to log(SPENDS), the adjusted R^2 increased to 0.099, which is the best one in all of our trials. Although 10% adjusted R^2 is not a good result in the general case. However, refer to randomness of customer buying behavior, and some economics effect, (i.e. income level, personal utility, customer age etc.) we believe this result is fair good based on our data. Compare to some similar research of customer behavior, Liang's [5] fitted model shows 10.8% adjusted R^2 in his result, using 'product knowledge', 'price consciousness', 'age', 'materialism' as independent variable. So we decided this one as our final fitted model.

Regression model	**Adjusted R^2**	**Test MSE**
MSLR, log(SPENDS) ~ .	0.07995	759.2073
MSLR, SPENDS ~ poly(x,4)	0.08243	765.4136
MSLR, log(SPENDS) ~ .	0.09922	0.9492816

Table 1: Comparison of different regression results.

For 'Lasso', we used K-fold cross validation method to choose the best lambda, with K = 10. The result is shown below.

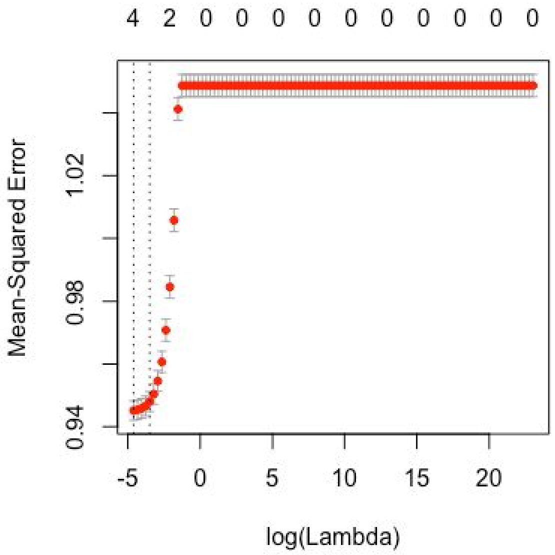
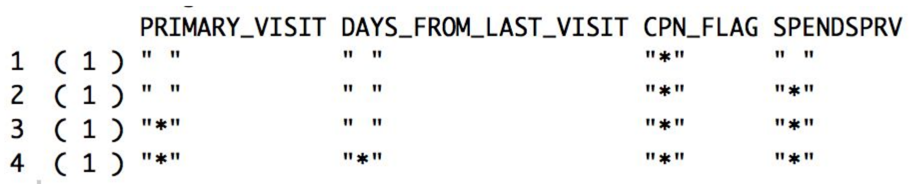


Figure 4: Lasso resulting plot.

Therefore, based on the above results and observation, we determined that the best lambda is 0.01, the respective test MSE is 0.9492816, and that the best subset to select is all four predictor variables. This is further backed by our feature importance variable results for 'Forward Subset Selection.' The first important variable, as we deduced based on historical spending patterns, was 'CPN_FLAG' with an adjusted R^2 value of 0.0518. However if we used all four important variables, we end up with a resulting adjusted R^2 value of 0.099377. The team decided to choose the group of variables with the best adjusted R^2.

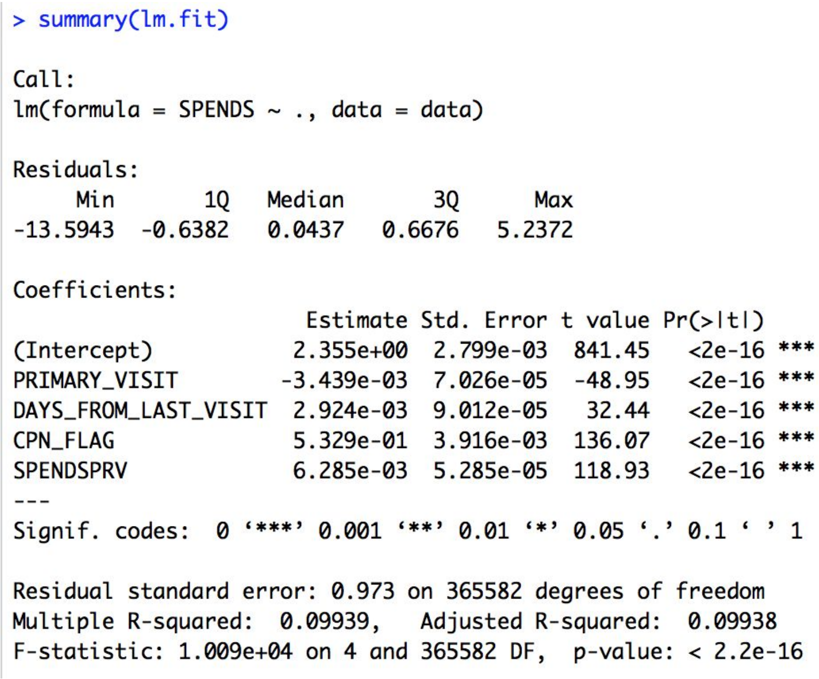


```
         PRIMARY_VISIT DAYS_FROM_LAST_VISIT CPN_FLAG SPENDSPRV
1  ( 1 ) " "           " "                  "*"      " "
2  ( 1 ) " "           " "                  "*"      "*"
3  ( 1 ) "*"           " "                  "*"      "*"
4  ( 1 ) "*"           "*"                  "*"      "*"
```

Figure 5: Feature importance for each variable.

Both 'Lasso' and 'Forward Subset Selection' suggested to use all four predictors. We then decided to fit our data with our determined logarithm model, using all four predictors. The final multiple linear regression formula and resulting summary table can be found below:

$$\log Spends = \beta_0 + \beta_1 * PrimaryVisit + \beta_2 * DaysFromLastVisit + \beta_3 * SpendsPrv + \beta_4 * CpnFlag + \epsilon$$



```
> summary(lm.fit)

Call:
lm(formula = SPENDS ~ ., data = data)

Residuals:
     Min       1Q   Median       3Q      Max 
-13.5943  -0.6382   0.0437   0.6676   5.2372 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)          2.355e+00  2.799e-03  841.45   <2e-16 ***
PRIMARY_VISIT       -3.439e-03  7.026e-05  -48.95   <2e-16 ***
DAYS_FROM_LAST_VISIT  2.924e-03  9.012e-05   32.44   <2e-16 ***
CPN_FLAG             5.329e-01  3.916e-03  136.07   <2e-16 ***
SPENDSPRV            6.285e-03  5.285e-05  118.93   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.973 on 365582 degrees of freedom
Multiple R-squared:  0.09939,	Adjusted R-squared:  0.09938 
F-statistic: 1.009e+04 on 4 and 365582 DF,  p-value: < 2.2e-16
```

Figure 6: MSLR formula and summary results of regression analysis using logarithm model.

From the summary, all predictors and the whole model are significant at more than 99% level. The adjusted R^2 shows us that consider the dimension of predictors, our model can explain almost 10% of customer spending amount in CVS store. Instead of stopping here, we tried to plot the errors to testify our model, which is shown as below.

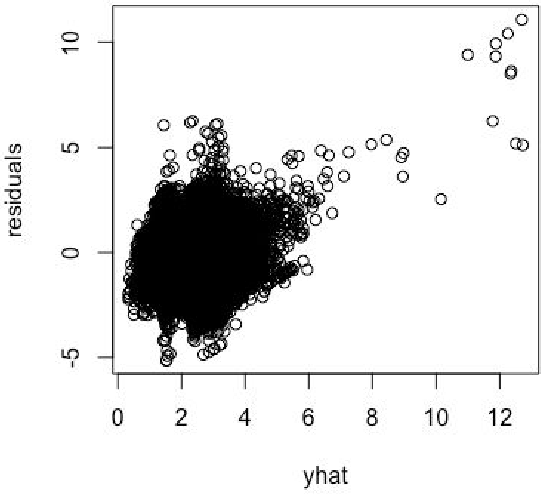


Figure 7: Residuals plot of MSLR and where response y is log(SPENDS).

Except for some outliers in the upper right, all residuals are positioned around 0. And the more closed to 0, the denser of residuals. This plot shows us that the residual follow a normal distribution with mean 0. As what we learned from class, a good fitted model should have normally distributed residuals with mean 0, which is consistent with our result. Based on our analysis of different model and the residuals, we believe our model is reasonable and fair good.

Classification Results

When we performed the random forest classification on the training data, we got a classification tree that would classify the customer in the best predicted spend bin which is the estimated spends for his current visit. The figure below is a part of the pictorial representation of the random forest tree that was generated, followed by a plot showing the optimal depth of tree to choose for random forest classification:

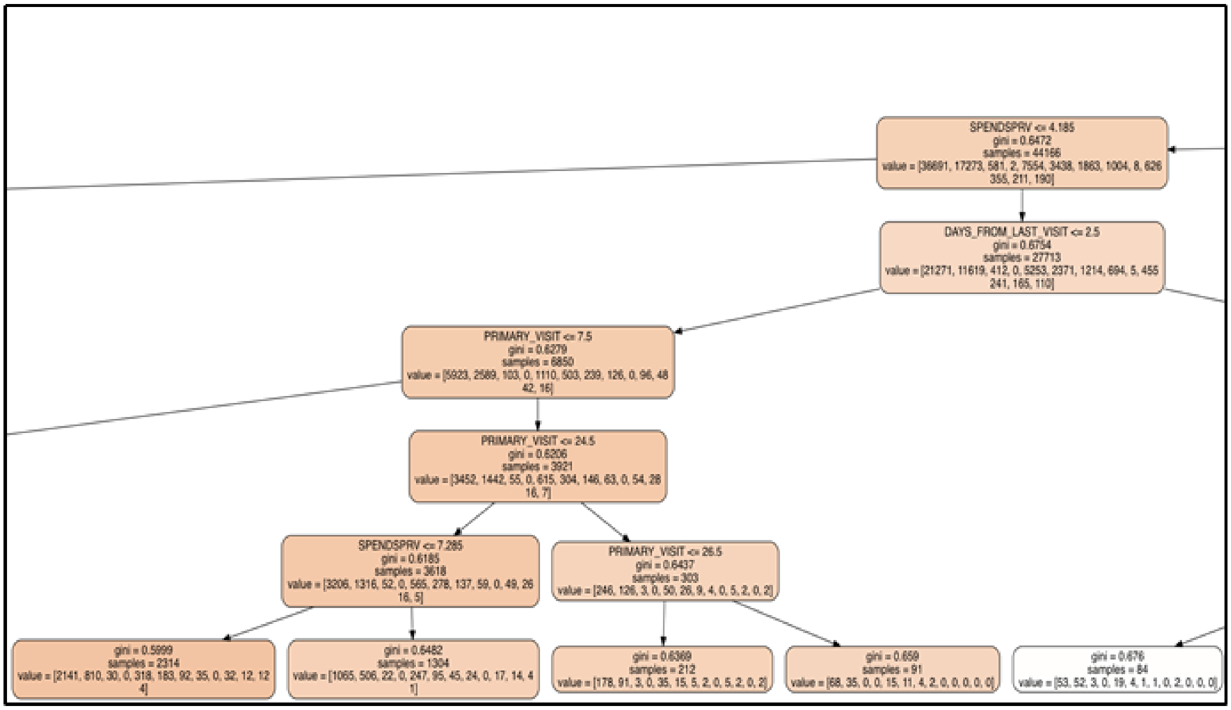

SPENDSPRV <= 4.185
gini = 0.6472
samples = 44166
value = [36691, 17273, 581, 2, 7554, 3438, 1863, 1004, 8, 626
355, 211, 190]
DAYS_FROM_LAST_VISIT <= 2.5
gini = 0.6754
samples = 27713
value = [21271, 11619, 412, 0, 5253, 2371, 1214, 694, 5, 455
241, 165, 110]
PRIMARY_VISIT <= 7.5
gini = 0.6279
samples = 6850
value = [5923, 2589, 103, 0, 1110, 503, 239, 126, 0, 96, 48
42, 16]
PRIMARY_VISIT <= 24.5
gini = 0.6206
samples = 3921
value = [3452, 1442, 55, 0, 615, 304, 146, 63, 0, 54, 28
16, 7]
SPENDSPRV <= 7.285
gini = 0.6185
samples = 3618
value = [3206, 1316, 52, 0, 565, 278, 137, 59, 0, 49, 26
16, 5]
PRIMARY_VISIT <= 26.5
gini = 0.6437
samples = 303
value = [246, 126, 3, 0, 50, 26, 9, 4, 0, 5, 2, 0, 2]
gini = 0.5999
samples = 2314
value = [2141, 810, 30, 0, 318, 183, 92, 35, 0, 32, 12, 12
4]
gini = 0.6482
samples = 1304
value = [1065, 506, 22, 0, 247, 95, 45, 24, 0, 17, 14, 4
1]
gini = 0.6369
samples = 212
value = [178, 91, 3, 0, 35, 15, 5, 2, 0, 5, 2, 0, 2]
gini = 0.659
samples = 91
value = [68, 35, 0, 0, 15, 11, 4, 2, 0, 0, 0, 0, 0]
gini = 0.676
samples = 84
value = [53, 52, 3, 0, 19, 4, 1, 1, 0, 2, 0, 0, 0]

Figure 8: A sliced example view of the random forest classification output.

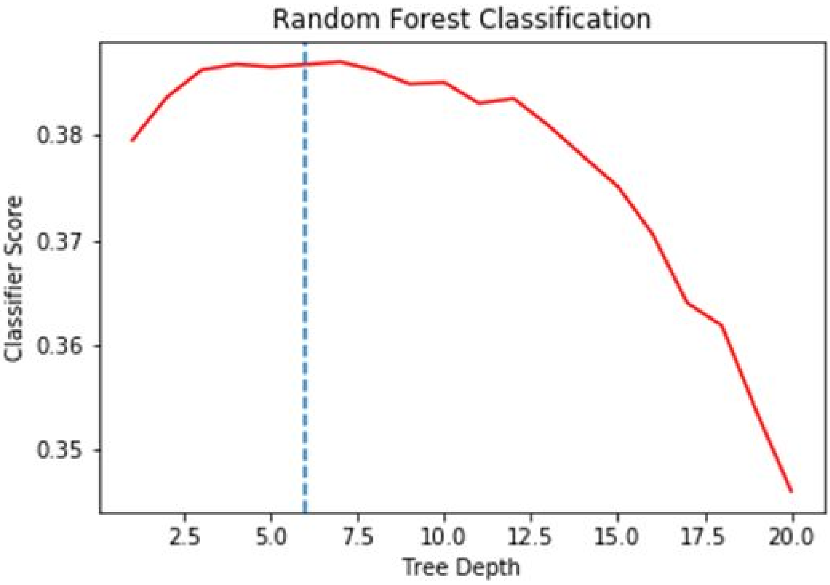


Figure 9: Tree depth versus classification score plot.

Based on the above results, we noticed that at tree depth of about 6, the model gives the best accuracy of 40%. That is, for every 10 data points, at least 4 will be correctly classified in their respective spending bins. However, there are more advancements that can be made to further improve the accuracy of the model.

Discussion and Conclusion

When we received the data from CVS, we had team discussions where we brainstormed the methods that could be used to model this data. Since this was customer trending data that we were dealing with, which is considered as one of the most complicated data fields to perform predictive analysis along with the enormity of the dataset, figuring out the most optimal model selection was a challenging task.

Our final adjusted R^2 for the multiple linear regression model is not very high compared to the regression model we talked about in class which is just below 10%. But we believe this is because the consumers’ buying behavior is relatively hard to predict. We searched and discovered a similar research project related to our project model; their adjusted R^2 are also around 10%, which indicates that our model is somewhat reasonable. [5] In daily life, unplanned impulse buying behavior accounts for a large proportion of consumer purchases. That means people always buy items without a historical reason and it can happen randomly. Therefore, it is hard to predict if they will purchase tomorrow or next week, and especially difficult to predict the amount of future purchase.

One interesting discovery from our data analysis was that the coupon flag, which is a dummy variable, is the most important predictor in our model (as shown in figures 3 and 5). This judgement is further backed because according to a published article it was mentioned that discounted items (or coupon that can make it discounted) would generate a lot additional sales or profit for the retail stores. [4]

Finally, we discovered some potential factors we could add-on to our model for future improvements. For example, we can include the category of the products, sales, etc. Also we can try to get more identifying information about the customers to help with specialized targeting such as their age, gender, income, and other demographic information. Seasonality or holidays also influence a lot; people always tend to buy more during holiday times or during season-shifting time people may need to prepare more medicines to prevent getting sick. We believe our work serves as a strong foundation for specialized CVS customer advertising and for future work in predictive customer analytics.

References

[1] Annual report of retail pharmacy (Chain Drug Review, Aug 2017). Web: http://go.galegroup.com/ps/i.do?p=AONE&u=mlin_c_worpoly&id=GALE%7CA501077927&v=2.1&it=r&sid=summon&ugroup=outside&authCount=1. Retrieved on November 27, 2017.

[2] Kapner, Suzanne. “Retailers’ Emails Are Misfires for Many Holiday Shoppers.” Web: https://www.wsj.com/articles/retailers-emails-are-misfires-for-many-holiday-shoppers-1511778600. Retrieved on November 27, 2017.

[3] Emdad, Fatemeh. Statistical Methods in Data Science (DS 502/MA 543). WPI Fall 2017.

[4] An Empirical Analysis of the Impact of Promotional Discounts on Store Performance (Sept 2017). Web: https://search.proquest.com/docview/1929060379?pq-origsite=summon&accountid=29120

[5] Liang, Ying-Ping. "The relationship between consumer product involvement, product knowledge and impulsive buying behavior." Procedia-Social and Behavioral Sciences 57 (2012): 325-330.

Appendix

Appendix A: R script file for regression analysis

```
# MA 543/DS 502
# CVS CASESTUDY PROJECT
# Team 4: Alex Shoop, Ming Min, Mohita Goplani, Yijiang (Chuck) Xu

# Importing the CVS raw dataset
rawDF <- read.csv(file="mainCVSdata.csv", header=TRUE, sep=",")

# Setting up the additional column of SPENDSNEXT
newDF <- rawDF
# removing first row since we will have no use for it
newDF <- newDF[-1,]
# getting the previous visit's SPENDS for the customer
newDF$SPENDSPRV <- rawDF$SPENDS[-nrow(newDF)]

# loop to make NA the SPENDSNEXT where it's the customer's last visit
# this takes about 5 minutes to process
#for (i in 1:nrow(newDF)){
#  if (newDF$XTRA_CARD_NBR[i] != newDF$XTRA_CARD_NBR[i+1] ||
newDF$PRIMARY_VISIT[i] + 1 == newDF$PRIMARY_VISIT[i+1]){
#    newDF$SPENDSNEXT[i] <- NA
#  }
#}
for (i in 2:nrow(newDF)){
  # check if card number A != card number B. If yes, then it means we are at card A's last entry.
  if (newDF$XTRA_CARD_NBR[i-1] != newDF$XTRA_CARD_NBR[i]){
    newDF$SPENDSPRV[i] <- NA

  }
  # check if card B != card A. If yes, check card B's first visit.
  if (newDF$XTRA_CARD_NBR[i] != newDF$XTRA_CARD_NBR[i-1]){
    # check if card B's first visit is not 1. If yes, then bad card B entry. Thus, NA.
    if (newDF$PRIMARY_VISIT[i] != 1){
      newDF$PRIMARY_VISIT[i] <- NA
    }

  }
  # if there's an NA entry previously for SPENDSNEXT (ie, bad card entry), then make the rest NA
  else
```

```
    if (is.na(newDF$PRIMARY_VISIT[i-1])){
    newDF$PRIMARY_VISIT[i] <- NA
    }
  # else then card B's entry is good (eg, they have a good first visit entry, and subsequent entries)

}

# removing the rows that have an NA entry
analysisDF <- na.omit(newDF)
# output analysis data
write.csv(analysisDF, file = 'analysis_data.csv')



#library('caret')
library('scatterplot3d')
library('corrplot')
library('lubridate')
set.seed(1)

# Importing the pre-setup dataset
df.CVS <- read.csv(file="analysis_data_ming.csv", header=TRUE, sep=",")
# removing unnecessary columns
newDFwithoutQNTY <- df.CVS[-c(1,2,3,6)]
newDFwithQNTY <- df.CVS[-c(1,2,3)]

# looking at basic plots, and observing any visual trend
col1 <- colorRampPalette(c("red", "grey", "blue"))
dfCVScor = cor(df.CVS)
corrplot(dfCVScor, method = "number", col = col1(100))
plot(df.CVS$DATE_VISIT, df.CVS$SPENDS)

# centered
#primvisit.c = scale(df.CVS$PRIMARY_VISIT, center=TRUE, scale=FALSE)
#daysfromlast.c = scale(df.CVS$DAYS_FROM_LAST_VISIT, center=TRUE, scale=FALSE)
#spendprev.c = scale(df.CVS$SPENDSPRV, center=TRUE, scale=FALSE)
#newvars.c = cbind(primvisit.c, daysfromlast.c, spendprev.c)
#newDF.CVS = cbind(df.CVS, newvars.c)
#names(newDF.CVS)[10:12] = c("primVisit.c", "daysFromLast.c", "spendPrev.c" )
#summary(newDF.CVS)



plot(df.CVS$TTL_QNTY, df.CVS$SPENDS, xlab = "TTL_QNTY", ylab = "SPENDS", main =
"TTL_QNTY vs SPENDS")
```

```
plot(df.CVS$PRIMARY_VISIT, df.CVS$SPENDS, xlab = "PRIMARY_VISIT", ylab = "SPENDS",
main = "PRIMARY_VISIT vs SPENDS")
boxplot(df.CVS$SPENDS ~ df.CVS$CPN_FLAG)
pairs(data.frame(df.CVS$SPENDS, df.CVS[3:5], df.CVS$CPN_FLAG), labels =
c("SPENDS","PRIMARY_VISIT","DAYS_FROM_LAST_VISIT","TTL_QNTY","CPN_FLAG"),
lower.panel = NULL)


# setting up training and test datasets
# random sampling
training = sample(nrow(df.CVS), nrow(df.CVS)/2)
#train.CVS = df.CVS[training,]
#test.CVS = df.CVS[-training,]
train.withoutQNTY = newDFwithoutQNTY[training,]
test.withoutQNTY = newDFwithoutQNTY[-training,]
train.withQNTY = newDFwithQNTY[training,]
test.withQNTY = newDFwithQNTY[-training,]

# performing basic regression analysis

# linear regression, SEMI-GOOD CHOICE
lin.regr.noQNTY <- lm(SPENDS ~ ., data = train.withoutQNTY)
lin.regr.QNTY <- lm(SPENDS ~., data = train.withQNTY)
# the version below is for the variable importance function
#lin.regr <- train(SPENDS ~ PRIMARY_VISIT + DAYS_FROM_LAST_VISIT + CPN_FLAG +
SPENDSPRV + QNTYPRV, data = train.CVS, method = "lm")
summary(lin.regr.noQNTY)
summary(lin.regr.QNTY)
# diagnosis plots
# prediction test
lin.pred.noQNTY = predict(lin.regr.noQNTY, test.withoutQNTY)
lin.pred.QNTY = predict(lin.regr.QNTY, test.withQNTY)
mean((lin.pred.noQNTY - test.withoutQNTY$SPENDS)^2)
mean((lin.pred.QNTY - test.withQNTY$SPENDS)^2)

# residual plot
plot(lin.pred[-length(lin.pred)], lin.regr$residuals)


# polynomial fit, BAD CHOICE
poly.fit = lm(SPENDS ~ PRIMARY_VISIT + DAYS_FROM_LAST_VISIT + CPN_FLAG +
SPENDSPRV*QNTYPRV, data = train.CVS)
summary(poly.fit)
```

```
poly.pred = predict(poly.fit, test.CVS)
mean((poly.pred - test.CVS$SPENDS)^2)


# log response fit, GOOD CHOICE
logresp.fit.noQNTY <- lm(log(SPENDS) ~ ., data = train.withoutQNTY)
logresp.fit.QNTY <- lm(log(SPENDS) ~ ., data = train.withQNTY)
logresp.fit.invQNTY <- lm(log(SPENDS) ~ PRIMARY_VISIT + DAYS_FROM_LAST_VISIT +
I(1/TTL_QNTY) + CPN_FLAG + SPENDSPRV, data = train.withQNTY)
summary(logresp.fit.noQNTY)
summary(logresp.fit.QNTY)
summary(logresp.fit.invQNTY)

logresp.pred.noQNTY <- predict(logresp.fit.noQNTY, test.withoutQNTY)
logresp.pred.QNTY <- predict(logresp.fit.QNTY, test.withQNTY)
logresp.pred.invQNTY <- predict(logresp.fit.invQNTY, test.withQNTY)

mean((logresp.pred.noQNTY - log(test.withoutQNTY$SPENDS))^2)
mean((logresp.pred.QNTY - log(test.withQNTY$SPENDS))^2)
mean((logresp.pred.invQNTY - log(test.withQNTY$SPENDS))^2)

# residual plots
plot(lin.regr.noQNTY, which = 1, main = "MSLR, no QNTY")
plot(lin.regr.QNTY, which = 1, main = "MSLR, w/ QNTY")
plot(logresp.fit.noQNTY, which = 1, main = "MSLR, where response = log(SPENDS), no QNTY")
plot(logresp.fit.QNTY, which = 1, main = "MSLR, where response = log(SPENDS), w/ QNTY")
plot(logresp.fit.invQNTY, which = 1, main = "MSLR, where response = log(SPENDS), w/ (1/QNTY)")

randomSampling = sample(nrow(test.withoutQNTY),10)
testSPENDS = test.withoutQNTY$SPENDS[randomSampling]
#testPRIMVISIT = test.withoutQNTY$PRIMARY_VISIT[randomSampling]
#testDAYSLASTVISIT = test.withoutQNTY$DAYS_FROM_LAST_VISIT[randomSampling]
#testCPN = test.withoutQNTY$CPN_FLAG[randomSampling]
#testPREVSPENDS = test.withoutQNTY$SPENDSPRV[randomSampling]
logresp.noQNTY.coefs = coef(logresp.fit.noQNTY)

logresp.pred.testing = predict(logresp.fit.noQNTY, test.withoutQNTY[randomSampling,])

inverse.pred = predict(inverse.fit, test.CVS)
mean((inverse.pred - test.CVS$SPENDS)^2)
plot(inverse.pred[-length(inverse.pred)], inverse.fit$residuals)
plot(inverse.fit, pch = 16, which = 1)
# variable importance, for linear regression model
```

```
important.regr = varImp(lin.regr, scale = FALSE)
plot(important.regr)
```

Appendix B: R script file for additional error metrics

```
data = read.csv("analysis_data.csv")
data = data[-c(1,2,3)]
data$SPENDS = log(data$SPENDS)

set.seed(1)
training = sample(nrow(data), nrow(data)/2)
train = data[training, ]
test = data[-training, ]

# lasso
library(glmnet)
y.train = train$SPENDS
x.train = model.matrix(SPENDS~., train)[, -1]
x.test = model.matrix(SPENDS~., test)[, -1]
y.test = test$SPENDS
x = model.matrix(SPENDS~., data)[, -1]
y = data$SPENDS
grid=10^seq(10,-2,length=100)
lasso.mod = glmnet(x.train, y.train, alpha=1, lambda = grid)
cv.out = cv.glmnet(x.train, y.train,alpha = 1, lambda = grid, nfolds = 10)
plot(cv.out)
best.lam = cv.out$lambda.min
lasso.pred = predict(lasso.mod, s=best.lam, newx=x.test)
print("This is our test error with lasso")
mean((lasso.pred-y.test)^2)

# use whole data set to train our model
out = glmnet(x, y, alpha = 1, lambda = grid)
lasso.coef = predict(out, type = "coefficients", s=best.lam)
print(lasso.coef)
# we can plot the residuals to see if linear regression is a proper method
yhat = predict(out, s=best.lam, newx=x)
residuals = yhat - y
plot(yhat,residuals)
# it turns out not, we'll try the log because of the shape of residuals plot

# try subset selectionm, for simple linear
```

```
library(leaps)
regfit.full = regsubsets(SPENDS~., data)
reg.summary = summary(regfit.full)
print(reg.summary)
print(reg.summary$adjr2)
# From the adjusted R2, is not a very good model

# use linear regression to see the summary
lm.fit = lm(SPENDS~.,data = data)



##################################
# coupon takes value of only 0&1, so we just try inverse from pairs(data)
#data$SPENDSPRV = 1/data$SPENDSPRV
#data$PRIMARY_VISIT = 1/data$PRIMARY_VISIT
# seperate the data into train and test part
set.seed(1)
training = sample(nrow(data), nrow(data)/2)
train = data[training, ]
test = data[-training, ]
# lasso
library(glmnet)
y.train = train$SPENDS
x.train = model.matrix(SPENDS~., train)[, -1]
x.test = model.matrix(SPENDS~., test)[, -1]
y.test = test$SPENDS
x = model.matrix(SPENDS~., data)[, -1]
y = data$SPENDS
grid=10^seq(10,-2,length=100)
lasso.mod = glmnet(x.train, y.train, alpha=1, lambda = grid)
cv.out = cv.glmnet(x.train, y.train,alpha = 1, lambda = grid)
plot(cv.out)
best.lam = cv.out$lambda.min
lasso.pred = predict(lasso.mod, s=best.lam, newx=x.test)
print("This is our test error with lasso")
mean((lasso.pred-y.test)^2)

# use whole data set to train our model
out = glmnet(x, y, alpha = 1, lambda = grid)
lasso.coef = predict(out, type = "coefficients", s=best.lam)
print(lasso.coef)
# we can plot the residuals to see if linear regression is a proper method
yhat = predict(out, s=best.lam, newx=x)
```

```
residuals = -yhat + y

plot(x[,'LogSpends'],residuals)
plot(x[, 'SPENDS'], residuals)

# try subset selection
library(leaps)
regfit.full = regsubsets(SPENDS~., data)
reg.summary = summary(regfit.full)
print(reg.summary)
print(reg.summary$adjr2)
```