Book case study

Thomas Farrell, Leo Salazar, Mia Brito, Raneen Aljamal

2/28/2022

Contents

Executive Summary 2
The Problem 2
Related Literature 2
Methodology 3
Loading libraries 3
Load in data 3
Checking for any missing values 3
Remove first column is data 4
Convert the response variable to a factor 4
Checking correlation within X-Variables 4
Results and findings 8
Linear model 8
Logit model 11
SVM Model 15
Conclusions and recommendations 17
References 18

Executive Summary

The Bookbinders club is a distribution company that sells specialty books through direct marketing. There are three types of predictive modeling used within this case to determine which customers the company should target for their next direct marketing campaign. The models utilized are Linear Regression, Logistic Regression, and Support Vector Machines. The results of the study show that the logistic regression model was the most accurate at predicting which customers would make a purchase with a sensitivity of 73%. When compared to the Support Vector machine model we only achieved a sensitivity of 28.5%. The most significant predictor variables found in the Logistic regression model were Gender, Time period since last purchase and their frequency.

The Problem

BBBC is exploring whether to use predictive modeling approaches to improve the efficacy of its direct mail program. For a recent mailing, the company selected 20,000 customers in Pennsylvania, New York, and Ohio from its database and included with their regular mailing a specially produced brochure for the book The Art History of Florence. This resulted in a 9.03% response rate (1806 orders) for the purchase of the book. BBBC then developed a database to calibrate a response model to identify the factors that influences these purchases. For this case analysis, we will use a subset of the database available to BBBC. It consists of data for 400 customers who purchased the book and 1200 customers who did not, thereby overrepresenting the response group. The dependent variable for the analysis is Choice – purchase or no purchase of the book. BBBC also selected several independent variables that it thought might explain the observed choice behavior. In this study we explored the variables that would be most significant to accurately predict who would purchase a book or not. We used the Linear Regression. Logit model, and Support Vector Machine to analyze the data.

Related Literature

In the previous case study, we explored that Mitchell Dayton concluded that logistical regression analysis is often described as the response variable, which must be of numerical value. In this case study, we wanted to explore the use of linear regression. According to —, linear regression and logistic regression have many similarities. However, there are a few differences. For example, linear regression more commonly handles regression problems while logistic regression tackles the process of classifying problems.

In Kyung-Shik Shin, Taik Soo Lee, and Hyun-jung Kim's case study An Application of Support Vector Machines in Bankruptcy Prediction Model, they explored the use of support vector machines in a Bankruptcy Prediction Model. As Noel Bambrick, a Support Vector Machines helps with classification and regression through a machine learning algorithm. It uses the notion of finding a hyperplane to divide data into two groups. In Shin, Lee, and Kim's case of corporate bankruptcy prediction, the Support Vector Model method is more effective than the Back-Propagation Neural Network method. The Support Vector Model proved to be more accurate and have better performance. Therefore, we wanted to explore the use of SVMs in our case study.

In addition to SVMs and linear regression, in this case study we will also explore the use of Logit Models. A Logit Regression Model is essentially the generalized linear model when discussing its link function. It is commonly compared to the logistical regression model, however, the logistic regression is the generalized linear model when discussing its activation function. In this case study, we explored the use of linear regression, support vector model, and a logit model to evaluate which variable would be most significant when predicting who would purchase a book.

Methodology

choice: whether the customer purchased the art history of Florence. 1- represents a purchase 0- representa a nonpurchase

Gender: 0 = Female , 1 = Male Amount_Purchased - Total money spent on BBBC books

Frequency: Total number of purchases in the chosen period

Last_Purchase: Months since last purchase

First_Purchase: Months since first purchase

P_Child: Number of children's books purchased

P_Youth: Number of youth books purchased

P_Cook: Number of cookbooks purchased

P_DIY: Number of do-it-yourself books purchased

P_Art: Number of art books purchased

Loading libraries

```
library(readxl)
library(e1071)
library(caret)
library(ROCR)
library(ggplot2)
library(tidyverse)
library(MASS)
library(ggThemeAssist)
library(esquisse)
library(gridExtra)
library(trackdown)
library(corrplot)
source('/Users/thomasfarrell/Downloads/optim_threshold.R')
```

Load in data

```
bbtrain = read_excel("/Users/thomasfarrell/Downloads/BBBC-Train.xlsx")
bbtest = read_excel("/Users/thomasfarrell/Downloads/BBBC-Test.xlsx")
```

Checking for any missing values

```
colSums(is.na(bbtest))
```

```
##      Observation           Choice           Gender Amount_purchased
##                0                0                0                0
##        Frequency    Last_purchase   First_purchase          P_Child
##                0                0                0                0
##          P_Youth           P_Cook            P_DIY            P_Art
##                0                0                0                0
```

Remove first column is data

We remove the first variable in the data due to the fact that this column just held a placeholder number.

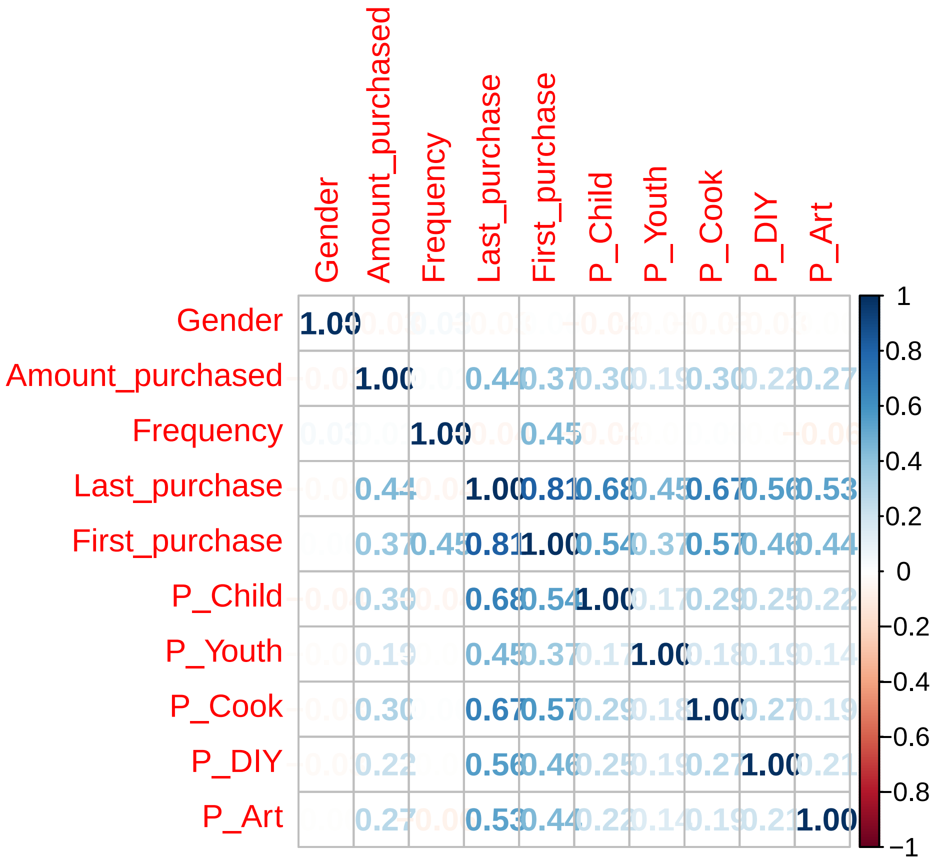
```
bbtrain = bbtrain[-c(1)]
bbtest = bbtest[-c(1)]
```

Convert the response variable to a factor

```
bbtrain$Choice = as.factor(bbtrain$Choice)
bbtest$Choice = as.factor(bbtest$Choice)
```

Checking correlation within X-Variables

```
b1_num = dplyr::select_if(bbtrain, is.numeric)
M = cor(b1_num)
corrplot(M, method = "number")
```



```
cor(bbtrain[sapply(bbtrain, is.numeric)])
```

```
##                        Gender Amount_purchased     Frequency Last_purchase
## Gender             1.000000000      -0.03060700  0.0321704951   -0.02896341
## Amount_purchased  -0.030607000       1.00000000  0.0136664846    0.44070127
## Frequency          0.032170495       0.01366648  1.0000000000   -0.04194328
## Last_purchase     -0.028963412       0.44070127 -0.0419432803    1.00000000
## First_purchase     0.001026138       0.37481393  0.4459457457    0.81467469
## P_Child           -0.041475936       0.29931372 -0.0433279437    0.67913392
## P_Youth           -0.014130306       0.18755727 -0.0095854745    0.45325891
## P_Cook            -0.026673876       0.30425340  0.0004968833    0.67250539
## P_DIY             -0.025946174       0.22331539 -0.0089634125    0.55816739
## P_Art             -0.003500037       0.27248948 -0.0613754066    0.53433415
##                   First_purchase     P_Child      P_Youth        P_Cook
## Gender               0.001026138 -0.04147594 -0.014130306 -0.0266738763
## Amount_purchased     0.374813928  0.29931372  0.187557270  0.3042533969
## Frequency            0.445945746 -0.04332794 -0.009585474  0.0004968833
## Last_purchase        0.814674687  0.67913392  0.453258910  0.6725053933
## First_purchase       1.000000000  0.54482083  0.367892128  0.5710547918
## P_Child              0.544820825  1.00000000  0.174826719  0.2947065185
## P_Youth              0.367892128  0.17482672  1.000000000  0.1816566401
## P_Cook               0.571054792  0.29470652  0.181656640  1.0000000000
## P_DIY                0.462018843  0.25383708  0.188683456  0.2717251256
## P_Art                0.442082061  0.22451285  0.141751220  0.1916807611
##                          P_DIY        P_Art
## Gender            -0.025946174 -0.003500037
## Amount_purchased   0.223315392  0.272489483
## Frequency         -0.008963412 -0.061375407
## Last_purchase      0.558167395  0.534334145
## First_purchase     0.462018843  0.442082061
## P_Child            0.253837077  0.224512850
## P_Youth            0.188683456  0.141751220
## P_Cook             0.271725126  0.191680761
## P_DIY              1.000000000  0.207791065
## P_Art              0.207791065  1.000000000
```

Highest correlations are between First_purchase and Last_purchase, which makes sense in a way. An only customer makes a purchase and walks out, that may be both a First purchase and Last purchase observation. Repeat customers will have a longer time in months for their first First purchase than their Last purchase.

("Linear Regression Vs Logistic Regression - Javatpoint," n.d.)(Shin, Lee, and Kim 2004)("Support Vector Machines: A Simple Explanation," n.d.)(n.d.) ## Exploratory Analysis

```
p1 = ggplot(data = bbtrain, mapping = aes(x = Frequency, fill = Choice))+
  geom_histogram() + theme(plot.title = element_text(face = "italic"),
    panel.background = element_rect(fill = "gray90",
        colour = "antiquewhite1", linetype = "dotted"),
    plot.background = element_rect(fill = "white",
        linetype = "dashed"))
```

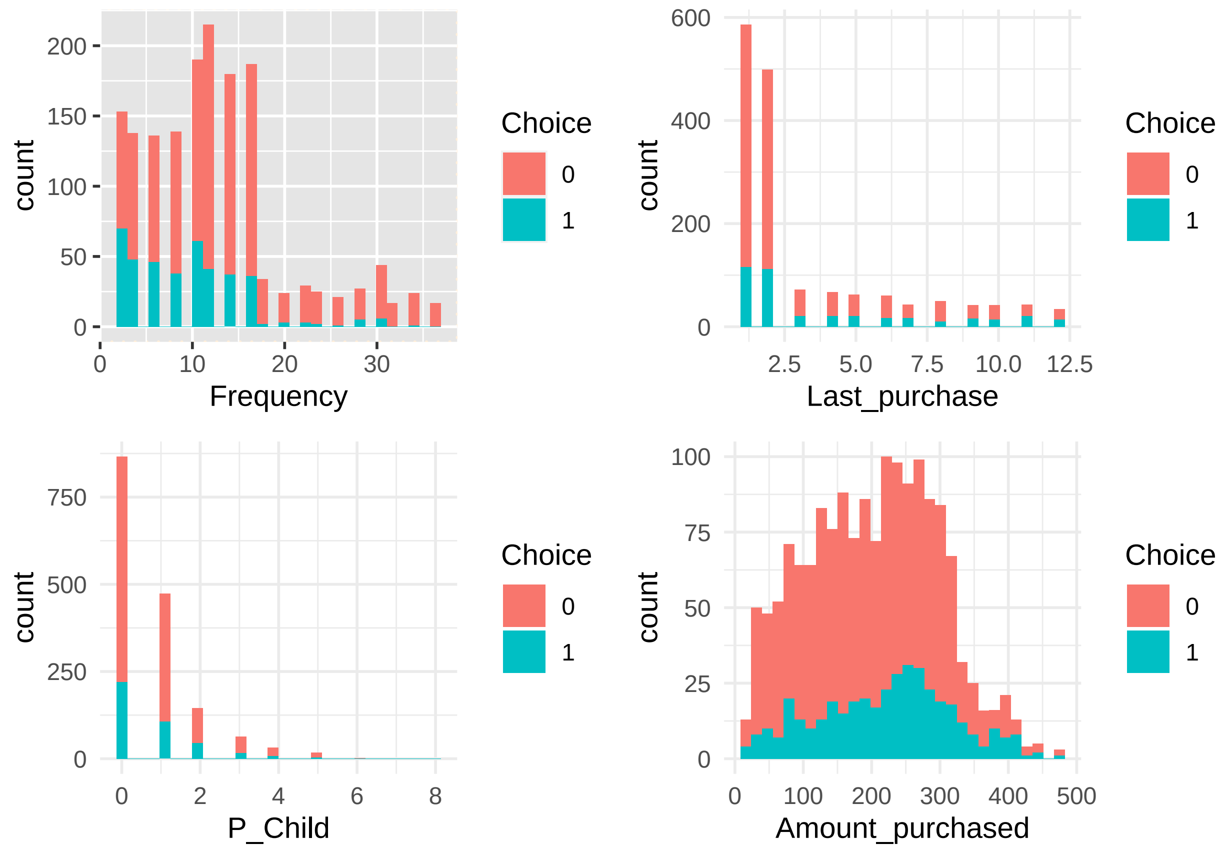
```
p2 = ggplot(bbtrain) +
  aes(x = Last_purchase, fill = Choice) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()
```

```
p3 = ggplot(bbtrain) +
  aes(x = P_Child, fill = Choice) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()
```

```
p4 = ggplot(bbtrain) +
  aes(x = Amount_purchased, fill = Choice) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()
```

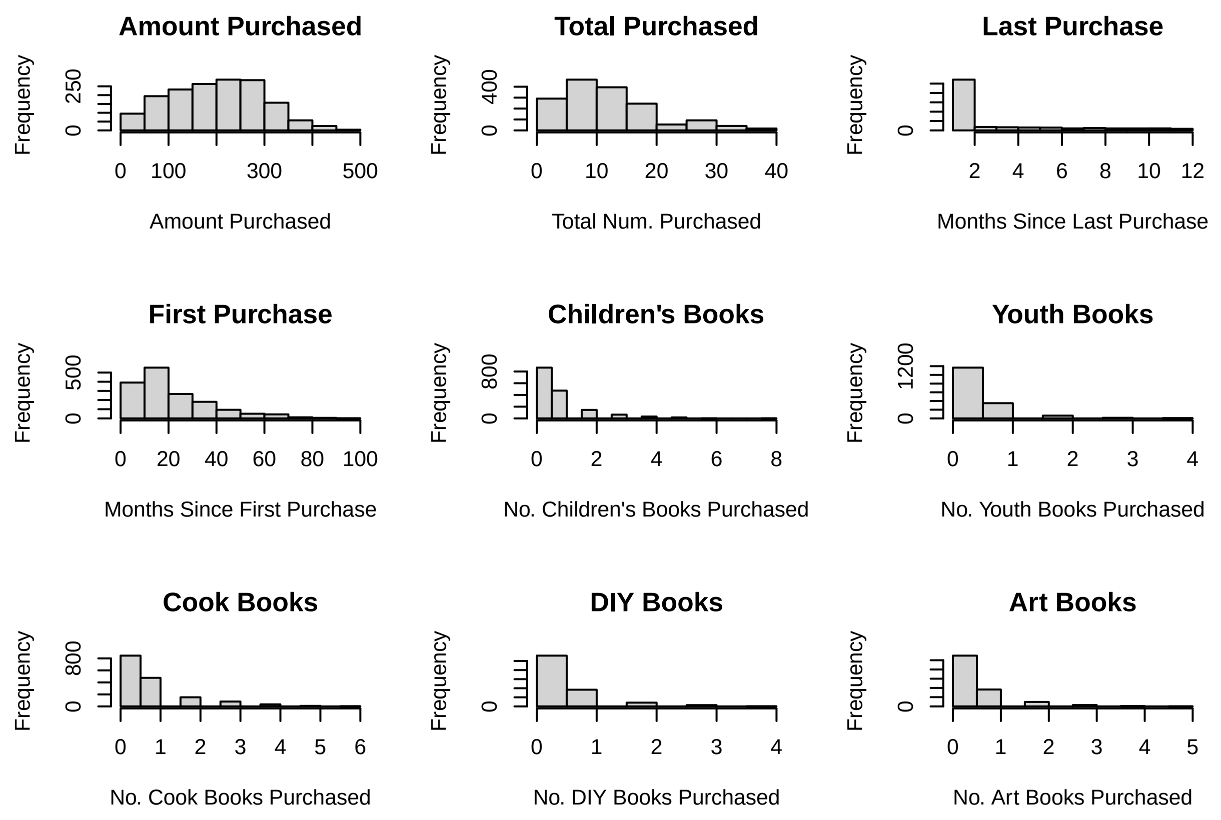
```
grid.arrange(p1,p2,p3,p4)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
par(mfrow = c(3,3))
hist(bbtrain$Amount_purchased, xlab = "Amount Purchased", main = "Amount Purchased")
hist(bbtrain$Frequency, xlab = "Total Num. Purchased", main = "Total Purchased")
hist(bbtrain$Last_purchase, xlab = "Months Since Last Purchase", main = "Last Purchase")
hist(bbtrain$First_purchase, xlab = "Months Since First Purchase", main = "First Purchase")
hist(bbtrain$P_Child, xlab = "No. Children's Books Purchased", main = "Children's Books")
hist(bbtrain$P_Youth, xlab = "No. Youth Books Purchased", main = "Youth Books")
hist(bbtrain$P_Cook, xlab = "No. Cook Books Purchased", main = "Cook Books")
```

```r
hist(bbtrain$P_DIY, xlab = "No. DIY Books Purchased", main = "DIY Books")
hist(bbtrain$P_Art, xlab = "No. Art Books Purchased", main = "Art Books")
```



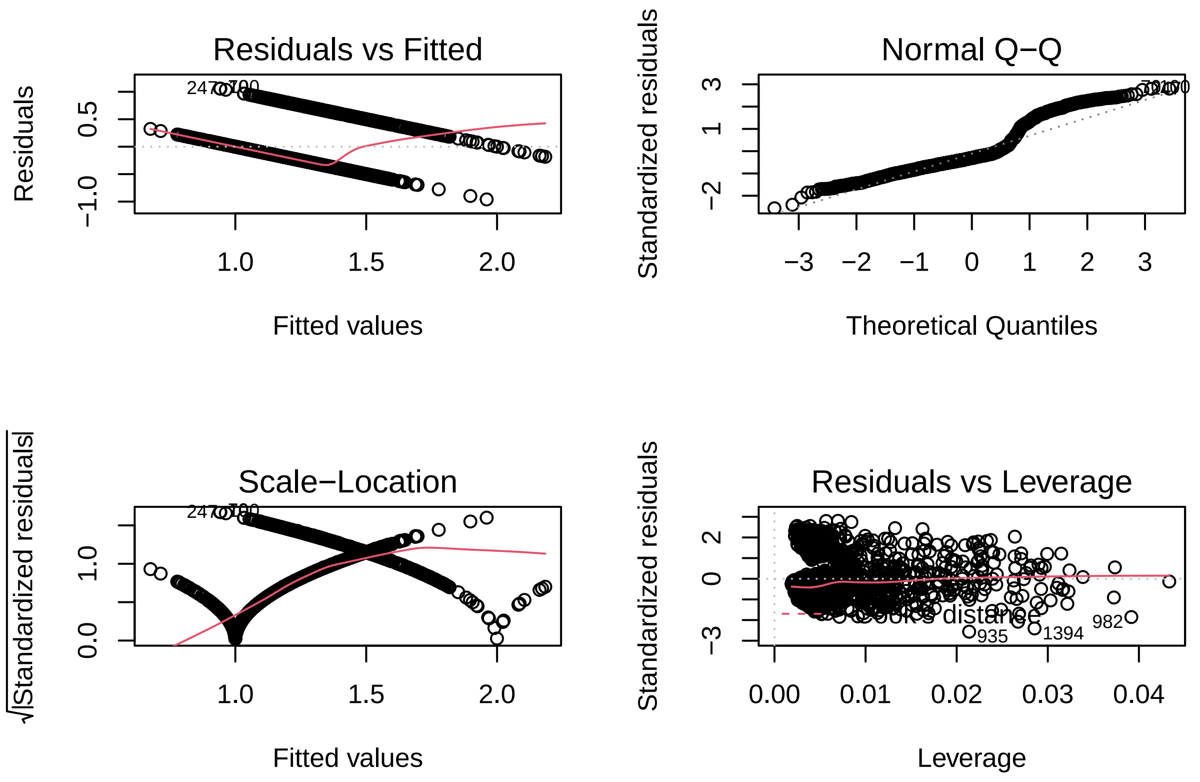
From these charts we can see that the Amount Purchased variable is the only variable that looks normally distributed and they other variables appear to have a right skew in their results.

Results and findings

The three modeling methods that we used were a linear regression model, a logit model (glm), and a support vector machine (SVM). The accuracy and sensitivity for each of the techniques vary, as well as the confusion matrices that tell us what each model predicts the best. However, the linear regression model would not be appropriate to use in this instance because a linear model can only be applied when using a continuous dependent variable, rather than a binary one that uses 0 and 1's. Logistic regression is typically used when the objective of the prediction is to project an outcome variable versus seeing how every predictor variable is related to the outcome variable.

Linear model

```
par(mfrow=c(2,2))
mod_1 = lm(as.numeric(Choice) ~., data = bbtrain)
plot(mod_1)
```



- Advantages:
  - Simple Explanation
  - Linear regression fits linearly separable data sets almost perfectly and is often used to find the nature of the relationship between variables.
  - Overfitting of data can reduced by regularization.
- Disadvantages:
  - Prone to under fitting
  - Sensitive to outliers
  - Does not work well with categorical variables as the dependent (unless given a scaled value). Assumes the information is independent

```
summary(mod_1)
```

```
##
## Call:
## lm(formula = as.numeric(Choice) ~ ., data = bbtrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9603 -0.2462 -0.1161  0.1622  1.0588
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.3642284  0.0307411  44.378  < 2e-16 ***
## Gender           -0.1309205  0.0200303  -6.536 8.48e-11 ***
## Amount_purchased  0.0002736  0.0001110   2.464   0.0138 *
## Frequency        -0.0090868  0.0021791  -4.170 3.21e-05 ***
## Last_purchase     0.0970286  0.0135589   7.156 1.26e-12 ***
## First_purchase   -0.0020024  0.0018160  -1.103   0.2704
## P_Child          -0.1262584  0.0164011  -7.698 2.41e-14 ***
## P_Youth          -0.0963563  0.0201097  -4.792 1.81e-06 ***
## P_Cook           -0.1414907  0.0166064  -8.520  < 2e-16 ***
## P_DIY            -0.1352313  0.0197873  -6.834 1.17e-11 ***
## P_Art             0.1178494  0.0194427   6.061 1.68e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3788 on 1589 degrees of freedom
## Multiple R-squared:  0.2401, Adjusted R-squared:  0.2353
## F-statistic:  50.2 on 10 and 1589 DF,  p-value: < 2.2e-16
```

#VIF for linear model

```
car::vif(mod_1)
```

```
##           Gender Amount_purchased        Frequency    Last_purchase
##         1.005801         1.248066         3.253860        18.770402
##   First_purchase          P_Child          P_Youth           P_Cook
##         9.685333         3.360349         1.775022         3.324928
##            P_DIY            P_Art
##         2.016910         2.273771
```

Last_Purchase and First_Purchase have a GVIF over 5 which tells us that we must remove these variables due to multiculinarity.

```
bbtrain = dplyr::select(bbtrain, - Last_purchase)
bbtrain = dplyr::select(bbtrain, - First_purchase)
```

```
mod_1 = lm(as.numeric(Choice) ~., data = bbtrain)
car::vif(mod_1)
```

```
##          Gender Amount_purchased        Frequency          P_Child
##        1.003526         1.232595         1.007587         1.212223
##         P_Youth           P_Cook            P_DIY            P_Art
##        1.083475         1.214043         1.163794         1.138879
```

Our new model has improved as all our variables are under 5 GVIF.

Logit model

```
glm.fit = glm(Choice ~ ., data = bbtrain, family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Choice ~ ., family = binomial, data = bbtrain)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.31846  -0.69097  -0.47171  -0.02488   2.84182
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -0.286380   0.202966  -1.411  0.15825
## Gender           -0.811948   0.134579  -6.033 1.61e-09 ***
## Amount_purchased  0.002406   0.000771   3.120  0.00181 **
## Frequency        -0.088625   0.010385  -8.534  < 2e-16 ***
## P_Child          -0.194796   0.072207  -2.698  0.00698 **
## P_Youth          -0.031928   0.109605  -0.291  0.77082
## P_Cook           -0.292392   0.072998  -4.005 6.19e-05 ***
## P_DIY            -0.279282   0.108094  -2.584  0.00977 **
## P_Art             1.245842   0.099062  12.576  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1799.5  on 1599  degrees of freedom
## Residual deviance: 1445.0  on 1591  degrees of freedom
## AIC: 1463
##
## Number of Fisher Scoring iterations: 5
```

```
glm2.fit = step(glm.fit,direction = "backward")
```

```
## Start:  AIC=1462.98
## Choice ~ Gender + Amount_purchased + Frequency + P_Child + P_Youth +
##     P_Cook + P_DIY + P_Art
##
##                    Df Deviance    AIC
## - P_Youth           1   1445.1 1461.1
## <none>                  1445.0 1463.0
## - P_DIY             1   1451.9 1467.9
## - P_Child           1   1452.6 1468.6
## - Amount_purchased  1   1454.8 1470.8
## - P_Cook            1   1462.1 1478.1
## - Gender            1   1481.5 1497.5
## - Frequency         1   1534.5 1550.5
## - P_Art             1   1639.8 1655.8
##
```

```
## Step:  AIC=1461.07
## Choice ~ Gender + Amount_purchased + Frequency + P_Child + P_Cook +
##     P_DIY + P_Art
##
##                    Df Deviance    AIC
## <none>                 1445.1 1461.1
## - P_DIY             1   1452.2 1466.2
## - P_Child           1   1452.9 1466.9
## - Amount_purchased  1   1454.8 1468.8
## - P_Cook            1   1462.5 1476.5
## - Gender            1   1481.6 1495.6
## - Frequency         1   1534.5 1548.5
## - P_Art             1   1640.1 1654.1
```

```
summary(glm2.fit)
```

```
##
## Call:
## glm(formula = Choice ~ Gender + Amount_purchased + Frequency +
##     P_Child + P_Cook + P_DIY + P_Art, family = binomial, data = bbtrain)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.30792  -0.69156  -0.47311  -0.02466   2.84228
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -0.2894506  0.2026211  -1.429  0.15314
## Gender           -0.8120440  0.1345723  -6.034  1.6e-09 ***
## Amount_purchased  0.0023859  0.0007678   3.108  0.00189 **
## Frequency        -0.0885491  0.0103772  -8.533  < 2e-16 ***
## P_Child          -0.1964495  0.0720116  -2.728  0.00637 **
## P_Cook           -0.2940503  0.0727520  -4.042  5.3e-05 ***
## P_DIY            -0.2823065  0.1076089  -2.623  0.00870 **
## P_Art             1.2441330  0.0988603  12.585  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1799.5  on 1599  degrees of freedom
## Residual deviance: 1445.1  on 1592  degrees of freedom
## AIC: 1461.1
##
## Number of Fisher Scoring iterations: 5
```

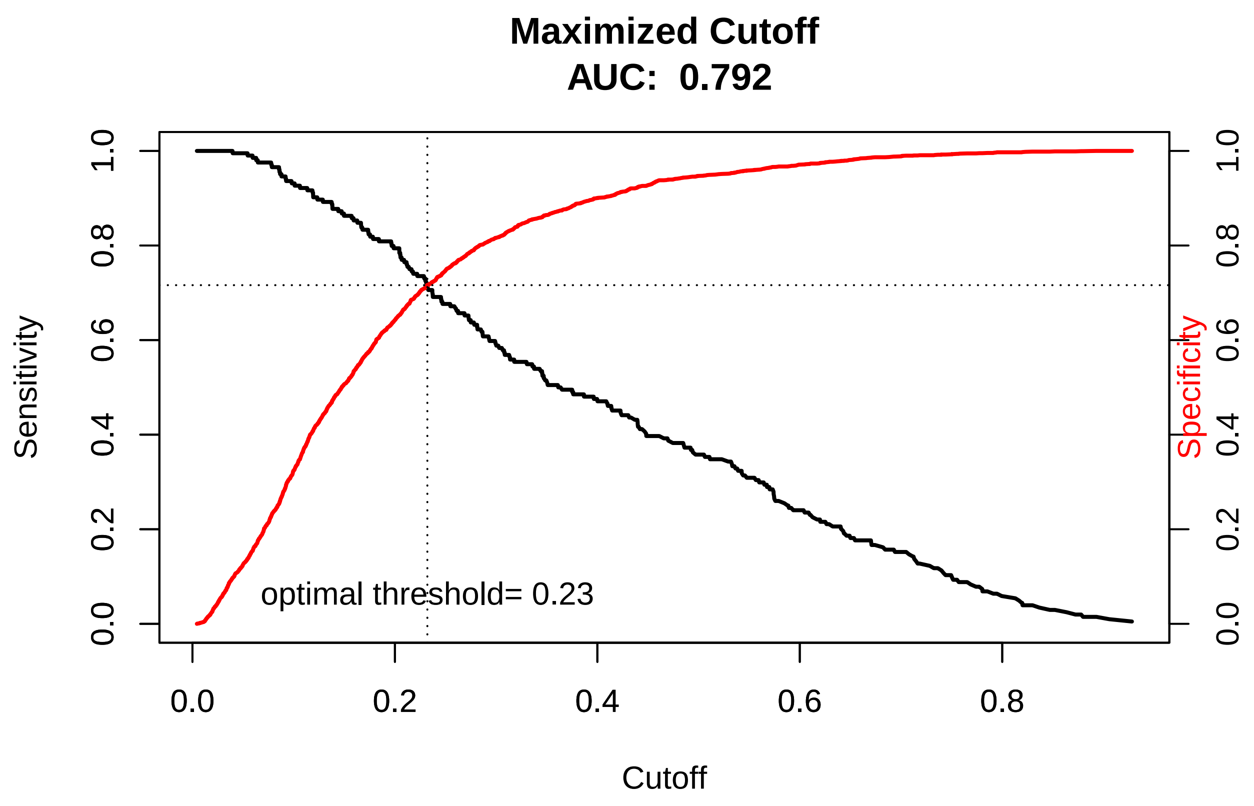
```
car::vif(glm2.fit)
```

```
##           Gender Amount_purchased        Frequency          P_Child
##         1.020262         1.204294         1.015220         1.208836
##           P_Cook            P_DIY            P_Art
##         1.221915         1.169499         1.225022
```

- Advantages:
  - Relatively easy to interpret and allows a clear understanding of how each of the predictors are influencing the outcome.
  - We do not need to transform the response to have a normal distribution.
  - Able to deal with categorical predictors.
- Disadvantages:
  - Predictor variables need to be uncorrelated.
  - Strict assumptions around distribution shape and randomness of error terms.
  - Sensitive to outliers

finding the optimal threshold

```
optim_threshold(glm2.fit,bbtest, bbtest$Choice)
```



```
predprob = predict.glm(glm2.fit, newdata = bbtest, type = "response")
predict.glm = ifelse(predprob >= .23, 1, 0)
caret::confusionMatrix(as.factor(predict.glm), as.factor(bbtest$Choice), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1490   55
##          1  606  149
##
##                Accuracy : 0.7126
##                  95% CI : (0.6936, 0.731)
##     No Information Rate : 0.9113
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1989
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.73039
##             Specificity : 0.71088
##          Pos Pred Value : 0.19735
##          Neg Pred Value : 0.96440
##              Prevalence : 0.08870
##          Detection Rate : 0.06478
##    Detection Prevalence : 0.32826
##       Balanced Accuracy : 0.72064
##
##        'Positive' Class : 1
##
```

SVM Model

Tuning the SVM Model

```
set.seed(1)
tuned = tune.svm(Choice ~ ., data = bbtrain, kernel = 'linear',gamma = seq(.01,.1,by = .025), cost = se
tuned$best.parameters
```

```
##    gamma cost
## 13  0.01  0.4
```

- Advantages:
  - It works really well with a clear margin of separation
  - It is effective in cases where the number of dimensions is greater than the number of samples
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Disadvantages:
  - It doesn't perform well when we have large data set because the required training time is higher
  - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation
  - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping

#creating SVM using tuned parameters

```
svm1 = svm(Choice ~ ., data = bbtrain, kernel = 'linear', gamma = tuned$best.parameters$gamma, cost = t
```

#Make predictions on training data set

```
predSVM = predict(svm1, bbtrain)
caret::confusionMatrix(predSVM, bbtrain$Choice, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1153  286
##          1   47  114
##
##                Accuracy : 0.7919
##                  95% CI : (0.7711, 0.8115)
##     No Information Rate : 0.75
##     P-Value [Acc > NIR] : 4.577e-05
##
##                   Kappa : 0.307
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.28500
##             Specificity : 0.96083
##          Pos Pred Value : 0.70807
##          Neg Pred Value : 0.80125
##              Prevalence : 0.25000
##          Detection Rate : 0.07125
##    Detection Prevalence : 0.10063
##       Balanced Accuracy : 0.62292
##
##        'Positive' Class : 1
##
```

The accuracy of our tuned model is 79.19% with a sensitivity of .285 and specificity of .96.

#make prediticions on the testing data set

```
predSVM = predict(svm1, bbtest)
caret::confusionMatrix(predSVM, bbtest$Choice, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2016  147
##          1   80   57
##
##                Accuracy : 0.9013
##                  95% CI : (0.8884, 0.9132)
##     No Information Rate : 0.9113
##     P-Value [Acc > NIR] : 0.9559
##
##                   Kappa : 0.2832
##
##  Mcnemar's Test P-Value : 1.184e-05
##
##             Sensitivity : 0.27941
##             Specificity : 0.96183
##          Pos Pred Value : 0.41606
##          Neg Pred Value : 0.93204
##              Prevalence : 0.08870
##          Detection Rate : 0.02478
##    Detection Prevalence : 0.05957
##       Balanced Accuracy : 0.62062
##
##        'Positive' Class : 1
##
```

With our tuned model against the testing data our model achieves an accuracy of 90.13% and a sensitivity of .279 and specificity of .961.

Conclusions and recommendations

After evaluating all three models, we concluded that the SVM was best at predicting which customers did not purchase a book, however it did not accurately predict the customers who did purchase a book. This is shown through the confusion matrix of the testing data set, where the SVM confusion matrix predicted a large number of true negatives and a specificity of 96.1% and sensitivity of 27.9% with an overall accuracy of 90.1%.

The logistic regression model best predicted a balance between who did purchase a book and who did not. The sensitivity of the logistic regression model was 73%, the specificity was 71%, and the accuracy was 71.2%. From this information, we see that the proportion of the predictions are more accurate at choosing which respondents would purchase a book or not.

There are two different methods to try to balance the data. The first method is to take a smaller sample of the data with more even yes and no results. Then we could run a logistic regression to produce a more accurate model with similar specificity and sensitivity numbers. To make a more accurate SVM model would could use a weighted SVM model, which would assign weights to the dependent variable classes. The differences in weights will influence the classification outcome during the training phase.

References

n.d. https://data.princeton.edu/wws509/notes/c3.pdf.

"Linear Regression Vs Logistic Regression - Javatpoint." n.d. *Www.javatpoint.com*. https://www.javatpoint.com/linear-regression-vs-logistic-regression-in-machine-learning.

Shin, Kyung-Shik, Taik Soo Lee, and Hyun-jung Kim. 2004. "An Application of Support Vector Machines in Bankruptcy Prediction Model." *Expert Systems with Applications*, September. https://www.sciencedirect.com/science/article/pii/S095741740400096X?casa_token=K3p5kjEkq3AAAAAA%3ASl8umHKq_UMCTMCNzXbAyXhXK1E6Q6vm2O5AD6lkuxrVsDY_i1c22L8ELn4byU8QudazbSRmYg.

"Support Vector Machines: A Simple Explanation." n.d. *KDnuggets*. https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html.