Not just who but where? Combining FSA data with transactions data to predict payment choice.

Previous research on payment choice has primarily focused on using transaction size and demographic information to predict payment method. While these variables have been shown to be important predictors, it is possible that more complex features may improve our understanding of payment choice. However, it is unclear whether adding more parameters necessarily leads to better predictive performance or improved understanding of demand. To think about this problem, I use a variety of machine learning methods to investigate whether adding new parameters to a model can improve its predictive performance. Then compare the results from higher dimensional datasets against the traditional parameters used in payment research. By doing so, I hope to first gain insight into the relative importance of different variables in predicting payment method and determine whether adding more complex features can lead to more accurate predictions, and to think about which learning methods might be most appropriate for working with payments and transactions data. Following the approach used by Oz Shy to model payment choice using data from the 2021 Survey and Diary of Consumer Payment Choice and machine learning methods I will estimate models using learning methods to classify methods of payment. To this end, I will utilize data from the Financial Diaries project, as well as census FSA data, to account for supply-side factors that may influence demand. In addition, I will also incorporate demographic characteristics of the community into the model. The hypothesis is that bank branches are more likely to provide services in areas with higher average incomes, and as such, income levels will be included as a parameter in the model. By including these additional parameters, I hope to improve our understanding of payment choice and potentially identify new factors that influence consumer behavior in this area.

Question 1: Describing the Data

The data set I will be using is the Financial Diaries dataset, which consists of six months of transaction data from low-income individuals residing in Winnipeg's urban core. Participants in the survey were instructed to keep logs of their weekly transactions and met with researchers from Menno Simons College discuss their spending habits. Their transactions were eventually transcribed into a CSV file, which comprises the data set I will be using. The data will be used to first replicate Oz Shy's models from his 2021 paper on payment choice, and subsequently, to explore the potential for additional parameters to improve model performance. The Financial Diaries dataset differs from the data set used by Shy in a few ways. The Financial Diaries dataset only tracks three methods of payment: cash, debit, and credit. Additionally, the data focuses on a specific subset of the population - those with low incomes. The data used by Shy contains a larger number of participants, but a shorter period where each participant was involved with the survey. In total the data set contains roughly 16000 observations, filtering out inflows and within account transactions I have 8572 total observations. These are randomly split into training and testing sets which are then used to build and test two sets of models. The first set uses the parameters set out in the Oz Shy paper, the second set includes all census and boundary characteristics of the participant FSAs. Adding the additional parameters leads to an increase in the dimensionality of the data which creates specific problems for the effectiveness of different machine learning methods. KNN, for example, is an algorithm that would likely struggle with the added parameters, so for the purpose of this assignment I will focus on algorithms that can work effectively with high dimensional data.

Question 2: Cleaning and Splitting the Data

```
Master <- read.csv("C:/Users/wardc/OneDrive/Desktop/Bank_of_Canada_project/OriginalData/Master_file_codes_namesCSV.csv")


SocioEcon <- read.csv("C:/Users/wardc/OneDrive/Desktop/Bank_of_Canada_project/OriginalData/soc & econ data by participantCSV.csv", header=FALSE)


FSA <- read.csv("~/BoClogitmodel/DemandModel/Soc&EconWFSA&Balances.csv", header=TRUE)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
## 
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
## 
##     combine
```

```
## 
## Attaching package: 'psych'
```

```
## The following object is masked from 'package:matlib':
## 
##     tr
```

```
## The following objects are masked from 'package:ggplot2':
## 
##     %+%, alpha
```

```
## Loading required package: bestglm
## Loading required package: leaps
```

```
## Loading required package: VGAM
```

```
## Loading required package: stats4
```

```
## Loading required package: splines
```

```
##
## Attaching package: 'VGAM'
```

```
## The following objects are masked from 'package:psych':
##
##     fisherz, logistic, logit
```

```
## Loading required package: rpart
```

```
## Loading required package: randomForest
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:psych':
##
##     outlier
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
## Important regclass change from 1.3:
## All functions that had a . in the name now have an _
## all.correlations -> all_correlations, cor.demo -> cor_demo, etc.
```

```
##
## Attaching package: 'class'
```

```
## The following objects are masked from 'package:FNN':
##
##     knn, knn.cv
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-4
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

Data cleaning, preperation and vizualation

```
#generate vector of participant ID's
ParticipantID <- matrix(SocioEcon$V1)
ParticipantID <- matrix(ParticipantID[2:29,])
ParticipantID <- matrix(as.numeric(ParticipantID))
colnames(ParticipantID) <- "ID"
#seperate outflows
Typesplit <- split(Master , f = Master$Type)
Outflows <- Typesplit$Outflow
rm(Typesplit)

Outflows <- subset(Outflows, Standard_item != "Credit Card Repayment"
                   & Custom.Credt == "Excluding Repayments"
                   & Standard_item != "Bank Deposit"
                   & Standard_item != "Investments"
                   & Item_category != "Financial")
```

```
SocioEcon <- SocioEcon[1:29,1:48]
colnames(SocioEcon)<- as.character(SocioEcon[1,])
SocioEcon <- SocioEcon[-1,]

#create a matrix to match demographic characteristics to
DemoChar <- matrix(, nrow = nrow(Outflows),ncol = 6)
colnames(DemoChar) <- c("Gender", "Income", "Indigenous", "Refugee", "Age", "Education")


for (j in 1:nrow(ParticipantID)) {
  a <- SocioEcon[j,1]
  a <- as.numeric(a)
  for (i in 1:nrow(Outflows)) {
  ifelse(Outflows[i,1] == a , DemoChar[i,1] <- SocioEcon[j,7],  "")
  ifelse(Outflows[i,1] == a , DemoChar[i,2] <- SocioEcon[j,26], "")
  ifelse(Outflows[i,1] == a , DemoChar[i,3] <- SocioEcon[j,30], "")
  ifelse(Outflows[i,1] == a , DemoChar[i,4] <- SocioEcon[j,31], "")
  ifelse(Outflows[i,1] == a , DemoChar[i,5] <- SocioEcon[j,43], "")
  ifelse(Outflows[i,1] == a , DemoChar[i,6] <- SocioEcon[j,47], "")
  }
}
#mix
DemoChar <- as.data.frame(DemoChar)
#Specify Variable Types

DemoChar$Gender <- as.factor(DemoChar$Gender)
DemoChar$Indigenous <- as.factor(DemoChar$Indigenous)
DemoChar$Refugee <- as.factor(DemoChar$Refugee)
DemoChar$Age <- as.numeric(DemoChar$Age)
DemoChar$Education <- as.numeric(DemoChar$Education)
```

```
## Warning: NAs introduced by coercion
```

```
DemoChar$Education[is.na(DemoChar$Education)] <- 10
#convert $ into numeric

cash_values <- DemoChar$Income
cash_values <- gsub("\\$", "", cash_values)
cash_values <- as.numeric(gsub(",", "", cash_values))
Income <- matrix(cash_values, ncol = 1 )
colnames(Income) <- "Income"


Amount <- matrix(Outflows$Amount, ncol = 1)
colnames(Amount) <- "Amount"

#create new dataframe for regressions
Logitdf <- cbind.data.frame(Outflows, DemoChar)

rm(Amount)
# create dataset that contains only usefull parameters


Lassodf <- Logitdf[,c(7,8,10,12,17,27,28,29,30,31,32)]
```

Diary participants

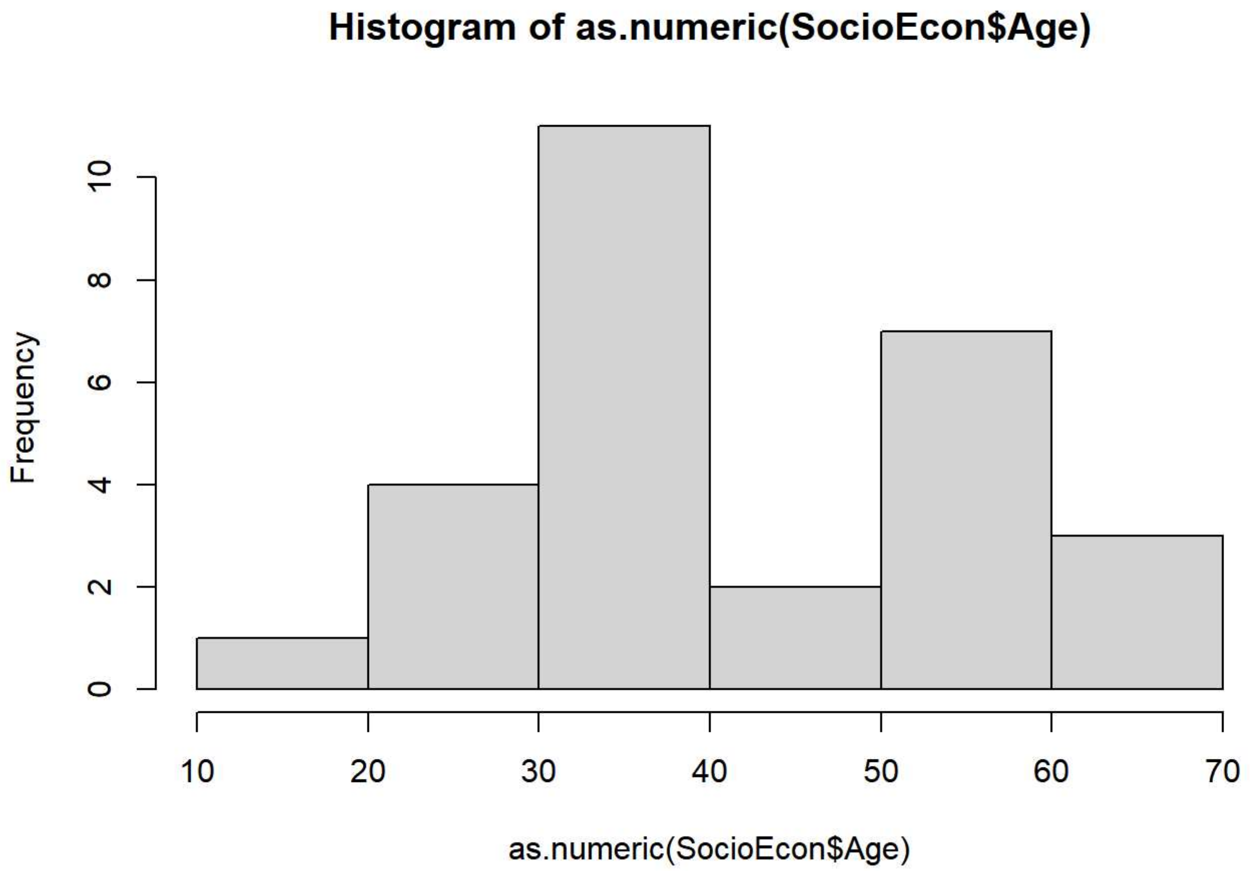
```
#Gender
table(SocioEcon$Gender)
```

```
##
## Female   Male
##     23      5
```

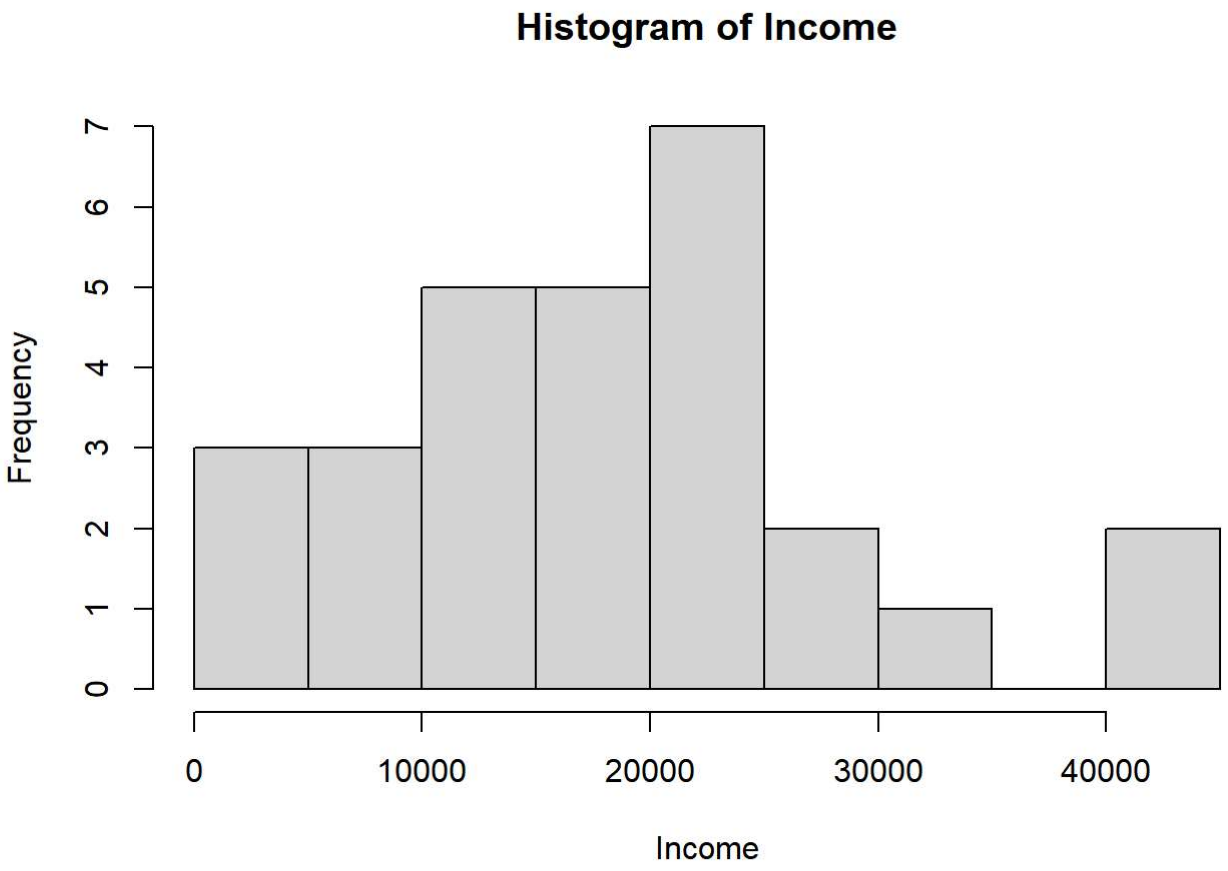
```
#Indigenous
table(SocioEcon$`Indigeneous*`)
```

```
##
##  No Yes
##  22   6
```

```
hist(as.numeric(SocioEcon$Age))
```



```
cash_values <- SocioEcon$`Income Over diary`
cash_values <- gsub("\\$", "", cash_values)
cash_values <- as.numeric(gsub(",", "", cash_values))
Income <- matrix(cash_values, ncol = 1 )
hist(Income)
```


Histogram of Income
Frequency
0
1
2
3
4
5
6
7
0
10000
20000
30000
40000
Income

Merging census Data

```
FSAProccessed <- read.csv("~/BoClogitmodel/DemandModel/FSAProccessed.csv", header=TRUE)
#FSA <- FSA[,49:71]

FSA <- FSA[1:28,]
ParticipantFSA <- as.matrix(FSA$FSA)

Census.Participant <- matrix(,ncol = ncol(FSAProccessed), nrow = 0)
colnames(Census.Participant) <- colnames(FSAProccessed)

for (i in 1:nrow(ParticipantFSA)) {
  a <- ParticipantFSA[i,1]
  b <- as.matrix(subset(FSAProccessed, fsa_name == a))
  Census.Participant <- rbind(Census.Participant, b)

}
Census.Participant <- cbind(FSA[,1:2], Census.Participant)


#attach census data to observations

Cencus.char <- matrix(, nrow = nrow(Outflows),ncol = ncol(Census.Participant))
colnames(Cencus.char) <- colnames(Census.Participant)


for (j in 1:nrow(ParticipantID)) {
  a <-as.numeric(SocioEcon[j,1])
  for (i in 1:nrow(Outflows)) {
  b <- as.matrix(subset(Census.Participant, Code == a))
  ifelse(Outflows[i,1] == a, Cencus.char[i,1:45] <- b[1:45], "")

  }
}



Cencus.char[,c(13,14,15,42)] <- as.numeric(Cencus.char[,c(13,14,15,42)])/100
Cencus.char[,c(4:6, 9:42)]<- as.numeric(Cencus.char[,c(4:6, 9:42)])

lassodf.census <- cbind(Lassodf, Cencus.char[,c(-1,-2,-3,-7,-8)])



df.final <- lassodf.census[lassodf.census$Method %in% c("Cash", "Credit Card", "Debit Card"), ]
df.final$Method <- as.factor(df.final$Method)
df.final <- na.omit(df.final)
df.final[,5] <- as.factor(df.final[,5])

for (i in 12:51) {
  df.final[,i] <- as.numeric(df.final[,i])
}
```

```
#converting cash values into numeric
cash_values <- df.final$Income
cash_values <- gsub("\\$", "", cash_values)
cash_values <- as.numeric(gsub(",", "", cash_values))
Income <- matrix(cash_values, ncol = 1 )
colnames(Income) <- "Income"

df.final$Income <- Income
df.final <- df.final[,c(-2,-4)]

Cash <- ifelse(df.final$Method == "Cash", 1, 0)
df.final.cashonly <- df.final
df.final.cashonly$Method <- Cash
df.final.cashonly$Method <- as.factor(df.final.cashonly$Method)

Credit <- ifelse(df.final$Method == "Credit Card" , 1, 0)
df.final.creditonly <- df.final
df.final.creditonly$Method <- Credit
df.final.creditonly$Method <- as.factor(df.final.creditonly$Method)
```

split into training and testing sets

```
set.seed(500)
df.split <- initial_split(df.final,0.5)
df.train <- training(df.split)
df.test <- testing(df.split)



df.split.cashonly <- initial_split(df.final.cashonly,0.5)
df.train.cashonly <- training(df.split.cashonly)
df.test.cashonly <- testing(df.split.cashonly)





df.train$Method <- relevel(df.train$Method , ref = "Debit Card")
```

Randomly splitting the cleaned data into two groups using the initial split function.

Question #3: Modeling the data

##The Oz Shy Models The variables Shy uses in his paper are commonly used in payments research to classify payment choice using transaction data. I re-create the Shy models as a baseline to compare the models with added covariates. The models I refer to as the Shy models are models estimated using the following set of covariates: - Transaction size, which is given by Amount in the data. - Gender, the number of males and females in my data set is highly unbalanced, with - Household income - Marital status - Education level, shy uses a three-level factor variable where I use a years of education measure. - Household income In comparing the Payment Survey data and the Financial Diaries data, Oz Shy uses a data set more representative of the underlying population using the payment survey data, the financial diaries provide an interesting test for various machine learning approaches due to its limited and unbalanced features. In particular, the Financial Diaries data presents a challenge for machine learning models because of its high-dimensional nature, unbalanced features, and small sample size. For this reason, I avoid estimating KNN models, and instead estimate a multinomial logit model using both the full set of covariates and the Oz Shy set of covariates then tree and forest models, and SVMs.

##multinomial logistic model Since Multinomial models are common in payments research, I have estimated two sets multinomial models. As I add more covariates to the data set, the dimensionality of the data increases, which may negatively impact the performance of the multinomial logit model. However, the multinomial model can still serve as a baseline model for comparison with more complex models like random forests and support vector machines. To assess the performance of these models, I will mainly focus on the accuracy of out-of-sample predictions on the testing set. By evaluating the models' predictive accuracy, I can determine which model performs the best at accurately predicting outcomes for new data points. Overall, while the increase in dimensionality of the data set may pose a challenge for the multinomial

```r
library(nnet)
LogModel.shy <- multinom(Method ~ Amount + Gender + Income + Age + Education, data = df.train )
```

```
## # weights:  21 (12 variable)
## initial  value 4708.652269
## iter  10 value 4091.647321
## iter  20 value 3831.096725
## final  value 3831.096116
## converged
```

```r
LogModel.open <- multinom(Method~.,  data = df.train)
```

```
## # weights:  183 (120 variable)
## initial  value 4708.652269
## iter  10 value 4123.997623
## iter  20 value 3641.760723
## iter  30 value 3327.726905
## iter  40 value 2935.831473
## iter  50 value 2825.002945
## iter  60 value 2783.913548
## iter  70 value 2777.137590
## iter  80 value 2776.339215
## iter  90 value 2775.629367
## iter 100 value 2775.579470
## final  value 2775.579470
## stopped after 100 iterations
```

Both the multinomial logit and the Shy models overestimate debit card usage in the data set. This could be due to a problem with multinomial logit models and unbalanced data, which can lead to a biased overestimation of the majority response variable. In this case, debit cards are the most used item in the survey, and the multinomial models predict debit card usage most frequently, which could be causing the overestimation of debit card usage.

Tree Models

Regression Tree

The classification tree models may be more effective at handling unbalanced data compared to the multinomial logit model, and they may be better equipped to predict outcomes given the added parameters. Classification trees do not require any assumptions about the underlying data distribution which gives them an advantage over the multinomial logit for prediction. However, as I add covariates to the by adding FSA data, it becomes more likely that the tree overfits the data and the out of sample predictions get worse, so its possible that the simple model outperforms in this case.

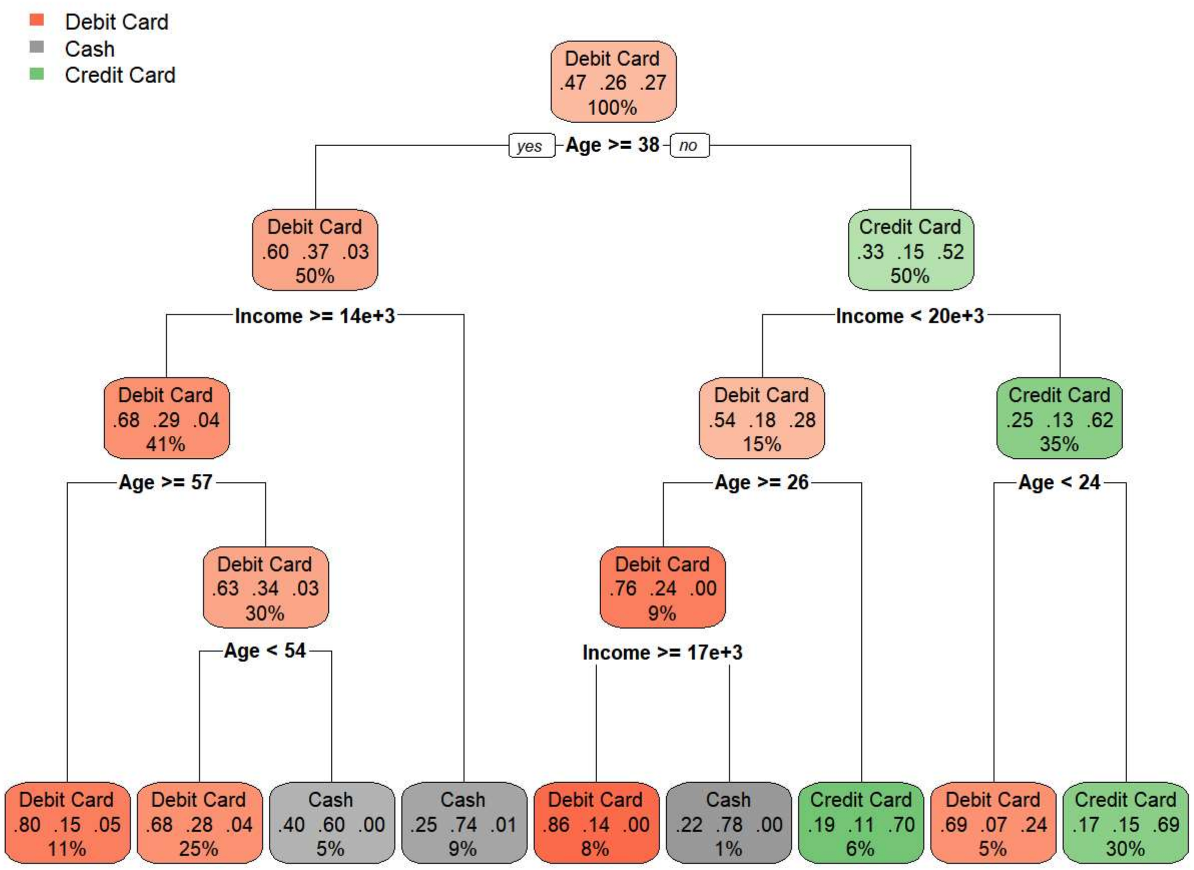
```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.1.3
```

```
df.train.tree <- df.train
df.test.tree <- df.test

y.tree <- df.train[,1]
x.tree <- model.matrix(Method ~., data = df.train.tree)
df.tree <- as.data.frame(cbind(y.tree, x.tree[,c(-1,-2)]))


Treemodel.shy <- rpart(Method ~ Amount + Gender + Income + Age + Education, data = df.train.tre
e, method = "class")
rpart.plot(Treemodel.shy)
```


Debit Card
Cash
Credit Card
Debit Card .47 .26 .27 100%
yes
Age >= 38
no
Debit Card .60 .37 .03 50%
Credit Card .33 .15 .52 50%
Income >= 14e+3
Income < 20e+3
Debit Card .68 .29 .04 41%
Debit Card .54 .18 .28 15%
Credit Card .25 .13 .62 35%
Age >= 57
Age >= 26
Age < 24
Debit Card .63 .34 .03 30%
Debit Card .76 .24 .00 9%
Age < 54
Income >= 17e+3
Debit Card .80 .15 .05 11%
Debit Card .68 .28 .04 25%
Cash .40 .60 .00 5%
Cash .25 .74 .01 9%
Debit Card .86 .14 .00 8%
Cash .22 .78 .00 1%
Credit Card .19 .11 .70 6%
Debit Card .69 .07 .24 5%
Credit Card .17 .15 .69 30%

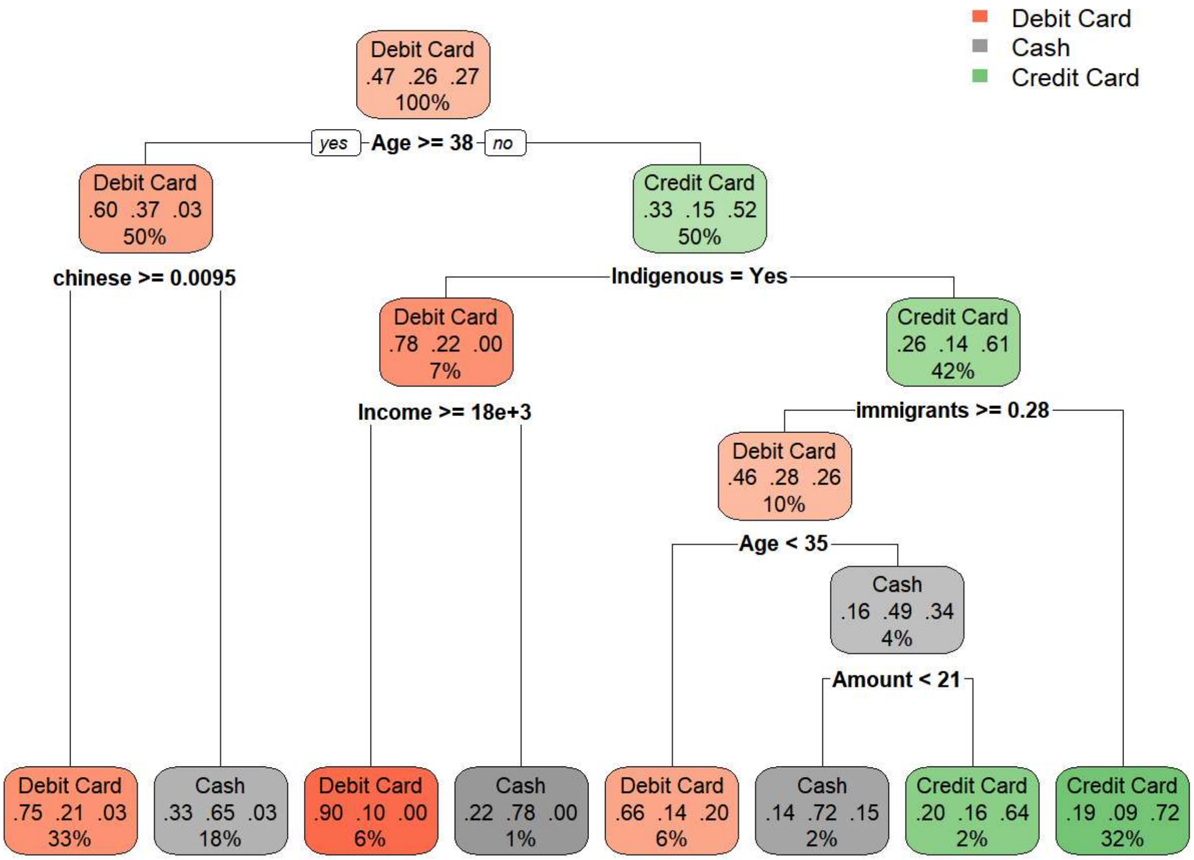
```
preds.shy <- predict(Treemodel.shy, type = "class")

table(df.train.tree$Method, preds.shy)
```

```
##              preds.shy
##               Debit Card Cash Credit Card
##   Debit Card        1549  197         261
##   Cash               434  461         213
##   Credit Card        118    4        1049
```

```
Treemodel.open <- rpart(Method~ ., data = df.train.tree,  method = "class")

rpart.plot(Treemodel.open)
```


Debit Card
Cash
Credit Card
Debit Card .47 .26 .27 100%
yes
Age >= 38
no
Debit Card .60 .37 .03 50%
Credit Card .33 .15 .52 50%
chinese >= 0.0095
Indigenous = Yes
Debit Card .78 .22 .00 7%
Credit Card .26 .14 .61 42%
Income >= 18e+3
immigrants >= 0.28
Debit Card .46 .28 .26 10%
Age < 35
Cash .16 .49 .34 4%
Amount < 21
Debit Card .75 .21 .03 33%
Cash .33 .65 .03 18%
Debit Card .90 .10 .00 6%
Cash .22 .78 .00 1%
Debit Card .66 .14 .20 6%
Cash .14 .72 .15 2%
Credit Card .20 .16 .64 2%
Credit Card .19 .09 .72 32%

```
preds.open <- predict(Treemodel.open, type = "class")

table(df.train.tree$Method, preds.open)
```

```
##                preds.open
##                 Debit Card Cash Credit Card
##    Debit Card         1452  275         280
##    Cash                361  608         139
##    Credit Card         101   36        1034
```

```
# 1- debit card
# 2- cash
# 3- credit card
```

Random Forrest

The random forest model should solve a lot of the problems that may lead to poor performance in the regression trees. They may be effective at reducing overfitting that occurs when I add the extra covariates. Unlike classification trees, which can easily be overfit to noisy or irrelevant predictors, random forests construct an ensemble of decision trees that are trained on different subsets of the data and features. By aggregating the predictions of multiple trees, the model should be able to reduce the variance and improve the stability of the predictions, while also capturing the non-linear interactions and correlations among predictors.

```
library(randomForest)

df.train.forrest <- df.train
df.train.forrest.cashonly <- df.train.cashonly

RandomForestModel.shy <- randomForest(Method ~ Amount + Gender+ Income + Age+ Education, data =
df.train.forrest, ntree = 300, mtry = 2)

RandomForestModel.shy$confusion
```

```
##             Debit Card Cash Credit Card class.error
## Debit Card        1579  212         216   0.2132536
## Cash               362  602         144   0.4566787
## Credit Card        111   41        1019   0.1298036
```

```
RandomForestModel.open <- randomForest(Method~., data = df.train.forrest, ntree = 300 , mtry =
2)
RandomForestModel.open$confusion
```

```
##             Debit Card Cash Credit Card class.error
## Debit Card        1439  284         284   0.2830095
## Cash               368  605         135   0.4539711
## Credit Card        116   80         975   0.1673783
```

Support Vector Machines

Support vector machines are another machine learning method that should be effective for high dimensional data with a low number of observations, particularly when there are clear decision boundaries. Unlike the multinomial logit models, support vector machines can capture non-linear decision boundaries and handle unbalanced data by minimizing a hinge loss function that penalizes misclassification errors. Because they don’t rely on fitting a distribution, they should perform reasonably well compared to multinomial logit models for the payment choice dataset. I’ve chosen radial kernel which maps the input features to a higher dimensional space, where the data can be more easily separated by a hyperplane. In the case of payment choice, the radial kernel can help to capture the complex and non-linear relationships between the covariates and the payment choice outcomes, such as the effects of income, age, education, and other factors on the likelihood of using different payment methods.

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.1.3
```

```
##
## Attaching package: 'e1071'
## The following object is masked from 'package:rsample':
##
##     permutations
```

```
svm.mod.shy <- svm(Method ~ Amount + Gender+ Income + Age+ Education, data = df.train.forrest, k
ernel = "radial", cost =1)

svm.mod.open <- svm(Method ~ ., data = df.train.forrest, kernel = "radial", cost = 1)
```

Evaluating Models on the testing set

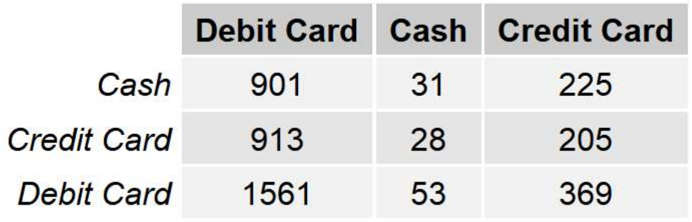
#Multinomial Logit Models

```
library(grid)
library(gridExtra)

Log.pred.1 <- predict(LogModel.shy, newx = df.test.cashonly[,c(2,4,5,8,9)], type = "class")
conf.log.shy <- table(df.test[,1], Log.pred.1)
conf.log.shy
```

```
##              Log.pred.1
##               Debit Card Cash Credit Card
##   Cash               901   31         225
##   Credit Card        913   28         205
##   Debit Card        1561   53         369
```

```
a <- grid.table(conf.log.shy)
```



	Debit Card	Cash	Credit Card
Cash	901	31	225
Credit Card	913	28	205
Debit Card	1561	53	369

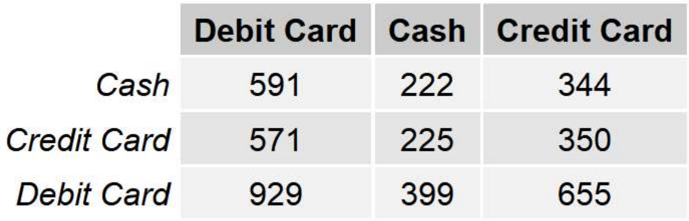
```
grid.newpage()
a
```

```
## NULL
```

```
Log.pred.2 <- predict(LogModel.open, newx = df.test, type = "class")
conf.log.open<- table(df.test[,1], Log.pred.2)
conf.log.open
```

```
##                Log.pred.2
##                 Debit Card Cash Credit Card
##   Cash                 591  222         344
##   Credit Card          571  225         350
##   Debit Card           929  399         655
```

```
a <- grid.table(conf.log.open)
```



	Debit Card	Cash	Credit Card
Cash	591	222	344
Credit Card	571	225	350
Debit Card	929	399	655

```
grid.newpage()
a
```

```
## NULL
```

Both the multinomial logit and the Shy models overestimate debit card usage in the data set. This could be due to a problem with multinomial logit models and unbalanced data, which can lead to a biased overestimation of the majority response variable. In this case, debit cards are the most used item in the survey, and the multinomial models predict debit card usage most frequently, which could be causing the overestimation of debit card usage.

Tree Models

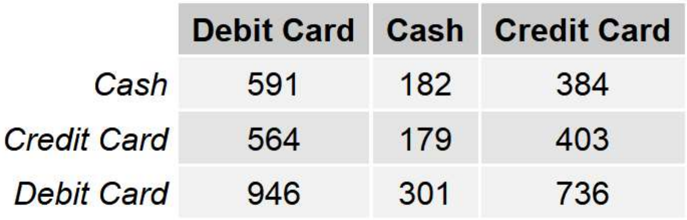
```
Tree.pred.shy <- predict(Treemodel.shy, newx = df.test[,-1], type = "class")

# Function to find column name with highest value in a row


conf.tree.shy <- table(df.test[,1], Tree.pred.shy)
conf.tree.shy
```

```
##              Tree.pred.shy
##               Debit Card Cash Credit Card
##   Cash               591  182         384
##   Credit Card        564  179         403
##   Debit Card         946  301         736
```

```
a <- grid.table(conf.tree.shy)
```



	Debit Card	Cash	Credit Card
Cash	591	182	384
Credit Card	564	179	403
Debit Card	946	301	736

```
grid.newpage()
a
```

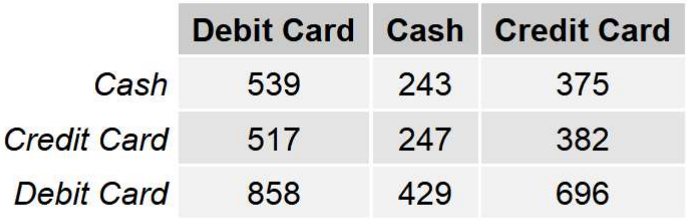
```
## NULL
```

```
Tree.pred.open <- predict(Treemodel.open, newx = df.test[,-1], type = "class")


conf.tree.open <- table(df.test[,1], Tree.pred.open)
conf.tree.open
```

```
##              Tree.pred.open
##               Debit Card Cash Credit Card
##   Cash               539  243         375
##   Credit Card        517  247         382
##   Debit Card         858  429         696
```

```
a <- grid.table(conf.tree.open)
```



	Debit Card	Cash	Credit Card
Cash	539	243	375
Credit Card	517	247	382
Debit Card	858	429	696

```
grid.newpage()
a
```

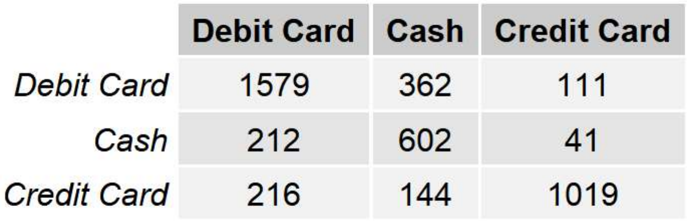
```
## NULL
```

random forrest model

```
Randomforestpred.shy <- predict(RandomForestModel.shy, newx = df.train)
conf.matrix.shy <- table(Randomforestpred.shy, df.train[,1])
conf.matrix.shy
```

```
##
## Randomforestpred.shy Debit Card Cash Credit Card
##          Debit Card        1579  362         111
##          Cash               212  602          41
##          Credit Card        216  144        1019
```

```
a <- grid.table(conf.matrix.shy)
```



	Debit Card	Cash	Credit Card
Debit Card	1579	362	111
Cash	212	602	41
Credit Card	216	144	1019

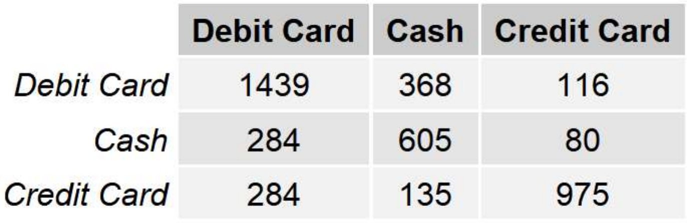
```
grid.newpage()
a
```

```
## NULL
```

```
Randomforestpred.open <- predict(RandomForestModel.open ,newx = df.train)
conf.matrix.open <- table(Randomforestpred.open, df.train[,1])
conf.matrix.open
```

```
##
## Randomforestpred.open Debit Card Cash Credit Card
##           Debit Card        1439  368         116
##           Cash               284  605          80
##           Credit Card        284  135         975
```

```
a <- grid.table(conf.matrix.open)
```



	Debit Card	Cash	Credit Card
Debit Card	1439	368	116
Cash	284	605	80
Credit Card	284	135	975

```
grid.newpage()
a
```

```
## NULL
```

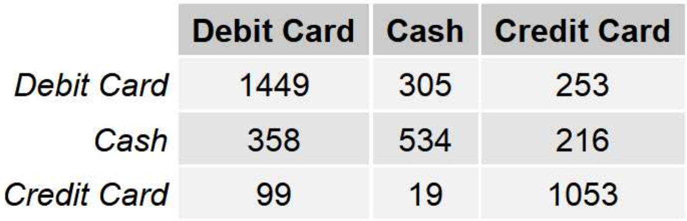
Comparing the two models and the different sets of covariates, by looking at their confusion matrices. The model with fewer covariates outperforms the model with the added covariates in terms of out of sample prediction.

support vector machines

```
SVM.pred.shy <- predict( svm.mod.shy, newdata = df.train[,-1] )
conf.matrix.svm.shy <- table(df.train[,1], SVM.pred.shy)
conf.matrix.svm.shy
```

```
##                SVM.pred.shy
##                 Debit Card Cash Credit Card
##   Debit Card         1449  305         253
##   Cash                358  534         216
##   Credit Card          99   19        1053
```

```
a <- grid.table(conf.matrix.svm.shy)
```



	Debit Card	Cash	Credit Card
Debit Card	1449	305	253
Cash	358	534	216
Credit Card	99	19	1053

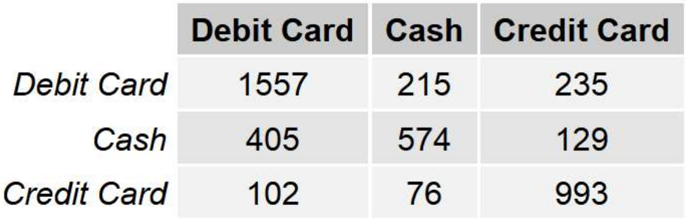
```
grid.newpage()
a
```

```
## NULL
```

```
SVM.pred.open <- predict(svm.mod.open, newdata = df.train[,-1])
conf.matrix.open <- table(df.train[,1], SVM.pred.open)
conf.matrix.open
```

```
##              SVM.pred.open
##               Debit Card Cash Credit Card
##   Debit Card        1557  215         235
##   Cash               405  574         129
##   Credit Card        102   76         993
```

```
a <- grid.table(conf.matrix.open)
```



	Debit Card	Cash	Credit Card
Debit Card	1557	215	235
Cash	405	574	129
Credit Card	102	76	993

```
grid.newpage()
a
```

```
## NULL
```

The support vector machines preform much better than the classification tree and the multinomial logit models, but slightly worse than the random forests when using the reduced number of covariates. Assessing the confusion matrix for the model with the added covariates

We see again that the added covariates don’t necessarily improve model performance which means the model is likely being overfitted. Even with the support vector machines and Random Forest methods the models with fewer covariates consistently outperform the models with more added covariates.

Question #4

From a high-level view working through question three highlighted the importance of thoroughly understanding the structure of a dataset before attempting to model it. In this case, the data had several problematic features, such as high correlation between covariates, many covariates, unbalanced dummy variables, and an unbalanced response variable. Thus, thinking about what methods are best equipped to deal with those issues was an interesting exercise. But by developing a thorough understanding of data structure, we can select the appropriate machine learning algorithms and techniques to optimize model performance. This exercise emphasizes the importance of exploring and preprocessing data before applying machine learning methods to it. In evaluating how the models preformed when predicting out of sample and with the models that had the added covariates, I suspect that overfitting was leading to their underperformance compared to the simple models with limited covariates. To address the overfitting issue, I would have liked to have tried a couple different strategies. The first strategy would be to use a multinomial logistic lasso to choose the most important covariates from the FSA pool before implementing additional models. One of the problems with this approach would be to take the factor variables and create multiple grouping for the lasso to select from. Another approach that may improve the performance of the regression trees is pruning the trees and eliminating covariates based on economic intuition. Finally, performing principal component analysis before the using trees could be helpful, as many of the additional FSA covariates are highly collinear. Using PCA may have allowed me to extract some variation across the transactions that could be useful to identify factors that contribute to payment preference. By using these strategies, I think I could have reduced overfitting which I suspect led to worse model performance. While these methods may have helped improve the predictive performance of the various models, except for group lasso for FSA characteristics, they would lead to interpretability problems which would limit the practical application of any of the above techniques. Using a method like PCA or Random Forests lead to models that can’t be easily interpreted. Using a grouped lasso method may be more appropriate as it allows you to select variables according to their relevance, which would shed some light on the relationship between FSA characteristics and payment choice.

From these models the added covariates don’t seem to make a significant impact on any of the models predictive performance. We also see that the relationships established by Shy between payment method, and his chosen set of covariates seem to hold when tested on this data set.