Project 2: Data Mining, Classification, Prediction

SDS322E

Mining, Classification, Prediction

Lindsay Chu

Introduction

My dataset, "credit_card_data", comes from Kaggle; it summarizes credit card usage of 8950 unique users based on 18 variables. Most are fairly straightforward, such as CUSTID, BALANCE, PURCHASES, and CREDIT LIMIT; others are scores from 0-1 and measure frequency, such as BALANCE_FREQUENCY (how often the balance is updated) and CASH_ADVANCE_FREQUENCY (how often payments are made in cash in advance).

I created the categorical variable, "AT_RISK", which assigns the value "TRUE" to customers whose credit utilization rates are greater than 30%, and "FALSE" otherwise. Credit utilization was calculated by dividing "BALANCE" by "CREDIT LIMIT", and the minimum threshold of 30% was chosen based on external research on "good" versus "poor" credit scores. Consequently, there are 2347 users "at-risk" of low credit scores / default ("TRUE" category), and 2128 users who are not at risk.

NOTE: I ended up cutting dataset in half (i.e. using the first 4475 rows) because R would not load any visualizations with all 8950 observations.

```
library(tidyverse)
library(dplyr)

# read dataset
ccdata <- read_csv("credit_card_data.csv")

glimpse(ccdata)
```

```
## Rows: 8,950
## Columns: 18
## $ CUST_ID                          <chr> "C10001", "C10002", "C10003", "C10004~
## $ BALANCE                          <dbl> 40.90075, 3202.46742, 2495.14886, 166~
## $ BALANCE_FREQUENCY                <dbl> 0.818182, 0.909091, 1.000000, 0.63636~
## $ PURCHASES                        <dbl> 95.40, 0.00, 773.17, 1499.00, 16.00, ~
## $ ONEOFF_PURCHASES                 <dbl> 0.00, 0.00, 773.17, 1499.00, 16.00, 0~
## $ INSTALLMENTS_PURCHASES           <dbl> 95.40, 0.00, 0.00, 0.00, 0.00, 1333.2~
## $ CASH_ADVANCE                     <dbl> 0.0000, 6442.9455, 0.0000, 205.7880, ~
## $ PURCHASES_FREQUENCY              <dbl> 0.166667, 0.000000, 1.000000, 0.08333~
## $ ONEOFF_PURCHASES_FREQUENCY       <dbl> 0.000000, 0.000000, 1.000000, 0.08333~
## $ PURCHASES_INSTALLMENTS_FREQUENCY <dbl> 0.083333, 0.000000, 0.000000, 0.00000~
## $ CASH_ADVANCE_FREQUENCY           <dbl> 0.000000, 0.250000, 0.000000, 0.08333~
## $ CASH_ADVANCE_TRX                 <dbl> 0, 4, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ PURCHASES_TRX                    <dbl> 2, 0, 12, 1, 1, 8, 64, 12, 5, 3, 12, ~
## $ CREDIT_LIMIT                     <dbl> 1000, 7000, 7500, 7500, 1200, 1800, 1~
## $ PAYMENTS                         <dbl> 201.8021, 4103.0326, 622.0667, 0.0000~
```

```
## $ MINIMUM_PAYMENTS                  <dbl> 139.50979, 1072.34022, 627.28479, NA,~
## $ PRC_FULL_PAYMENT                  <dbl> 0.000000, 0.222222, 0.000000, 0.00000~
## $ TENURE                            <dbl> 12, 12, 12, 12, 12, 12, 12, 12, 12, 1~
```

```
# tidying data

# select first half of the dataset because R would not load any visualizations with all of the observat
ccdata <- ccdata[1:4475,] %>%
  # shorten variable names for readability in visualizations
  rename(BAL_FREQ = BALANCE_FREQUENCY,
         ONEOFF = ONEOFF_PURCHASES,
         INSTALLMENTS = INSTALLMENTS_PURCHASES,
         PURCH_FREQ = PURCHASES_FREQUENCY,
         ONEOFF_FREQ = ONEOFF_PURCHASES_FREQUENCY,
         INSTALL_FREQ = PURCHASES_INSTALLMENTS_FREQUENCY,
         CASHADV_FREQ = CASH_ADVANCE_FREQUENCY,
         MIN_PAY = MINIMUM_PAYMENTS) %>%

  # create categorical variable "AT_RISK", which will be used later as the response variable in the Cla
  mutate(CREDIT_UTIL = round(BALANCE / CREDIT_LIMIT, 2),
         AT_RISK = ifelse(CREDIT_UTIL > 0.30, "TRUE", "FALSE"))

ccdata %>% filter(AT_RISK=="TRUE") %>%
  summarize(customers_at_risk = n(), not_at_risk = 4475-customers_at_risk)
```

```
## # A tibble: 1 x 2
##   customers_at_risk not_at_risk
##               <int>       <dbl>
## 1              2347        2128
```

```
# check for NA values
ccdata %>% summarize_all(function(x)sum(is.na(x)))
```

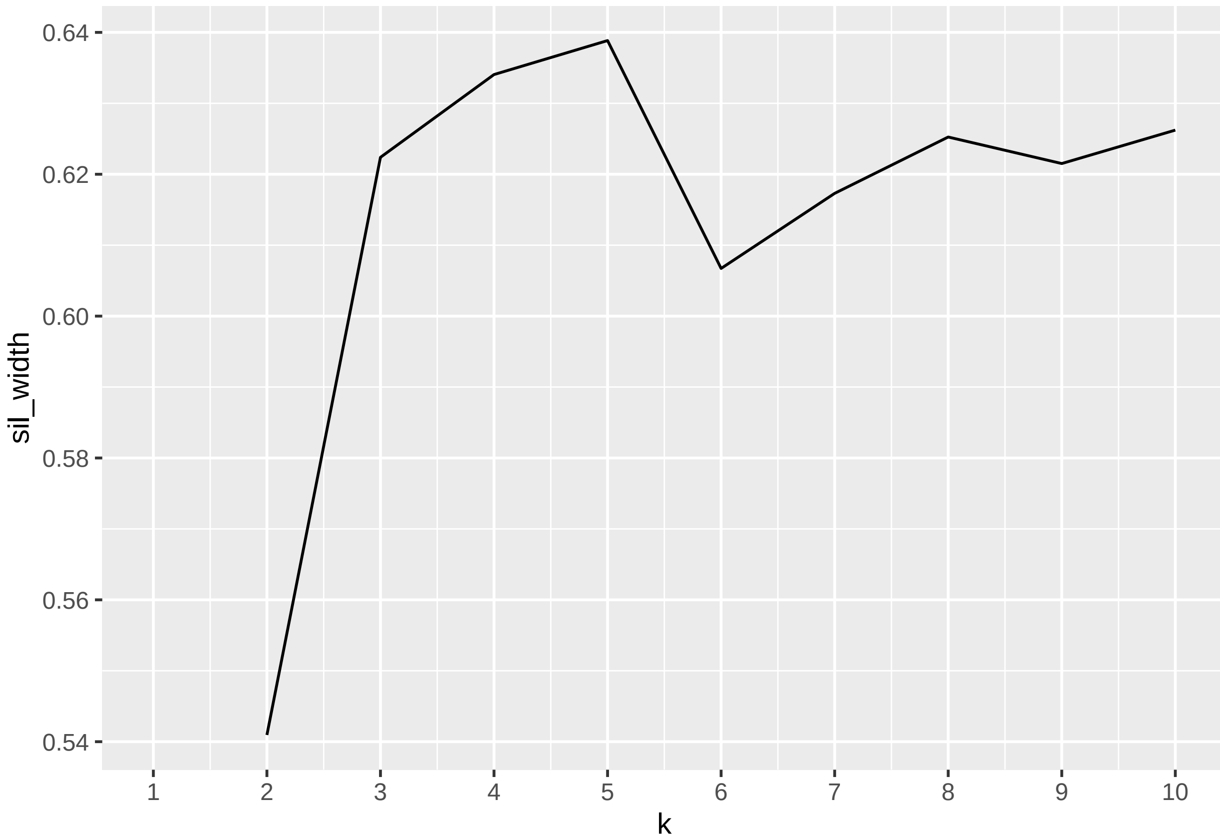
```
## # A tibble: 1 x 20
##   CUST_ID BALANCE BAL_F~1 PURCH~2 ONEOFF INSTA~3 CASH_~4 PURCH~5 ONEOF~6 INSTA~7
##     <int>   <int>   <int>   <int>  <int>   <int>   <int>   <int>   <int>   <int>
## 1       0       0       0       0      0       0       0       0       0       0
## # ... with 10 more variables: CASHADV_FREQ <int>, CASH_ADVANCE_TRX <int>,
## #   PURCHASES_TRX <int>, CREDIT_LIMIT <int>, PAYMENTS <int>, MIN_PAY <int>,
## #   PRC_FULL_PAYMENT <int>, TENURE <int>, CREDIT_UTIL <int>, AT_RISK <int>, and
## #   abbreviated variable names 1: BAL_FREQ, 2: PURCHASES, 3: INSTALLMENTS,
## #   4: CASH_ADVANCE, 5: PURCH_FREQ, 6: ONEOFF_FREQ, 7: INSTALL_FREQ
## # i Use `colnames()` to see all variable names
```

```
# for variables "CREDIT_LIMIT" and "MINIMUM_PAY", replace NA values with mean
ccdata$CREDIT_LIMIT[is.na(ccdata$CREDIT_LIMIT)] <- mean(ccdata$CREDIT_LIMIT, na.rm = TRUE)
ccdata$MIN_PAY[is.na(ccdata$MIN_PAY)] <- mean(ccdata$MIN_PAY, na.rm = TRUE)
```

Cluster Analysis

```
library(cluster)

pam_dat <- ccdata %>% select(BAL_FREQ, PURCH_FREQ, INSTALL_FREQ, ONEOFF_FREQ, CASHADV_FREQ, MIN_PAY)
sil_width <- vector()
for(i in 2:10){
  pam_fit <- pam(pam_dat, k = i)
```

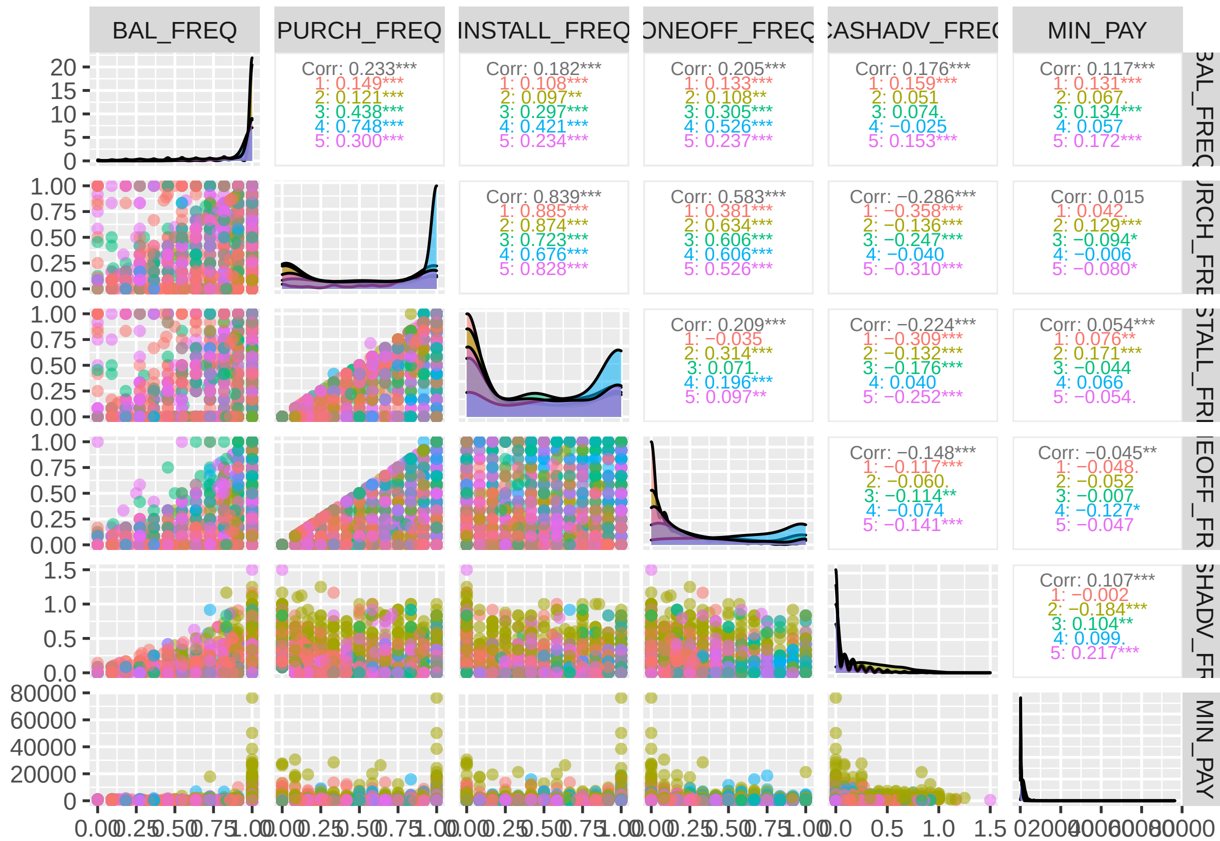
```
  sil_width[i] <-pam_fit$silinfo$avg.width
}
ggplot()+
  geom_line(aes(x=1:10,y=sil_width))+
  scale_x_continuous(name="k", breaks=1:10)
```




```
# looks like five clusters is best!

pam1 <- pam_dat %>% pam(k=5)
pam1$silinfo$avg.width # 0.6275
```

```
## [1] 0.6388471
```

```
# cluster visualization
library(GGally)
clust <- ccdata %>% pam(k=5)
ccdata %>%
  select(BAL_FREQ, PURCH_FREQ, INSTALL_FREQ, ONEOFF_FREQ, CASHADV_FREQ, MIN_PAY) %>%
  mutate(cluster=as.factor(clust$clustering)) %>%
  ggpairs(columns=1:6, aes(color=cluster, alpha = 0.5),
          upper = list(continuous = wrap("cor", size = 2.5)))
```



I performed PAM clustering on the following variables: balance frequency, purchase frequency, installment frequency, oneoff frequency, cash advance frequency, and minimum payments. To determine the ideal number of k clusters, I calculated the silhouette width from k=2 to k=10. In my analysis, 5 clusters returned the highest average silhouette width, at 0.63. This value indicates a reasonable clustering structure.

Subsequently, I used ggpairs() to visualize all pairwise variable combinations and colored them by the 5 cluster assignments. The pair with the highest positive correlation is installment frequency and purchase frequency (corr=0.839), and the pair with the most negative correlation is cash advance frequency and purchase frequency (corr=-0.286).

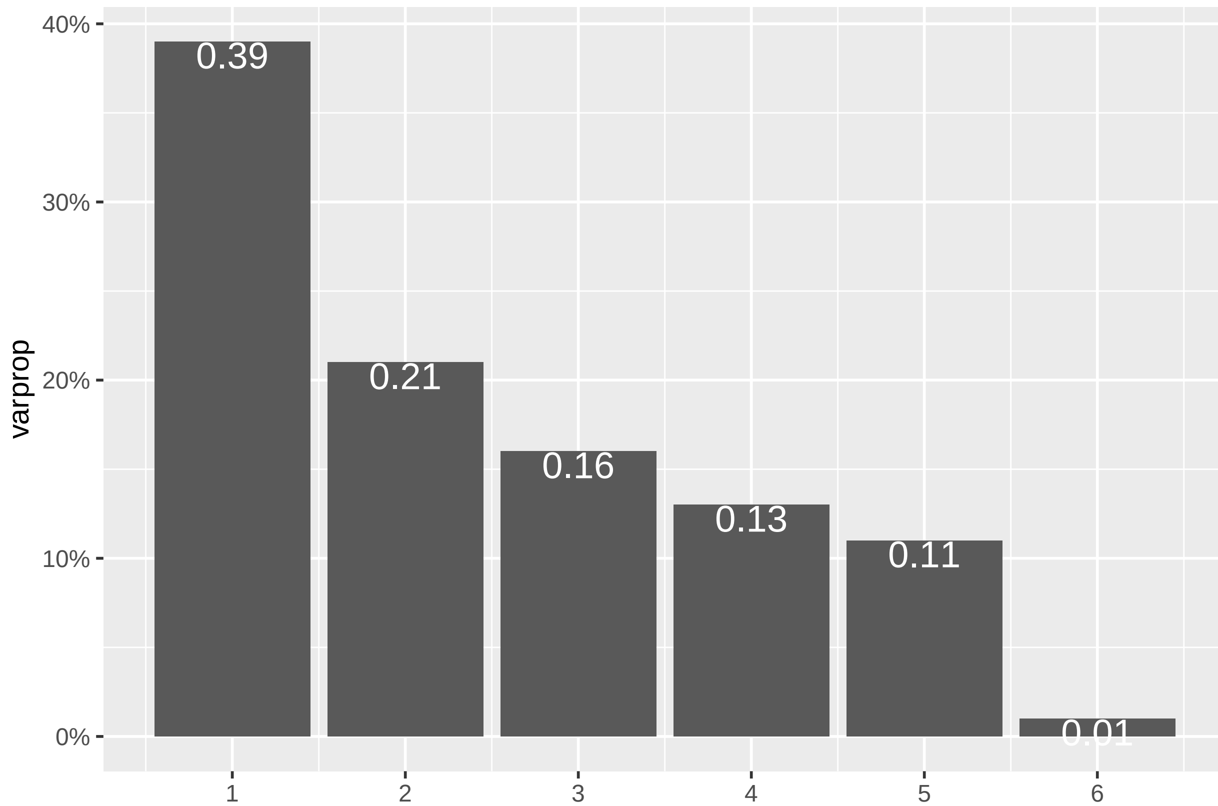
Dimensionality Reduction with PCA

```r
# scale data and summarize PCA results
ccdata %>% select(BAL_FREQ, PURCH_FREQ, INSTALL_FREQ, ONEOFF_FREQ, CASHADV_FREQ, MIN_PAY) %>% scale %>%

summary(ccdata_pca, loadings=T)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4    Comp.5
## Standard deviation     1.5211153 1.1266278 0.9775104 0.8701242 0.7971574
## Proportion of Variance 0.3857182 0.2115957 0.1592900 0.1262142 0.1059337
## Cumulative Proportion  0.3857182 0.5973138 0.7566039 0.8828181 0.9887517
##                            Comp.6
## Standard deviation     0.25975852
## Proportion of Variance 0.01124826
## Cumulative Proportion  1.00000000
##
## Loadings:
```

```
##              Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## BAL_FREQ      0.224  0.620  0.330  0.124  0.664
## PURCH_FREQ    0.634                      -0.191 -0.742
## INSTALL_FREQ  0.547        -0.271  0.512 -0.152  0.585
## ONEOFF_FREQ   0.431         0.414 -0.688 -0.252  0.325
## CASHADV_FREQ -0.249  0.602  0.271  0.256 -0.660
## MIN_PAY              0.502 -0.756 -0.420
```

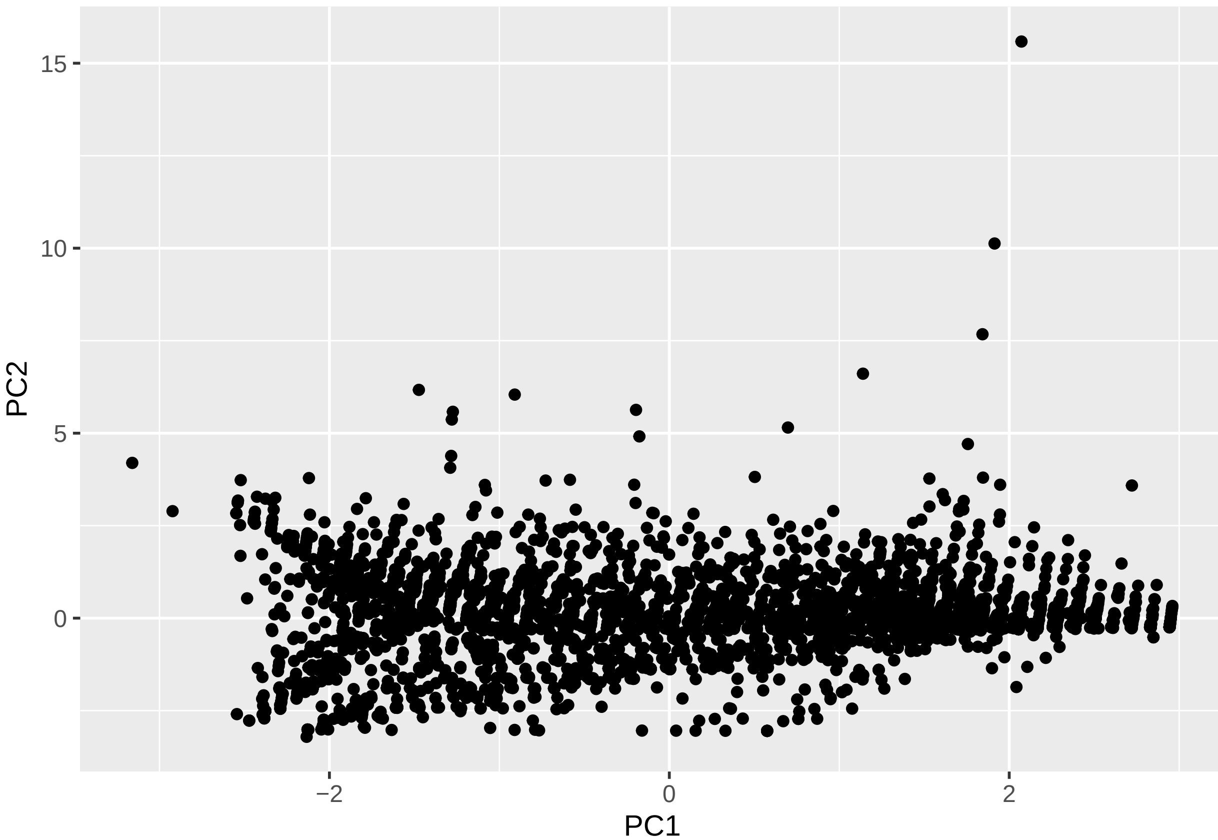
```
# determine number of PCs to keep
eigval <- ccdata_pca$sdev^2
varprop=round(eigval/sum(eigval), 2)

eigval<-ccdata_pca$sdev^2
varprop=round(eigval/sum(eigval), 2)
ggplot() + geom_bar(aes(y=varprop, x=1:6), stat="identity") + xlab("") +
  geom_text(aes(x=1:6, y=varprop, label=round(varprop, 2)), vjust=1, col="white", size=5) +
  scale_y_continuous(breaks=seq(0, .5, .1), labels = scales::percent) +
  scale_x_continuous(breaks=1:6)
```



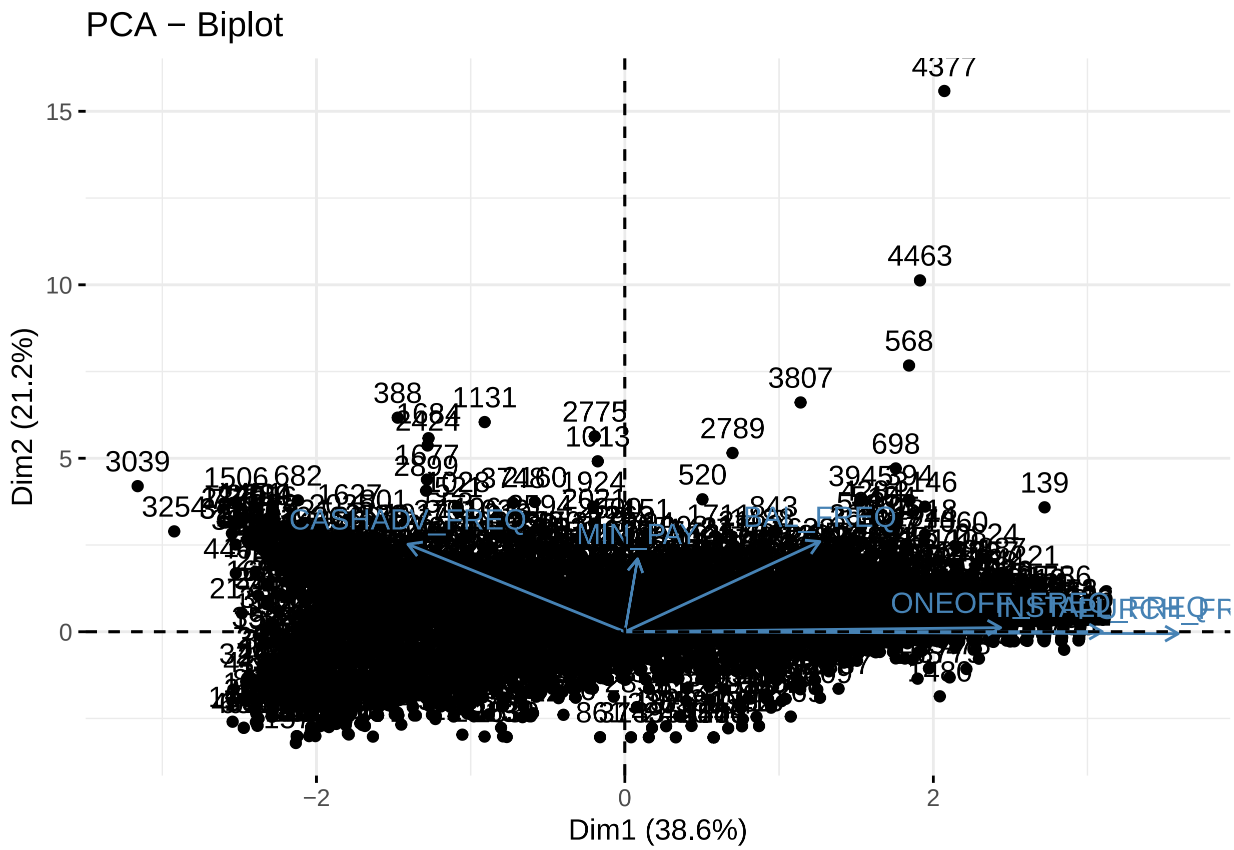

```
eigval # eigenvalues
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
## 2.31379178 1.26929031 0.95552661 0.75711613 0.63545990 0.06747449
```

```
# plot scores to show data with respect to 2 PCs
ccdataf <- data.frame(Customer=ccdata$CUST_ID, PC1=ccdata_pca$scores[,1], PC2=ccdata_pca$scores[,2])

ggplot(ccdataf, aes(PC1, PC2)) + geom_point()
```




```
library(factoextra)
fviz_pca_biplot(ccdata_pca)
```



To determine the number of Principal Components (PCs) to keep, I used a scree barplot to analyze the proportion of variance explained by each. Based on the graph, the plot flattens / creates an elbow at around PC2, suggesting that only 2 PCs should be kept. This is supported by Kaiser's rule, which holds that PCs with eigenvalues > 1 should be kept; PC1 and PC2 have eigenvalues of 2.3135 and 1.2689, respectively, while the others have values below 1. Based on the proportions, PC1 explains about 38.57% of the total variance and PC2 explains 21.15%.

Each PC's loadings (eigenvectors) indicate the strength and direction of the association between the PC and a particular variable. PC1 has a loading of 0.635 for purchase_frequency and 0.547 in install_frequency; the positive associations for both indicate that this component contains individuals who make purchases very frequently, both in immediate transactions and in installments. PC2 has fairly large positive loadings for balance frequency and cash in advance frequency, suggesting that individuals in this component tend to update their balances and make advanced payments frequently.

Linear Classifier

```
# logistic regression
logistic_fit <- glm(AT_RISK == "TRUE" ~ ONEOFF + INSTALLMENTS + CASH_ADVANCE + BAL_FREQ + PURCH_FREQ + (

log_score <- predict(logistic_fit, type="response")

class_diag(log_score, ccdata$AT_RISK, positive="TRUE")
```

```
##            acc  sens   spec    ppv     f1     ba    auc
## Metrics 0.8704 0.896 0.8421 0.8622 0.8788 0.8691 0.9435
```

```
# AUC: 0.9435
```

```r
# report a confusion matrix
table(actual = ccdata$AT_RISK, predicted = log_score > .5) %>% addmargins()
```

```
##        predicted
## actual  FALSE TRUE  Sum
##   FALSE  1792  336 2128
##   TRUE    244 2103 2347
##   Sum    2036 2439 4475
```

```r
TNR <- 1792 / 2128 # 0.842 (Specificity)
FP <- 1 - TNR       # 0.158
TPR <- 2103 / 2347 # 0.896 (Sensitivity / Recall)
FN <- 1 - TPR       # 0.104

# Positive Predictive Value (PPV) / Precision
PPV <- TPR / (TPR + FP)
PPV #0.850
```

```
## [1] 0.8501851
```

```r
# perform k-fold CV on this same model
set.seed(1234)
k=10
data <- ccdata[sample(nrow(ccdata)),] #randomly order rows
folds<-cut(seq(1:nrow(data)),breaks=k,labels=F) #create folds
diags<-NULL

for(i in 1:k){
  ## create training and test sets
  train<-data[folds!=i,]
  test<-data[folds==i,]
  truth<-test$AT_RISK

  ## train model on training set
  fit<-glm(AT_RISK == "TRUE" ~ ONEOFF + INSTALLMENTS + CASH_ADVANCE + BAL_FREQ + PURCH_FREQ + ONEOFF_FRE
  probs<-predict(fit,newdata = test,type="response")

  ## test model on test set (save all k results)
  diags<-rbind(diags,class_diag(probs,truth, positive="TRUE"))
}
summarize_all(diags,mean) # AUC: 0.9427
```

```
##       acc    sens    spec     ppv      f1      ba     auc
## 1 0.86996 0.89599 0.84104 0.86175 0.87833 0.86853 0.94275
```

For my classifier methods, I used "AT_RISK" as my response variable and the following variables as my predictors: ONEOFF, INSTALLMENTS, CASH_ADVANCE, BAL_FREQ, PURCH_FREQ, ONEOFF_FREQ, INSTALL_FREQ, CASHADV_FREQ, MIN_PAY, and PRC_FULL_PAYMENT. Logistic regression had an area under the curve (AUC) value of 0.9435, indicating that its predictive performance was very good. The confusion matrix also found that the logistic model could correctly classify AT_RISK (low credit score individuals) about 89.6% of the time, and non-risk individuals 84.2% of the time. Overall precision was about 85%. The AUC value for cross-validation (CV) was 0.9427; since this was only slightly lower than the original AUC value, the model did not appear to show signs of overfitting.

Non-Parametric Classifier

```
library(caret)

knn_fit <- knn3(AT_RISK == "TRUE" ~ ONEOFF + INSTALLMENTS + CASH_ADVANCE + BAL_FREQ + PURCH_FREQ + ONEO

y_hat_knn <- predict(knn_fit, ccdata)

class_diag(y_hat_knn[,2], ccdata$AT_RISK, positive="TRUE") # AUC: 0.9475
```

```
##            acc   sens   spec    ppv     f1     ba    auc
## Metrics 0.8708 0.8769 0.8642 0.8769 0.8769 0.8705 0.9475
```

```
# confusion matrix
table(actual = ccdata$AT_RISK, predicted = y_hat_knn[,2] > .5) %>% addmargins
```

```
##        predicted
## actual  FALSE TRUE  Sum
##   FALSE  1839  289 2128
##   TRUE    289 2058 2347
##   Sum    2128 2347 4475
```

```
TNR <- 1840 / 2128 # 0.865 (Specificity)
FP <- 1 - TNR       # 0.135
TPR <- 2058 / 2347 # 0.877 (Sensitivity / Recall)
FN <- 1 - TPR       # 0.123

# Positive Predictive Value (PPV) / Precision
PPV <- TPR / (TPR + FP)
PPV #0.866
```

```
## [1] 0.8662932
```

```
# k-fold CV on the same model
k=10
data<-ccdata[sample(nrow(ccdata)),]
folds<-cut(seq(1:nrow(ccdata)),breaks=k,labels=F)
diags<-NULL
for(i in 1:k){
  train<-data[folds!=i,]
  test<-data[folds==i,]
  truth<-test$AT_RISK

  fit<-knn3(AT_RISK == "TRUE" ~ ONEOFF + INSTALLMENTS + CASH_ADVANCE + BAL_FREQ + PURCH_FREQ + ONEOFF_F
  probs<-predict(fit,newdata = test)[,2]

  diags<-rbind(diags,class_diag(probs,truth, positive="TRUE"))
}
summarize_all(diags,mean) # AUC: 0.9012
```

```
##       acc    sens    spec    ppv      f1      ba     auc
## 1 0.84871 0.86918 0.82528 0.8462 0.85738 0.84724 0.90202
```

For my nonparametric model, I used k-nearest-neighbors (KNN). This method returned an AUC value of 0.9475, indicating good predictive performance. However, for its cross-validation, the AUC value was 0.9012, a noticeable decrease and potential result of overfitting. It appeared that the logistic method (linear classifier) had a better CV performance than kNN.

Regression/Numeric Prediction

```r
# Fit a linear regression model or regression tree to your entire dataset, predicting one of your numer
fit <- lm(AT_RISK == "TRUE" ~ ONEOFF + INSTALLMENTS + CASH_ADVANCE + BAL_FREQ + PURCH_FREQ + ONEOFF_FRE
probs <- predict(fit)
class_diag(probs, ccdata$AT_RISK, positive="TRUE") # AUC: 0.9035
```

```
##            acc   sens   spec    ppv    f1     ba    auc
## Metrics 0.8322 0.9433 0.7096 0.7818 0.855 0.8265 0.9036
```

```r
lm_summary <- summary(fit)

# calculate MSE for the overall dataset
mean(lm_summary$residuals^2)
```

```
## [1] 0.138627
```

```r
k=10
data<-ccdata[sample(nrow(ccdata)),]
folds<-cut(seq(1:nrow(ccdata)),breaks=k,labels=F)
diags<-NULL
for(i in 1:k){
  train<-data[folds!=i,]
  test<-data[folds==i,]
  truth<-test$AT_RISK

  fit<-lm(AT_RISK == "TRUE" ~ ONEOFF + INSTALLMENTS + CASH_ADVANCE + BAL_FREQ + PURCH_FREQ + ONEOFF_FRE
  probs<-predict(fit,newdata = test)

  diags<-rbind(diags,class_diag(probs,truth, positive="TRUE"))
}
summarize_all(diags,mean) # AUC: 0.9027
```

```
##      acc    sens    spec     ppv      f1      ba    auc
## 1 0.8286 0.94301 0.70182 0.77802 0.85241 0.82242 0.9025
```

The dataset overall had a mean standard error (MSE) of 0.1372. The linear regression had an AUC of 0.9035 and the cross-validation reported an AUC of 0.9027; this was only a slight decrease, so overfitting was unlikely, but overall this classifier method exhibited the weakest predictive performance.