credit card segmentation

Jeevan Hemmanna B

17 January, 2020

## Problem Statement

This case requires trainees to develop a customer segmentation to define marketing strategy. The sample dataset summarizes the usage behaviour of about 9000 active credit card holders during the last 6 months. The file is at a customer level with 18 behavioural variables.

## Data PreProcessing

Here we are now doing a data preprocessing and we will be cresting the New KPI’s and also we will be doing data scaling and treating the outliers.

set.seed(25)  
credit = read.csv('credit-card-data.csv',header = T , na.strings = c(" ","","NA","NAN"))  
  
str(credit)

## 'data.frame': 8950 obs. of 18 variables:  
## $ CUST\_ID : Factor w/ 8950 levels "C10001","C10002",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ BALANCE : num 40.9 3202.5 2495.1 1666.7 817.7 ...  
## $ BALANCE\_FREQUENCY : num 0.818 0.909 1 0.636 1 ...  
## $ PURCHASES : num 95.4 0 773.2 1499 16 ...  
## $ ONEOFF\_PURCHASES : num 0 0 773 1499 16 ...  
## $ INSTALLMENTS\_PURCHASES : num 95.4 0 0 0 0 ...  
## $ CASH\_ADVANCE : num 0 6443 0 206 0 ...  
## $ PURCHASES\_FREQUENCY : num 0.1667 0 1 0.0833 0.0833 ...  
## $ ONEOFF\_PURCHASES\_FREQUENCY : num 0 0 1 0.0833 0.0833 ...  
## $ PURCHASES\_INSTALLMENTS\_FREQUENCY: num 0.0833 0 0 0 0 ...  
## $ CASH\_ADVANCE\_FREQUENCY : num 0 0.25 0 0.0833 0 ...  
## $ CASH\_ADVANCE\_TRX : int 0 4 0 1 0 0 0 0 0 0 ...  
## $ PURCHASES\_TRX : int 2 0 12 1 1 8 64 12 5 3 ...  
## $ CREDIT\_LIMIT : num 1000 7000 7500 7500 1200 1800 13500 2300 7000 11000 ...  
## $ PAYMENTS : num 202 4103 622 0 678 ...  
## $ MINIMUM\_PAYMENTS : num 140 1072 627 NA 245 ...  
## $ PRC\_FULL\_PAYMENT : num 0 0.222 0 0 0 ...  
## $ TENURE : int 12 12 12 12 12 12 12 12 12 12 ...

summary(credit)

## CUST\_ID BALANCE BALANCE\_FREQUENCY PURCHASES   
## C10001 : 1 Min. : 0.0 Min. :0.0000 Min. : 0.00   
## C10002 : 1 1st Qu.: 128.3 1st Qu.:0.8889 1st Qu.: 39.63   
## C10003 : 1 Median : 873.4 Median :1.0000 Median : 361.28   
## C10004 : 1 Mean : 1564.5 Mean :0.8773 Mean : 1003.20   
## C10005 : 1 3rd Qu.: 2054.1 3rd Qu.:1.0000 3rd Qu.: 1110.13   
## C10006 : 1 Max. :19043.1 Max. :1.0000 Max. :49039.57   
## (Other):8944   
## ONEOFF\_PURCHASES INSTALLMENTS\_PURCHASES CASH\_ADVANCE PURCHASES\_FREQUENCY  
## Min. : 0.0 Min. : 0.0 Min. : 0.0 Min. :0.00000   
## 1st Qu.: 0.0 1st Qu.: 0.0 1st Qu.: 0.0 1st Qu.:0.08333   
## Median : 38.0 Median : 89.0 Median : 0.0 Median :0.50000   
## Mean : 592.4 Mean : 411.1 Mean : 978.9 Mean :0.49035   
## 3rd Qu.: 577.4 3rd Qu.: 468.6 3rd Qu.: 1113.8 3rd Qu.:0.91667   
## Max. :40761.2 Max. :22500.0 Max. :47137.2 Max. :1.00000   
##   
## ONEOFF\_PURCHASES\_FREQUENCY PURCHASES\_INSTALLMENTS\_FREQUENCY  
## Min. :0.00000 Min. :0.0000   
## 1st Qu.:0.00000 1st Qu.:0.0000   
## Median :0.08333 Median :0.1667   
## Mean :0.20246 Mean :0.3644   
## 3rd Qu.:0.30000 3rd Qu.:0.7500   
## Max. :1.00000 Max. :1.0000   
##   
## CASH\_ADVANCE\_FREQUENCY CASH\_ADVANCE\_TRX PURCHASES\_TRX CREDIT\_LIMIT   
## Min. :0.0000 Min. : 0.000 Min. : 0.00 Min. : 50   
## 1st Qu.:0.0000 1st Qu.: 0.000 1st Qu.: 1.00 1st Qu.: 1600   
## Median :0.0000 Median : 0.000 Median : 7.00 Median : 3000   
## Mean :0.1351 Mean : 3.249 Mean : 14.71 Mean : 4494   
## 3rd Qu.:0.2222 3rd Qu.: 4.000 3rd Qu.: 17.00 3rd Qu.: 6500   
## Max. :1.5000 Max. :123.000 Max. :358.00 Max. :30000   
## NA's :1   
## PAYMENTS MINIMUM\_PAYMENTS PRC\_FULL\_PAYMENT TENURE   
## Min. : 0.0 Min. : 0.02 Min. :0.0000 Min. : 6.00   
## 1st Qu.: 383.3 1st Qu.: 169.12 1st Qu.:0.0000 1st Qu.:12.00   
## Median : 856.9 Median : 312.34 Median :0.0000 Median :12.00   
## Mean : 1733.1 Mean : 864.21 Mean :0.1537 Mean :11.52   
## 3rd Qu.: 1901.1 3rd Qu.: 825.49 3rd Qu.:0.1429 3rd Qu.:12.00   
## Max. :50721.5 Max. :76406.21 Max. :1.0000 Max. :12.00   
## NA's :313

missing\_val = data.frame(apply(credit, 2, function(x){sum(is.na(x))}))  
missing\_val

## apply.credit..2..function.x...  
## CUST\_ID 0  
## BALANCE 0  
## BALANCE\_FREQUENCY 0  
## PURCHASES 0  
## ONEOFF\_PURCHASES 0  
## INSTALLMENTS\_PURCHASES 0  
## CASH\_ADVANCE 0  
## PURCHASES\_FREQUENCY 0  
## ONEOFF\_PURCHASES\_FREQUENCY 0  
## PURCHASES\_INSTALLMENTS\_FREQUENCY 0  
## CASH\_ADVANCE\_FREQUENCY 0  
## CASH\_ADVANCE\_TRX 0  
## PURCHASES\_TRX 0  
## CREDIT\_LIMIT 1  
## PAYMENTS 0  
## MINIMUM\_PAYMENTS 313  
## PRC\_FULL\_PAYMENT 0  
## TENURE 0

credit$CREDIT\_LIMIT[is.na(credit$CREDIT\_LIMIT)] = median(credit$CREDIT\_LIMIT, na.rm = T)  
credit$MINIMUM\_PAYMENTS[is.na(credit$MINIMUM\_PAYMENTS)] = median(credit$MINIMUM\_PAYMENTS, na.rm = T)  
sum(is.na(credit))

## [1] 0

## Deriving New KPI’s

Here we are deriving New KPI as asked in a problem statements.

* Monthly average perchase is purchases by tenure
* monthly cash advance is cash advance by tenure
* creating categories by using One\_off purchase and installment purchases
* limit usage is basically ratio of balance to limit usage.Lower limit usage implies cutomers are maintaing thier balance properly and it also mean that they have a good credit score
* Pay\_minPay is the ratio of payments to minimum payments

These are the new KPI’s formed

#New Variables creation#   
credit$Monthly\_Avg\_PURCHASES <- credit$PURCHASES/(credit$TENURE)  
credit$Monthly\_CASH\_ADVANCE <- credit$CASH\_ADVANCE/(credit$TENURE)  
credit$LIMIT\_USAGE <- credit$BALANCE/credit$CREDIT\_LIMIT  
credit$MIN\_PAYMENTS\_RATIO <- credit$PAYMENTS/credit$MINIMUM\_PAYMENTS  
  
credit$PURCHASE\_TYPE <- dplyr::case\_when(  
 credit$ONEOFF\_PURCHASES == 0 & credit$INSTALLMENTS\_PURCHASES == 0 ~ 'none',  
 credit$ONEOFF\_PURCHASES > 0 & credit$INSTALLMENTS\_PURCHASES == 0 ~ 'oneoff',  
 credit$ONEOFF\_PURCHASES > 0 & credit$INSTALLMENTS\_PURCHASES > 0 ~ 'both\_oneoff\_installment',  
 credit$ONEOFF\_PURCHASES ==0 & credit$INSTALLMENTS\_PURCHASES > 0 ~ 'installment'  
)  
credit$PURCHASE\_TYPE <- as.factor(credit$PURCHASE\_TYPE)  
  
colnames(credit)

## [1] "CUST\_ID" "BALANCE"   
## [3] "BALANCE\_FREQUENCY" "PURCHASES"   
## [5] "ONEOFF\_PURCHASES" "INSTALLMENTS\_PURCHASES"   
## [7] "CASH\_ADVANCE" "PURCHASES\_FREQUENCY"   
## [9] "ONEOFF\_PURCHASES\_FREQUENCY" "PURCHASES\_INSTALLMENTS\_FREQUENCY"  
## [11] "CASH\_ADVANCE\_FREQUENCY" "CASH\_ADVANCE\_TRX"   
## [13] "PURCHASES\_TRX" "CREDIT\_LIMIT"   
## [15] "PAYMENTS" "MINIMUM\_PAYMENTS"   
## [17] "PRC\_FULL\_PAYMENT" "TENURE"   
## [19] "Monthly\_Avg\_PURCHASES" "Monthly\_CASH\_ADVANCE"   
## [21] "LIMIT\_USAGE" "MIN\_PAYMENTS\_RATIO"   
## [23] "PURCHASE\_TYPE"

str(credit)

## 'data.frame': 8950 obs. of 23 variables:  
## $ CUST\_ID : Factor w/ 8950 levels "C10001","C10002",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ BALANCE : num 40.9 3202.5 2495.1 1666.7 817.7 ...  
## $ BALANCE\_FREQUENCY : num 0.818 0.909 1 0.636 1 ...  
## $ PURCHASES : num 95.4 0 773.2 1499 16 ...  
## $ ONEOFF\_PURCHASES : num 0 0 773 1499 16 ...  
## $ INSTALLMENTS\_PURCHASES : num 95.4 0 0 0 0 ...  
## $ CASH\_ADVANCE : num 0 6443 0 206 0 ...  
## $ PURCHASES\_FREQUENCY : num 0.1667 0 1 0.0833 0.0833 ...  
## $ ONEOFF\_PURCHASES\_FREQUENCY : num 0 0 1 0.0833 0.0833 ...  
## $ PURCHASES\_INSTALLMENTS\_FREQUENCY: num 0.0833 0 0 0 0 ...  
## $ CASH\_ADVANCE\_FREQUENCY : num 0 0.25 0 0.0833 0 ...  
## $ CASH\_ADVANCE\_TRX : int 0 4 0 1 0 0 0 0 0 0 ...  
## $ PURCHASES\_TRX : int 2 0 12 1 1 8 64 12 5 3 ...  
## $ CREDIT\_LIMIT : num 1000 7000 7500 7500 1200 1800 13500 2300 7000 11000 ...  
## $ PAYMENTS : num 202 4103 622 0 678 ...  
## $ MINIMUM\_PAYMENTS : num 140 1072 627 312 245 ...  
## $ PRC\_FULL\_PAYMENT : num 0 0.222 0 0 0 ...  
## $ TENURE : int 12 12 12 12 12 12 12 12 12 12 ...  
## $ Monthly\_Avg\_PURCHASES : num 7.95 0 64.43 124.92 1.33 ...  
## $ Monthly\_CASH\_ADVANCE : num 0 536.9 0 17.1 0 ...  
## $ LIMIT\_USAGE : num 0.0409 0.4575 0.3327 0.2222 0.6814 ...  
## $ MIN\_PAYMENTS\_RATIO : num 1.447 3.826 0.992 0 2.771 ...  
## $ PURCHASE\_TYPE : Factor w/ 4 levels "both\_oneoff\_installment",..: 2 3 4 4 4 2 1 2 1 4 ...

credit\_copy = credit

## preparing for Machine Learning

Here we will be preparing the data for machine learning algorithm. we Need to convert the purchase\_type column into binary variable.

transformed\_variables <- c("BALANCE","BALANCE\_FREQUENCY",   
 "PURCHASES", "ONEOFF\_PURCHASES", "INSTALLMENTS\_PURCHASES",   
 "CASH\_ADVANCE","PURCHASES\_FREQUENCY","ONEOFF\_PURCHASES\_FREQUENCY",  
 "PURCHASES\_INSTALLMENTS\_FREQUENCY","CASH\_ADVANCE\_FREQUENCY",   
 "CASH\_ADVANCE\_TRX", "PURCHASES\_TRX", "CREDIT\_LIMIT",   
 "PAYMENTS", "MINIMUM\_PAYMENTS","PRC\_FULL\_PAYMENT","TENURE", "Monthly\_Avg\_PURCHASES","Monthly\_CASH\_ADVANCE",  
 "LIMIT\_USAGE","MIN\_PAYMENTS\_RATIO") # vector of variables to be log transformed  
  
credit\_transformed\_data <- credit %>% # preserve original dataset  
 .[,-c(1,23)] %>% #dropping CUST\_ID and PURCHASE\_TYPE  
 mutate\_at(vars(transformed\_variables), funs(log(1 + .))) %>% # add 1 to each value to avoid log0  
 mutate\_at(c(2:17), funs(c(scale(.)))) # scale all numeric variables to mean of 0 & sd = 1

## Warning: funs() is soft deprecated as of dplyr 0.8.0  
## Please use a list of either functions or lambdas:   
##   
## # Simple named list:   
## list(mean = mean, median = median)  
##   
## # Auto named with `tibble::lst()`:   
## tibble::lst(mean, median)  
##   
## # Using lambdas  
## list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))  
## This warning is displayed once per session.

#taking only the variable which we use for KPI  
credit\_transformed\_selected <- credit\_transformed\_data[,-c(1,3,6,17,14,15,16,13)]  
  
colnames(credit\_transformed\_selected)

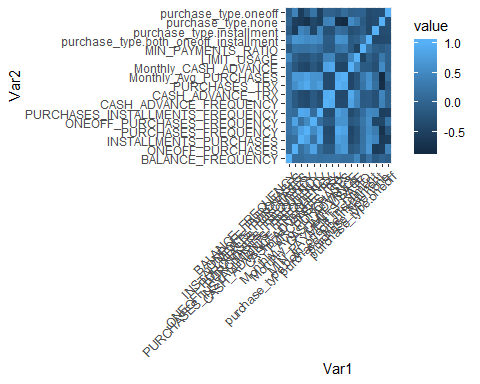
## [1] "BALANCE\_FREQUENCY" "ONEOFF\_PURCHASES"   
## [3] "INSTALLMENTS\_PURCHASES" "PURCHASES\_FREQUENCY"   
## [5] "ONEOFF\_PURCHASES\_FREQUENCY" "PURCHASES\_INSTALLMENTS\_FREQUENCY"  
## [7] "CASH\_ADVANCE\_FREQUENCY" "CASH\_ADVANCE\_TRX"   
## [9] "PURCHASES\_TRX" "Monthly\_Avg\_PURCHASES"   
## [11] "Monthly\_CASH\_ADVANCE" "LIMIT\_USAGE"   
## [13] "MIN\_PAYMENTS\_RATIO"

binary\_purchase\_type <- to.dummy(credit$PURCHASE\_TYPE, "purchase\_type")  
  
credit\_transformed\_selected <- cbind(credit\_transformed\_selected,binary\_purchase\_type)  
  
colnames(credit\_transformed\_selected)

## [1] "BALANCE\_FREQUENCY"   
## [2] "ONEOFF\_PURCHASES"   
## [3] "INSTALLMENTS\_PURCHASES"   
## [4] "PURCHASES\_FREQUENCY"   
## [5] "ONEOFF\_PURCHASES\_FREQUENCY"   
## [6] "PURCHASES\_INSTALLMENTS\_FREQUENCY"   
## [7] "CASH\_ADVANCE\_FREQUENCY"   
## [8] "CASH\_ADVANCE\_TRX"   
## [9] "PURCHASES\_TRX"   
## [10] "Monthly\_Avg\_PURCHASES"   
## [11] "Monthly\_CASH\_ADVANCE"   
## [12] "LIMIT\_USAGE"   
## [13] "MIN\_PAYMENTS\_RATIO"   
## [14] "purchase\_type.both\_oneoff\_installment"  
## [15] "purchase\_type.installment"   
## [16] "purchase\_type.none"   
## [17] "purchase\_type.oneoff"

cormat <- round(cor(credit\_transformed\_selected),3)  
credit\_corr <- melt(cormat)

ggplot(data = credit\_corr,aes (x = Var1,y = Var2,fill = value)) +  
theme(axis.text.x = element\_text(angle = 45, vjust = 1,size = 9, hjust = 1)) +  
geom\_tile()

 By seeing the heatmap we can conclude that number of variable is too much and we need to reduce the number of variable.

## Standardrizing and scaling of the data

We can reduce the number of variable by using the PCA (principal Component Algorithm). For PCA to work we need to scale the data and then we will be using the PCA. Before Applying the PCA we need to find the optimal value for number of component. we will be using the elbow method to find the optimal number of component

credit\_scale = scale(credit\_transformed\_selected)  
  
credit\_scale = as.data.frame(credit\_scale)  
dim(credit\_scale)

## [1] 8950 17

credit\_pca <- prcomp(credit\_scale)  
  
names(credit\_pca)

## [1] "sdev" "rotation" "center" "scale" "x"

credit\_pca$center

## BALANCE\_FREQUENCY ONEOFF\_PURCHASES   
## -7.713026e-18 9.636922e-19   
## INSTALLMENTS\_PURCHASES PURCHASES\_FREQUENCY   
## -3.785593e-17 4.267691e-17   
## ONEOFF\_PURCHASES\_FREQUENCY PURCHASES\_INSTALLMENTS\_FREQUENCY   
## 3.670365e-17 6.558999e-18   
## CASH\_ADVANCE\_FREQUENCY CASH\_ADVANCE\_TRX   
## 1.181938e-17 -1.832178e-17   
## PURCHASES\_TRX Monthly\_Avg\_PURCHASES   
## -1.997006e-17 -4.203419e-17   
## Monthly\_CASH\_ADVANCE LIMIT\_USAGE   
## -2.942343e-18 1.892884e-18   
## MIN\_PAYMENTS\_RATIO purchase\_type.both\_oneoff\_installment   
## -1.318545e-17 1.190854e-18   
## purchase\_type.installment purchase\_type.none   
## -9.923781e-17 3.101182e-17   
## purchase\_type.oneoff   
## -1.935137e-17

credit\_pca$scale

## [1] FALSE

credit\_pca$rotation

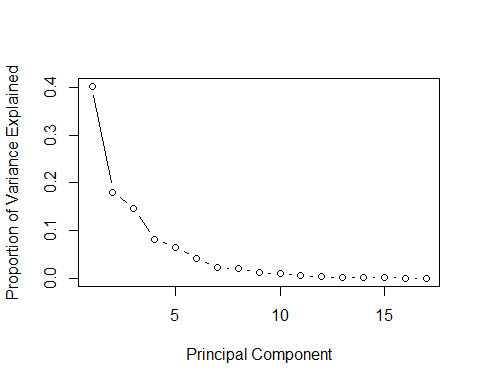
## PC1 PC2 PC3  
## BALANCE\_FREQUENCY 0.02970710 0.240072217 -0.26313956  
## ONEOFF\_PURCHASES 0.21410721 0.406078184 0.23916545  
## INSTALLMENTS\_PURCHASES 0.31205063 -0.098403659 -0.31562528  
## PURCHASES\_FREQUENCY 0.34582323 0.015813410 -0.16284334  
## ONEOFF\_PURCHASES\_FREQUENCY 0.21470195 0.362207806 0.16322234  
## PURCHASES\_INSTALLMENTS\_FREQUENCY 0.29545081 -0.112002477 -0.33002935  
## CASH\_ADVANCE\_FREQUENCY -0.21433583 0.286073833 -0.27858553  
## CASH\_ADVANCE\_TRX -0.22939348 0.291555606 -0.28508932  
## PURCHASES\_TRX 0.35550341 0.106624771 -0.10274279  
## Monthly\_Avg\_PURCHASES 0.34599214 0.141635391 0.02398613  
## Monthly\_CASH\_ADVANCE -0.24386117 0.264317622 -0.25742721  
## LIMIT\_USAGE -0.14630190 0.235709775 -0.25127813  
## MIN\_PAYMENTS\_RATIO 0.11963162 0.021328016 0.13635689  
## purchase\_type.both\_oneoff\_installment 0.24139215 0.273675977 -0.13193476  
## purchase\_type.installment 0.08220908 -0.443375470 -0.20868331  
## purchase\_type.none -0.31028278 -0.005213539 -0.09691138  
## purchase\_type.oneoff -0.04213752 0.167737275 0.47274947  
## PC4 PC5 PC6  
## BALANCE\_FREQUENCY 0.353548687 0.228680754 0.693815814  
## ONEOFF\_PURCHASES -0.001519524 0.023197008 -0.129094254  
## INSTALLMENTS\_PURCHASES -0.087982599 0.002181369 -0.115223015  
## PURCHASES\_FREQUENCY 0.074616666 -0.115947716 0.081878633  
## ONEOFF\_PURCHASES\_FREQUENCY -0.036303132 0.051279437 0.097299004  
## PURCHASES\_INSTALLMENTS\_FREQUENCY -0.023502025 -0.025871312 -0.006730740  
## CASH\_ADVANCE\_FREQUENCY -0.096352556 -0.360132311 -0.066588914  
## CASH\_ADVANCE\_TRX -0.103484065 -0.332752510 -0.082307035  
## PURCHASES\_TRX 0.054296296 -0.104970995 0.009401792  
## Monthly\_Avg\_PURCHASES 0.079372898 -0.194146654 -0.015878030  
## Monthly\_CASH\_ADVANCE -0.135291816 -0.268026007 -0.058258298  
## LIMIT\_USAGE 0.431681652 0.181884667 -0.024297511  
## MIN\_PAYMENTS\_RATIO -0.591561004 -0.215445825 0.572466894  
## purchase\_type.both\_oneoff\_installment -0.254709583 0.340849320 -0.294708307  
## purchase\_type.installment 0.190829116 -0.353821446 0.086086800  
## purchase\_type.none -0.245103766 0.342221935 0.176809261  
## purchase\_type.oneoff 0.338548865 -0.362584507 0.060697768  
## PC7 PC8 PC9  
## BALANCE\_FREQUENCY 0.09173787 -0.431064339 -0.112621289  
## ONEOFF\_PURCHASES -0.11207671 -0.108809159 -0.007506005  
## INSTALLMENTS\_PURCHASES -0.15801822 -0.074729483 -0.179761704  
## PURCHASES\_FREQUENCY 0.26571934 0.163098977 0.317796727  
## ONEOFF\_PURCHASES\_FREQUENCY 0.47928010 0.481208548 -0.342413071  
## PURCHASES\_INSTALLMENTS\_FREQUENCY 0.02066155 -0.018221837 0.572047213  
## CASH\_ADVANCE\_FREQUENCY 0.10158695 -0.104466837 -0.151573653  
## CASH\_ADVANCE\_TRX 0.06177460 -0.060774470 -0.023484354  
## PURCHASES\_TRX 0.10871589 0.181945908 0.111716677  
## Monthly\_Avg\_PURCHASES -0.15503284 0.008362292 -0.065560225  
## Monthly\_CASH\_ADVANCE 0.02757110 0.028309616 0.169043764  
## LIMIT\_USAGE -0.57028318 0.542110667 0.012883535  
## MIN\_PAYMENTS\_RATIO -0.44103227 0.155002599 -0.004476100  
## purchase\_type.both\_oneoff\_installment -0.15864522 -0.261627280 -0.125737623  
## purchase\_type.installment 0.02287131 0.129173667 -0.403572317  
## purchase\_type.none 0.22616541 0.269451649 0.254485821  
## purchase\_type.oneoff -0.07736328 -0.118462881 0.311377731  
## PC10 PC11 PC12  
## BALANCE\_FREQUENCY 0.072681049 0.003270399 -0.019471900  
## ONEOFF\_PURCHASES 0.041911211 -0.186745099 0.104968406  
## INSTALLMENTS\_PURCHASES 0.070318830 -0.111173241 0.013310855  
## PURCHASES\_FREQUENCY -0.087597670 0.224448083 0.264717157  
## ONEOFF\_PURCHASES\_FREQUENCY 0.024551408 0.193346653 0.141212772  
## PURCHASES\_INSTALLMENTS\_FREQUENCY -0.146495038 0.080048402 0.172078415  
## CASH\_ADVANCE\_FREQUENCY -0.574926430 -0.119684198 0.235813990  
## CASH\_ADVANCE\_TRX -0.081652414 0.012850292 -0.364328421  
## PURCHASES\_TRX 0.004815034 -0.132382711 -0.772376064  
## Monthly\_Avg\_PURCHASES 0.154011522 -0.676758212 0.239721419  
## Monthly\_CASH\_ADVANCE 0.753475887 0.120702202 0.144827843  
## LIMIT\_USAGE -0.098819934 0.094187403 0.037285750  
## MIN\_PAYMENTS\_RATIO -0.069078416 0.135305619 -0.011157306  
## purchase\_type.both\_oneoff\_installment -0.018073429 0.294116264 -0.016241612  
## purchase\_type.installment 0.102429411 0.042105274 0.050139408  
## purchase\_type.none -0.030061463 -0.475576361 -0.001061546  
## purchase\_type.oneoff -0.057825292 0.111239217 -0.033982732  
## PC13 PC14 PC15  
## BALANCE\_FREQUENCY -0.026773693 0.010774081 -0.011799331  
## ONEOFF\_PURCHASES -0.060703616 -0.062922863 -0.565615937  
## INSTALLMENTS\_PURCHASES -0.273825563 0.039339520 0.581150706  
## PURCHASES\_FREQUENCY 0.617722661 -0.248151197 0.106667765  
## ONEOFF\_PURCHASES\_FREQUENCY -0.351837251 -0.009092037 0.100497914  
## PURCHASES\_INSTALLMENTS\_FREQUENCY -0.555235612 0.024011574 -0.247817722  
## CASH\_ADVANCE\_FREQUENCY 0.090805500 0.445192487 -0.000859998  
## CASH\_ADVANCE\_TRX -0.129237775 -0.698803099 0.007735549  
## PURCHASES\_TRX 0.142454298 0.373509839 -0.076034254  
## Monthly\_Avg\_PURCHASES 0.102521416 -0.158659754 0.110384146  
## Monthly\_CASH\_ADVANCE 0.040175744 0.275211333 -0.008129671  
## LIMIT\_USAGE 0.022563468 -0.007199258 -0.002074872  
## MIN\_PAYMENTS\_RATIO 0.022377693 -0.003482872 -0.016949179  
## purchase\_type.both\_oneoff\_installment 0.152577987 0.004855332 0.008592734  
## purchase\_type.installment -0.022251309 -0.030846725 -0.360433603  
## purchase\_type.none 0.007432856 -0.029148013 0.043512405  
## purchase\_type.oneoff -0.157334673 0.057482336 0.330224339  
## PC16 PC17  
## BALANCE\_FREQUENCY -0.002840445 -2.095296e-16  
## ONEOFF\_PURCHASES 0.561410322 -1.455831e-14  
## INSTALLMENTS\_PURCHASES 0.526958878 1.546913e-14  
## PURCHASES\_FREQUENCY 0.216924086 -5.779553e-15  
## ONEOFF\_PURCHASES\_FREQUENCY -0.110616285 5.527948e-15  
## PURCHASES\_INSTALLMENTS\_FREQUENCY -0.184999143 7.653131e-15  
## CASH\_ADVANCE\_FREQUENCY 0.008681222 -1.272357e-15  
## CASH\_ADVANCE\_TRX -0.013183035 1.595619e-15  
## PURCHASES\_TRX -0.023146276 -6.874869e-15  
## Monthly\_Avg\_PURCHASES -0.447156988 2.597173e-15  
## Monthly\_CASH\_ADVANCE 0.005866518 7.366525e-16  
## LIMIT\_USAGE 0.003086279 -2.670582e-16  
## MIN\_PAYMENTS\_RATIO -0.001062543 1.690909e-16  
## purchase\_type.both\_oneoff\_installment -0.272490715 -5.360651e-01  
## purchase\_type.installment 0.065374536 -5.035909e-01  
## purchase\_type.none 0.175643204 -4.864236e-01  
## purchase\_type.oneoff 0.058763213 -4.716169e-01

dim(credit\_pca$x)

## [1] 8950 17

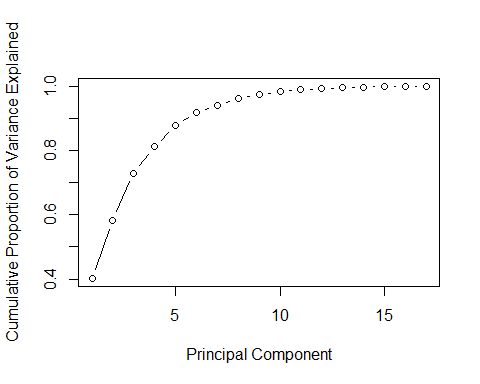
std\_dev <- credit\_pca$sdev  
pca\_var <- std\_dev^2  
  
pca\_varex <- pca\_var/sum(pca\_var)

plot(pca\_varex, xlab = "Principal Component",  
 ylab = "Proportion of Variance Explained",  
 type = "b")



This plot tells us that selecting 5 components we can preserve something around 87% of the total variance of the data. It makes sense, we’ll not use 100% of our variance, because it denotes all components, and we want only the principal ones. With this information in our hands, we can implement the PCA for 5 best components.

plot(cumsum(pca\_varex), xlab = "Principal Component",  
 ylab = "Cumulative Proportion of Variance Explained",  
 type = "b")



This plot shows that 5 components results in variance close to ~ 87%. Therefore, in this case, we’ll select number of components as 5 [PC1 to PC5] and proceed to the modeling stage. This completes the steps to implement PCA on train data. For modeling, we’ll use these 5 components as predictor variables and follow the normal procedures.

credit\_pca\_data <- data.frame(credit\_pca$x)  
credit\_pca\_data <- credit\_pca\_data[,1:5]  
dim(credit\_pca\_data)

## [1] 8950 5

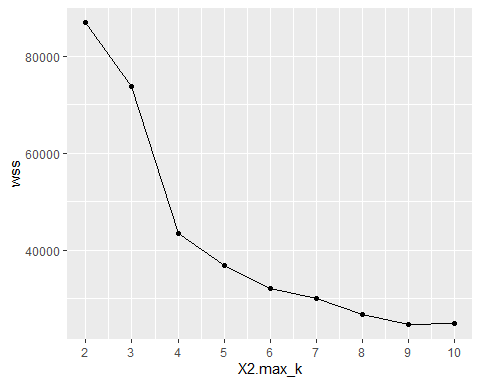
## Clustering Algorithm - K Means Clustering

Now Our data is ready after we have done the feature selection and scaling of the data.

Before we start the KMeans we need find the optimal cluster value.for the we will be elbow method to find the optimal number of cluster.

kmean\_withinss <- function(k) {  
cluster <- kmeans(credit\_pca\_data, k)  
return (cluster$tot.withinss)  
}  
  
max\_k <- 10  
wss <- sapply(2:max\_k, kmean\_withinss)  
  
# Create a data frame to plot the graph  
elbow <-data.frame(2:max\_k, wss)

ggplot(elbow, aes(x = X2.max\_k, y = wss)) +  
geom\_point() +  
geom\_line() +  
scale\_x\_continuous(breaks = seq(1, 20, by = 1))



From the graph, you can see the optimal k is 4, where the curve is starting to have a diminishing return. Once you have our optimal k, you run the algorithm with k equals to 4 and evaluate the clusters.

## Applying the K value 4

pc\_cluster\_4 <-kmeans(credit\_pca\_data, 4)  
cluster\_df <- (pc\_cluster\_4$cluster)  
pc\_cluster\_4$centers

## PC1 PC2 PC3 PC4 PC5  
## 1 -0.5469527 0.99148953 2.3044494 0.9049383 -0.7747818  
## 2 1.0192177 -2.37441234 -0.8745912 0.4503346 -0.6573398  
## 3 -3.8635229 -0.02076308 -0.4779493 -0.6005808 0.6456682  
## 4 2.4760461 1.26016466 -0.4971120 -0.5235628 0.5681826

pc\_cluster\_4$size

## [1] 1874 2228 2090 2758

center <-pc\_cluster\_4$centers  
center

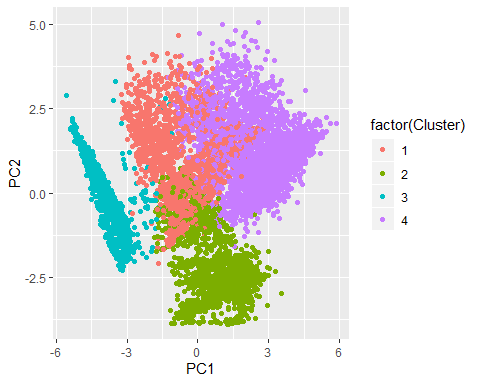
## PC1 PC2 PC3 PC4 PC5  
## 1 -0.5469527 0.99148953 2.3044494 0.9049383 -0.7747818  
## 2 1.0192177 -2.37441234 -0.8745912 0.4503346 -0.6573398  
## 3 -3.8635229 -0.02076308 -0.4779493 -0.6005808 0.6456682  
## 4 2.4760461 1.26016466 -0.4971120 -0.5235628 0.5681826

# create dataset with the cluster number  
cluster <- c(1: 4)  
center\_df <- data.frame(cluster, center)  
# Reshape the data  
center\_reshape <- gather(center\_df, features, values, PC1:PC5)  
head(center\_reshape)

## cluster features values  
## 1 1 PC1 -0.5469527  
## 2 2 PC1 1.0192177  
## 3 3 PC1 -3.8635229  
## 4 4 PC1 2.4760461  
## 5 1 PC2 0.9914895  
## 6 2 PC2 -2.3744123

cluster\_scatter <- cbind(credit\_pca\_data,pc\_cluster\_4$cluster)  
names(cluster\_scatter)[6] <- "Cluster"

ggplot(data=cluster\_scatter, aes(x=PC1, y=PC2, color=factor(Cluster))) +   
geom\_point()

 From the scatter plot we can see that we have 4 cluster here and we each data point classified into different cluster.

To get a clear idea of how observation are classified into different cluster we will using the original data without the varibale which are used to create new KPI’s and then adding the cluster classification which we get from the kMeans algorithm and then we will be grouping the data by cluster and we will be summarizing the other variable by mean and checking how well all the observation are classified.

col\_kpi=c('CASH\_ADVANCE\_TRX', 'PURCHASES\_TRX', 'Monthly\_Avg\_PURCHASES',  
 'Monthly\_CASH\_ADVANCE', 'LIMIT\_USAGE', 'MIN\_PAYMENTS\_RATIO')  
cluster\_4 <- credit[,col\_kpi]  
cluster\_4 <- cbind(cluster\_4,binary\_purchase\_type)  
cluster\_4 <- cbind(cluster\_4,cluster\_df)  
cluster\_4[1:10,]

## CASH\_ADVANCE\_TRX PURCHASES\_TRX Monthly\_Avg\_PURCHASES Monthly\_CASH\_ADVANCE  
## 1 0 2 7.950000 0.0000  
## 2 4 0 0.000000 536.9121  
## 3 0 12 64.430833 0.0000  
## 4 1 1 124.916667 17.1490  
## 5 0 1 1.333333 0.0000  
## 6 0 8 111.106667 0.0000  
## 7 0 64 590.917500 0.0000  
## 8 0 12 36.350000 0.0000  
## 9 0 5 71.790833 0.0000  
## 10 0 3 106.800000 0.0000  
## LIMIT\_USAGE MIN\_PAYMENTS\_RATIO purchase\_type.both\_oneoff\_installment  
## 1 0.04090075 1.4465084 0  
## 2 0.45749535 3.8262415 0  
## 3 0.33268651 0.9916815 0  
## 4 0.22222274 0.0000000 0  
## 5 0.68142861 2.7710745 0  
## 6 1.00546042 0.5816014 0  
## 7 0.04646376 32.0818198 1  
## 8 0.79289250 1.2763566 0  
## 9 0.14498950 2.2062798 1  
## 10 0.01383872 11.6126054 0  
## purchase\_type.installment purchase\_type.none purchase\_type.oneoff cluster\_df  
## 1 1 0 0 2  
## 2 0 1 0 3  
## 3 0 0 1 1  
## 4 0 0 1 1  
## 5 0 0 1 1  
## 6 1 0 0 2  
## 7 0 0 0 4  
## 8 1 0 0 2  
## 9 0 0 0 4  
## 10 0 0 1 1

dim(cluster\_4)

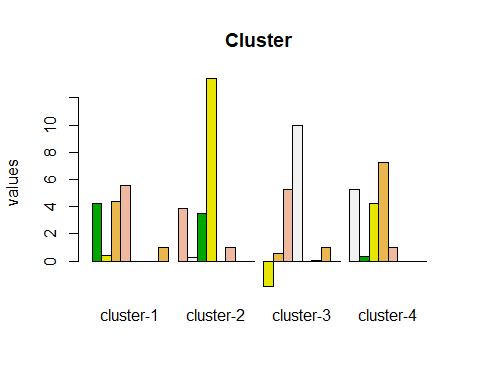
## [1] 8950 11

cluster\_group\_by <- group\_by(cluster\_4,cluster\_df)  
cluster\_summ <- summarise(cluster\_group\_by,CASH\_ADVANCE\_TRX = mean(CASH\_ADVANCE\_TRX),  
 PURCHASES\_TRX= mean(PURCHASES\_TRX),  
 Monthly\_Avg\_PURCHASES = mean(Monthly\_Avg\_PURCHASES),  
 LIMIT\_USAGE = mean(LIMIT\_USAGE),  
 Monthly\_CASH\_ADVANCE = mean(Monthly\_CASH\_ADVANCE),  
 MIN\_PAYMENTS\_RATIO = mean(MIN\_PAYMENTS\_RATIO),  
 purchase\_type.both\_oneoff\_installment = mean(purchase\_type.both\_oneoff\_installment),  
 purchase\_type.installment = mean(purchase\_type.installment),  
 purchase\_type.none = mean(purchase\_type.none),  
 purchase\_type.oneoff = mean(purchase\_type.oneoff))  
cluster\_summ <- t(cluster\_summ)  
colnames(cluster\_summ) <- c("cluster-1","cluster-2","cluster-3","cluster-4")  
cluster\_summ

## cluster-1 cluster-2 cluster-3  
## cluster\_df 1.000000000 2.000000000 3.000000e+00  
## CASH\_ADVANCE\_TRX 2.864994664 1.019299820 6.552632e+00  
## PURCHASES\_TRX 7.118996798 12.053859964 4.593301e-02  
## Monthly\_Avg\_PURCHASES 69.758275903 47.573597589 1.593372e-01  
## LIMIT\_USAGE 0.378726746 0.264274761 5.762172e-01  
## Monthly\_CASH\_ADVANCE 77.843484865 33.489845985 1.862980e+02  
## MIN\_PAYMENTS\_RATIO 5.561421013 13.402659725 9.927979e+00  
## purchase\_type.both\_oneoff\_installment 0.003735326 0.001795332 2.392344e-03  
## purchase\_type.installment 0.000000000 0.998204668 1.722488e-02  
## purchase\_type.none 0.000000000 0.000000000 9.770335e-01  
## purchase\_type.oneoff 0.996264674 0.000000000 3.349282e-03  
## cluster-4  
## cluster\_df 4.0000000  
## CASH\_ADVANCE\_TRX 2.8071066  
## PURCHASES\_TRX 33.1254532  
## Monthly\_Avg\_PURCHASES 193.6960834  
## LIMIT\_USAGE 0.3544875  
## Monthly\_CASH\_ADVANCE 67.6200059  
## MIN\_PAYMENTS\_RATIO 7.2686054  
## purchase\_type.both\_oneoff\_installment 1.0000000  
## purchase\_type.installment 0.0000000  
## purchase\_type.none 0.0000000  
## purchase\_type.oneoff 0.0000000

cluster\_summ <- cluster\_summ[-c(1,2,3),]  
  
cluster\_summ <- t(cluster\_summ)  
cluster\_summ <- as.data.frame(cluster\_summ)  
cluster\_summ$Monthly\_Avg\_PURCHASES <- log(cluster\_summ$Monthly\_Avg\_PURCHASES)  
cluster\_summ$Monthly\_CASH\_ADVANCE <- log(cluster\_summ$Monthly\_CASH\_ADVANCE)  
cluster\_summ <- t(cluster\_summ)  
cluster\_summ <- as.data.frame(cluster\_summ)

barplot(as.matrix(cluster\_summ), main="Cluster", ylab="values", beside=TRUE,   
 col=terrain.colors(5))



cluster\_count <- cluster\_scatter %>%  
 group\_by(Cluster) %>%  
 summarise(count\_value = n())  
cluster\_count

## # A tibble: 4 x 2  
## Cluster count\_value  
## <int> <int>  
## 1 1 1874  
## 2 2 2228  
## 3 3 2090  
## 4 4 2758

cluster\_percentage <- cluster\_count %>%  
 mutate(percentage = (count\_value/sum(count\_value)\*100))  
cluster\_percentage

## # A tibble: 4 x 3  
## Cluster count\_value percentage  
## <int> <int> <dbl>  
## 1 1 1874 20.9  
## 2 2 2228 24.9  
## 3 3 2090 23.4  
## 4 4 2758 30.8

**Some insight on the KMeans cluster-4**

from the above graph we can see that each cluster is clearly showing a distinguishing behaviour within the customers.

* Cluster 4 is the group of customers who have highest Monthly\_avg purchases and doing both installment as well as one\_off purchases, have comparatively good credit score. This group is about 31% of the total customer base.
* cluster 3 is taking maximum advance\_cash and is paying comparatively less minimum payment and poor credit\_score & doing no purchase transaction. This group is about 23% of the total customer base.
* Cluster 1 customers are doing maximum One\_Off transactions and least payment ratio and credit\_score on lower side This group is about 21% of the total customer base.
* Cluster 2 customers have maximum credit score and are paying dues and are doing maximum installment purchases. This group is about 25% of the total customer base

## Finding behaviour with 5 cluster

Even though we found that optimal value of K is 4 we will just see how the data will behave with number of cluster as 5.

For this cluster also we will be doing all the methods that are done for cluster 4 and finding how does the classification of the data point are done.

pc\_cluster\_5 <-kmeans(credit\_pca\_data, 5)  
cluster\_df\_5 <- (pc\_cluster\_5$cluster)  
pc\_cluster\_5$centers

## PC1 PC2 PC3 PC4 PC5  
## 1 1.0812539 -2.47673214 -0.77418789 0.4475856 -0.5827494  
## 2 1.0057281 2.21215340 -1.98116120 -0.2677740 -0.7054710  
## 3 -3.8694145 -0.02903736 -0.47070767 -0.6003124 0.6534036  
## 4 -0.5401119 0.97258462 2.32077056 0.9100699 -0.7579760  
## 5 2.9577683 0.78548084 0.03998517 -0.5831136 0.9670140

pc\_cluster\_5$size

## [1] 2131 891 2084 1860 1984

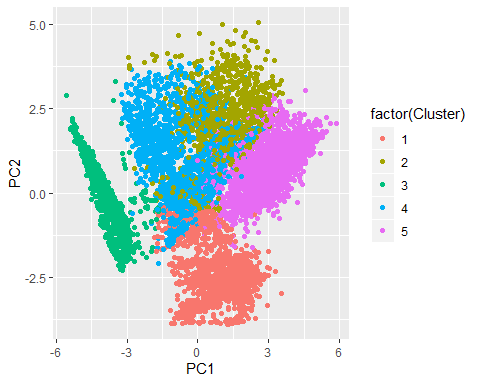
center\_5 <-pc\_cluster\_5$centers  
center\_5

## PC1 PC2 PC3 PC4 PC5  
## 1 1.0812539 -2.47673214 -0.77418789 0.4475856 -0.5827494  
## 2 1.0057281 2.21215340 -1.98116120 -0.2677740 -0.7054710  
## 3 -3.8694145 -0.02903736 -0.47070767 -0.6003124 0.6534036  
## 4 -0.5401119 0.97258462 2.32077056 0.9100699 -0.7579760  
## 5 2.9577683 0.78548084 0.03998517 -0.5831136 0.9670140

cluster\_5 <- c(1: 5)  
center\_df\_5 <- data.frame(cluster\_5, center\_5)  
# Reshape the data  
center\_reshape\_5 <- gather(center\_df\_5, features, values, PC1:PC5)  
head(center\_reshape\_5)

## cluster\_5 features values  
## 1 1 PC1 1.0812539  
## 2 2 PC1 1.0057281  
## 3 3 PC1 -3.8694145  
## 4 4 PC1 -0.5401119  
## 5 5 PC1 2.9577683  
## 6 1 PC2 -2.4767321

cluster\_scatter\_5 <- cbind(credit\_pca\_data,pc\_cluster\_5$cluster)  
names(cluster\_scatter\_5)[6] <- "Cluster"  
ggplot(data=cluster\_scatter\_5, aes(x=PC1, y=PC2, color=factor(Cluster))) +   
 geom\_point()



col\_kpi=c('CASH\_ADVANCE\_TRX', 'PURCHASES\_TRX', 'Monthly\_Avg\_PURCHASES',  
 'Monthly\_CASH\_ADVANCE', 'LIMIT\_USAGE', 'MIN\_PAYMENTS\_RATIO')  
cluster\_5 <- credit[,col\_kpi]  
cluster\_5 <- cbind(cluster\_5,binary\_purchase\_type)  
cluster\_5 <- cbind(cluster\_5,cluster\_df\_5)  
cluster\_5[1:10,]

## CASH\_ADVANCE\_TRX PURCHASES\_TRX Monthly\_Avg\_PURCHASES Monthly\_CASH\_ADVANCE  
## 1 0 2 7.950000 0.0000  
## 2 4 0 0.000000 536.9121  
## 3 0 12 64.430833 0.0000  
## 4 1 1 124.916667 17.1490  
## 5 0 1 1.333333 0.0000  
## 6 0 8 111.106667 0.0000  
## 7 0 64 590.917500 0.0000  
## 8 0 12 36.350000 0.0000  
## 9 0 5 71.790833 0.0000  
## 10 0 3 106.800000 0.0000  
## LIMIT\_USAGE MIN\_PAYMENTS\_RATIO purchase\_type.both\_oneoff\_installment  
## 1 0.04090075 1.4465084 0  
## 2 0.45749535 3.8262415 0  
## 3 0.33268651 0.9916815 0  
## 4 0.22222274 0.0000000 0  
## 5 0.68142861 2.7710745 0  
## 6 1.00546042 0.5816014 0  
## 7 0.04646376 32.0818198 1  
## 8 0.79289250 1.2763566 0  
## 9 0.14498950 2.2062798 1  
## 10 0.01383872 11.6126054 0  
## purchase\_type.installment purchase\_type.none purchase\_type.oneoff  
## 1 1 0 0  
## 2 0 1 0  
## 3 0 0 1  
## 4 0 0 1  
## 5 0 0 1  
## 6 1 0 0  
## 7 0 0 0  
## 8 1 0 0  
## 9 0 0 0  
## 10 0 0 1  
## cluster\_df\_5  
## 1 1  
## 2 3  
## 3 4  
## 4 4  
## 5 4  
## 6 1  
## 7 5  
## 8 1  
## 9 5  
## 10 4

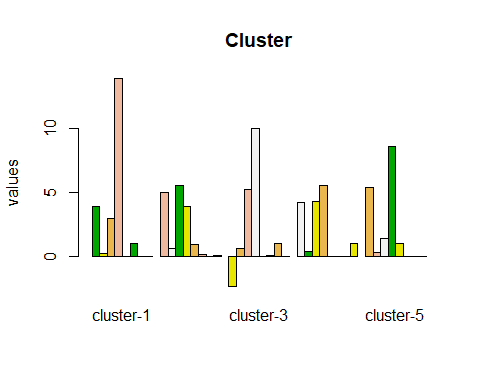
dim(cluster\_5)

## [1] 8950 11

cluster\_group\_by\_5 <- group\_by(cluster\_5,cluster\_df\_5)  
cluster\_summ <- summarise(cluster\_group\_by\_5,CASH\_ADVANCE\_TRX = mean(CASH\_ADVANCE\_TRX),PURCHASES\_TRX= mean(PURCHASES\_TRX),  
 Monthly\_Avg\_PURCHASES = mean(Monthly\_Avg\_PURCHASES),LIMIT\_USAGE = mean(LIMIT\_USAGE),Monthly\_CASH\_ADVANCE = mean(Monthly\_CASH\_ADVANCE),  
 MIN\_PAYMENTS\_RATIO = mean(MIN\_PAYMENTS\_RATIO),purchase\_type.both\_oneoff\_installment = mean(purchase\_type.both\_oneoff\_installment),  
 purchase\_type.installment = mean(purchase\_type.installment),purchase\_type.none = mean(purchase\_type.none),  
 purchase\_type.oneoff = mean(purchase\_type.oneoff))  
cluster\_summ <- t(cluster\_summ)  
colnames(cluster\_summ) <- c("cluster-1","cluster-2","cluster-3","cluster-4","cluster-5")  
cluster\_summ

## cluster-1 cluster-2 cluster-3  
## cluster\_df\_5 1.0000000 2.00000000 3.000000e+00  
## CASH\_ADVANCE\_TRX 0.4842797 10.51178451 6.454894e+00  
## PURCHASES\_TRX 11.8967621 27.56677890 3.550864e-02  
## Monthly\_Avg\_PURCHASES 47.2438247 141.90647352 9.657167e-02  
## LIMIT\_USAGE 0.2467329 0.59531009 5.762603e-01  
## Monthly\_CASH\_ADVANCE 19.1550482 252.43152371 1.851095e+02  
## MIN\_PAYMENTS\_RATIO 13.8619374 3.91707728 9.950170e+00  
## purchase\_type.both\_oneoff\_installment 0.0000000 0.87991021 0.000000e+00  
## purchase\_type.installment 1.0000000 0.10549944 1.679463e-02  
## purchase\_type.none 0.0000000 0.00000000 9.798464e-01  
## purchase\_type.oneoff 0.0000000 0.01459035 3.358925e-03  
## cluster-4 cluster-5  
## cluster\_df\_5 4.000000000 5.0000000  
## CASH\_ADVANCE\_TRX 2.648387097 0.1517137  
## PURCHASES\_TRX 7.067741935 34.5357863  
## Monthly\_Avg\_PURCHASES 68.685725177 209.7761264  
## LIMIT\_USAGE 0.377563147 0.2626540  
## Monthly\_CASH\_ADVANCE 73.635702543 3.9750396  
## MIN\_PAYMENTS\_RATIO 5.540102055 8.5730188  
## purchase\_type.both\_oneoff\_installment 0.003225806 1.0000000  
## purchase\_type.installment 0.000000000 0.0000000  
## purchase\_type.none 0.000000000 0.0000000  
## purchase\_type.oneoff 0.996774194 0.0000000

cluster\_summ <- cluster\_summ[-c(1,2,3),]  
  
cluster\_summ <- t(cluster\_summ)  
cluster\_summ <- as.data.frame(cluster\_summ)  
cluster\_summ$Monthly\_Avg\_PURCHASES <- log(cluster\_summ$Monthly\_Avg\_PURCHASES)  
cluster\_summ$Monthly\_CASH\_ADVANCE <- log(cluster\_summ$Monthly\_CASH\_ADVANCE)  
cluster\_summ <- t(cluster\_summ)  
cluster\_summ <- as.data.frame(cluster\_summ)  
  
barplot(as.matrix(cluster\_summ), main="Cluster", ylab="values", beside=TRUE,   
 col=terrain.colors(5))



cluster\_count <- cluster\_scatter\_5 %>%  
 group\_by(Cluster) %>%  
 summarise(count\_value = n())  
cluster\_count

## # A tibble: 5 x 2  
## Cluster count\_value  
## <int> <int>  
## 1 1 2131  
## 2 2 891  
## 3 3 2084  
## 4 4 1860  
## 5 5 1984

cluster\_percentage <- cluster\_count %>%  
 mutate(percentage = (count\_value/sum(count\_value)\*100))  
cluster\_percentage

## # A tibble: 5 x 3  
## Cluster count\_value percentage  
## <int> <int> <dbl>  
## 1 1 2131 23.8   
## 2 2 891 9.96  
## 3 3 2084 23.3   
## 4 4 1860 20.8   
## 5 5 1984 22.2

**Insights on cluster-5**

* we have a group of customers (cluster 5) having highest avergae purchases but there is Cluster 2 also having highest cash advance & secong highest purchase behaviour but their type of purchases are same.
* Cluster 4 and Cluster 2 are behaving similar in terms of Credit\_limit and have cash transactions is on higher side.

**Since we cannot draw a proper conclusion with K value 5. We take K value 4 as optimal K and we will moving ahead considering K as 5 and drawing our marketing strategies.**

## Marketing strategies that can be drawn from above analysis

We Came to know that we have 4 kind of customers

**1. Group 1**

* This group is has minimum paying ratio and using card for just oneoff transactions (may be for utility bills only). This group seems to be risky group.

**2. Group 2**

* This group is performing best among all as cutomers are maintaining good credit score and paying dues on time. – Giving rewards point will make them perform more purchases.

**3. Group 3**

* They have poor credit score and taking only cash on advance. We can target them by providing less interest rate on purchase transaction.

**4. Group 4**

* They are potential target customers who are paying dues and doing purchases and maintaining comparatively good credit score ) – we can increase credit limit or can lower down interest rate – Can be given premium card /loyality cards to increase transactions.

**Group 4 is the potential Target customer**