3–Relations\_continuous\_variables.R

r366736

2023-02-23

# R code snippets from slides for Chapman & Feit 2015  
# Slide file: Chapter4/Chapter4-ChapmanFeit  
  
# All code is (c) 2015, Springer. http://r-marketing.r-forge.r-project.org/  
  
# ==========  
  
  
# Load CRM data  
# ==========  
cust.df <- read.csv("http://goo.gl/PmPkaG")  
str(cust.df)

## 'data.frame': 1000 obs. of 12 variables:  
## $ cust.id : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ age : num 22.9 28 35.9 30.5 38.7 ...  
## $ credit.score : num 631 749 733 830 734 ...  
## $ email : chr "yes" "yes" "yes" "yes" ...  
## $ distance.to.store: num 2.58 48.18 1.29 5.25 25.04 ...  
## $ online.visits : int 20 121 39 1 35 1 1 48 0 14 ...  
## $ online.trans : int 3 39 14 0 11 1 1 13 0 6 ...  
## $ online.spend : num 58.4 756.9 250.3 0 204.7 ...  
## $ store.trans : int 4 0 0 2 0 0 2 4 0 3 ...  
## $ store.spend : num 140.3 0 0 95.9 0 ...  
## $ sat.service : int 3 3 NA 4 1 NA 3 2 4 3 ...  
## $ sat.selection : int 3 3 NA 2 1 NA 3 3 2 2 ...

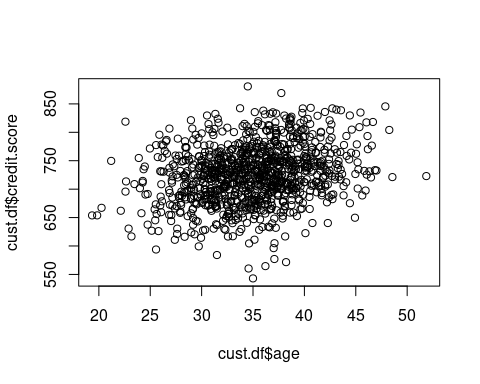
# Converting data to factors  
# ==========  
str(cust.df$cust.id)

## int [1:1000] 1 2 3 4 5 6 7 8 9 10 ...

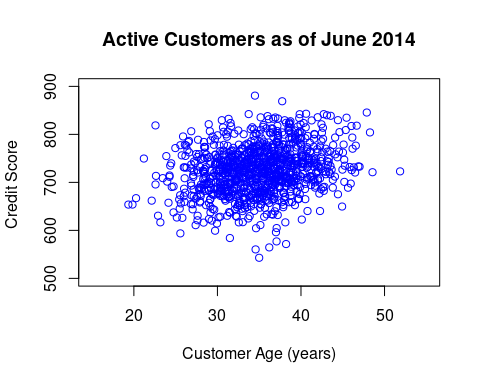
cust.df$cust.id <- factor(cust.df$cust.id)  
  
str(cust.df$cust.id)

## Factor w/ 1000 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...

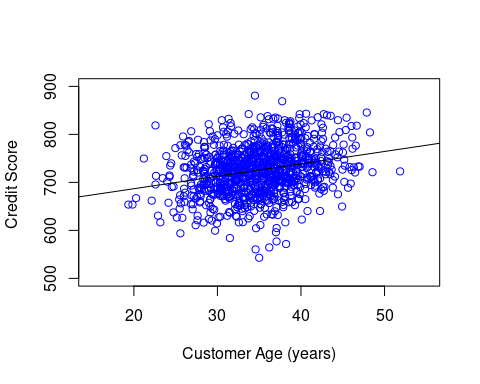
# Basic scatterplot  
# ==========  
plot(x=cust.df$age, y=cust.df$credit.score)



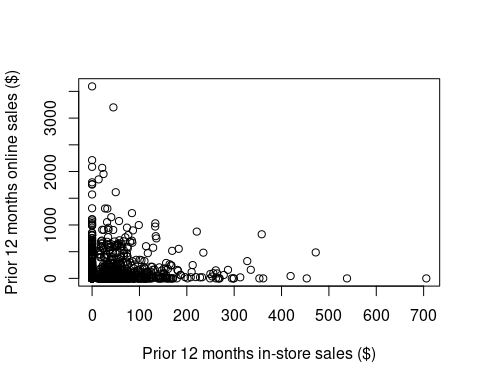
# A better plot  
# ==========  
plot(cust.df$age, cust.df$credit.score,   
 col="blue",  
 xlim=c(15, 55), ylim=c(500, 900),   
 main="Active Customers as of June 2014",  
 xlab="Customer Age (years)", ylab="Credit Score ")



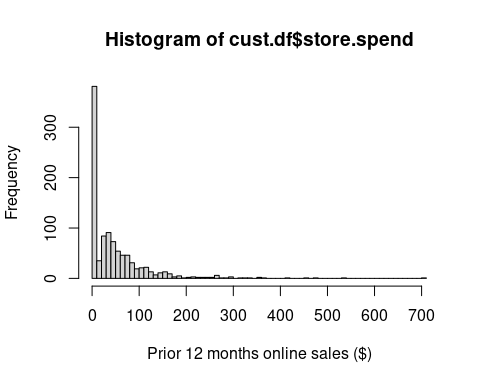
# Add a regression line  
# ==========  
plot(cust.df$age, cust.df$credit.score,   
 col="blue", xlim=c(15, 55), ylim=c(500, 900),   
 xlab="Customer Age (years)", ylab="Credit Score ")  
  
abline(lm(cust.df$credit.score ~ cust.df$age))



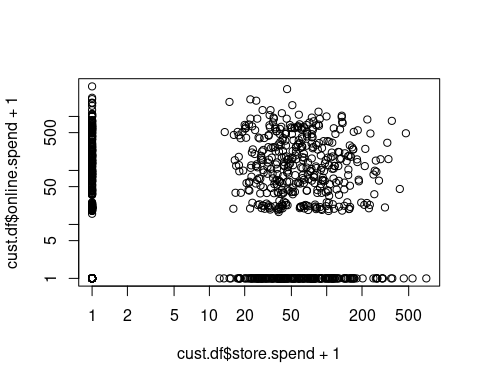
# Scatterplot with skew  
# ==========  
plot(cust.df$store.spend, cust.df$online.spend,   
 xlab="Prior 12 months in-store sales ($)",   
 ylab="Prior 12 months online sales ($)")



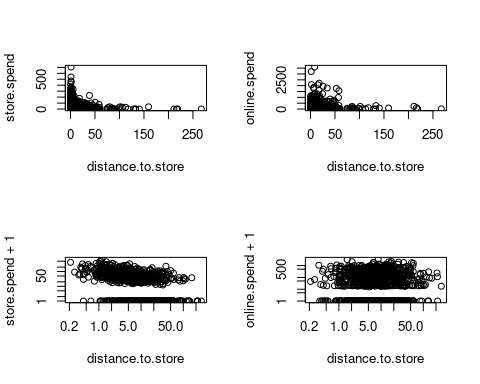
# Looking at the skew  
# ==========  
hist(cust.df$store.spend,   
 breaks=(0:ceiling(max(cust.df$store.spend)/10))\*10,  
 xlab="Prior 12 months online sales ($)" )



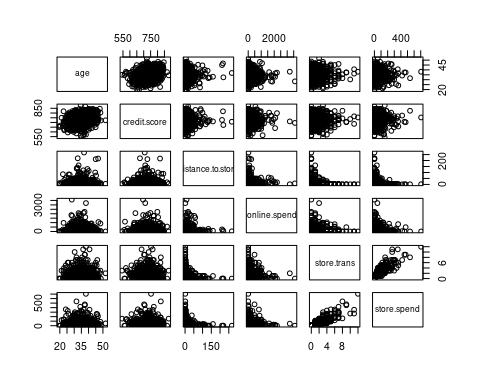
# Using logarithmic axes  
# ==========  
plot(cust.df$store.spend + 1, cust.df$online.spend + 1,  
 log="xy")



# Multi-panel plots  
# ==========  
par(mfrow=c(2, 2))  
with(cust.df, plot(distance.to.store, store.spend))  
with(cust.df, plot(distance.to.store, online.spend))  
with(cust.df, plot(distance.to.store, store.spend+1, log="xy"))  
with(cust.df, plot(distance.to.store, online.spend+1, log="xy"))



# Scatterplot matrix: Quick 2-way Visualization  
# ==========  
pairs(formula = ~ age + credit.score + distance.to.store +   
 online.spend + store.trans + store.spend,  
 data=cust.df)

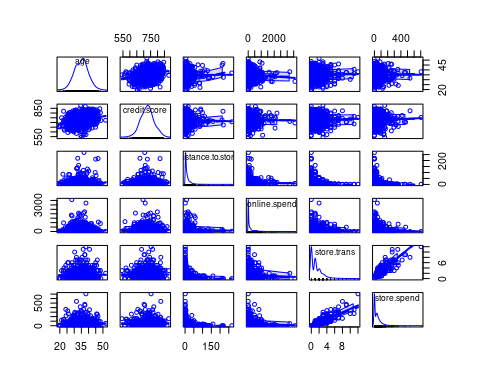


# Fancy alternative: scatterplotMatrix in "car"  
# ==========  
library(car) # install if needed

## Loading required package: carData

scatterplotMatrix(formula = ~ age + credit.score +   
 distance.to.store + online.spend +   
 store.trans + store.spend,  
 data=cust.df, diagonal="histogram")

## Warning in applyDefaults(diagonal, defaults = list(method =  
## "adaptiveDensity"), : unnamed diag arguments, will be ignored



# Correlation  
# ==========  
cor.test(cust.df$age, cust.df$credit.score)

##   
## Pearson's product-moment correlation  
##   
## data: cust.df$age and cust.df$credit.score  
## t = 8.3138, df = 998, p-value = 3.008e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1955974 0.3115816  
## sample estimates:  
## cor   
## 0.2545045

# Correlation matrix  
# ==========  
cor(cust.df[, c(2, 3, 5:12)]) # only numeric cols

## age credit.score distance.to.store online.visits  
## age 1.000000000 0.254504457 0.00198741 -0.06138107  
## credit.score 0.254504457 1.000000000 -0.02326418 -0.01081827  
## distance.to.store 0.001987410 -0.023264183 1.00000000 -0.01460036  
## online.visits -0.061381070 -0.010818272 -0.01460036 1.00000000  
## online.trans -0.063019935 -0.005018400 -0.01955166 0.98732805  
## online.spend -0.060685729 -0.006079881 -0.02040533 0.98240684  
## store.trans 0.024229708 0.040424158 -0.27673229 -0.03666932  
## store.spend 0.003841953 0.042298123 -0.24149487 -0.05068554  
## sat.service NA NA NA NA  
## sat.selection NA NA NA NA  
## online.trans online.spend store.trans store.spend  
## age -0.06301994 -0.060685729 0.02422971 0.003841953  
## credit.score -0.00501840 -0.006079881 0.04042416 0.042298123  
## distance.to.store -0.01955166 -0.020405326 -0.27673229 -0.241494870  
## online.visits 0.98732805 0.982406842 -0.03666932 -0.050685537  
## online.trans 1.00000000 0.993346657 -0.04024588 -0.052244650  
## online.spend 0.99334666 1.000000000 -0.04089133 -0.051690053  
## store.trans -0.04024588 -0.040891332 1.00000000 0.892756851  
## store.spend -0.05224465 -0.051690053 0.89275685 1.000000000  
## sat.service NA NA NA NA  
## sat.selection NA NA NA NA  
## sat.service sat.selection  
## age NA NA  
## credit.score NA NA  
## distance.to.store NA NA  
## online.visits NA NA  
## online.trans NA NA  
## online.spend NA NA  
## store.trans NA NA  
## store.spend NA NA  
## sat.service 1 NA  
## sat.selection NA 1

# Redoing that with complete cases  
# ==========  
cor(cust.df[, c(2, 3, 5:12)], use="complete.obs")

## age credit.score distance.to.store online.visits  
## age 1.00000000 0.27384005 0.04606521 -0.06334468  
## credit.score 0.27384005 1.00000000 -0.03444605 -0.04337523  
## distance.to.store 0.04606521 -0.03444605 1.00000000 -0.02680514  
## online.visits -0.06334468 -0.04337523 -0.02680514 1.00000000  
## online.trans -0.07282280 -0.03041161 -0.03046099 0.98349553  
## online.spend -0.06857108 -0.03344978 -0.03224989 0.97645451  
## store.trans 0.01917930 0.07147923 -0.28777128 -0.01833510  
## store.spend -0.01101162 0.07319630 -0.25249002 -0.06022874  
## sat.service -0.05846361 -0.05095454 0.02561875 -0.01614200  
## sat.selection -0.07411506 -0.02350937 0.01293211 -0.01837661  
## online.trans online.spend store.trans store.spend  
## age -0.07282280 -0.06857108 0.019179304 -0.011011624  
## credit.score -0.03041161 -0.03344978 0.071479231 0.073196297  
## distance.to.store -0.03046099 -0.03224989 -0.287771277 -0.252490015  
## online.visits 0.98349553 0.97645451 -0.018335097 -0.060228738  
## online.trans 1.00000000 0.99306906 -0.026717771 -0.063219201  
## online.spend 0.99306906 1.00000000 -0.025572587 -0.061704685  
## store.trans -0.02671777 -0.02557259 1.000000000 0.892855470  
## store.spend -0.06321920 -0.06170469 0.892855470 1.000000000  
## sat.service -0.01762744 -0.01187352 0.001821736 0.007466294  
## sat.selection -0.01846859 -0.02114840 0.001309098 0.008642354  
## sat.service sat.selection  
## age -0.058463613 -0.074115061  
## credit.score -0.050954538 -0.023509365  
## distance.to.store 0.025618746 0.012932114  
## online.visits -0.016142000 -0.018376612  
## online.trans -0.017627444 -0.018468588  
## online.spend -0.011873515 -0.021148403  
## store.trans 0.001821736 0.001309098  
## store.spend 0.007466294 0.008642354  
## sat.service 1.000000000 0.587855775  
## sat.selection 0.587855775 1.000000000

dev.off()

## null device   
## 1

# Visualize correlation matrix  
# ==========  
# install.packages("corrplot")  
library(corrplot) # install if needed

## corrplot 0.92 loaded

corrplot(corr=cor(cust.df[ , c(2, 3, 5:12)],   
 use="complete.obs"),   
 method ="ellipse")  
  
par(mar=c(5.1, 4.1, 4.1, 2.1))  
# Data Transformation  
# ==========  
cor(cust.df$distance.to.store, cust.df$store.spend)

## [1] -0.2414949

cor(1/cust.df$distance.to.store, cust.df$store.spend)

## [1] 0.4329997

cor(1/sqrt(cust.df$distance.to.store), cust.df$store.spend)

## [1] 0.4843334

# Polychoric correlation  
# ==========  
plot(cust.df$sat.service, cust.df$sat.selection,   
 xlab="Sat, Service", ylab="Sat, Selection")  
  
  
# Polychoric correlation test  
# ==========  
resp <- !is.na(cust.df$sat.service)  
  
cor(cust.df$sat.service[resp], cust.df$sat.selection[resp])

## [1] 0.5878558

library(psych) # install if needed

##   
## Attaching package: 'psych'

## The following object is masked from 'package:car':  
##   
## logit

polychoric(cbind(cust.df$sat.service[resp],   
 cust.df$sat.selection[resp]))

## Call: polychoric(x = cbind(cust.df$sat.service[resp], cust.df$sat.selection[resp]))  
## Polychoric correlations   
## C1 C2   
## R1 1.00   
## R2 0.67 1.00  
##   
## with tau of   
## 1 2 3 4  
## [1,] -1.83 -0.72 0.54 1.7  
## [2,] -0.99 0.12 1.26 2.4

# Exercise!  
# ==========  
library(car) # install.packages("car") if needed  
data(Salaries)  
  
  
# Answers (1)  
# ==========  
with(Salaries, plot(yrs.since.phd, salary))  
  
  
# Answers (2)  
# ==========  
with(Salaries, cor(salary, yrs.since.phd))

## [1] 0.4192311

with(Salaries, cor(salary, yrs.service))

## [1] 0.3347447

with(Salaries, cor.test(salary, yrs.since.phd))

##   
## Pearson's product-moment correlation  
##   
## data: salary and yrs.since.phd  
## t = 9.1775, df = 395, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.3346160 0.4971402  
## sample estimates:  
## cor   
## 0.4192311

with(Salaries, cor.test(salary, yrs.service))

##   
## Pearson's product-moment correlation  
##   
## data: salary and yrs.service  
## t = 7.0602, df = 395, p-value = 7.529e-12  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.2443740 0.4193506  
## sample estimates:  
## cor   
## 0.3347447

# Answers (3)  
# ==========  
library(car)  
scatterplotMatrix(Salaries) # could use pairs() insteadv