Exercise 1

```
#Exercise 1 Missing Data----
##Number of Students----
length(datstu$X)
```

```
## [1] 340823
```

```
##Number of Schools----
length(na.omit(unique(datsss$schoolcode)))
```

```
## [1] 898
```

```
length(na.omit(unique(unlist(datstu[5:10]))))
```

```
## [1] 640
```

```
##Number of Programs----
length(unique(na.omit(unlist(datstu[11:16]))))
```

```
## [1] 33
```

```
##Number of Choices----
datstu=datstu %>% mutate(choice1=paste(schoolcode1, choicepgm1, sep=" "),
                         choice2=paste(schoolcode2, choicepgm2, sep=" "),
                         choice3=paste(schoolcode3, choicepgm3, sep=" "),
                         choice4=paste(schoolcode4, choicepgm4, sep=" "),
                         choice5=paste(schoolcode5, choicepgm5, sep=" "),
                         choice6=paste(schoolcode6, choicepgm6, sep=" "))
dat=select(datstu, choice1, choice2, choice3, choice4, choice5, choice6)
dat_long=gather(dat, 'key', 'value')
dat_long <- subset(dat_long, value!= "NA ")
length(unique(na.omit(dat_long$value)))
```

```
## [1] 3085
```

```
##Number of Missing Test Scores----
sum(is.na(datstu$score))
```

```
## [1] 179887
```

```
##Number of Same School, Different Programs----
for (i in 1:nrow(datstu)){
datstu$sameschool[i]=length(unique(na.omit(unlist(datstu[i,5:10]))))
}
sum(datstu$sameschool<6-rowSums(is.na(datstu[5:10])))
```

```
## [1] 120071
```

```
#sum=result$`combine$dat2.program`
##Apply to less than 6 choices----
length(which(rowSums(dat=="NA ")!=0))
```

```
## [1] 17720
```

Number of students: 340823
Number of schools(Junior High): 898
Number of schools(Senior HIgh that students apply): 640
Number of programs: 33
Number of choices (school,program): 3085
Missing test score: 179887
Apply to the same school (different programs): 120071
Apply to less than 6 choices: 17720
#Note here some of the results may be different due to different manipulations on NA values in original dataset

Exercise 2

```
#Exercise 2 Data----
datstu$admitted_schoolcode=ifelse(datstu$rankplace==1, datstu$schoolcode1,
                       ifelse(datstu$rankplace==2, datstu$schoolcode2,
                              ifelse(datstu$rankplace==3, datstu$schoolcode3,
                                     ifelse(datstu$rankplace==4, datstu$schoolcode4,
                                            ifelse(datstu$rankplace==5, datstu$schoolcode5,
                                                   ifelse(datstu$rankplace==6, datstu$schoolcode6,
NA))))))
datstu$admitted=ifelse(datstu$rankplace==1, datstu$choice1,
                       ifelse(datstu$rankplace==2, datstu$choice2,
                              ifelse(datstu$rankplace==3, datstu$choice3,
                                     ifelse(datstu$rankplace==4, datstu$choice4,
                                            ifelse(datstu$rankplace==5, datstu$choice5,
                                                   ifelse(datstu$rankplace==6, datstu$choice6, NA
))))))
dataset=datstu %>% group_by(admitted) %>% summarise(schoolcode=admitted_schoolcode, minscore=min(s
core), average=mean(score), number=n(), .groups = 'drop')
dataset=dataset %>%
  rename(
    school_program = admitted
  )
dataset=unique(dataset)
datsss$X=NULL
datsss=unique(datsss)
datafinal<-merge(x=dataset,y=datsss,by="schoolcode",all.x=TRUE)
datafinal=na.omit(datafinal)
#Another way to left join
datatest=left_join(dataset, datsss, by="schoolcode")
datatest=na.omit(datatest)
datatest_final=select(datatest, school_program, sssdistrict, ssslat, ssslong, minscore, average, n
umber)
datatest_final=datatest_final %>%
  rename(
    district=sssdistrict,
    latitude=ssslat,
    longitude=ssslong,
    cutoff=minscore,
    quality=average,
    size=number
  )
head(datatest_final, 20)
```

```
## # A tibble: 20 x 7
##    school_program         district     latitude longitude cutoff quality  size
##    <chr>                  <chr>           <dbl>     <dbl>  <int>   <dbl> <int>
##  1 100101 General Arts    Wa Municipal     10.0     -2.29    198    244.    79
##  2 100101 Home Economics  Wa Municipal     10.0     -2.29    199    229.    40
##  3 100101 Technical       Wa Municipal     10.0     -2.29    201    235.    49
##  4 100102 Agriculture     Wa Municipal     10.0     -2.29    273    293.    90
##  5 100102 Business        Wa Municipal     10.0     -2.29    283    303.    90
##  6 100102 General Arts    Wa Municipal     10.0     -2.29    291    311.    90
##  7 100102 General Science Wa Municipal     10.0     -2.29    273    298.    90
##  8 100102 Home Economics  Wa Municipal     10.0     -2.29    262    279.    45
##  9 100102 Visual Arts     Wa Municipal     10.0     -2.29    250    275.    45
## 10 100104 General Arts    Wa Municipal     10.0     -2.29    319    337.    45
## 11 100104 General Science Wa Municipal     10.0     -2.29    313    334     45
## 12 100104 Home Economics  Wa Municipal     10.0     -2.29    282    309.    45
## 13 100105 Business        Wa Municipal     10.0     -2.29    251    268.    80
## 14 100105 General Arts    Wa Municipal     10.0     -2.29    258    275.    80
## 15 100105 Home Economics  Wa Municipal     10.0     -2.29    242    258.    80
## 16 100106 Agriculture     Wa Municipal     10.0     -2.29    223    241.    40
## 17 100106 Business        Wa Municipal     10.0     -2.29    238    254.    40
## 18 100106 General Arts    Wa Municipal     10.0     -2.29    248    269.    40
## 19 100201 Business        Lawra            10.5     -2.80    288    314.    80
## 20 100201 General Arts    Lawra            10.5     -2.80    319    339.    40
```

Exercise 3

```
#Exercise 3 Distance----
datjss=select(datjss, -X)
datstujss=left_join(datstu, datjss, by="jssdistrict")
dat_school=select(datstu, X, choice1:choice6)
dat_school=gather(dat_school, 'key', 'value', -X)
dat_school=left_join(dat_school, datstujss, by=c("X"="X"))
dat_school=select(dat_school, X, key, value, jssdistrict, point_x, point_y)
dat_school=cbind(dat_school, colsplit(dat_school$value," ",c("schoolcode","program")))

m=dat_school%>%group_by(X)

datstujsssss=left_join(dat_school, datsss, by=c("schoolcode"="schoolcode"))
datstujsssss$distance=sqrt(
  (69.172*(datstujsssss$ssslong-datstujsssss$point_x)*cos(datstujsssss$point_y/57.3))^2+(69.172*(datstujsssss$ssslat-datstujsssss$point_y)^2)
)
datstujsssss_omit=datstujsssss[!is.na(datstujsssss$distance),]

datstujsssss_omit=select(datstujsssss_omit, jssdistrict, point_x, point_y, schoolname, sssdistrict, ssslat, ssslong, distance)
datstujsssss_omit=datstujsssss_omit %>%
  rename(
    jsslat=point_y,
    jsslong=point_x,
    sssname=schoolname,
  )

head(datstujsssss_omit, 20)
```

```
##                              jssdistrict    jsslong    jsslat
## 1  Bosomtwe/Atwima/Kwanwoma (Kuntanase) -1.5627517  6.559323
## 3                          Ho Municipal  0.5261422  6.717607
## 4                   Kwabre (Mamponteng) -1.5414201  6.806778
## 6            Kassena/Nankani (Navrongo) -1.2174410 10.909423
## 8              Atwima Mponua (Nyinahin) -2.1771805  6.549507
## 10                         Kumasi Metro -1.5971872  6.682060
## 12             Nanumba North (Bimbilla) -0.1417642  8.816774
## 14                 Jomoro (Half Assini) -2.8032203  5.069508
## 16                     East Akim (Kibi) -0.4543442  6.178558
## 18            Ejura/Sekyedumase (Ejura) -1.3679653  7.462874
## 20               Sekyere West (Mampong) -1.1800768  7.199565
## 22           Kassena/Nankani (Navrongo) -1.2174410 10.909423
## 23                         Agona Swedru -0.7552425  5.617353
## 25               Tolon Kunbungu (Tolon) -1.1097199  9.527246
## 27                   Accra Metropolitan -0.1971153  5.607396
## 29          Mpohor-Wassa East (Daboase) -1.6975694  5.330796
## 31            Ejura/Sekyedumase (Ejura) -1.3679653  7.462874
## 33                   Ga West (Amasaman) -0.3975105  5.664688
## 35            Wassa Amenfi (Asankragwa) -2.3020179  5.725518
## 36                                 Bole -2.2666752  8.629696
##                                                sssname
## 1            KUMASI SENIOR HIGH./TECH. SCHOOL, KUMASI
## 3                      MAWULI SENIOR HIGH. SCHOOL, HO
## 4            SIMMS SENIOR HIGH. COMM. SCHOOL, FAWOADE
## 6          NAVRONGO SENIOR HIGH SCH, JAMANIA-NAVRONGO
## 8            TWENEBOA KODUA SENIOR HIGH. SCH., KUMAWU
## 10        ST. MARY’S SENIOR HIGH. SCHOOL, KORLE GONNO
## 12                  SALAGA SENIOR HIGH SCHOOL, SALAGA
## 14                   NSEIN SENIOR HIGH SCHOOL - NSEIN
## 16                        ABUAKWA STATE COLLEGE, KIBI
## 18                  TAMALE SENIOR HIGH SCHOOL, TAMALE
## 20           TWENEBOA KODUA SENIOR HIGH. SCH., KUMAWU
## 22                  NANDOM SENIOR HIGH SCHOOL, NANDOM
## 23     OBRACHIRE SENIOR HIGH./TECH. SCHOOL, OBRACHIRE
## 25                  TAMALE SENIOR HIGH SCHOOL, TAMALE
## 27 SANDEMA SENIOR HIGH/TECHNICAL SCH, BILINSA-SANDEMA
## 29       FIASEMAN SENIOR HIGH. SCHOOL, BENKYIM TARKWA
## 31                  GHANA SENIOR HIGH. SCHOOL, TAMALE
## 33               GHANATTA SENIOR HIGH. SCHOOL, DODOWA
## 35                BEREKUM SENIOR HIGH SCHOOL, BEREKUM
## 36                          WA SENIOR HIGH SCHOOL, WA
##                    sssdistrict    ssslat    ssslong   distance
## 1                 Kumasi Metro  6.682060 -1.5971872   2.577169
## 3                 Ho Municipal  6.717607  0.5261422   0.000000
## 4          Kwabre (Mamponteng)  6.806778 -1.5414201   0.000000
## 6   Kassena/Nankani (Navrongo) 10.909423 -1.2174410   0.000000
## 8     Sekyere East (Effiduase)  7.210829 -0.8442360  91.765769
## 10          Accra Metropolitan  5.607396 -0.1971153  96.602382
## 12         East Gonja (Salaga)  8.729157 -0.5339396  26.816957
## 14           Nzema East (Axim)  5.141226 -2.3118021  33.864677
## 16            East Akim (Kibi)  6.178558 -0.4543442   0.000000
## 18                      Tamale  9.383351 -0.7843482  43.097181
## 20    Sekyere East (Effiduase)  7.210829 -0.8442360  23.047843
## 22                       Lawra 10.546398 -2.8009412 107.597001
## 23 Awutu/Efutu/Senya (Winneba)  5.544896 -0.5086389  16.986853
## 25                      Tamale  9.383351 -0.7843482  22.228458
## 27            Builsa (Sandema) 10.557073 -1.3374945  88.643584
## 29         Wassa West (Tarkwa)  5.276049 -1.9888532  20.066721
## 31                      Tamale  9.383351 -0.7843482  43.097181
## 33        Dangme West (Dodowa)  5.786251  0.5123865  62.640244
## 35                     Berekum  7.503565 -2.6317439  27.086959
## 36                Wa Municipal 10.030622 -2.2850304  11.718883
```

Exercise 4

```
#Exercise 4 Descriptive Characteristics----
datstujssssss=left_join(datstujssssss, dataset, by=c("value"="school_program"))
score=data.frame(datstu[,1:2])
datstujssssss=left_join(datstujssssss, score, by=c("X"="X"))


#choice1
rank1=na.omit(datstujssssss[datstujssssss$key=="choice1",])
sd_cutoff_choice1=sd(rank1$minscore)
mean_cutoff_choice1=mean(rank1$minscore)
sd_quality_choice1=sd(rank1$average)
mean_quality_choice1=mean(rank1$average)
sd_distance_choice1=sd(rank1$distance)
mean_distance_choice1=mean(rank1$distance)

#choice2
rank2=na.omit(datstujssssss[datstujssssss$key=="choice2",])
sd_cutoff_choice2=sd(rank2$minscore)
mean_cutoff_choice2=mean(rank2$minscore)
sd_quality_choice2=sd(rank2$average)
mean_quality_choice2=mean(rank2$average)
sd_distance_choice2=sd(rank2$distance)
mean_distance_choice2=mean(rank2$distance)

#choice3
rank3=na.omit(datstujssssss[datstujssssss$key=="choice3",])
sd_cutoff_choice3=sd(rank3$minscore)
mean_cutoff_choice3=mean(rank3$minscore)
sd_quality_choice3=sd(rank3$average)
mean_quality_choice3=mean(rank3$average)
sd_distance_choice3=sd(rank3$distance)
mean_distance_choice3=mean(rank3$distance)

#choice4
rank4=na.omit(datstujssssss[datstujssssss$key=="choice4",])
sd_cutoff_choice4=sd(rank4$minscore)
mean_cutoff_choice4=mean(rank4$minscore)
sd_quality_choice4=sd(rank4$average)
mean_quality_choice4=mean(rank4$average)
sd_distance_choice4=sd(rank4$distance)
mean_distance_choice4=mean(rank4$distance)

#choice5
rank5=na.omit(datstujssssss[datstujssssss$key=="choice5",])
sd_cutoff_choice5=sd(rank5$minscore)
mean_cutoff_choice5=mean(rank5$minscore)
sd_quality_choice5=sd(rank5$average)
mean_quality_choice5=mean(rank5$average)
sd_distance_choice5=sd(rank5$distance)
mean_distance_choice5=mean(rank5$distance)

#choice6
rank6=na.omit(datstujssssss[datstujssssss$key=="choice6",])
```

```
sd_cutoff_choice6=sd(rank6$minscore)
mean_cutoff_choice6=mean(rank6$minscore)
sd_quality_choice6=sd(rank6$average)
mean_quality_choice6=mean(rank6$average)
sd_distance_choice6=sd(rank6$distance)
mean_distance_choice6=mean(rank6$distance)

mean_sd_table=cbind(c(sd_cutoff_choice1, sd_cutoff_choice2, sd_cutoff_choice3, sd_cutoff_choice4, sd_cutoff_choice5, sd_cutoff_choice6), c(mean_cutoff_choice1, mean_cutoff_choice2, mean_cutoff_choice3, mean_cutoff_choice4, mean_cutoff_choice5, mean_cutoff_choice6), c(sd_quality_choice1, sd_quality_choice2, sd_quality_choice3, sd_quality_choice4, sd_quality_choice5, sd_quality_choice6), c(mean_quality_choice1, mean_quality_choice2, mean_quality_choice3, mean_quality_choice4, mean_quality_choice5, mean_quality_choice6), c(sd_distance_choice1, sd_distance_choice2, sd_distance_choice3, sd_distance_choice4, sd_distance_choice5, sd_distance_choice6), c(mean_distance_choice1, mean_distance_choice2, mean_distance_choice3, mean_distance_choice4, mean_distance_choice5, mean_distance_choice6))

colnames(mean_sd_table) = c("sd_cutoff", "mean_cutoff","sd_quality","mean_quality", "sd_distance", "mean_distance")
rownames(mean_sd_table) = c("choice1", "choice2","choice3","choice4", "choice5", "choice6")




#datstu_omit=datstu[!is.na(datstu$score),]

#choice1_quantile
rank1_q <- rank1 %>%
  mutate(group = cut(score, c(quantile(score)),
                     labels = 1:4, include.lowest=T))
rank1_q1=rank1_q[rank1_q$group==1, ]
rank1_q2=rank1_q[rank1_q$group==2, ]
rank1_q3=rank1_q[rank1_q$group==3, ]
rank1_q4=rank1_q[rank1_q$group==4, ]

sd_cutoff_r1q1=sd(rank1_q1$minscore)
mean_cutoff_r1q1=mean(rank1_q1$minscore)
sd_cutoff_r1q2=sd(rank1_q2$minscore)
mean_cutoff_r1q2=mean(rank1_q2$minscore)
sd_cutoff_r1q3=sd(rank1_q3$minscore)
mean_cutoff_r1q3=mean(rank1_q3$minscore)
sd_cutoff_r1q4=sd(rank1_q4$minscore)
mean_cutoff_r1q4=mean(rank1_q4$minscore)


sd_quality_r1q1=sd(rank1_q1$average)
mean_quality_r1q1=mean(rank1_q1$average)
sd_quality_r1q2=sd(rank1_q2$average)
mean_quality_r1q2=mean(rank1_q2$average)
sd_quality_r1q3=sd(rank1_q3$average)
mean_quality_r1q3=mean(rank1_q3$average)
sd_quality_r1q4=sd(rank1_q4$average)
mean_quality_r1q4=mean(rank1_q4$average)
```

```
sd_distance_r1q1=sd(rank1_q1$distance)
mean_distance_r1q1=mean(rank1_q1$distance)
sd_distance_r1q2=sd(rank1_q2$distance)
mean_distance_r1q2=mean(rank1_q2$distance)
sd_distance_r1q3=sd(rank1_q3$distance)
mean_distance_r1q3=mean(rank1_q3$distance)
sd_distance_r1q4=sd(rank1_q4$distance)
mean_distance_r1q4=mean(rank1_q4$distance)



#choice2_quantile
rank2_q <- rank2 %>%
  mutate(group = cut(score, c(quantile(score)),
                     labels = 1:4, include.lowest=T))
rank2_q1=rank2_q[rank2_q$group==1, ]
rank2_q2=rank2_q[rank2_q$group==2, ]
rank2_q3=rank2_q[rank2_q$group==3, ]
rank2_q4=rank2_q[rank2_q$group==4, ]

sd_cutoff_r2q1=sd(rank2_q1$minscore)
mean_cutoff_r2q1=mean(rank2_q1$minscore)
sd_cutoff_r2q2=sd(rank2_q2$minscore)
mean_cutoff_r2q2=mean(rank2_q2$minscore)
sd_cutoff_r2q3=sd(rank2_q3$minscore)
mean_cutoff_r2q3=mean(rank2_q3$minscore)
sd_cutoff_r2q4=sd(rank2_q4$minscore)
mean_cutoff_r2q4=mean(rank2_q4$minscore)


sd_quality_r2q1=sd(rank2_q1$average)
mean_quality_r2q1=mean(rank2_q1$average)
sd_quality_r2q2=sd(rank2_q2$average)
mean_quality_r2q2=mean(rank2_q2$average)
sd_quality_r2q3=sd(rank2_q3$average)
mean_quality_r2q3=mean(rank2_q3$average)
sd_quality_r2q4=sd(rank2_q4$average)
mean_quality_r2q4=mean(rank2_q4$average)

sd_distance_r2q1=sd(rank2_q1$distance)
mean_distance_r2q1=mean(rank2_q1$distance)
sd_distance_r2q2=sd(rank2_q2$distance)
mean_distance_r2q2=mean(rank2_q2$distance)
sd_distance_r2q3=sd(rank2_q3$distance)
mean_distance_r2q3=mean(rank2_q3$distance)
sd_distance_r2q4=sd(rank2_q4$distance)
mean_distance_r2q4=mean(rank2_q4$distance)

#choice3_quantile
rank3_q <- rank3 %>%
  mutate(group = cut(score, c(quantile(score)),
                     labels = 1:4, include.lowest=T))
rank3_q1=rank3_q[rank3_q$group==1, ]
rank3_q2=rank3_q[rank3_q$group==2, ]
```

```
rank3_q3=rank3_q[rank3_q$group==3, ]
rank3_q4=rank3_q[rank3_q$group==4, ]

sd_cutoff_r3q1=sd(rank3_q1$minscore)
mean_cutoff_r3q1=mean(rank3_q1$minscore)
sd_cutoff_r3q2=sd(rank3_q2$minscore)
mean_cutoff_r3q2=mean(rank3_q2$minscore)
sd_cutoff_r3q3=sd(rank3_q3$minscore)
mean_cutoff_r3q3=mean(rank3_q3$minscore)
sd_cutoff_r3q4=sd(rank3_q4$minscore)
mean_cutoff_r3q4=mean(rank3_q4$minscore)


sd_quality_r3q1=sd(rank3_q1$average)
mean_quality_r3q1=mean(rank3_q1$average)
sd_quality_r3q2=sd(rank3_q2$average)
mean_quality_r3q2=mean(rank3_q2$average)
sd_quality_r3q3=sd(rank3_q3$average)
mean_quality_r3q3=mean(rank3_q3$average)
sd_quality_r3q4=sd(rank3_q4$average)
mean_quality_r3q4=mean(rank3_q4$average)

sd_distance_r3q1=sd(rank3_q1$distance)
mean_distance_r3q1=mean(rank3_q1$distance)
sd_distance_r3q2=sd(rank3_q2$distance)
mean_distance_r3q2=mean(rank3_q2$distance)
sd_distance_r3q3=sd(rank3_q3$distance)
mean_distance_r3q3=mean(rank3_q3$distance)
sd_distance_r3q4=sd(rank3_q4$distance)
mean_distance_r3q4=mean(rank3_q4$distance)

#choice4_quantile
rank4_q <- rank4 %>%
  mutate(group = cut(score, c(quantile(score)),
                     labels = 1:4, include.lowest=T))
rank4_q1=rank4_q[rank4_q$group==1, ]
rank4_q2=rank4_q[rank4_q$group==2, ]
rank4_q3=rank4_q[rank4_q$group==3, ]
rank4_q4=rank4_q[rank4_q$group==4, ]

sd_cutoff_r4q1=sd(rank4_q1$minscore)
mean_cutoff_r4q1=mean(rank4_q1$minscore)
sd_cutoff_r4q2=sd(rank4_q2$minscore)
mean_cutoff_r4q2=mean(rank4_q2$minscore)
sd_cutoff_r4q3=sd(rank4_q3$minscore)
mean_cutoff_r4q3=mean(rank4_q3$minscore)
sd_cutoff_r4q4=sd(rank4_q4$minscore)
mean_cutoff_r4q4=mean(rank4_q4$minscore)


sd_quality_r4q1=sd(rank4_q1$average)
mean_quality_r4q1=mean(rank4_q1$average)
sd_quality_r4q2=sd(rank4_q2$average)
mean_quality_r4q2=mean(rank4_q2$average)
```

```
sd_quality_r4q3=sd(rank4_q3$average)
mean_quality_r4q3=mean(rank4_q3$average)
sd_quality_r4q4=sd(rank4_q4$average)
mean_quality_r4q4=mean(rank4_q4$average)

sd_distance_r4q1=sd(rank4_q1$distance)
mean_distance_r4q1=mean(rank4_q1$distance)
sd_distance_r4q2=sd(rank4_q2$distance)
mean_distance_r4q2=mean(rank4_q2$distance)
sd_distance_r4q3=sd(rank4_q3$distance)
mean_distance_r4q3=mean(rank4_q3$distance)
sd_distance_r4q4=sd(rank4_q4$distance)
mean_distance_r4q4=mean(rank4_q4$distance)

#choice5_quantile
rank5_q <- rank5 %>%
  mutate(group = cut(score, c(quantile(score)),
                     labels = 1:4, include.lowest=T))
rank5_q1=rank5_q[rank5_q$group==1, ]
rank5_q2=rank5_q[rank5_q$group==2, ]
rank5_q3=rank5_q[rank5_q$group==3, ]
rank5_q4=rank5_q[rank5_q$group==4, ]

sd_cutoff_r5q1=sd(rank5_q1$minscore)
mean_cutoff_r5q1=mean(rank5_q1$minscore)
sd_cutoff_r5q2=sd(rank5_q2$minscore)
mean_cutoff_r5q2=mean(rank5_q2$minscore)
sd_cutoff_r5q3=sd(rank5_q3$minscore)
mean_cutoff_r5q3=mean(rank5_q3$minscore)
sd_cutoff_r5q4=sd(rank5_q4$minscore)
mean_cutoff_r5q4=mean(rank5_q4$minscore)


sd_quality_r5q1=sd(rank5_q1$average)
mean_quality_r5q1=mean(rank5_q1$average)
sd_quality_r5q2=sd(rank5_q2$average)
mean_quality_r5q2=mean(rank5_q2$average)
sd_quality_r5q3=sd(rank5_q3$average)
mean_quality_r5q3=mean(rank5_q3$average)
sd_quality_r5q4=sd(rank5_q4$average)
mean_quality_r5q4=mean(rank5_q4$average)

sd_distance_r5q1=sd(rank5_q1$distance)
mean_distance_r5q1=mean(rank5_q1$distance)
sd_distance_r5q2=sd(rank5_q2$distance)
mean_distance_r5q2=mean(rank5_q2$distance)
sd_distance_r5q3=sd(rank5_q3$distance)
mean_distance_r5q3=mean(rank5_q3$distance)
sd_distance_r5q4=sd(rank5_q4$distance)
mean_distance_r5q4=mean(rank5_q4$distance)

#choice6_quantile
rank6_q <- rank6 %>%
  mutate(group = cut(score, c(quantile(score)),
                     labels = 1:4, include.lowest=T))
rank6_q1=rank6_q[rank6_q$group==1, ]
rank6_q2=rank6_q[rank6_q$group==2, ]
rank6_q3=rank6_q[rank6_q$group==3, ]
rank6_q4=rank6_q[rank6_q$group==4, ]

sd_cutoff_r6q1=sd(rank6_q1$minscore)
mean_cutoff_r6q1=mean(rank6_q1$minscore)
sd_cutoff_r6q2=sd(rank6_q2$minscore)
mean_cutoff_r6q2=mean(rank6_q2$minscore)
sd_cutoff_r6q3=sd(rank6_q3$minscore)
mean_cutoff_r6q3=mean(rank6_q3$minscore)
sd_cutoff_r6q4=sd(rank6_q4$minscore)
mean_cutoff_r6q4=mean(rank6_q4$minscore)


sd_quality_r6q1=sd(rank6_q1$average)
mean_quality_r6q1=mean(rank6_q1$average)
sd_quality_r6q2=sd(rank6_q2$average)
mean_quality_r6q2=mean(rank6_q2$average)
sd_quality_r6q3=sd(rank6_q3$average)
mean_quality_r6q3=mean(rank6_q3$average)
sd_quality_r6q4=sd(rank6_q4$average)
mean_quality_r6q4=mean(rank6_q4$average)

sd_distance_r6q1=sd(rank6_q1$distance)
mean_distance_r6q1=mean(rank6_q1$distance)
sd_distance_r6q2=sd(rank6_q2$distance)
mean_distance_r6q2=mean(rank6_q2$distance)
sd_distance_r6q3=sd(rank6_q3$distance)
mean_distance_r6q3=mean(rank6_q3$distance)
sd_distance_r6q4=sd(rank6_q4$distance)
mean_distance_r6q4=mean(rank6_q4$distance)

mean_sd_table_quantile_1=cbind(c(sd_cutoff_r1q1, sd_cutoff_r2q1, sd_cutoff_r3q1, sd_cutoff_r4q1, sd_cutoff_r5q1, sd_cutoff_r6q1), c(mean_cutoff_r1q1, mean_cutoff_r2q1, mean_cutoff_r3q1, mean_cutoff_r4q1, mean_cutoff_r5q1, mean_cutoff_r6q1), c(sd_quality_r1q1, sd_quality_r2q1, sd_quality_r3q1, sd_quality_r4q1, sd_quality_r5q1, sd_quality_r6q1), c(mean_quality_r1q1, mean_quality_r2q1, mean_quality_r3q1, mean_quality_r4q1, mean_quality_r5q1, mean_quality_r6q1), c(sd_distance_r1q1, sd_distance_r2q1, sd_distance_r3q1, sd_distance_r4q1, sd_distance_r5q1, sd_distance_r6q1), c(mean_distance_r1q1, mean_distance_r2q1, mean_distance_r3q1, mean_distance_r4q1, mean_distance_r5q1, mean_distance_r6q1))

colnames(mean_sd_table_quantile_1) = c("sd_cutoff", "mean_cutoff","sd_quality","mean_quality", "sd_distance", "mean_distance")
rownames(mean_sd_table_quantile_1) = c("choice1", "choice2","choice3","choice4", "choice5", "choice6")

mean_sd_table_quantile_2=cbind(c(sd_cutoff_r1q2, sd_cutoff_r2q2, sd_cutoff_r3q2, sd_cutoff_r4q2, sd_cutoff_r5q2, sd_cutoff_r6q2), c(mean_cutoff_r1q2, mean_cutoff_r2q2, mean_cutoff_r3q2, mean_cutoff_r4q2, mean_cutoff_r5q2, mean_cutoff_r6q2), c(sd_quality_r1q2, sd_quality_r2q2, sd_quality_r3q2, sd_quality_r4q2, sd_quality_r5q2, sd_quality_r6q2), c(mean_quality_r1q2, mean_quality_r2q2, mean_quality_r3q2, mean_quality_r4q2, mean_quality_r5q2, mean_quality_r6q2), c(sd_distance_r1q2, sd_distance_r2q2, sd_distance_r3q2, sd_distance_r4q2, sd_distance_r5q2, sd_distance_r6q2), c(mean_distan
ce_r1q2, mean_distance_r2q2, mean_distance_r3q2, mean_distance_r4q2, mean_distance_r5q2, mean_dist
ance_r6q2))

colnames(mean_sd_table_quantile_2) = c("sd_cutoff", "mean_cutoff","sd_quality","mean_quality", "sd
_distance", "mean_distance")
rownames(mean_sd_table_quantile_2) = c("choice1", "choice2","choice3","choice4", "choice5", "choic
e6")

mean_sd_table_quantile_3=cbind(c(sd_cutoff_r1q3, sd_cutoff_r2q3, sd_cutoff_r3q3, sd_cutoff_r4q3, s
d_cutoff_r5q3, sd_cutoff_r6q3), c(mean_cutoff_r1q3, mean_cutoff_r2q3, mean_cutoff_r3q3, mean_cutof
f_r4q3, mean_cutoff_r5q3, mean_cutoff_r6q3), c(sd_quality_r1q3, sd_quality_r2q3, sd_quality_r3q3,
 sd_quality_r4q3, sd_quality_r5q3, sd_quality_r6q3), c(mean_quality_r1q3, mean_quality_r2q3, mean_
quality_r3q3, mean_quality_r4q3, mean_quality_r5q3, mean_quality_r6q3), c(sd_distance_r1q3, sd_dis
tance_r2q3, sd_distance_r3q3, sd_distance_r4q3, sd_distance_r5q3, sd_distance_r6q3), c(mean_distan
ce_r1q3, mean_distance_r2q3, mean_distance_r3q3, mean_distance_r4q3, mean_distance_r5q3, mean_dist
ance_r6q3))

colnames(mean_sd_table_quantile_3) = c("sd_cutoff", "mean_cutoff","sd_quality","mean_quality", "sd
_distance", "mean_distance")
rownames(mean_sd_table_quantile_3) = c("choice1", "choice2","choice3","choice4", "choice5", "choic
e6")

mean_sd_table_quantile_4=cbind(c(sd_cutoff_r1q4, sd_cutoff_r2q4, sd_cutoff_r3q4, sd_cutoff_r4q4, s
d_cutoff_r5q4, sd_cutoff_r6q4), c(mean_cutoff_r1q4, mean_cutoff_r2q4, mean_cutoff_r3q4, mean_cutof
f_r4q4, mean_cutoff_r5q4, mean_cutoff_r6q4), c(sd_quality_r1q4, sd_quality_r2q4, sd_quality_r3q4,
 sd_quality_r4q4, sd_quality_r5q4, sd_quality_r6q4), c(mean_quality_r1q4, mean_quality_r2q4, mean_
quality_r3q4, mean_quality_r4q4, mean_quality_r5q4, mean_quality_r6q4), c(sd_distance_r1q4, sd_dis
tance_r2q4, sd_distance_r3q4, sd_distance_r4q4, sd_distance_r5q4, sd_distance_r6q4), c(mean_distan
ce_r1q4, mean_distance_r2q4, mean_distance_r3q4, mean_distance_r4q4, mean_distance_r5q4, mean_dist
ance_r6q4))

colnames(mean_sd_table_quantile_4) = c("sd_cutoff", "mean_cutoff","sd_quality","mean_quality", "sd
_distance", "mean_distance")
rownames(mean_sd_table_quantile_4) = c("choice1", "choice2","choice3","choice4", "choice5", "choic
e6")

mean_sd_table
```

```
##         sd_cutoff mean_cutoff sd_quality mean_quality sd_distance mean_distance
## choice1  52.37614    315.5782   47.07851     336.6502    28.33323      20.84311
## choice2  49.11026    298.3855   43.35033     320.3354    27.60721      20.26696
## choice3  47.31512    286.0724   41.27931     309.2015    26.41314      18.98818
## choice4  46.01771    272.6580   39.71273     297.6116    25.50565      16.57650
## choice5  31.88871    255.9669   25.72390     283.7554    21.12574      19.29866
## choice6  31.54638    250.9539   25.57021     279.3014    21.13284      19.56513
```

```
mean_sd_table_quantile_1
```

```
##         sd_cutoff mean_cutoff sd_quality mean_quality sd_distance mean_distance
## choice1  44.58961    281.9352   38.90879     305.4819    26.07438      17.15765
## choice2  41.42142    268.7200   35.91141     293.2921    25.95436      17.62458
## choice3  40.44757    260.1197   34.95345     285.6953    25.52405      17.17507
## choice4  39.64101    250.1035   34.16101     277.2046    25.04035      15.72636
## choice5  31.07920    246.4197   25.64508     273.9869    21.83625      19.07736
## choice6  30.58503    242.0347   25.44333     270.1349    21.83437      19.36456
```

```
mean_sd_table_quantile_2
```

```
##         sd_cutoff mean_cutoff sd_quality mean_quality sd_distance mean_distance
## choice1  45.24089    299.2065   39.38205     321.2432    27.19540      19.04230
## choice2  42.45549    283.6100   36.66318     306.7683    26.92121      18.95797
## choice3  41.40262    272.4466   35.48793     296.8900    26.27882      18.21268
## choice4  40.78319    260.6210   34.61951     286.8788    25.58141      16.18064
## choice5  31.45672    252.7596   25.39447     280.3306    21.90799      19.50463
## choice6  31.02043    248.1978   25.13213     276.3406    21.77805      19.77792
```

```
mean_sd_table_quantile_3
```

```
##         sd_cutoff mean_cutoff sd_quality mean_quality sd_distance mean_distance
## choice1  44.10649    321.2082   38.70195     341.4040    27.87830      20.73886
## choice2  42.73055    302.5646   36.88891     323.9980    27.31597      20.35313
## choice3  41.87320    289.1472   35.85557     311.9115    25.90766      19.06081
## choice4  41.46432    274.8737   35.11525     299.6255    25.52078      16.78128
## choice5  31.26999    259.3355   24.50818     286.8612    21.02212      19.57518
## choice6  31.20494    253.8833   24.73753     282.0121    21.18129      19.85557
```

```
mean_sd_table_quantile_4
```

```
##         sd_cutoff mean_cutoff sd_quality mean_quality sd_distance mean_distance
## choice1  39.08711    360.8319   35.61869     379.2900    31.12758      26.54233
## choice2  39.02456    339.3293   34.53791     357.9088    29.70252      24.19745
## choice3  40.22921    323.1614   34.86815     342.8397    27.71604      21.54451
## choice4  42.03751    305.9293   35.92919     327.5434    25.84435      17.64777
## choice5  30.53845    265.4006   22.88918     293.8943    19.64661      19.03640
## choice6  30.59588    259.7650   23.11111     288.7887    19.64730      19.25976
```

Exercise 5

```
#Exercise 5 Data Creation----
set.seed(100)
x1=runif(10000, 1, 3)
x2=rgamma(10000, 3, scale=2)
x3=rbinom(10000, 1, prob=0.3)
e=rnorm(10000, 2, 1)

y=0.5+1.2*x1-0.9*x2+0.1*x3+e
#hist(y)
ydum=as.numeric((y>mean(y)))
```

Exercise 6

```
#Exercise 6 OLS----
cor(y, x1)
```

```
## [1] 0.2162074
```

```
X=cbind(1, x1, x2, x3)
b=solve(t(X)%*%X)%*%(t(X)%*%y)
y_hat=X%*%b
e_hat_2=(y-X%*%b)^2
b
```

```
##          [,1]
##     2.4561034
## x1  1.2158000
## x2 -0.8984434
## x3  0.1018762
```

```
Var=sum(e_hat_2)/(nrow(X)-ncol(X))*solve(t(X)%*%X)

standard_error=sqrt(diag(Var))
standard_error
```

```
##                      x1          x2          x3
## 0.040982313 0.017491090 0.002952839 0.022040052
```

```
#check
#model=lm(y~X)
#summary(model)
```

Correlation of y and X_1 is shown above.
Coefficient of y on x_1 is 1.24059, it’s not very different from 1.2, and is statistically significant.
Coefficients of y on X is shown above as b.
Standard Error is shown above

Exercise 7

```
#Exercise 7 Discrete Choice----

#probit
reg1 = glm(ydum~x1+x2+x3,family = binomial(link = "probit"))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(reg1)
```

```
## 
## Call:
## glm(formula = ydum ~ x1 + x2 + x3, family = binomial(link = "probit"))
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -3.6273  -0.1177   0.0086   0.2557   3.8444  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  2.81677    0.09726  28.962   <2e-16 ***
## x1           1.23905    0.04414  28.071   <2e-16 ***
## x2          -0.89214    0.01804 -49.457   <2e-16 ***
## x3           0.04804    0.04686   1.025    0.305    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 13721.5  on 9999  degrees of freedom
## Residual deviance:  4372.7  on 9996  degrees of freedom
## AIC: 4380.7
## 
## Number of Fisher Scoring iterations: 7
```

```
flike = function(par,x1,x2,x3,ydum)
{
  xbeta           = par[1] + par[2]*x1 + par[3]*x2 + par[4]*x3
  pr              = pnorm(xbeta)
  #  pr              = exp(beta)/(1+exp(beta)) logit
  pr[pr>0.999999] = 0.999999
  pr[pr<0.000001] = 0.000001
  like           = ydum*log(pr) + (1-ydum)*log(1-pr)
  return(-sum(like))
}

start = runif(4)
res  = optim(start,fn=flike,method="BFGS",control=list(trace=6,REPORT=1,maxit=1000),x1=x1,x2=x2,x3
=x3,ydum=ydum,hessian=TRUE)
```

```
## initial  value 58837.284212 
## iter   2 value 18055.824932
## iter   3 value 17942.782391
## iter   4 value 17665.040324
## iter   5 value 17408.016147
## iter   6 value 14998.783573
## iter   7 value 13222.508184
## iter   8 value 9224.372033
## iter   9 value 7834.882778
## iter  10 value 6104.742857
## iter  11 value 6010.742693
## iter  12 value 5007.939605
## iter  13 value 3809.706013
## iter  14 value 2586.378126
## iter  15 value 2367.133140
## iter  16 value 2269.511192
## iter  17 value 2205.008428
## iter  18 value 2188.106920
## iter  19 value 2186.459286
## iter  20 value 2186.449395
## iter  21 value 2186.396385
## iter  22 value 2186.360480
## iter  23 value 2186.358410
## iter  24 value 2186.348331
## iter  24 value 2186.348329
## iter  24 value 2186.348329
## final  value 2186.348329 
## converged
```

```
fisher_info_probit = solve(res$hessian)       # standard formula is -res$hessian but flike is retu
rn -like
prop_sigma_probit  = sqrt(diag(fisher_info_probit))
#prop_sigma


#logit
reg2 = glm(ydum~x1+x2+x3,family = binomial(link = "logit"))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(reg2)
```

```
##
## Call:
## glm(formula = ydum ~ x1 + x2 + x3, family = binomial(link = "logit"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2817  -0.1535   0.0401   0.2656   3.4123
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.06602    0.18221  27.803   <2e-16 ***
## x1           2.23094    0.08252  27.037   <2e-16 ***
## x2          -1.60595    0.03612 -44.466   <2e-16 ***
## x3           0.08672    0.08425   1.029    0.303
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 13721.5  on 9999  degrees of freedom
## Residual deviance:  4381.2  on 9996  degrees of freedom
## AIC: 4389.2
##
## Number of Fisher Scoring iterations: 7
```

```
flike_logit = function(par,x1,x2,x3,ydum)
{
  xbeta_logit           = par[1] + par[2]*x1 + par[3]*x2 + par[4]*x3
  pr_logit              = exp(xbeta_logit)/(1+exp(xbeta_logit))
  pr_logit[pr_logit>0.999999] = 0.999999
  pr_logit[pr_logit<0.000001] = 0.000001
  like_logit           = ydum*log(pr_logit) + (1-ydum)*log(1-pr_logit)
  return(-sum(like_logit))
}

start_logit = runif(4)
res_logit  = optim(start_logit,fn=flike_logit,method="BFGS",control=list(trace=6,REPORT=1,maxit=10
00),x1=x1,x2=x2,x3=x3,ydum=ydum,hessian=TRUE)
```

```
## initial  value 21598.058963
## iter   2 value 6409.437429
## iter   3 value 3895.442834
## iter   4 value 3606.474681
## iter   5 value 3525.163081
## iter   6 value 2337.939740
## iter   7 value 2208.140378
## iter   8 value 2198.334725
## iter   9 value 2196.788720
## iter  10 value 2190.741896
## iter  11 value 2190.636517
## iter  12 value 2190.617073
## iter  13 value 2190.609501
## iter  14 value 2190.609107
## iter  15 value 2190.593202
## iter  16 value 2190.592006
## iter  16 value 2190.592006
## iter  16 value 2190.592006
## final  value 2190.592006
## converged
```

```
fisher_info_logit = solve(res_logit$hessian)       # standard formula is -res$hessian but flike is
return -like
prop_sigma_logit  = sqrt(diag(fisher_info_logit))
#prop_sigma_logit

#Linear
linear=lm(ydum~x1+x2+x3)
summary(linear)
```

```
##
## Call:
## lm(formula = ydum ~ x1 + x2 + x3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.90570 -0.26599  0.05805  0.24995  2.35722
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.8795230  0.0134596   65.345   <2e-16 ***
## x1           0.1520890  0.0057445   26.476   <2e-16 ***
## x2          -0.1055427  0.0009698 -108.831   <2e-16 ***
## x3           0.0105571  0.0072385    1.458    0.145
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3305 on 9996 degrees of freedom
## Multiple R-squared:  0.5571, Adjusted R-squared:  0.557
## F-statistic:  4191 on 3 and 9996 DF,  p-value: < 2.2e-16
```

```
est_probit = cbind(summary(reg1)$coefficients[, 1],summary(reg1)$coefficients[, 2],res$par,prop_si
gma_probit)
colnames(est_probit) = c("Probit : est","Probit :se","Probit: own : est","Probit: own :se")
est_probit
```

```
##             Probit : est Probit :se Probit: own : est Probit: own :se
## (Intercept)   2.81677032 0.09725690        2.81678796      0.09744932
## x1            1.23905407 0.04413997        1.23906199      0.04429841
## x2           -0.89214080 0.01803881       -0.89214661      0.01800619
## x3            0.04803623 0.04686155        0.04803662      0.04693303
```

```
est_logit = cbind(summary(reg2)$coefficients[, 1],summary(reg2)$coefficients[, 2],res_logit$par,pr
op_sigma_logit)
colnames(est_logit) = c("Logit : est","Logit :se","Logit: own : est","Logit: own :se")
est_logit
```

```
##             Logit : est  Logit :se Logit: own : est Logit: own :se
## (Intercept)  5.06601765 0.18221378        5.0660224     0.18221706
## x1           2.23093874 0.08251564        2.2309371     0.08251716
## x2          -1.60595037 0.03611664       -1.6059501     0.03611765
## x3           0.08672068 0.08425283        0.0867122     0.08425381
```

Table est_probit shows the point estimation and SE of porbit model and optimization results.
Table est_logit shows the point estimation and SE of logit model and optimization results.
The value of the point estimation of probit and logit model is quite different, which doesn't matter because the point estimation doesn't tell us the marginal effect.
The sign of the point estimation of probit and logit model is the same, which makes sense, because although we cannot interpret the magnitude, we can interpret if it's more likely or less likely for the dependent variable to be 1 in this case keeping all else constant.

In terms of significance, by calculating the p value, point estimation of probit and logit model is significant. In linear model, however, coefficient for x3 is not significant, coefficients for x1&x2 are significant though.

Exercise 8

```
#Exercise 8 Marginal Effects----
#Probit Model Average Marginal Effects
probit_scalar=mean(dnorm(X%*%res$par))
probit_margin=as.matrix(probit_scalar*res$par)
probit_margin
```

```
##              [,1]
## [1,]  0.342005646
## [2,]  0.150443058
## [3,] -0.108321670
## [4,]  0.005832458
```

```
#Logit Model Average Marginal Effects
logit_scalar=mean(dlogis(X%*%res_logit$par))
logit_margin=as.matrix(logit_scalar*res_logit$par)
logit_margin
```

```
##              [,1]
## [1,]  0.340762203
## [2,]  0.150062314
## [3,] -0.108023028
## [4,]  0.005832631
```

```
#SE
X_all=as.data.frame(cbind(ydum,x1,x2,x3))
x_mean=as.matrix(colMeans(X_all))
mat=as.matrix(res$par)
lll=length(res$par)
xb=t(x_mean)%*%mat
vcv=solve(res$hessian)
gr=apply(cbind(1, x1, x2, x3), 1, function(x){
as.numeric(as.numeric(dnorm(x %*% mat))*(diag(lll) - as.numeric(x %*% mat)*(mat %*% t(x))))
})
gr = matrix(apply(gr,1,mean),nrow=lll)
Probit_marg_SE = sqrt(diag(gr %*% vcv %*% t(gr)))
Probit_marg_SE
```

```
## [1] 0.0096665150 0.0044774860 0.0003842982 0.0056974086
```

```
X_all=as.data.frame(cbind(ydum,x1,x2,x3))
x_mean=as.matrix(colMeans(X_all))
mat=as.matrix(res_logit$par)
lll=length(res_logit$par)
xb=t(x_mean)%*%mat
vcv=solve(res_logit$hessian)
gr = apply(cbind(1,x1,x2,x3), 1, function(x){
as.numeric(as.numeric(plogis(x %*% mat)*(1-plogis(x %*% mat)))*
(diag(lll) - (1 - 2*as.numeric(plogis(x %*% mat)))*(mat %*% t(x))))
})
gr = matrix(apply(gr,1,mean),nrow=lll)
Logit_marg_SE = sqrt(diag(gr %*% vcv %*% t(gr)))
Logit_marg_SE
```

```
## [1] 0.017689861 0.007873916 0.004707562 0.005671145
```

Marginal effect of probit and logit model is calculated in probit_margin and logit_margin.
Standard error of probit and logit marginal effects is shown in Probit_marg_SE and Logit_marg_SE.