Final Project

Advait Rajeshkumar Shah

04/04/2022

START

Section 1: R Programming Questions: ####################################

Q.1.:->

We have been provided with following information in the question: A clinic has three doctors. Patients come into the clinic at random, starting at 9 a.m. The time after opening at which the first patient appears follows an exponential distribution with expectation of 10 minutes and then, after each patient arrives, the waiting time until the next patient is independently exponentially distributed, also with expectation of 10 minutes. When a patient arrives, he or she waits until a doctor is available. The amount of time spent by each doctor with each patient is a random variable, uniformly distributed between 5 and 20 minutes. The office stops admitting new patients at 4 p.m. and closes when the last patient is through with the doctor.

Q.1.(a)

In this part of question, we need to simulate the above process once.

```
#setting a random seed value of 243 to get same/ reproducible results every time we run this code
set.seed(243)

# expected arrival waiting time = 10 mins
# and arrivals are following an independent exponential distribution
#therefore, (1/lambda) = 10,
#therefore, lambda = 0.1

#therefore, we can estimate the arrival times of patient during these total of 420 mins (between 9 a.m.
and 4 p.m.)
#for this, we will use a built-in function REXP in R for random exponential numbers generation with lam
bda = 0.1

AT <- rexp(100, rate = 0.1)

#calculating cumulative arrival times
CAT <- cumsum(AT)

#from this cumulative sum vector "CAT", we will find number of patients who arrived before 420 minutes
or 4 p.m.
A <- sum(CAT<420)
A
```

```
## [1] 32
```

```
#so , from the result of vector "A", we can observe that in this case, with set.seed(243), 32 number of
patients arrived before 420 minutes or 4 p.m.
# This gives us answer to part (i) of our question

########################################################################

#now, we will generate data regarding amount of time spent by each patient during consultation with the
doctor
DT <- runif(A, min = 5, max = 20)

#creating a matrix to determine how each patient will be assigned with a doctor and how much will be th
e wait time

#first creating a matrix of NA values
W_MAT <- matrix(NA, nrow = A, ncol = 10)

#assigning names to all columns of the matrix
colnames(W_MAT) <- c("Arr Time", "Dr6/7/8", "Starts", "Dur", "Ends", "Dr6", "Dr7", "Dr8", "W", "WT")

#note that we have designated 3 doctors of the clinic with Dr6, Dr7 and Dr8 names respectively as they
are placed in the 6th, 7th and 8th column of the matrix

#we will assign doctor Dr6/7/8 to patient such that:
#(a) if all doctors are busy, he need to wait for a doctor only until any one doctor becomes available
#(b) if multiple doctors are available for a patient, he will go to the doctor who was available from t
he earliest time frame

#for the first patient (i.e. first row), assigning values for all respective columns 1 to 10 named as
"Arr Time", "Dr6/7/8", "Starts", "Dur", "Ends", "Dr6", "Dr7", "Dr8", "W", "WT"
W_MAT[1,1] <- CAT[1]
W_MAT[1,2] <- 6
W_MAT[1,3] <- W_MAT[1,1]
W_MAT[1,4] <- DT[1]
W_MAT[1,5] <- sum(W_MAT[1,3],W_MAT[1,4])
W_MAT[1,6] <- W_MAT[1,5]
W_MAT[1,7] <- 0
W_MAT[1,8] <- 0
W_MAT[1,9] <- 0
W_MAT[1,10] <- 0

#now assigning other patient data in similar way with above stated concepts
for(i in 2:A){

  #using cumulative arrival times of patients
  W_MAT[i,1] <- CAT[i]

  #we assign doctor Dr6/7/8 to patient such that:
  #(a) if all doctors are busy, he need to wait for a doctor only until any one doctor becomes availabl
e
  #(b) if multiple doctors are available for a patient, he will go to the doctor who was available from
the earliest time frame
  W_MAT[i,2] <- sum(5,which.min(c(W_MAT[i-1,6],W_MAT[i-1,7],W_MAT[i-1,8])))

  if(W_MAT[i,1] < W_MAT[i-1,W_MAT[i,2]]){
    W_MAT[i,3] <- W_MAT[i-1,W_MAT[i,2]]
  }else{
    W_MAT[i,3] <- W_MAT[i,1]
  }
```

```
  #using time spent by patients with doctor
  W_MAT[i,4] <- DT[i]

  #exit time of that patient from clinic
  W_MAT[i,5] <- sum(W_MAT[i,3],W_MAT[i,4])

  #updating Dr6 occupancy timestamp if this patient is in consultation with doctor
  if(W_MAT[i,2] == 6){
    W_MAT[i,6] <- W_MAT[i,5]
  }else{
    W_MAT[i,6] <- W_MAT[i-1,6]
  }

  #updating Dr7 occupancy timestamp if this patient is in consultation with doctor
  if(W_MAT[i,2] == 7){
    W_MAT[i,7] <- W_MAT[i,5]
  }else{
    W_MAT[i,7] <- W_MAT[i-1,7]
  }

  #updating Dr8 occupancy timestamp if this patient is in consultation with doctor
  if(W_MAT[i,2] == 8){
    W_MAT[i,8] <- W_MAT[i,5]
  }else{
    W_MAT[i,8] <- W_MAT[i-1,8]
  }

  #determining if this patient had to wait or not
  W_MAT[i,9] <- (W_MAT[i,"Arr Time"] < W_MAT[i,"Starts"]) * 1

  #determining wait time if this patient had to wait
  W_MAT[i,10] <- (W_MAT[i,"Starts"]-W_MAT[i,"Arr Time"]) * W_MAT[i,9]

}

#so now we can see the matrix with all updated data in it
W_MAT
```

```
##        Arr Time Dr6/7/8    Starts       Dur      Ends       Dr6       Dr7
##  [1,]  11.73359       6  11.73359  8.075296  19.80888  19.80888   0.00000
##  [2,]  20.39953       7  20.39953 15.286732  35.68626  19.80888  35.68626
##  [3,]  25.66015       8  25.66015  7.571070  33.23122  19.80888  35.68626
##  [4,]  31.29808       6  31.29808 15.449553  46.74764  46.74764  35.68626
##  [5,]  53.49361       8  53.49361 11.638936  65.13255  46.74764  35.68626
##  [6,]  67.16264       7  67.16264 14.496379  81.65902  46.74764  81.65902
##  [7,]  79.27071       6  79.27071 19.121291  98.39200  98.39200  81.65902
##  [8,]  84.98348       8  84.98348 12.319853  97.30333  98.39200  81.65902
##  [9,]  85.01312       7  85.01312 16.079053 101.09218  98.39200 101.09218
## [10,] 114.27862       8 114.27862  7.585585 121.86421  98.39200 101.09218
## [11,] 127.24158       6 127.24158  7.807285 135.04886 135.04886 101.09218
## [12,] 140.48698       7 140.48698 18.269386 158.75637 135.04886 158.75637
## [13,] 146.48946       8 146.48946 15.179458 161.66892 135.04886 158.75637
## [14,] 149.70361       6 149.70361 10.993399 160.69701 160.69701 158.75637
## [15,] 152.13597       7 158.75637 11.419941 170.17631 160.69701 170.17631
## [16,] 162.24503       6 162.24503 17.920499 180.16553 180.16553 170.17631
## [17,] 170.03512       8 170.03512 13.314208 183.34933 180.16553 170.17631
## [18,] 170.52518       7 170.52518  8.979134 179.50431 180.16553 179.50431
## [19,] 176.99011       7 179.50431 15.048659 194.55297 180.16553 194.55297
## [20,] 216.54503       6 216.54503 14.694457 231.23949 231.23949 194.55297
## [21,] 224.75679       8 224.75679 19.472750 244.22954 231.23949 194.55297
## [22,] 254.48005       7 254.48005 13.555716 268.03576 231.23949 268.03576
## [23,] 260.07441       6 260.07441 11.168827 271.24324 271.24324 268.03576
## [24,] 281.44525       8 281.44525 14.408563 295.85382 271.24324 268.03576
## [25,] 330.46214       7 330.46214 13.323931 343.78607 271.24324 343.78607
## [26,] 354.79602       6 354.79602  8.192454 362.98847 362.98847 343.78607
## [27,] 365.85396       8 365.85396  9.006508 374.86047 362.98847 343.78607
## [28,] 374.73370       7 374.73370 18.247882 392.98158 362.98847 392.98158
## [29,] 384.27567       6 384.27567  8.863893 393.13956 393.13956 392.98158
## [30,] 398.26934       8 398.26934 13.312021 411.58136 393.13956 392.98158
## [31,] 402.35521       7 402.35521 13.906389 416.26160 393.13956 416.26160
## [32,] 417.56405       6 417.56405 14.929635 432.49369 432.49369 416.26160
##             Dr8 W       WT
##  [1,]   0.00000 0 0.000000
##  [2,]   0.00000 0 0.000000
##  [3,]  33.23122 0 0.000000
##  [4,]  33.23122 0 0.000000
##  [5,]  65.13255 0 0.000000
##  [6,]  65.13255 0 0.000000
##  [7,]  65.13255 0 0.000000
##  [8,]  97.30333 0 0.000000
##  [9,]  97.30333 0 0.000000
## [10,] 121.86421 0 0.000000
## [11,] 121.86421 0 0.000000
## [12,] 121.86421 0 0.000000
## [13,] 161.66892 0 0.000000
## [14,] 161.66892 0 0.000000
## [15,] 161.66892 1 6.620393
## [16,] 161.66892 0 0.000000
## [17,] 183.34933 0 0.000000
## [18,] 183.34933 0 0.000000
## [19,] 183.34933 1 2.514206
## [20,] 183.34933 0 0.000000
## [21,] 244.22954 0 0.000000
## [22,] 244.22954 0 0.000000
## [23,] 244.22954 0 0.000000
## [24,] 295.85382 0 0.000000
## [25,] 295.85382 0 0.000000
```

```
## [26,] 295.85382 0 0.000000
## [27,] 374.86047 0 0.000000
## [28,] 374.86047 0 0.000000
## [29,] 374.86047 0 0.000000
## [30,] 411.58136 0 0.000000
## [31,] 411.58136 0 0.000000
## [32,] 411.58136 0 0.000000
```

```
# now, using this "W_MAT" matrix results, we will find How many patients had to wait for a doctor
#for this we will take a sum of values in "W" column of our matrix
A2 <- sum(W_MAT[,"W"])
A2
```

```
## [1] 2
```

```
#so , from the result of "A2", we can observe that in this case, with set.seed(243), only 2 patients ha
d to wait for a doctor
# This gives us answer to part (ii) of our question

######################################################################

# now, using this "W_MAT" matrix results, we will find what was patients average wait time
#for this we will take average of values in "WT" column of our matrix
A3 <- sum(W_MAT[,"WT"])/ A
A3
```

```
## [1] 0.2854562
```

```
#so , from the result of "A3", we can observe that in this case, with set.seed(243), patients average w
ait time was only 0.2854562 minutes
# This gives us answer to part (iii) of our question

######################################################################

# now, using this "W_MAT" matrix results, we will find when did the office close or when the last patie
nt is through with the doctor

#for this, we will use the maximum of doctors time-stamps values in the last row of the matrix
#because it is possible that patient admitted earlier than last patient may leave clinic last due to hi
s higher consultation time with doctor, so it is better to find at what time all doctors complete their
consultation and office can be closed
#also we will ensure that in cases where this value is less than 420 min, our answer should be 420 min
and not anything less than that, because clinic can't be closed before 420 min or 4 p.m. (even though a
ll existing patients completes their consultation with doctors before 4 p.m. and we are expecting no on
e to arrive before 4 pm as per out predicted arrival time)

A4 <- max(max(W_MAT[A,c("Dr6","Dr7","Dr8")]), 420)
A4
```

```
## [1] 432.4937
```

```
#this A4 gives us answer in minutes starting from 9 a.m.

#we can convert this A4 minutes to clock time in following way, while doing so we will be rounding of a fraction of minute to higher side by applying ceiling() function
A4TH <- cat(floor((A4/60)-3),":",ceiling(A4%%60),"p.m.")
```

```
## 4 : 13 p.m.
```

```
#so , from the result of "A4TH", we can observe that in this case, with set.seed(243), the office will be closed at 4:13 p.m.
# This gives us answer to part (iv) of our question

#######################################################################
```

so, as derived and specified in the above code, considering set.seed(243) arbitrarily, we have got the following results for the questions asked in Q.1.(a):

i. How many patients came to the office? Ans. 32

ii. How many had to wait for a doctor? Ans. 2

iii. What was their average wait? Ans. (0.2854562) minutes

iv. When did the office close? Ans. 4:13 p.m.

Q.1.(b)

In this part of question, we need to simulate the above process 1000 times and estimate the median for each of the summaries in Q.1.(a)

```r
# creating empty vectors and a list which we will fill with values we receive in each simulation
A1 <- c()
A2 <- c()
A3 <- c()
A4 <- c()
All_W_MAT <- list()

#as asked in the question, simulating the process done in Q.1.(a) above, 1000 times
for(j in 1:1000){

  #using random seed value of 1 to 1000 so that our results stays reproducible for this experiment of 1
000 simulations
  set.seed(j)

  ##########
  # NOTE: just repeating same code as used and explained above in Q.1.(a), so no need to explain all li
nes of code again
  ##########

  AT <- rexp(100, rate = 0.1)
  CAT <- cumsum(AT)
  A <- sum(CAT<420)

  A1[j] <- A
  DT <- runif(A, min = 5, max = 20)
  W_MAT <- matrix(NA, nrow = A, ncol = 10)
  colnames(W_MAT) <- c("Arr Time", "Dr6/7/8", "Starts", "Dur", "Ends", "Dr6", "Dr7", "Dr8", "W", "WT")

  W_MAT[1,1] <- CAT[1]
  W_MAT[1,2] <- 6
  W_MAT[1,3] <- W_MAT[1,1]
  W_MAT[1,4] <- DT[1]
  W_MAT[1,5] <- sum(W_MAT[1,3],W_MAT[1,4])
  W_MAT[1,6] <- W_MAT[1,5]
  W_MAT[1,7] <- 0
  W_MAT[1,8] <- 0
  W_MAT[1,9] <- 0
  W_MAT[1,10] <- 0

  ##########
  # NOTE: just repeating same code as used and explained above in Q.1.(a), so no need to explain all li
nes of code again
  ##########

  for(i in 2:A){

    W_MAT[i,1] <- CAT[i]
    W_MAT[i,2] <- sum(5,which.min(c(W_MAT[i-1,6],W_MAT[i-1,7],W_MAT[i-1,8])))

    if(W_MAT[i,1] < W_MAT[i-1,W_MAT[i,2]]){
      W_MAT[i,3] <- W_MAT[i-1,W_MAT[i,2]]
    }else{
      W_MAT[i,3] <- W_MAT[i,1]
    }

    #using time spent by patients with doctor
    W_MAT[i,4] <- DT[i]

    #exit time of that patient from clinic
```

```
    W_MAT[i,5] <- sum(W_MAT[i,3],W_MAT[i,4])

    #updating Dr6 occupancy timestamp if this patient is in consultation with doctor
    if(W_MAT[i,2] == 6){
      W_MAT[i,6] <- W_MAT[i,5]
    }else{
     W_MAT[i,6] <- W_MAT[i-1,6]
    }

    #updating Dr7 occupancy timestamp if this patient is in consultation with doctor
    if(W_MAT[i,2] == 7){
      W_MAT[i,7] <- W_MAT[i,5]
    }else{
      W_MAT[i,7] <- W_MAT[i-1,7]
    }

    #updating Dr8 occupancy timestamp if this patient is in consultation with doctor
    if(W_MAT[i,2] == 8){
      W_MAT[i,8] <- W_MAT[i,5]
    }else{
      W_MAT[i,8] <- W_MAT[i-1,8]
    }

    #determining if this patient had to wait or not
    W_MAT[i,9] <- (W_MAT[i,"Arr Time"] < W_MAT[i,"Starts"]) * 1

    #determining wait time if this patient had to wait
    W_MAT[i,10] <- (W_MAT[i,"Starts"]-W_MAT[i,"Arr Time"]) * W_MAT[i,9]

  }

  #in following manner, we will save each of this W_MAT matrix result of each simulation in a list, whi
ch if required can be checked later:->
  All_W_MAT[[j]] <- W_MAT

  ##########
  # NOTE: just repeating same code as used and explained above in Q.1.(a), so no need to explain all li
nes of code again
  ##########

  A2[j] <- sum(W_MAT[,"W"])


  A3[j] <- sum(W_MAT[,"WT"])/ A


  A4[j] <- max(max(W_MAT[A,c("Dr6","Dr7","Dr8")]),420)

}

# in case required, we can use following list to check the results matrix of a particular simulation, f
or eg. to see 291th simulation matrix we can use following code line:->
#All_W_MAT[[291]]

# in case required, we can use following vectors to check the summary of each questions asked in Q.1.
(a) for all of 1000 simulations
#A1
#A2
#A3
#A4
```

```
#now, as requested in the question Q.1.(b), we can estimate the median for each of the above summaries
vector A1 to A4 in the following manner:->
B1 <- median(A1)
B1
```

```
## [1] 42
```

```
B2 <- median(A2)
B2
```

```
## [1] 5
```

```
B3 <- median(A3)
B3
```

```
## [1] 0.4640054
```

```
B4 <- median(A4)
B4
```

```
## [1] 426.1528
```

```
#we can also convert above answer of B4 from minutes to clock time in following manner, please note tha
t we will be rounding of a fraction of minute to higher side by applying ceiling() function:->
B4TH <- cat(floor((B4/60)-3),":",ceiling(B4%%60),"p.m.")
```

```
## 4 : 7 p.m.
```

so, as seen from running above code, by considering seed values of 1 to 1000 arbitrarily, we are getting following answers by simulating the process 1000 times and estimating the median for each of the summary questions asked in Q.1.(a):->

i. How many patients came to the office? (median of 1000 simulations) Ans. 42

ii. How many had to wait for a doctor? (median of 1000 simulations) Ans. 5

iii. What was their average wait? (median of 1000 simulations) Ans. (0.4640054) minutes

iv. When did the office close? (median of 1000 simulations) Ans. 4:07 p.m. (as mentioned in the code, we have rounded of 4:06.1528 p.m. to 4:07 p.m.)

Section 2: Data Analysis Questions: ####################################

Q.1.:->

```r
# The first data (Q1Data1.csv) is Pew Research Center polls taken during the 2008 election campaign. We
will name this dataset as "dat1".

# The second data (Q1Data2.csv) is about 2008 election result in the US. We will name this dataset as
"dat2"

#need to use "foreign" library package for loading dataset from the file with ".dta" extension
library(foreign)

#Loading the given data file
dat1 <- read.dta("Q1Data1.dta")

#Loading the given data file
dat2 <- read.csv("Q1Data2.csv", header = T, stringsAsFactors = F)

#displaying head of loaded datasets dat1 and dat2
head(dat1)
```

```
##            survey   rid  date                      sample
## 1     june08voter  1720 62708                    landline
## 2      aug08relig   668 80208                    landline
## 3      aug08relig    50 73108                    landline
## 4      aug08relig 50533 80208                        cell
## 5     june08voter 30091 62108 18-29 oversample (landline)
## 6 july08poli-econ    60 72708                    landline
##                                            phoneuse zipcode   msa usr   form
## 1        dual-all or almost all calls on home phone   01007 44140   S form 2
## 2        dual-all or almost all calls on home phone   01010 44140   2 form 1
## 3 dual-some calls on cell phone, some on home phone   01013 44140   2 form 1
## 4 dual-some calls on cell phone, some on home phone   01013 44140     form 2
## 5        dual-all or almost all calls on cell phone   01020 44140   S form 1
## 6                              no cell phone sample   01020 44140   S form 2
##   thoughtpres     regist          regicert       party       partyln    sex
## 1        <NA> registered absolutely certain    democrat          <NA>   male
## 2        <NA> registered absolutely certain  republican          <NA> female
## 3        <NA> registered absolutely certain independent lean democrat   male
## 4        <NA> registered absolutely certain independent      other/dk   male
## 5        <NA> registered absolutely certain    democrat          <NA> female
## 6        <NA> registered absolutely certain    democrat          <NA> female
##   age            educ hisp race              marital parent
## 1  58    post-graduate   no    1             divorced   <NA>
## 2  35 college graduate   no    1              married   <NA>
## 3  59     some college   no    1              married   <NA>
## 4  32     some college  yes    9 living with a partner   <NA>
## 5  23 college graduate   no    3         never married   <NA>
## 6  55     some college   no    1              widowed   <NA>
##                  relig                relig2 born                attend
## 1      roman catholic        roman catholic    2    a few times a year
## 2 no religion/atheist nothing in particular   NA                 never
## 3      roman catholic        roman catholic    2           once a week
## 4          protestant             christian    2 once or twice a month
## 5               other              buddhist   NA more than once a week
## 6      roman catholic        roman catholic    2    a few times a year
##               income ownrent         ideo    employ labor   weight density
## 1   $50,000-$74,999     own      liberal      <NA>  <NA> 1.326923       2
## 2 $100,000-$149,999    <NA>     moderate      <NA>  <NA> 0.822000       3
## 3        dk/refused    <NA> conservative      <NA>  <NA> 0.493000       3
## 4   $30,000-$39,999    <NA>      liberal      <NA>  <NA> 0.492000       3
## 5   $75,000-$99,999    rent     moderate      <NA>  <NA> 2.000000       3
## 6   $30,000-$39,999     own     moderate full-time  <NA> 1.800000       3
##   attempt fcall     thought heat2a heat2b intsex intrace area      niicamp
## 1       9 80624 quite a lot    dem   <NA> female       2  413 very closely
## 2      NA    NA quite a lot     dk     dk   <NA>      NA   NA         <NA>
## 3      NA    NA quite a lot    dem   <NA>   <NA>      NA   NA         <NA>
## 4      NA    NA quite a lot    dem   <NA>   <NA>      NA   NA         <NA>
## 5       3 80621 quite a lot    dem   <NA>   male       1  413 very closely
## 6       7 80723      little    rep   <NA> female       1  413         <NA>
##            heat2c                                chancer
## 1        strongly decided not to vote for republican
## 2            <NA>                             dk/refused
## 3 only moderately   chance might vote for republican
## 4        strongly decided not to vote for republican
## 5        strongly   chance might vote for republican
## 6 only moderately                                   <NA>
##                            chanced planto1            planto2   cheata cheatb
## 1                             <NA>     yes absolutely certain democrat     NA
## 2                       dk/refused     yes               <NA>     <NA>     NA
```

```
## 3                            <NA>      yes                 <NA>      <NA>      NA
## 4                            <NA>      yes                 <NA>      <NA>      NA
## 5                            <NA>      yes absolutely certain democrat      NA
## 6 chance might vote for democrat      yes                 <NA>      <NA>      NA
##   precinct       oftvote scale10 pvote08 inthisp where heat4 pvote04 heat4a
## 1     <NA>          <NA>      NA    <NA>    <NA>  <NA>  <NA>      NA   <NA>
## 2      yes nearly always      10    <NA>    <NA>  <NA>  <NA>      NA   <NA>
## 3      yes        always      10    <NA>    <NA>  <NA>  <NA>      NA   <NA>
## 4      yes        always      10    <NA>    <NA>  <NA>  <NA>      NA   <NA>
## 5     <NA>          <NA>      NA    <NA>    <NA>  <NA>  <NA>      NA   <NA>
## 6      yes        always      10    <NA>    <NA>  <NA>  <NA>      NA   <NA>
##   heat4b heat4c fips         state cregion                   partysum relign
## 1   <NA>   <NA>   15 massachusetts    east     democrat/lean democrat      5
## 2   <NA>   <NA>   13 massachusetts    east republican/lean republican     10
## 3   <NA>   <NA>   13 massachusetts    east     democrat/lean democrat      5
## 4   <NA>   <NA>   13 massachusetts    east            refused to lean      4
## 5   <NA>   <NA>   13 massachusetts    east     democrat/lean democrat      9
## 6   <NA>   <NA>   13 massachusetts    east     democrat/lean democrat      5
##          heat2        cheat  age2           educ2            income2
## 1 dem/lean dem dem/lean dem 50-64 college graduate $50,000 to $74,999
## 2     other-dk         <NA> 30-49 college graduate            $75,000+
## 3 dem/lean dem         <NA> 50-64     some college               <NA>
## 4 dem/lean dem         <NA> 30-49     some college $30,000 to $49,999
## 5 dem/lean dem dem/lean dem 18-29 college graduate            $75,000+
## 6 rep/lean rep         <NA> 50-64     some college $30,000 to $49,999
##        party4
## 1    democrat
## 2  republican
## 3 independent
## 4 independent
## 5    democrat
## 6    democrat
```

```
head(dat2)
```

```
##        state vote_Obama vote_Obama_pct vote_McCain vote_McCain_pct
## 1    Alabama     811764           38.8     1264879            60.4
## 2     Alaska     105650           37.7      168844            60.2
## 3    Arizona     948648           45.0     1132560            53.8
## 4   Arkansas     418049           38.8      632672            58.8
## 5 California    7245731           60.9     4434146            37.3
## 6   Colorado    1216793           53.5     1020135            44.9
##   electoral_vote_dem electoral_vote_rep
## 1                 NA                  9
## 2                 NA                  3
## 3                 NA                 10
## 4                 NA                  6
## 5                 55                 NA
## 6                  9                 NA
```

Q.1.(a)

Here we need to update “dat1” dataframe, loaded from the first data file (Q1Data1.csv), as per the instructions provided in the question:

```r
#--- 1) as per instructions given in the question, subsetting the data so that we have all states but H
awaii, Alaska, and Washington D.C and have only four columns "state," "marital," "heat2," and "heat4."

D1 <- subset(dat1, state != "hawaii" & state != "alaska" & state != "washington dc", select = c(state,
marital,heat2,heat4), stringsAsFactors = FALSE)

#--- 2) Here, If no data is available in "heat2," we are replacing that "NA" with the corresponding val
ue in "heat4."

#before transferring data from heat4 to heat2, first we need to create factor levels that are not exist
ing in heat2 but existing heat4, so that it does not give any error while transferring data
D1$heat2 <- factor(D1$heat2, levels = c(levels(D1$heat2), "3rd party/lean 3rd party (barr)", "4th part
y/lean 4th party (nader)"))

#now, If no data is available in "heat2," we are replacing that "NA" with the corresponding value in "h
eat4."
D1$heat2[which(is.na(D1$heat2))] <- D1$heat4[which(is.na(D1$heat2))]

#Furthermore, as instructed in the question, If neither "heat2" nor "heat4" has data, we are erasing th
e corresponding row.
D1 <- D1[-which(is.na(D1$heat2) & is.na(D1$heat4)),]

#--- 3) Now, we need to Subset the data so that we only have "dem/lean dem" and "rep/lean rep" in the
"heat2" column

D1 <- subset(D1, heat2 == "dem/lean dem" | heat2 == "rep/lean rep")

#--- 4) Here, we need to change the label of all the variables but 'married' (married people) in the "m
arital" column to 'other' (which indicates non-married people).

#before that, we need to remove the row when the marital variable is missing
D1 <- D1[-which(is.na(D1$marital)),]

#also, we need to convert D1$marital column from factor to character to allow relabelling
D1$marital <- as.character(D1$marital)

#now, relabelling variable value as per above mentioned philosophy
for(i in 1:length(D1$marital)){
  if(D1$marital[i] != "married"){
    D1$marital[i] = "other"
  }
}

#now, we need to convert D1$marital column back to factor from character
D1$marital <- as.factor(D1$marital)

#also, for ease during further data analysis in Q.1(b), updating state and heat2 variables to latest fa
ctors and removing unwanted factors by converting them once into character and then back to factors
D1$state <- as.character(D1$state)
D1$state <- as.factor(D1$state)

D1$heat2 <- as.character(D1$heat2)
D1$heat2 <- as.factor(D1$heat2)

#now, by running this code, we have our dataframe ready as requested in the question, and we will see h
ead of it now
head(D1)
```

```
##            state marital        heat2 heat4
## 1 massachusetts   other dem/lean dem  <NA>
## 3 massachusetts married dem/lean dem  <NA>
## 4 massachusetts   other dem/lean dem  <NA>
## 5 massachusetts   other dem/lean dem  <NA>
## 6 massachusetts   other rep/lean rep  <NA>
## 7 massachusetts   other dem/lean dem  <NA>
```

Q.1.(b)

Firstly, For each state, we need to first calculate following: 1) the proportion of the democratic supporters, 2) the proportion of the married people, 3) the ratio of the married people among the democratic supporters to the total married people, 4) the ratio of non-married among the democratic to the total non-married people, 5) the difference of 3) and 4).

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#by using below summarise function and state as grouping variable, and applying necessary formulae on a
bove dataframe D1, we get our new dataframe D1b with new variables as asked in the question

D1b <- D1 %>%
  group_by(state) %>%
  summarise(prop_dem = sum(heat2 == 'dem/lean dem')/ n(),
            prop_marr = sum(marital == 'married')/ n(),
            prop_marr_demo = sum(heat2 == 'dem/lean dem' & marital == 'married')/ sum(marital == 'marri
ed'),
            prop_other_demo = sum(heat2 == 'dem/lean dem' & marital == 'other')/ sum(marital == 'othe
r'),
            diff = prop_marr_demo-prop_other_demo)

# so, by running this code, we can see our new dataframe
D1b
```

```
## # A tibble: 48 x 6
##    state       prop_dem prop_marr prop_marr_demo prop_other_demo    diff
##    <fct>          <dbl>     <dbl>          <dbl>           <dbl>   <dbl>
##  1 alabama        0.349     0.604          0.257           0.489 -0.231
##  2 arizona        0.468     0.629          0.364           0.644 -0.281
##  3 arkansas       0.370     0.656          0.302           0.5   -0.198
##  4 california     0.573     0.549          0.473           0.695 -0.222
##  5 colorado       0.553     0.586          0.502           0.624 -0.122
##  6 connecticut    0.585     0.613          0.545           0.649 -0.103
##  7 delaware       0.531     0.605          0.510           0.562 -0.0523
##  8 florida        0.490     0.561          0.401           0.603 -0.202
##  9 georgia        0.455     0.624          0.366           0.601 -0.234
## 10 idaho          0.467     0.641          0.407           0.576 -0.169
## # ... with 38 more rows
```

Now, we need to multiply all values received in above dataframe by 100 to convert them to percentage and then we need to show the first 5 observations of these new variables.

```
#creating new dataframe which gives summary results in percentage, as asked in the question
Db <- D1 %>%
  group_by(state) %>%
  summarise(per_dem = sum(heat2 == 'dem/lean dem')/ n() *100,
            per_marr = sum(marital == 'married')/ n() *100,
            per_marr_demo = sum(heat2 == 'dem/lean dem' & marital == 'married')/ sum(marital == 'marrie
d') *100,
            per_other_demo = sum(heat2 == 'dem/lean dem' & marital == 'other')/ sum(marital == 'other')
*100,
            raw_marr_gap = (per_marr_demo-per_other_demo))

# so, by running this code, we can see head of our new dataframe with these new variables, as asked in
the question.
head(Db)
```

```
## # A tibble: 6 x 6
##   state       per_dem per_marr per_marr_demo per_other_demo raw_marr_gap
##   <fct>         <dbl>    <dbl>         <dbl>          <dbl>        <dbl>
## 1 alabama        34.9     60.4          25.7           48.9        -23.1
## 2 arizona        46.8     62.9          36.4           64.4        -28.1
## 3 arkansas       37.0     65.6          30.2           50          -19.8
## 4 california     57.3     54.9          47.3           69.5        -22.2
## 5 colorado       55.3     58.6          50.2           62.4        -12.2
## 6 connecticut    58.5     61.3          54.5           64.9        -10.3
```

Q.1.(c)

Here, we need to consider the second data file (Q1Data2.csv) i.e. “dat2” dataframe as created in the beginning.

```
# we need to Subset the data so that:
# 1) we have all but three states, Hawaii, Alaska, and District of Columbia (Washington D.C), and
# 2) our subset data shall have only two columns “state,” and “vote_Obama_pct” (Obama’s actual vote sha
re).

#so, we are using subset function for creating the required dataframe D2
D2 <- subset(dat2, state != "Hawaii" & state != "Alaska" & state != "District of Columbia", select = c
(state, vote_Obama_pct), stringsAsFactors = FALSE)

# so now, by running this code, we can see the head of the data set "D2", as asked in the question
head(D2)
```

```
##         state vote_Obama_pct
## 1     Alabama           38.8
## 3     Arizona           45.0
## 4    Arkansas           38.8
## 5  California           60.9
## 6    Colorado           53.5
## 7 Connecticut           60.5
```

Q.1.(d)

Here we need to use a logistic regression predicting vote intention given state, using the indicator for being married as a predictor by setting up a proper link function. We need to check this for three different assumptions as to the state-level heterogeneity.

#— Assumption 1: No state-level heterogeneity. All states have the same intercept and slope.

This means, it will be a complete pooling for state variable. So, we do not need to add state as a variable while making our model. Note: we are not using glmnet() lasso with very high lambda here because it will create coefficients of marital variable (x) also zero along with state variable. however, as we know that marital variable heterogeneity we still need to consider in the model, we will use glm() and that too with only marital as variable in this case.

so here, we will consider “logit(p)= ln(p/1-p)= alpha+beta(x)” as our link function for binomial logistic regression, where, p can be understood as voting intention towards democratic candidate and x as the indicator for being married

```
# Firstly, we need to update our reference category level, which will help us in interpreting our logis
tic model results as per variable terminology stated above
# so, we are assigning "rep/lean rep" as reference level (or 0 level in other words) for voting intenti
on variable column heat2
D1 <- within(D1, heat2 <- relevel(heat2, ref = "rep/lean rep"))

# and similarly, we are assigning "other" as reference level (or 0 level in other words) for marital st
atus variable column marital
D1 <- within(D1, marital <- relevel(marital, ref = "other"))

#now, fitting the binomial logistic regression model to our data, as per Assumption-1
F1 <- glm(heat2 ~ I(marital), data = D1, family = "binomial")

#generating the summary of the coefficients for above logistic regression fit model F1
summary(F1)$coef
```

```
##                       Estimate Std. Error   z value      Pr(>|z|)
## (Intercept)          0.4857758 0.02165630  22.43116 1.954914e-111
## I(marital)married   -0.7346133 0.02785963 -26.36838 3.159344e-153
```

From the summary results we can note that the estimate of the coefficient related to marital variable is about -0.73. This can be interpreted in the following way. As we increase one unit in x1 variable i.e. from 0 (other) to 1 (married), the log-odds of their vote intention leaning towards democratic candidate compared to leaning towards republican candidate, i.e. logit(p) or ln(p/1-p), decreases by 0.73.

Also, alpha = 0.48 means at x = 0 (other than married people), p = e^0.48/(1 + e^0.48) = 0.62 (vote intention leaning towards democratic). Also, by plugging alpha and beta values at x = 1 (married) in our above link function, we get p=0.44 (vote intention leaning towards democratic). This suggests that As we increase one unit in x1 variable i.e. from 0 (other) to 1 (married), the probability of their vote intention leaning towards democratic candidate decreases from 0.62 to 0.44

also, we can understand, beta = -0.73 suggests that that x and p has a negative relationship and makes p a monotonically decreasing function of x. Also, we can say that the logistic curve will center at x value of -alpha/beta = 0.657, and slope at center will be beta/4= -0.18, which is also known as a divide by 4 rule, which gives degree of variation in p (democratic leaning) wrt unit change in x (marital status) at centre.

#— Assumption 2: Complete state-level heterogeneity. All states have completely independent intercepts and slopes. No outlying coefficient is penalized.

This means, it will be a no-pooling model and we will add state also as a categorical predictor variable while making our model. so, our ink function will look like this: logit(p)= ln(p/1-p)= alpha + betas (marital as factors) + gammas(states as factor)

```r
library(glmnet)
```

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

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

```
## Loaded glmnet 4.1-3
```

```r
#creating preditor and outcome variable datasets
mm_predictor <- model.matrix(heat2 ~ marital + state, data = D1)
mm_outcome = D1$heat2

#now, in this case of no pooling, we will consider lambda=0 in our ridge method based binomial logistic regression model fitting to our data, as per Assumption-2, therefore, our model can be written as below
F2 <- glmnet(x = mm_predictor, y = mm_outcome, alpha = 0, lambda = 0, family = "binomial")

#generating the summary of the coefficients for above binomial logistic regression model fitting F2
coef(F2)
```

```
## 50 x 1 sparse Matrix of class "dgCMatrix"
##                              s0
## (Intercept)         -0.16112501
## (Intercept)          .
## maritalmarried      -0.72658737
## statearizona         0.48513417
## statearkansas        0.09069449
## statecalifornia      0.86428820
## statecolorado        0.80554291
## stateconnecticut     0.96241909
## statedelaware        0.72836912
## stateflorida         0.52581082
## stategeorgia         0.42706464
## stateidaho           0.49329291
## stateillinois        0.89280522
## stateindiana         0.52995160
## stateiowa            0.69265534
## statekansas          0.72628027
## statekentucky        0.42469961
## statelouisiana       0.13389962
## statemaine           0.95540995
## statemaryland        1.34142221
## statemassachusetts   0.99139890
## statemichigan        0.90560491
## stateminnesota       0.58903681
## statemississippi     0.36946871
## statemissouri        0.42579413
## statemontana         0.60967250
## statenebraska        0.48355809
## statenevada          0.67046255
## statenew hampshire   0.43028900
## statenew jersey      0.83008913
## statenew mexico      0.96696212
## statenew york        1.11688861
## statenorth carolina  0.54428077
## statenorth dakota    0.48224302
## stateohio            0.57542946
## stateoklahoma        0.19944432
## stateoregon          0.86222473
## statepennsylvania    0.85339491
## staterhode island    0.39958828
## statesouth carolina  0.15859874
## statesouth dakota    0.63590207
## statetennessee       0.18349974
## statetexas           0.31299641
## stateutah            0.25038507
## statevermont         1.38254237
## statevirginia        0.41292979
## statewashington      0.94636842
## statewest virginia   0.45715286
## statewisconsin       0.86230699
## statewyoming        -0.05149354
```

so, by running above code we can see the estimation of the coefficients for above binomial logistic regression model fit F2 with no pooling

#— Assumption 3: State-level heterogeneity is unknown a priori. States have partially pooled intercepts and slopes. Outlying coefficients are penalized.

This means, it will be a partial-pooling model and we will use Ridge regression model fit to penalise the coefficients. Here, although our link function will still look the same as used in previous case, logit(p)= ln(p/1-p)= alpha + betas (marital as factors) + gammas(states as factor), here we will penalise coefficients based on best Cross validated lambda value through ridge method of partial pooling

```
#creating predictor and outcome variable datasets
mm_predictor <- model.matrix(heat2 ~ marital + state, data = D1)
mm_outcome = D1$heat2

#finding the best lambda from cross-validation(CV) with ridge method in binomial logistic regression mo
del
cv_model_ridge <- cv.glmnet(x = mm_predictor, y = mm_outcome, alpha = 0, family = "binomial")
best_lambda_ridge <- cv_model_ridge$lambda.min

#now, fitting the binomial logistic regression model to our data with ridge method of partial pooling a
nd with best CV lambda value, as per Assumption-3
F3 <- glmnet(x = mm_predictor, y = mm_outcome, alpha = 0, lambda = best_lambda_ridge, family = "binomia
l")

#generating the summary of the coefficients for the above generated model F3
coef(F3)
```

```
## 50 x 1 sparse Matrix of class "dgCMatrix"
##                              s0
## (Intercept)          0.230613545
## (Intercept)          .
## maritalmarried      -0.698981121
## statearizona         0.074133552
## statearkansas       -0.305681459
## statecalifornia      0.442142448
## statecolorado        0.383630846
## stateconnecticut     0.533594682
## statedelaware        0.308771668
## stateflorida         0.115177476
## stategeorgia         0.018215902
## stateidaho           0.081480005
## stateillinois        0.468014376
## stateindiana         0.117851374
## stateiowa            0.272408452
## statekansas          0.305599661
## statekentucky        0.015910633
## statelouisiana      -0.263602344
## statemaine           0.526627334
## statemaryland        0.895635136
## statemassachusetts   0.562718181
## statemichigan        0.480102104
## stateminnesota       0.174562892
## statemississippi    -0.037073755
## statemissouri        0.017930798
## statemontana         0.191722289
## statenebraska        0.071135979
## statenevada          0.254604446
## statenew hampshire   0.018727332
## statenew jersey      0.406472473
## statenew mexico      0.537236093
## statenew york        0.684254223
## statenorth carolina  0.131021818
## statenorth dakota    0.069729003
## stateohio            0.162158518
## stateoklahoma       -0.201750485
## stateoregon          0.436327423
## statepennsylvania    0.429132548
## staterhode island   -0.007601347
## statesouth carolina -0.240322091
## statesouth dakota    0.217120763
## statetennessee      -0.215551425
## statetexas          -0.091503170
## stateutah           -0.157578466
## statevermont         0.935641115
## statevirginia        0.005255065
## statewashington      0.519646127
## statewest virginia   0.046254529
## statewisconsin       0.437148079
## statewyoming        -0.437815397
```

so, by running above code we can see the estimation of the coefficients for above binomial logistic regression model fit F3 with partial pooling

Q.1.(e)

Now here, using the estimation result from the model with Assumption 3, we need to plot our inference for the predicted vote share by state, along with the actual vote intention, and also need to plot them vs. Obama’s actual vote share. And we will be annotating each dot with the corresponding state name.

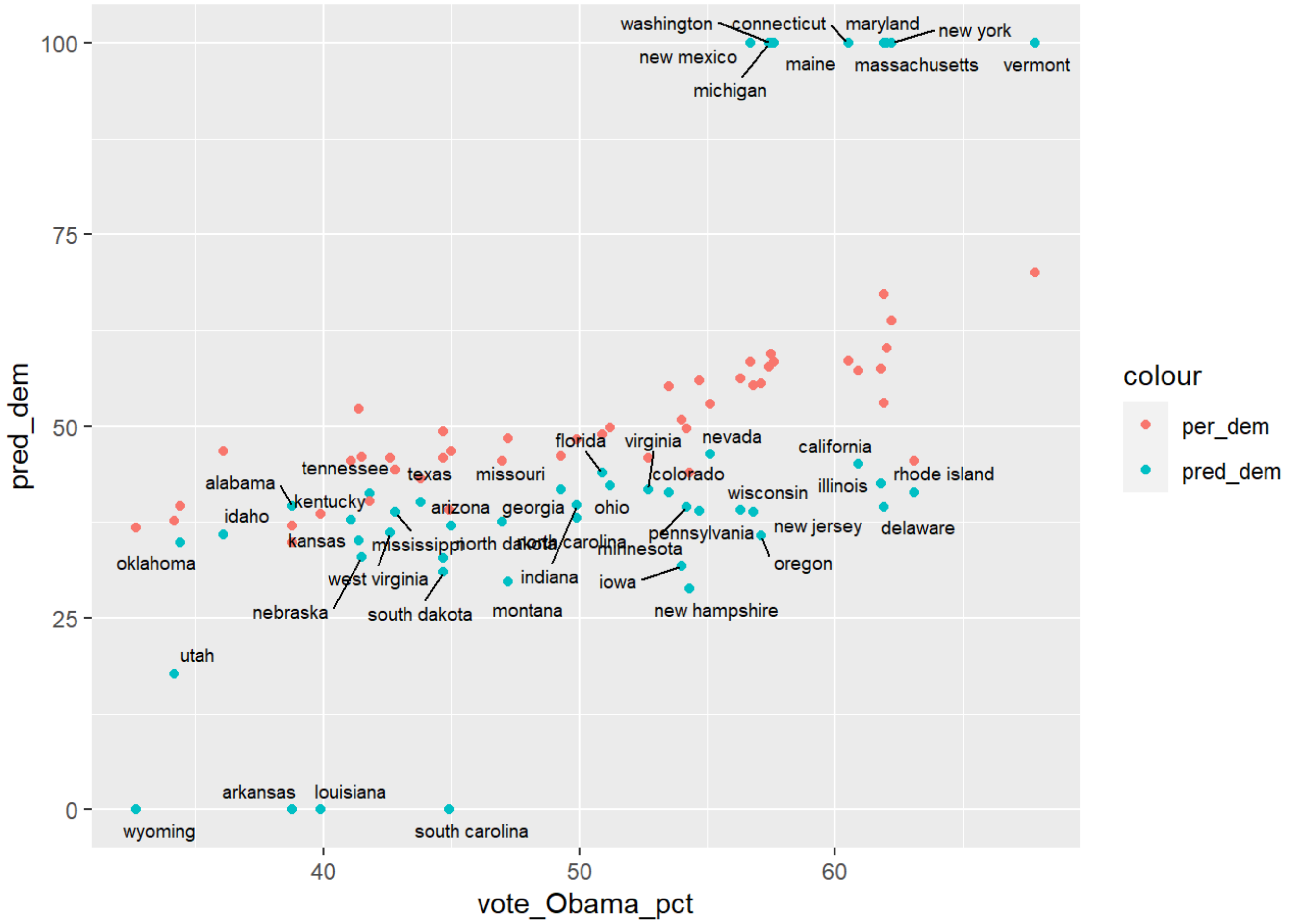
```
#Loading some Libraries necessary for plotting
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

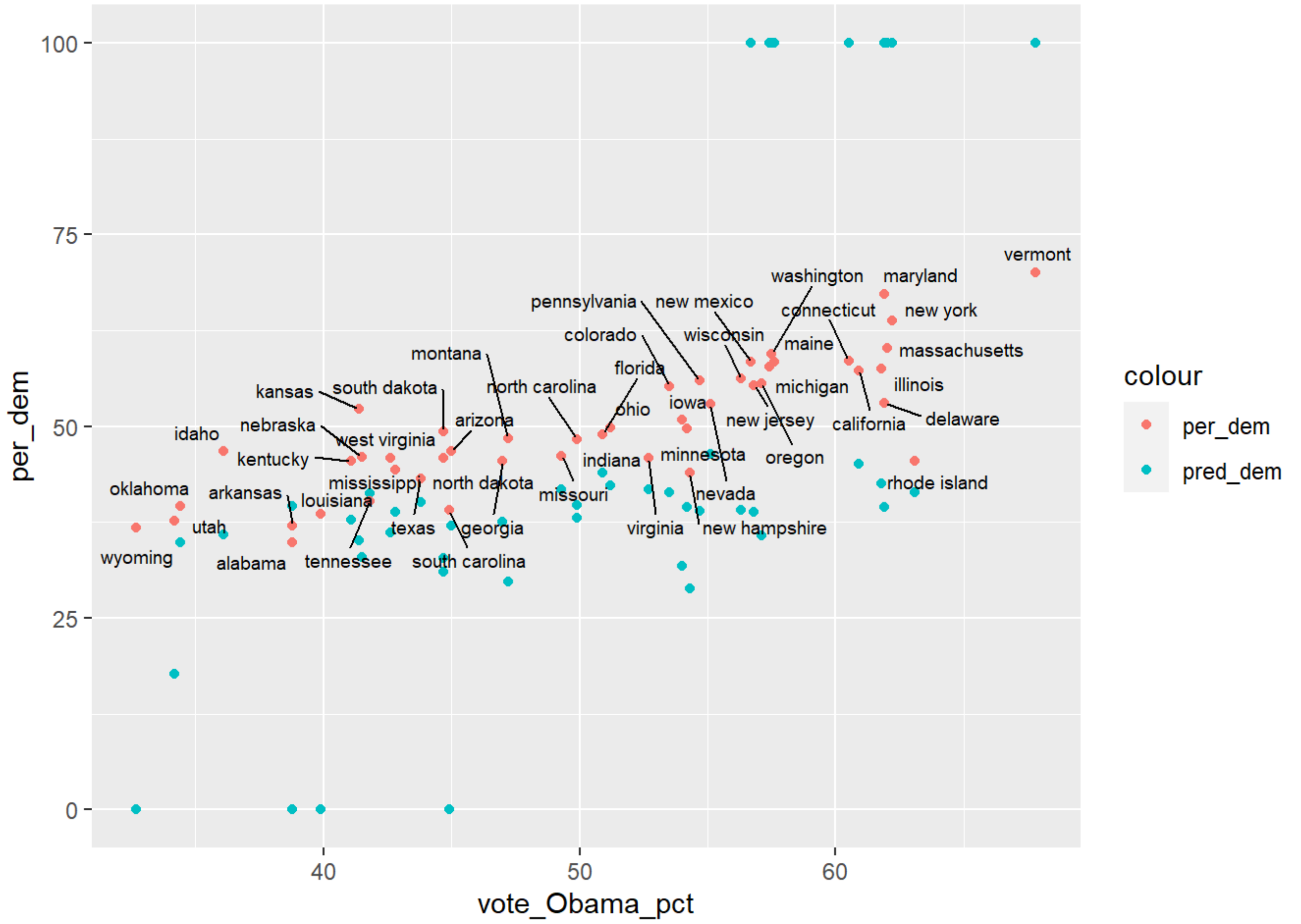
```
library(ggrepel)
```

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

```
#now, we can predict probability of y=1 (dem leaning) based on our existing dataset
dem_prob <- predict(F3, s = best_lambda_ridge, newx = mm_predictor, type = "response")

#assigning prediction to democratic or republic leaning based on criteria of probability value 0.5
dem_rep_lean <- ifelse(dem_prob>0.5, "dem/lean dem", "rep/lean rep")

#creating new dataframe covering this new predicted voting intention column, and our predictors column
of marital and state
DP <- data.frame(s1 = dem_rep_lean, marital = D1$marital, state = D1$state)

#now, creating new dataframe which gives us predicted democratic vote share by state
pred_Db <- DP %>%
  group_by(state) %>%
  summarise(pred_dem = sum(s1 == 'dem/lean dem')/ n() *100)

#now, creating dataframe covering state-wise predicted vote share and actual vote intention

EPD1 <- merge(x = pred_Db, y = Db[ , c("state", "per_dem")], by = "state", all.x=T)

# and before merging data from D2 in above dataframe, we need to update state variable from capital to
small letters
D2$state <- tolower(D2$state)

#now, we will add Obama’s actual vote share also in above dataframe EPD
EPD <- merge(x = EPD1, y = D2, by = "state", all.x=T)

#now, creating first plot of predicted vote share "pred_dem" vs Obama’s actual vote share "vote_Obama_p
ct". we have also added points for actual vote intention "per_dem" for better visualization. Also, we h
ave annotated states for predicted vote share "pred_dem"
EP1 <- ggplot(EPD, aes(x = vote_Obama_pct, y = pred_dem, per_dem)) +
  geom_point(aes(y = pred_dem, col = "pred_dem")) +
  geom_point(aes(y = per_dem, col = "per_dem")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
EP1
```




```
# similarly creating second plot of actual vote intention "per_dem" vs Obama's actual vote share "vote_Obama_pct". we have also added points for predicted vote share "pred_dem" for better visualization. Also, we have annotated states for actual vote intention "per_dem"
EP2 <- ggplot(EPD, aes(x = vote_Obama_pct, y = per_dem, pred_dem)) +
  geom_point(aes(y = pred_dem, col = "pred_dem")) +
  geom_point(aes(y = per_dem, col = "per_dem")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
EP2
```

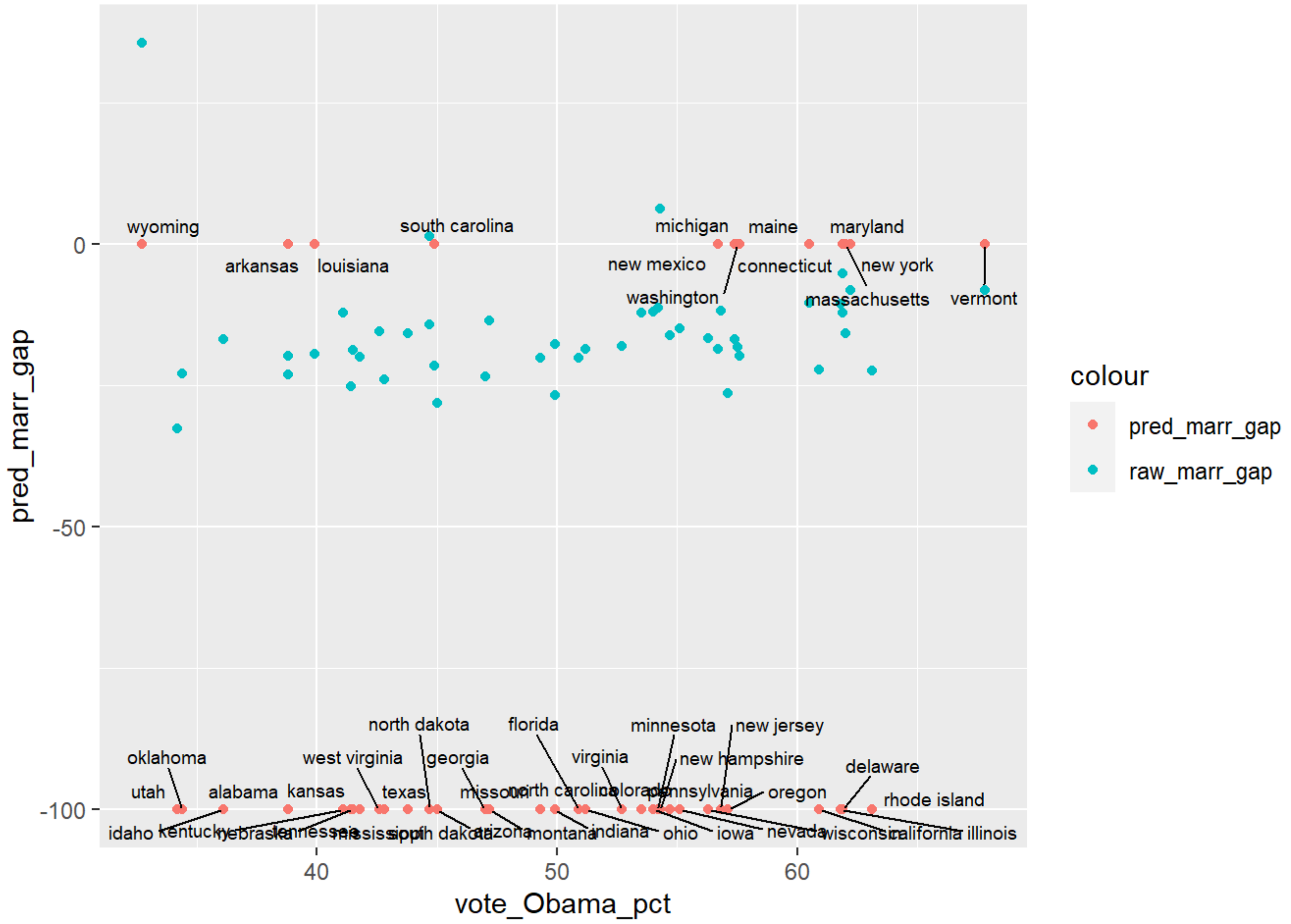


so, by running above code, we are getting the two necessary plots as asked in the question.

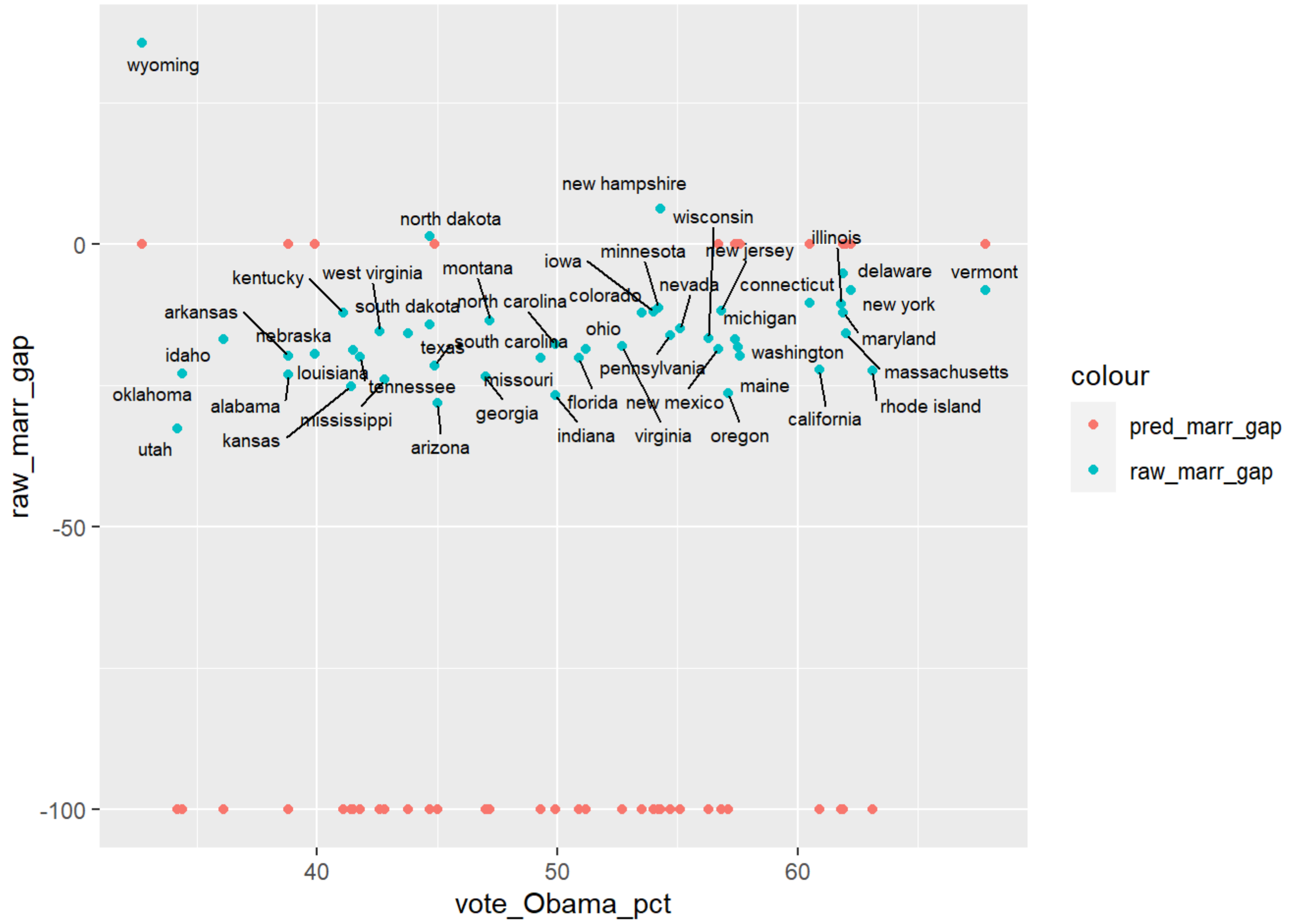
Q.1.(f)

As given in the question, The marriage gap is defined as the difference of Obama's vote share among married and non-married people ("other"). Based on this definition, we will first find out the marriage gap from our estimation result from the model with Assumption 3. And then we will be plotting our inference for the predicted marriage gap, along with the raw marriage gaps from the data, vs. Obama's actual vote share.

```
#now, creating new dataframe which gives us predicted democratic vote share by marital status, and usin
g that we are estimating the predicted marriage gap for each state. For this we are using the predicted
vote intention (s1) from DP dataset which we created above based on ridge best lambda method as per ass
umption3
pred_Db2 <- DP %>%
  group_by(state) %>%
  summarise(pred_marr_demo = sum(s1 == 'dem/lean dem' & marital == 'married')/ sum(marital == 'marrie
d') *100,
            pred_other_demo = sum(s1 == 'dem/lean dem' & marital == 'other')/ sum(marital == 'other') *
100,
            pred_marr_gap = (pred_marr_demo-pred_other_demo))

#now, creating dataframe covering state-wise predicted marriage gap and raw marriage gap

FPD1 <- merge(x = pred_Db2[ , c("state", "pred_marr_gap")], y = Db[ , c("state", "raw_marr_gap")], by =
"state", all.x=T)

# and before merging data from D2 in above dataframe, we need to update state variable from capital to
small case letters
D2$state <- tolower(D2$state)

#now, we will add Obama’s actual vote share "vote_Obama_pct" also in above dataframe FPD
FPD <- merge(x = FPD1, y = D2, by = "state", all.x=T)

#now, creating first plot of predicted marriage gap "pred_marr_gap" vs Obama’s actual vote share "vote_
Obama_pct". we have also added points for raw marriage gap "raw_marr_gap" for better visualization. Als
o, we have annotated states for predicted marriage gap "pred_marr_gap"
FP1 <- ggplot(FPD, aes(x = vote_Obama_pct, y = pred_marr_gap, raw_marr_gap)) +
  geom_point(aes(y = pred_marr_gap, col = "pred_marr_gap")) +
  geom_point(aes(y = raw_marr_gap, col = "raw_marr_gap")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
FP1
```




```
# similarly creating second plot of raw marriage gap "raw_marr_gap" vs Obama's actual vote share "vote_
Obama_pct". we have also added points for predicted marriage gap "pred_marr_gap" for better visualizati
on. Also, we have annotated states for raw marriage gap "raw_marr_gap"
FP2 <- ggplot(FPD, aes(x = vote_Obama_pct, y = raw_marr_gap, pred_marr_gap)) +
  geom_point(aes(y = raw_marr_gap, col = "raw_marr_gap")) +
  geom_point(aes(y = pred_marr_gap, col = "pred_marr_gap")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
FP2
```



so, by running above code, we are getting the two necessary plots as asked in the question.

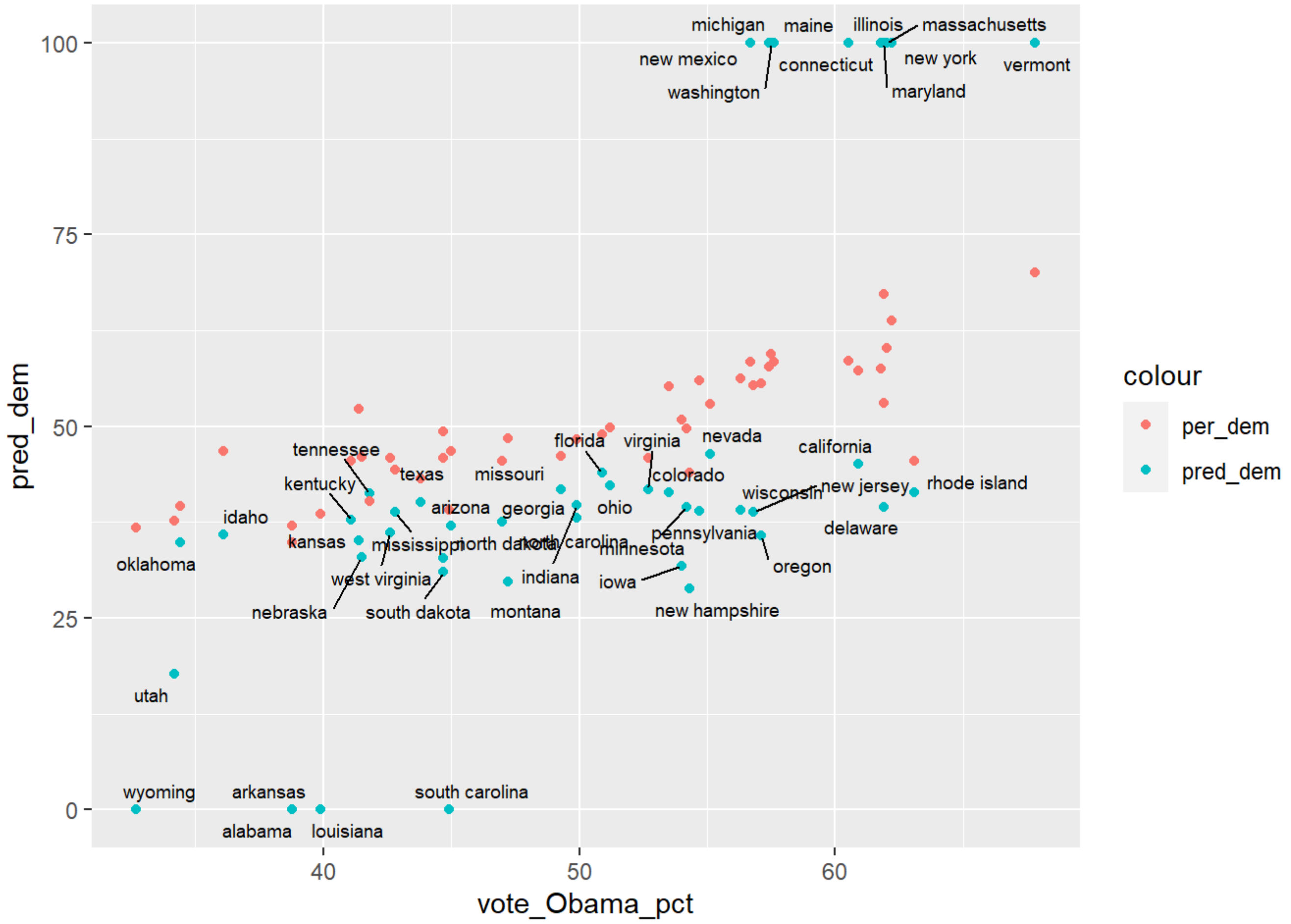
Q.1.(g)

Here we need to repeat Q.1.(e) & Q.1.(f) for the model with Assumption 2, and discuss our result.

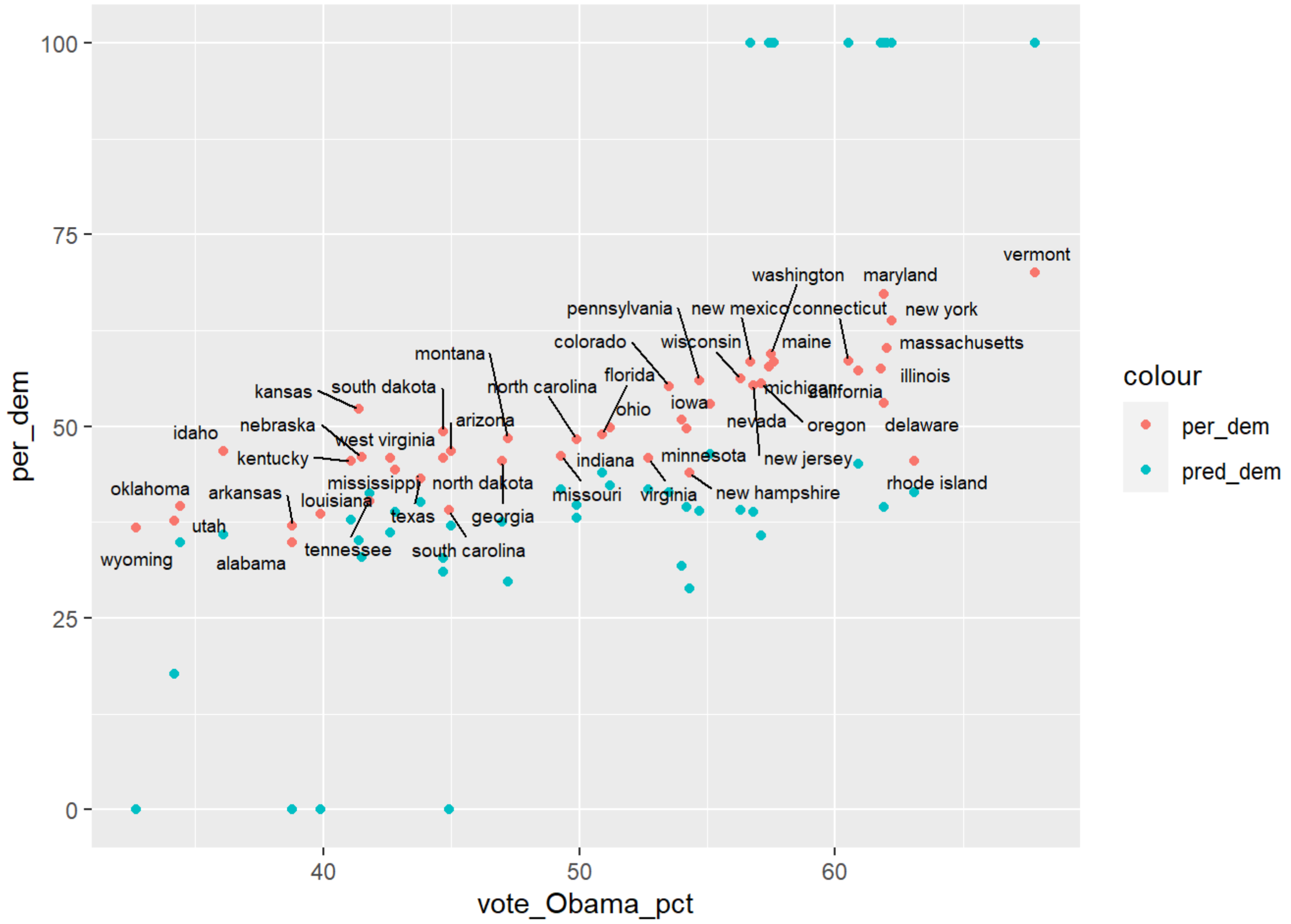
(i) repeat of Q.1.(e) for the model with Assumption 2

so here, using the estimation result from the model with Assumption 2, we need to plot our inference for the predicted vote share by state, along with the actual vote intention, and also need to plot them vs. Obama's actual vote share. And we will be annotating each dot with the corresponding state name.

```
#now, we can predict probability of y=1 (dem leaning) based on our existing dataset
dem_prob <- predict(F2, s = 0, newx = mm_predictor, type = "response")

#assigning prediction to democratic or republic leaning based on criteria of probability value 0.5
dem_rep_lean <- ifelse(dem_prob>0.5, "dem/lean dem", "rep/lean rep")

#creating new dataframe covering this new predicted voting intention column, and our predictors column
of marital and state
DP2 <- data.frame(s1 = dem_rep_lean, marital = D1$marital, state = D1$state)

#now, creating new dataframe which gives us predicted democratic vote share by state
pred_Db3 <- DP2 %>%
  group_by(state) %>%
  summarise(pred_dem = sum(s1 == 'dem/lean dem')/ n() *100)

#now, creating dataframe covering state-wise predicted vote share and actual vote intention

GEPD1 <- merge(x = pred_Db3, y = Db[ , c("state", "per_dem")], by = "state", all.x=T)

# and before merging data from D2 in above dataframe, we need to update state variable from capital to
small letters
D2$state <- tolower(D2$state)

#now, we will add Obama's actual vote share also in above dataframe GEPD
GEPD <- merge(x = GEPD1, y = D2, by = "state", all.x=T)

#now, creating first plot of predicted vote share "pred_dem" vs Obama's actual vote share "vote_Obama_p
ct". we have also added points for actual vote intention "per_dem" for better visualization. Also, we h
ave annotated states for predicted vote share "pred_dem"
GEP1 <- ggplot(GEPD, aes(x = vote_Obama_pct, y = pred_dem, per_dem)) +
  geom_point(aes(y = pred_dem, col = "pred_dem")) +
  geom_point(aes(y = per_dem, col = "per_dem")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
GEP1
```




```
# similarly creating second plot of actual vote intention "per_dem" vs Obama’s actual vote share "vote_
Obama_pct". we have also added points for predicted vote share "pred_dem" for better visualization. Als
o, we have annotated states for actual vote intention "per_dem"
GEP2 <- ggplot(GEPD, aes(x = vote_Obama_pct, y = per_dem, pred_dem)) +
  geom_point(aes(y = pred_dem, col = "pred_dem")) +
  geom_point(aes(y = per_dem, col = "per_dem")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
GEP2
```



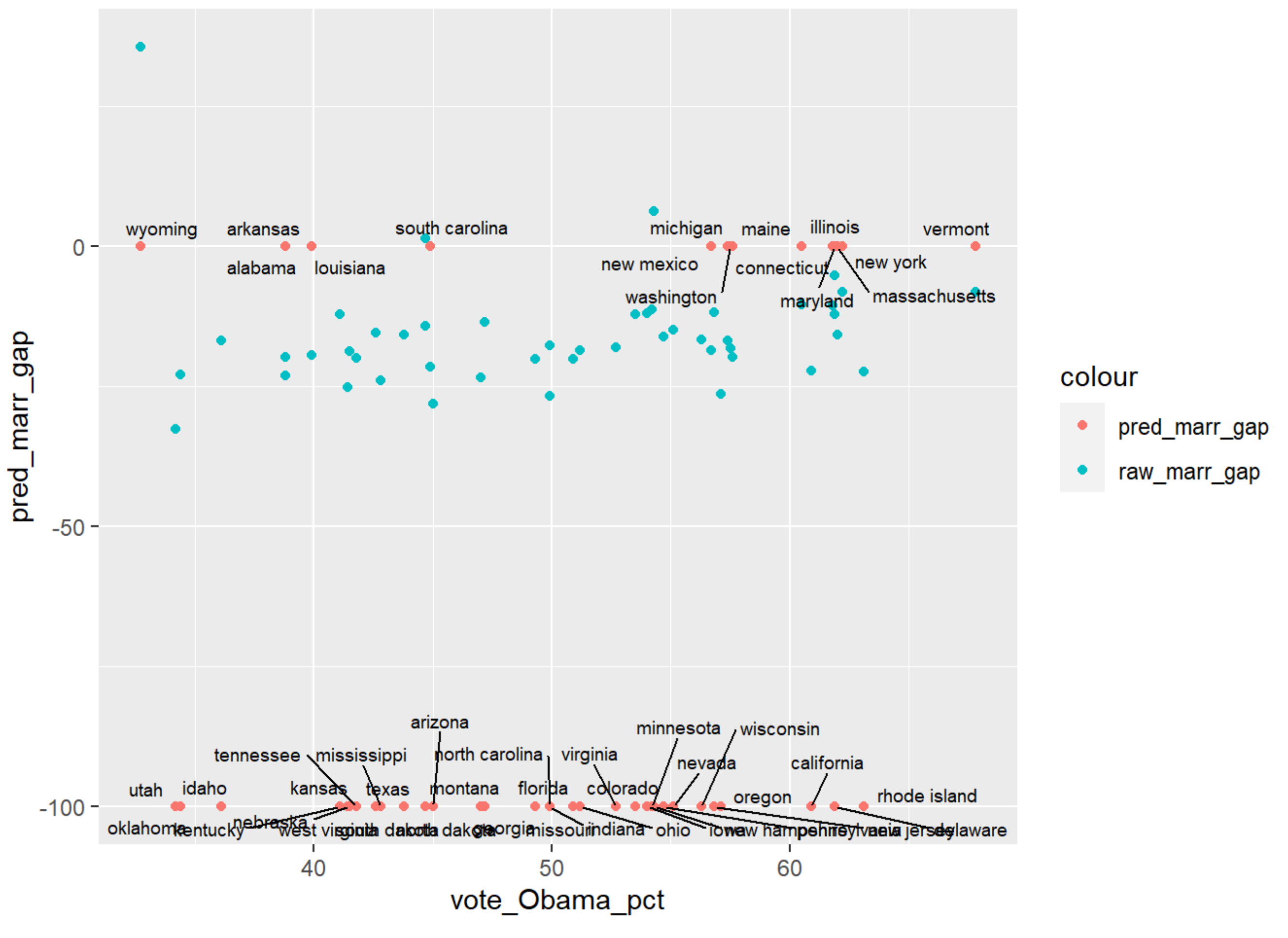
so, by running above code, we are getting the two necessary plots as asked in the question.

Also, from our these plot results, we can observe that in most of the states, our predicted democratic vote share is lesser but near to actual democratic vote intention. However, as actual vote share deviates from mean, the gap of predicted democratic vote share widens largely from the actual vote intention as our model predicts strongly towards or against the democratic at the extreme values of obama vote percents. Moreover, we can observe that compared to partial pooling based predicted democratic vote share, in this no pooling based model, we have higher gap between our predicted democratic vote share data and actual democratic intention data, suggesting higher prediction error. This suggests partial pooling model (assumption-3) has better predictive power compared to no pooling based model(assumption-2).

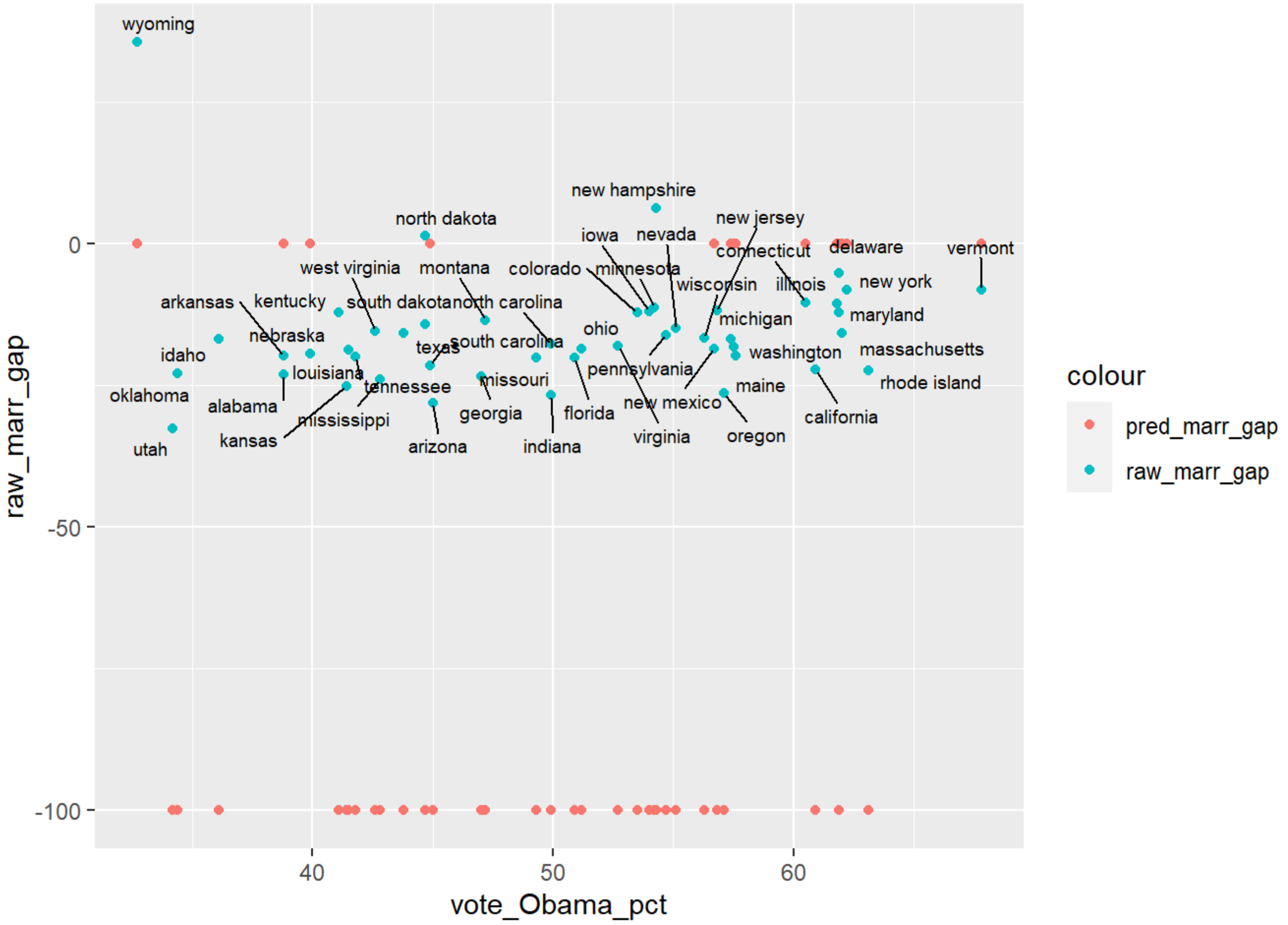
(ii) repeat of Q.1.(f) for the model with Assumption 2

As given in the question, The marriage gap is defined as the difference of Obama's vote share among married and non-married people ("other"). Based on this definition, we will first find out the marriage gap from our estimation result from the model with Assumption 2. And then we will be plotting our inference for the predicted marriage gap, along with the raw marriage gaps from the data, vs. Obama's actual vote share.

```
#now, creating new dataframe which gives us predicted democratic vote share by marital status, and using that we are estimating the predicted marriage gap for each state. For this we are using the predicted vote intention (s1) from DP2 dataset which we generated above based on our model with assumption2
pred_Db4 <- DP2 %>%
  group_by(state) %>%
  summarise(pred_marr_demo = sum(s1 == 'dem/lean dem' & marital == 'married')/ sum(marital == 'married') *100,
            pred_other_demo = sum(s1 == 'dem/lean dem' & marital == 'other')/ sum(marital == 'other') * 100,
            pred_marr_gap = (pred_marr_demo-pred_other_demo))

#now, creating dataframe covering state-wise predicted marriage gap and raw marriage gap

GFPD1 <- merge(x = pred_Db4[ , c("state", "pred_marr_gap")], y = Db[ , c("state", "raw_marr_gap")], by = "state", all.x=T)

# and before merging data from D2 in above dataframe, we need to update state variable from capital to small case letters
D2$state <- tolower(D2$state)

#now, we will add Obama’s actual vote share "vote_Obama_pct" also in above dataframe FPD
GFPD <- merge(x = GFPD1, y = D2, by = "state", all.x=T)

#now, creating first plot of predicted marriage gap "pred_marr_gap" vs Obama’s actual vote share "vote_Obama_pct". we have also added points for raw marriage gap "raw_marr_gap" for better visualization. Also, we have annotated states for predicted marriage gap "pred_marr_gap"
GFP1 <- ggplot(GFPD, aes(x = vote_Obama_pct, y = pred_marr_gap, raw_marr_gap)) +
  geom_point(aes(y = pred_marr_gap, col = "pred_marr_gap")) +
  geom_point(aes(y = raw_marr_gap, col = "raw_marr_gap")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
GFP1
```



```
# similarly creating second plot of raw marriage gap "raw_marr_gap" vs Obama's actual vote share "vote_Obama_pct". we have also added points for predicted marriage gap "pred_marr_gap" for better visualization. Also, we have annotated states for raw marriage gap "raw_marr_gap"
GFP2 <- ggplot(GFPD, aes(x = vote_Obama_pct, y = raw_marr_gap, pred_marr_gap)) +
  geom_point(aes(y = raw_marr_gap, col = "raw_marr_gap")) +
  geom_point(aes(y = pred_marr_gap, col = "pred_marr_gap")) +
  geom_text_repel(aes(label = state), size = 2.5, max.overlaps = 100)
GFP2
```



so, by running above code, we are getting the two necessary plots as asked in the question.

Also, from our these plot results, we can observe that in most of the states, our predicted marriage gap is negative and near to -100, meaning that in most of the states un-married (other) people are highly democratic leaning, whereas married people are mostly republican leaning. Moreover, we can observe that compared to partial pooling based predicted marriage gap, in this no pooling based model, our predicted marriage gap data has much lower values compared to raw marriage gap data, suggesting higher prediction error. This suggests partial pooling model (assumption-3) has better predictive power compared to no pooling based model(assumption-2).

END