[Tutorial] SCN2A mutations in neurodevelopmental disorders

1. Introduction

1.1 Background

SCN2A is a voltage-gated sodium channel gene that encodes the neuronal sodium channel NaV1.2 and plays a critical role in action potential initiation during early neurodevelopment. The latest study demonstrated that it is loss of function mutations that in SCN2A that lead to autism spectrum disorders (ASD), in contrast to gain of function, which leads to infantile seizures (Ben-Shalom 2018).

In this tutorial, we will handle genetic data for SCN2A mutations identified in latest genomic studies, and then explore the data format to describe genetic mutations using R basic functions. Our tutorial will utilize the summary data from Sanders et al. (2018).

1.2 Aims

What we will do with this dataset,

- Understand the dataset from a scientific journal
- Apply some functions you have learnt from the Chapter 2 and 3

2. Explore your data

2.1. Unboxing your dataset

Here we obtain the list of mutations in the **Supplementary Table 1** from Sanders et al. (2018).

Using the rio package, reading the excel file from the file link into your workspace. If you don't have the rio package in your system, please install as following:

```
# install.packages("rio")
```

Now you can read the file from the website. This will create the d object.

```
d <- rio::import("https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6015533/bin/NIHMS957592-supplement-1.xls
```

Let's explore the object you just loaded. How would you check the class of the object d?

```
class(d)
```

```
## [1] "data.frame"
```

It shows that the d object is data.frame.
Then, let's overview the data frame. We will use head function to print out first few lines.

```
head(d)
```

```
##      PatientID PatientSex PatientAgeAtAssessment Chr  Pos_hg19       Ref
## 1            .          M                     7y   2 154370122 166384278
## 2            .          F                     3y   2 163706754 166506754
## 3     Patient2          F                    18m   2 163875903 166478766
## 4 Case1,280269          F                     4y   2 165798270 166304847
## 5            .          M                     3y   2 166019786 166249879
## 6            .          F                    25y   2 166060054 166153823
##                 Alt Type         Effect c.DNA p.Protein  Inheritence
## 1  12Mb Duplication  CNV DuplicationCNV     .         . DeNovoMosaic
## 2    2.8Mb Deletion  CNV    DeletionCNV     .         .       DeNovo
## 3 2.6Mb Duplication  CNV DuplicationCNV     .         .    Inherited
## 4 507kb Duplication  CNV DuplicationCNV     .         .    Inherited
## 5    230kb Deletion  CNV    DeletionCNV     .         .      Unknown
## 6     94kb Deletion  CNV    DeletionCNV     .         .       DeNovo
##                 Source SourcePMID Ben-Shalom2017 Wolff2017 AnyRecurrence
## 1    Vecchi et al 2011   21893419              Y         N             1
## 2      Chen et al 2010   20346423              Y         N             1
## 3 Yoshitomi et al 2015   25843248              Y         N             1
## 4 Thuresson et al 2016   27153334              Y         N             1
## 5     Celle et al 2013   24080482              Y         N             1
## 6   Bartnik et al 2011   20807223              Y         N             1
##   UniqueSample/Family TrueRecurrence Seizures SeizureOnsetDays SeizureType
## 1                   Y              1        Y               90     Unknown
## 2                   Y              1        N                .        None
## 3                   Y              1        Y                3     Unknown
## 4                   Y              1        Y                3     Unknown
## 5                   Y              1        N                .        None
## 6                   Y              1        Y              365     Unknown
##       ASD DD/ID DD/ID severity                                    OtherFeatures
## 1       N     Y           Mild                                       clumsiness
## 2       Y     Y         Severe                 dysmorphia, immature myelination
## 3 Unknown     Y         Severe                                  cerebral atrophy
## 4       N     N              .                                                .
## 5       Y     Y         Severe                                     microcephaly
## 6       N     Y       Moderate hypotonia, bipolar disorder, behavioral problems
##    Classification
## 1 IEE_Mild/Ataxia
## 2          ASD/DD
## 3             IEE
## 4             BIS
## 5          ASD/DD
## 6          ASD/DD
```

When you execute code in a notebook chunk, an output will be visible immediately beneath the input. From this, you can see several rows and columns in the data frame.

Let's look at the first column PatientID and check which class it is.

```r
class(d$PatientID)
```

```
## [1] "character"
```

```r
# Character
```

Cool. Now you can see the TrueRecurrence column. What is the class of the column TrueRecurrence?

```r
class(d$TrueRecurrence)
```

```
## [1] "numeric"
```

```r
# Numeric
```

To check the class of columns, you don't need to type an individual column. We can overview the summary of the dataset using summary function. Which column has the character class?

```r
summary(d)
```

```
##   PatientID          PatientSex        PatientAgeAtAssessment      Chr
##  Length:293         Length:293         Length:293             Min.   :2
##  Class :character   Class :character   Class :character       1st Qu.:2
##  Mode  :character   Mode  :character   Mode  :character       Median :2
##                                                               Mean   :2
##                                                               3rd Qu.:2
##                                                               Max.   :2
##    Pos_hg19             Ref                 Alt                 Type
##  Length:293         Length:293         Length:293         Length:293
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##     Effect             c.DNA            p.Protein         Inheritence
##  Length:293         Length:293         Length:293         Length:293
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##     Source           SourcePMID        Ben-Shalom2017      Wolff2017
##  Length:293         Length:293         Length:293         Length:293
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  AnyRecurrence    UniqueSample/Family TrueRecurrence    Seizures
##  Min.   : 1.000   Length:293          Min.   :1.000   Length:293
##  1st Qu.: 1.000   Class :character    1st Qu.:1.000   Class :character
##  Median : 1.000   Mode  :character    Median :1.000   Mode  :character
##  Mean   : 2.119                       Mean   :1.768                          
##  3rd Qu.: 2.000                       3rd Qu.:1.000                          
##  Max.   :10.000                       Max.   :7.000                          
##  SeizureOnsetDays   SeizureType            ASD               DD/ID          
##  Length:293         Length:293         Length:293         Length:293        
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                             
##                                                                             
##                                                                             
##  DD/ID severity     OtherFeatures      Classification    
##  Length:293         Length:293         Length:293        
##  Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character  
##                                                          
##                                                          
##                                                          
```

```
# Every column except 'Chr','AnyRecurrene', 'TrueRecurrence'.
```

2.2 Difference between data frame and matrix

Here we will convert the data frame into a matrix, and compare which part will be different in this. To convert a data frame into a matrix, you can use the command called as.matrix.

```
m=as.matrix(d)
```

Let's overview the matrix object. Can you tell difference with data frame?

```
head(m[, "TrueRecurrence"])
```

```
##   1   2   3   4   5   6 
## "1" "1" "1" "1" "1" "1"
```

```
# I can see the difference between matrix and data frame!
# Matrices can only contain a single class of data, while
# data frames can consist of many different classes of
# data.
```

2.3. Subset and Sort

Some patients who have the SCN2A mutation (hereafter called "SCN2A patient") often have seizures. So we want to know when the seizure occurs in development.

Let's check the class first.

```
class(d$SeizureOnsetDays)
```

```
## [1] "character"
```

Why this column contains character? Let's head the first few lines.

```
head(d$SeizureOnsetDays)
```

```
## [1] "90"  "."   "3"   "3"   "."   "365"
```

t seems that some rows contain samples who do not have seizure or unknown information. It's represented by ".", and also recorded in another column called Seizures.

```
head(d$Seizures)
```

```
## [1] "Y" "N" "Y" "Y" "N" "Y"
```

So we want to subset rows where the seizure phenotype is available.

```
d1<- d[d$Seizures == "Y",]
```

Let's see how many samples have seizure phenotypes? Then, you can ask when is the earliest days for the representation of seizure phenotype? How can we check this? The fisrt, as seen previously the SeizureOnsetDays column is character so we cannot apply functions for numeric.

```
head(d1$SeizureOnsetDays)
```

```
## [1] "90"   "3"    "3"    "365"  "30"   "1825"
```

So we have to convert this into numeric first.

```
d1$SeizureOnsetDays2 <- as.numeric(d1$SeizureOnsetDays)
```

```
## Warning: NAs introduced by coercion
```

Hmm. There's an warning for NA introduction. This is because some rows do not have character that we can properly convert from character to numeric. So possible solutions are either you can bear with this in your downstream analyses or 2) convert character into an appropriate form of numeric conversion.

Then, the question is how can we find the rows with NA? We will ask whether the rows contains NA or not using is.na function. This will return boolean as to NA presence.

```
is.na(d1$SeizureOnsetDays2)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [193] FALSE FALSE FALSE FALSE FALSE
```

See we can find some rows with NA. One of them is the 7th row. Let's see how it looks like.

```
d1$SeizureOnsetDays[13]
```

```
## [1] "<365"
```

```
# In my data 'd1', 13th row have NA not 7th.
```

Here you have < (angle bracket) in the character so it won't properly converted to numeric information. Did you find more of these cases?

```
d1[is.na(d1$SeizureOnsetDays2),]$SeizureOnsetDays
```

```
##  [1] "<365" "<365" "<365" "<28"  "<30"  "<365" "<365" "<365" "<365" "<365"
```

Our NAs all contains <, which prevent converting a character into a numeric. We would fix for downstream analyses. For example, we can convert <365 into 365. One function we can try is gsub. This replace your string into a format that you may not get NA. For example,

```
# gsub('pattern in your character', 'new character you want
# to replace', vectors for your character)
d1$SeizureOnsetDays3 <- gsub("<", "", d1$SeizureOnsetDays)
head(d1$SeizureOnsetDays3)
```

```
## [1] "90"   "3"    "3"    "365"  "30"   "1825"
```

Let's convert them into numerics.

```
d1$SeizureOnsetDays3<-as.numeric(d1$SeizureOnsetDays3)
```

Did you get warning for this? Now we can ask our initial question. When is the earliest day for having seizure?

```
min(d1$SeizureOnsetDays3)
```

```
## [1] 0
```

3. Exercise

The dataset contains more details for genetic mutations in SCN2A patients. From this information, what can we analyze further?

Here I list up few questions you can examine further.

- Finding the position of the genetic mutations within SCN2A. Which information you would use? If you are not familiar with positional information on genetic variants (or mutations), please find the **Figure 1** or the slides for Mutation (BSMS205 Session 3-1).
- Counting the recurrent mutations at the same protein position (in other words, the same mutations seen across different patients), and examine whether the patients have similar phenotype.
- Finding the position where different consequences mutations occur. Please note that "consequences" are loss-of-function (Nonsense, Frameshift) or missense.
- Sketch a plot to visualize your analysis.

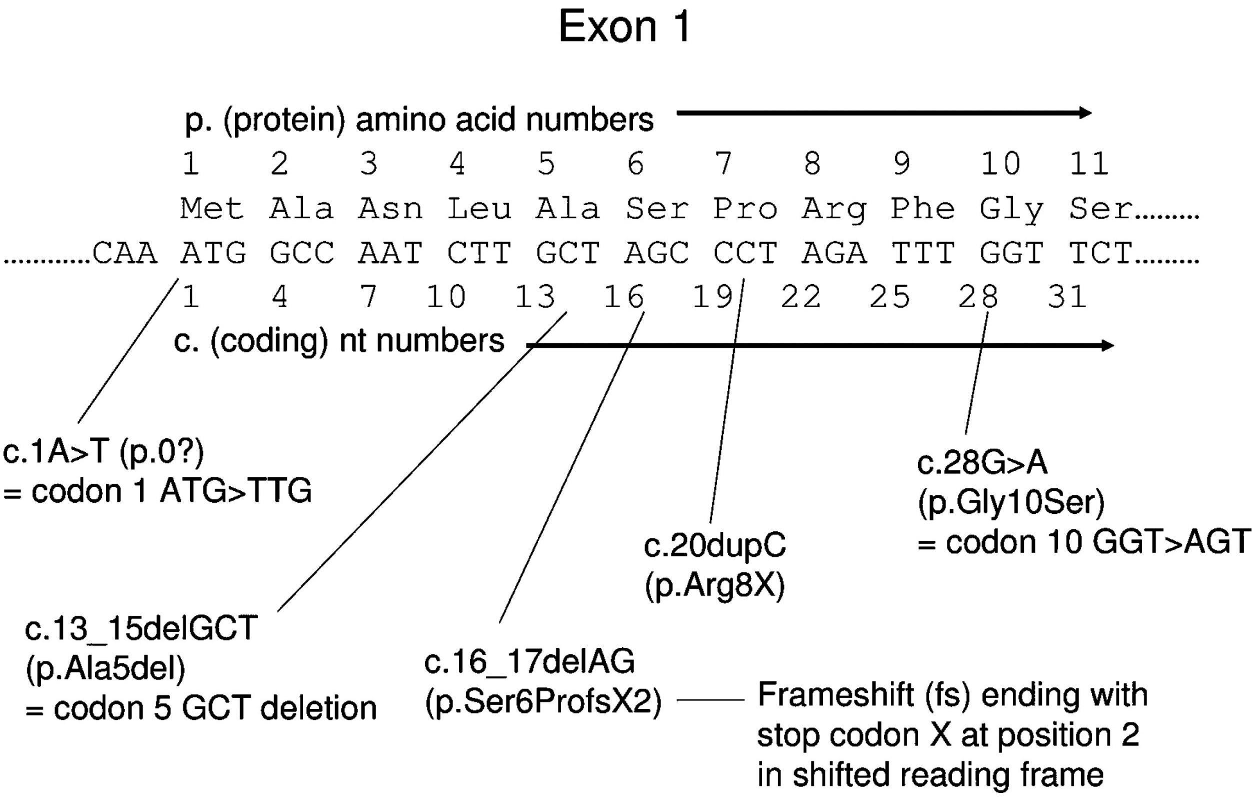

Figure 1: **Figure 1. Standard mutation nomenclature** (Ogino et al. 2007). According to this guideline, genetic mutations can be represented for coding DNA reference sequences (c. prefix) and protein-level amino acid sequences (p. prefix).

3.1 For-loops and Vectorization

Here we examine more details of genetic mutations as to their functional consequence and position of SCN2A mutations. In the dataset includes, there are two columns called c.DNA and p.Protein, containing the cDNA or protein position for the genetic mutations.

During these exercises, we will look at the concept of for-loop and vectorization, which you learn from the Chapter 3.4. Let's look at the column **p.Protein**. It contains protein positions from each patient. What would you check at the first place?

- Task 1. First, I want to overview this column using 'head()'
- Task 2. Overview this column using 'tail()'
- Task 3. Which class is it?
- Task 4. How many observations(samples)

- Task 5. How many samples with p.Protein information?
- . . . .
- Task N

Let's write down your code to explore this column.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
head(d$p.Protein)
```

```
## [1] "." "." "." "." "." "."
```

```
tail(d$p.Protein)
```

```
## [1] "p.R1902C" "p.R1902C" "p.Q1904E" "p.R1918H" "p.K1933M" "p.S1974L"
```

```
class(d$p.Protein)
```

```
## [1] "character"
```

```
nrow(d)
```

```
## [1] 293
```

```
d[d$p.Protein!=".",]%>%nrow()
```

```
## [1] 285
```

If you need more information on SCN2A, please visit the Uniprot description for SCN2A. The Uniprot database contains description for protein domains.
Then, I would remove the characters from the string so we can have only numerics for positions. Here I use gsub function to extract numbers from string. Let's remove non-numeric characters from the string.

```
gsub('[^0-9]', '', 'p.R102X')
```

```
## [1] "102"
```

In the dataset, we have many rows for protein positions. One way we might try is to set up for-loop to process each row.

```r
for(i in 1:293){
  a <- gsub('[^0-9]','',d[i,'p.Protein'])
}
```

Do you think this is an effective approach? As we have done in your assignment, for-loop is not a good choice to process vectors because R can do vectorization for this process with a shorter and clearer code. So this mean you can apply gsub on vector and return your output to another column (could be new assign).

```r
d$p.Protein1 <- gsub('[^0-9]','',d$p.Protein)
# I made new column named 'p.Protein1' in which outputs are saved.
```

3.2. Counting the recurrent mutations

Recurrent mutations are the ones that the same genetic mutations occur in multiple individuals. Recurrent mutations can be common 1) when the mutation does not affect on natural selection, 2) when the mutation is beneficial, 3) in the hotspot for a disease or strongly associated with trait. However, given we are dealing with the genetic mutations from rare disorders, the mutations in the dataset are supposed to be uniquely present in general population. Otherwise, the recurrent mutations can indicate strong association with the phenotype.

To assess the recurrent mutations, the first thing we can try is to examine whether the same mutations occur in multiple individuals. Since the dataset contains individual patients for each row, we can simply check the frequency using:

```r
c <- as.data.frame(table(d$p.Protein))
```

or we can check the number of unique variants in the dataset by:

```r
nrow(c)
```

```
## [1] 203
```

```r
# There are 203 unique variants in the dataset.
```

How many unique variants you can find? and which variants are occurred in multiple times?

```r
filter(c,Freq>1  )%>%
  nrow()
```

```
## [1] 40
```

```r
# Among 203 unique variants, 40 are occurred in multiple times.
# But I just find out that one of them is undefined variants.
# In other words, 39 variants are occurred in multiple individuals.
```

Then, you can use other columns to check frequency for different groups. Which columns you would use for more grouping?

If you find that, please check the recurrent mutations for each group.

```
# I can use 'c.DNA' column to check frequency for different
# groups.
d$c.DNA %>%
    table() %>%
    as.data.frame %>%
    filter(Freq > 1)
```

```
##             . Freq
## 1         ???    2
## 2           .    8
## 3    c.106A>G    3
## 4   c.1136G>A    2
## 5   c.1267G>C    2
## 6   c.2558G>A    9
## 7   c.2645G>A    2
## 8   c.2695G>A    2
## 9   c.2715G>C    2
## 10  c.2783T>G    2
## 11  c.2809C>T    2
## 12  c.2932T>C    2
## 13  c.2995G>A    6
## 14   c.304C>T    2
## 15 c.3057AA>A    2
## 16  c.3631G>A    3
## 17  c.3844G>T    2
## 18 c.386+2T>C    2
## 19  c.3956G>A    6
## 20  c.4007C>A    2
## 21  c.4025T>C    4
## 22   c.408G>T    2
## 23  c.4303C>T    3
## 24  c.4436A>C    2
## 25  c.4565G>C    2
## 26  c.4591C>A    2
## 27 c.476+1G>A    2
## 28  c.4777G>A    2
## 29  c.4886G>A    2
## 30  c.5318C>T    3
## 31   c.562C>T    2
## 32  c.5644C>G    2
## 33  c.5645G>A    6
## 34  c.5704C>T    2
## 35   c.632G>A    2
## 36   c.638T>A    2
## 37   c.710T>A    2
## 38   c.781G>A    2
## 39   c.788C>T    8
## 40   c.982T>G    2
```

```
# I got 40 recurrent mutations group using nrow() from
# here. But two groups are undefined DNA mutations group.
# Without them, 38 recurrent mutations are identified.
```

3.3 What is the proportion of diagnosis for SCN2A patient?

SCN2A mutation can have multiple different consequences for disease phenotypes. It can cause ASD but also other neurodevelopmental conditions. In total cases, how many phenotypes occur in SCN2A patients. Then, calculate the proportion of the phenotypes among total cases.

```
# I used column 'Classification' in which each patients are
# classified into different phenotypes.
d$Classification %>%
    table() %>%
    as.data.frame() %>%
    mutate(proportion = Freq/293)
```

```
##                  . Freq proportion
## 1           ASD/DD   92 0.31399317
## 2              BIS   36 0.12286689
## 3              IEE  111 0.37883959
## 4 IEE_Mild/Ataxia    7 0.02389078
## 5            Other    3 0.01023891
## 6    Schizophrenia    5 0.01706485
## 7          Unclear   39 0.13310580
```

Then, you might be intrigued to whether females and males have different occurrence in each disorder. Let's check it.

```
# Calculate total number of Female
Female_total <- d %>%
    filter(PatientSex == "F") %>%
    nrow()
Female_total  #114
```

```
## [1] 114
```

```
# Calculate total number of Male
Male_total <- d %>%
    filter(PatientSex == "M") %>%
    nrow()
Male_total  #119
```

```
## [1] 119
```

```
# Male and Female have difference occurences in each
# disorder. Check this below.
d %>%
    filter(PatientSex %in% c("F", "M")) %>%
    group_by(PatientSex, Classification) %>%
    count() %>%
    mutate(proportion = ifelse(PatientSex == "F", n/Female_total,
        n/Male_total))
```

```
## # A tibble: 13 x 4
## # Groups:   PatientSex, Classification [13]
##    PatientSex Classification      n proportion
##    <chr>      <chr>           <int>      <dbl>
##  1 F          ASD/DD             26    0.228
##  2 F          BIS                11    0.0965
##  3 F          IEE                57    0.5
##  4 F          IEE_Mild/Ataxia     3    0.0263
##  5 F          Other               2    0.0175
##  6 F          Schizophrenia       1    0.00877
##  7 F          Unclear            14    0.123
##  8 M          ASD/DD             44    0.370
##  9 M          BIS                12    0.101
## 10 M          IEE                43    0.361
## 11 M          IEE_Mild/Ataxia     4    0.0336
## 12 M          Other               1    0.00840
## 13 M          Unclear            15    0.126
```

Another question you can ask is whether different mutation consequences occur in each phenotype. Let's find out how many mutation consequences are observed in each phenotype.

```
d%>%
  group_by(Classification,Effect)%>%
  count()
```

```
## # A tibble: 25 x 3
## # Groups:   Classification, Effect [25]
##    Classification Effect                       n
##    <chr>          <chr>                    <int>
##  1 ASD/DD         DeletionCNV                  3
##  2 ASD/DD         DuplicationCNV               1
##  3 ASD/DD         Frameshift                  17
##  4 ASD/DD         Missense                    40
##  5 ASD/DD         Nonsense                    16
##  6 ASD/DD         PopulationVariantInExAC      3
##  7 ASD/DD         SpliceSite                  12
##  8 BIS            DuplicationCNV               1
##  9 BIS            Frameshift                   1
## 10 BIS            Missense                    34
## # ... with 15 more rows
```

3.4. Find the position where different consequences of mutations occur

If you checked the recurrent mutations, you might want to find a locus where two or more variants occur. Such loci might indicate functionally important position of the gene and you might find some insight as to a cause of disease.

```
r_locus <- d%>%
  group_by(c.DNA)%>%
  count()%>%
  filter(n>1)
r_locus$loci <- gsub('[^0-9]','',r_locus$c.DNA)%>%
  as.numeric()
r_locus
```

```
## # A tibble: 40 x 3
## # Groups:   c.DNA [40]
##    c.DNA         n  loci
##    <chr>     <int> <dbl>
##  1 ???           2    NA
##  2 .             8    NA
##  3 c.106A>G      3   106
##  4 c.1136G>A     2  1136
##  5 c.1267G>C     2  1267
##  6 c.2558G>A     9  2558
##  7 c.2645G>A     2  2645
##  8 c.2695G>A     2  2695
##  9 c.2715G>C     2  2715
## 10 c.2783T>G     2  2783
## # ... with 30 more rows
```

3.5. Sketch a plot to visualize your analysis

When you examine the dataset, you would draw something to show your output. Though we haven't learnt how to plot data yet, we can have a quick sketch for the dataset. There's no restriction on your suggestion. Please submit your hand-drawing for the plot you would like to show from this dataset.

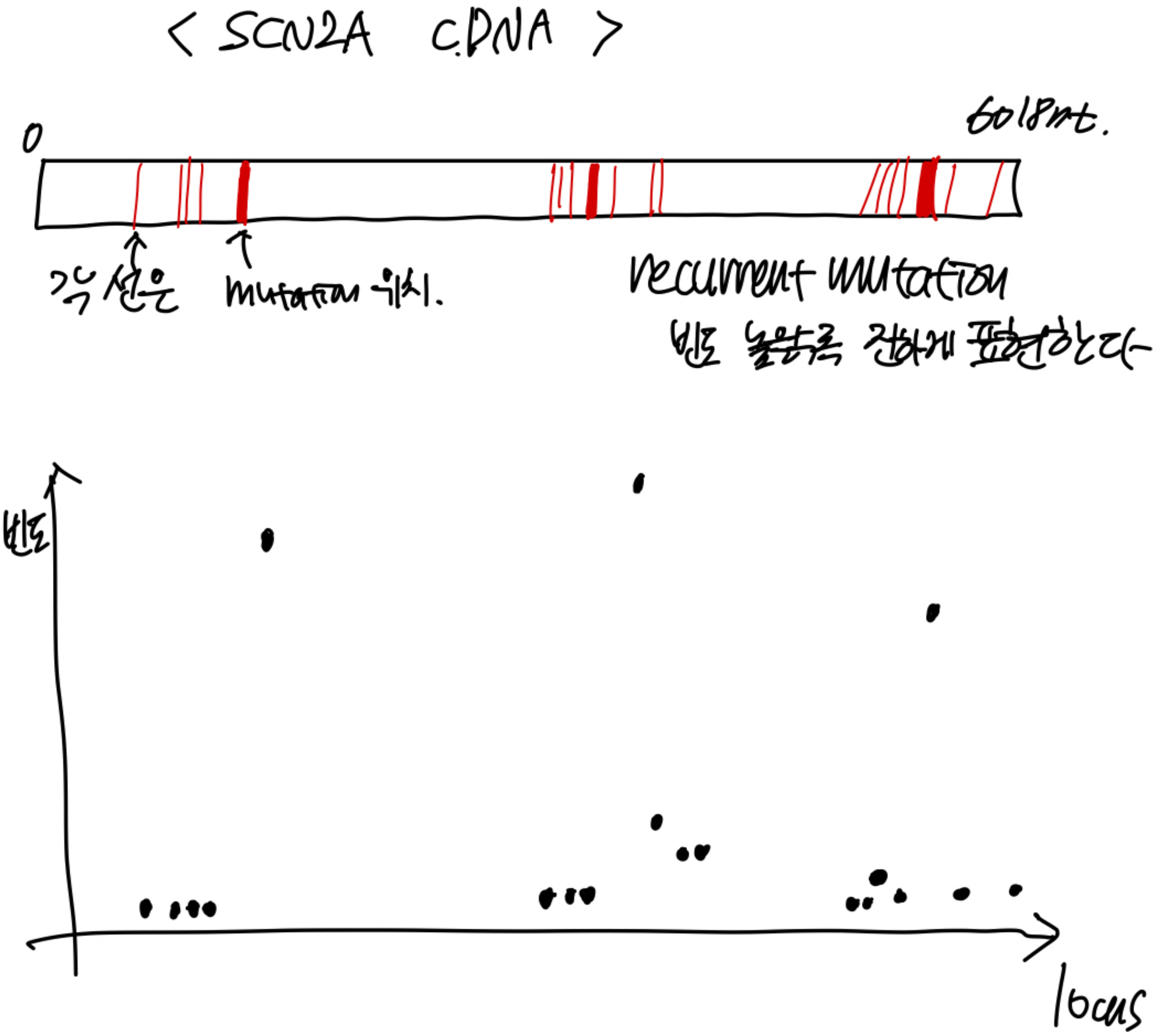

Figure 2: plot