[Tutorial] Human Genome Annotation

1. Introduction

1.1. What is gene annotation?

Over the past years, we have learnt that there are a number of chromosomes and genes in our genome. Counting the number of chromosomes is fairly easy but students might find difficult to say how many genes we have in our genome. If you can get an answer for this, could you tell how many genes encode protein and how many do not? To answer this question, we need to access the database for gene annotation. Gene annotation is the process of making nucleotide sequence meaningful - where genes are located? whether it is protein-coding or noncoding. If you would like to get an overview of gene annotation, please find this link. One of well-known collaborative efforts in gene annotation is the GENCODE consortium. It is a part of the Encyclopedia of DNA Elements (The ENCODE project consortium) and aims to identify all gene features in the human genome using a combination of computational analysis, manual annotation, and experimental validation (Harrow et al. 2012). You might find another database for gene annotation, like RefSeq, CCDS, and need to understand differences between them.

In this tutorial, we will access to gene annotation from the GENCODE consortium and explore genes and functional elements in our genome.

1.2. Aims

What we will do with this dataset:

- Be familiar with gene annotation modality.
- Tidy data and create a table for your analysis.
- Apply tidyverse functions for data munging.

Please note that there is better solution for getting gene annotation in R if you use a biomart. Our tutorial is only designed to have a practice on tidyverse exercise.

2. Explore your data

2.1. Unboxing your dataset

This tutorial will use a gene annotation file from the GENCODE. You will need to download the file from the GENCODE. If you are using terminal, please download file using wget:

```
# Run from your terminal, not R console
# wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_31/gencode.v31.basic.annotation.

# Once you downloaded the file, you won't need to download it again. So please comment out the command
```

Once you download the file, you can print out the first few lines using the following bash command (we will learn UNIX commands later):

```
# Run from your terminal, not R console
# gzcat gencode.v31.basic.annotation.gtf.gz | head -7
```

The file is the GFT file format, which you will find most commonly in gene annotation. Please read the file format thoroughly in the link above. For the tutorial, we need to load two packages. If the package is not installed in your system, please install it.

- **tidyverse**, a package you have learnt from the chapter 5.
- **readr**, a package provides a fast and friendly way to read. Since the file gencode.v31.basic.annotation.gtf.gz is pretty large, you will need some function to load data quickly into your workspace. readr in a part of tidyverse, so you can just load tidyverse to use readr functions.

Let's load the GTF file into your workspace. We will use read_delim function from the readr package. This is much faster loading than read.delim or read.csv from R base. However, please keep in mind that some parameters and output class for read_delim are slightly different from them.

```
library(tidyverse)
```

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

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

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

```
d = read_delim('/Users/choeseunghwan/Documents/GitHub/bsms222_105_choi/assignment/gencode.v31.basic.ann
              delim='\t', skip = 5, progress = F,
              col_names = F)
```

```
## Rows: 1756502 Columns: 9

## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (7): X1, X2, X3, X6, X7, X8, X9
## dbl (2): X4, X5

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Can you find out what the parameters mean? Few things to note are:

- The GTF file contains the first few lines for comments (#). In general, the file contains description, provider, date, format.

- The GTF file does not have column names so you will need to assign 'FALSE for col_names.

This is sort of canonical way to load your dataset into R. However, we are using a GTF format, which is specific to gene annotation so we can use a package to specifically handle a GTF file. Here I introduce the package rtracklayer. Let's install the package first.

```
# if (!requireNamespace("BiocManager", quietly = TRUE))
   # install.packages("BiocManager")

# BiocManager::install("rtracklayer")
```

Then, now you can read the GTF file using this package. Then, you can check the class of the object d.

```
d <- rtracklayer::import('gencode.v31.basic.annotation.gtf')
class(d)
```

```
## [1] "GRanges"
## attr(,"package")
## [1] "GenomicRanges"
```

You will find out that this is GRanges class. This is from the package Genomic Range, specifically dealing with genomic datasets but we are not heading into this in this tutorial. So please find this information if you are serious on this.

We are converting d into a data frame as following:

```
d <- d%>%as.data.frame()
```

Let's overview few lines from the data frame, and explore what you get in this object.

```
head(d)
```

```
##   seqnames start   end width strand source       type score phase
## 1     chr1 11869 14409  2541      + HAVANA       gene    NA    NA
## 2     chr1 11869 14409  2541      + HAVANA transcript    NA    NA
## 3     chr1 11869 12227   359      + HAVANA       exon    NA    NA
## 4     chr1 12613 12721   109      + HAVANA       exon    NA    NA
## 5     chr1 13221 14409  1189      + HAVANA       exon    NA    NA
## 6     chr1 12010 13670  1661      + HAVANA transcript    NA    NA
##             gene_id                          gene_type gene_name level
## 1 ENSG00000223972.5 transcribed_unprocessed_pseudogene   DDX11L1     2
## 2 ENSG00000223972.5 transcribed_unprocessed_pseudogene   DDX11L1     2
## 3 ENSG00000223972.5 transcribed_unprocessed_pseudogene   DDX11L1     2
## 4 ENSG00000223972.5 transcribed_unprocessed_pseudogene   DDX11L1     2
## 5 ENSG00000223972.5 transcribed_unprocessed_pseudogene   DDX11L1     2
## 6 ENSG00000223972.5 transcribed_unprocessed_pseudogene   DDX11L1     2
##      hgnc_id          havana_gene     transcript_id
## 1 HGNC:37102 OTTHUMG00000000961.2              <NA>
## 2 HGNC:37102 OTTHUMG00000000961.2 ENST00000456328.2
## 3 HGNC:37102 OTTHUMG00000000961.2 ENST00000456328.2
## 4 HGNC:37102 OTTHUMG00000000961.2 ENST00000456328.2
## 5 HGNC:37102 OTTHUMG00000000961.2 ENST00000456328.2
## 6 HGNC:37102 OTTHUMG00000000961.2 ENST00000450305.2
##                      transcript_type transcript_name transcript_support_level
## 1                               <NA>            <NA>                     <NA>
## 2                             lncRNA     DDX11L1-202                        1
## 3                             lncRNA     DDX11L1-202                        1
## 4                             lncRNA     DDX11L1-202                        1
## 5                             lncRNA     DDX11L1-202                        1
## 6 transcribed_unprocessed_pseudogene     DDX11L1-201                       NA
##     tag    havana_transcript exon_number           exon_id         ont
## 1  <NA>                 <NA>        <NA>              <NA>        <NA>
## 2 basic OTTHUMT00000362751.1        <NA>              <NA>        <NA>
## 3 basic OTTHUMT00000362751.1           1 ENSE00002234944.1        <NA>
## 4 basic OTTHUMT00000362751.1           2 ENSE00003582793.1        <NA>
## 5 basic OTTHUMT00000362751.1           3 ENSE00002312635.1        <NA>
## 6 basic OTTHUMT00000002844.2        <NA>              <NA> PGO:0000019
##   protein_id ccdsid
## 1       <NA>   <NA>
## 2       <NA>   <NA>
## 3       <NA>   <NA>
## 4       <NA>   <NA>
## 5       <NA>   <NA>
## 6       <NA>   <NA>
```

One thing you can find is that there is no columns in the data frame. Let's match which information is provided in columns. You can find the instruction page in the website (link).

Based on this, you can assign a name for 9 columns. One thing to remember is you should not use space for the column name. Spacing in the column name is actually working but not a good habit for your code. So please replace a space with underscore in the column name.

```
# Assign column names according to the GENCODE instruction.
cols = c('chrom', 'source', 'feature_type', 'start', 'end', 'score', 'strand', 'phase', 'info')
```

Now you can set up the column names into the col_names parameter, and load the file into a data frame.

```
d <- read_delim('gencode.v31.basic.annotation.gtf',
              delim="\t",
              skip = 5,
              progress = T,
              col_names = cols)
```

```
## indexing gencode.v31.basic.annotation.gtf [---------------] 218.12GB/s, eta:  0sindexing gencode.v31
```

```
## Rows: 1756502 Columns: 9
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (7): chrom, source, feature_type, score, strand, phase, info
## dbl (2): start, end
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

You can find the column names are now all set.

```
head(d)
```

```
## # A tibble: 6 x 9
##   chrom source feature_type start   end score strand phase info
##   <chr> <chr>  <chr>        <dbl> <dbl> <chr> <chr>  <chr> <chr>
## 1 chr1  HAVANA gene         11869 14409 .     +      .     "gene_id \"ENSG00000~
## 2 chr1  HAVANA transcript   11869 14409 .     +      .     "gene_id \"ENSG00000~
## 3 chr1  HAVANA exon         11869 12227 .     +      .     "gene_id \"ENSG00000~
## 4 chr1  HAVANA exon         12613 12721 .     +      .     "gene_id \"ENSG00000~
## 5 chr1  HAVANA exon         13221 14409 .     +      .     "gene_id \"ENSG00000~
## 6 chr1  HAVANA transcript   12010 13670 .     +      .     "gene_id \"ENSG00000~
```

When you loaded the file, you see the message about the data class. You might want to overview this data.

```
summary(d)
```

```
##     chrom              source          feature_type           start
##  Length:1756502     Length:1756502     Length:1756502     Min.   :      577
##  Class :character   Class :character   Class :character   1st Qu.: 32101517
##  Mode  :character   Mode  :character   Mode  :character   Median : 61732754
##                                                           Mean   : 75288563
##                                                           3rd Qu.:111760181
##                                                           Max.   :248936581
##       end               score              strand              phase
##  Min.   :      647   Length:1756502     Length:1756502     Length:1756502
##  1st Qu.: 32107331   Class :character   Class :character   Class :character
##  Median : 61738373   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 75292632
##  3rd Qu.:111763007
##  Max.   :248937043
##      info
##  Length:1756502
##  Class :character
##  Mode  :character
##
##
##
```

2.2 How many feature types in the GENCODE dataset?

As instructed in the GENCODE website, the GENCODE dataset provides a range of annotations for the feature type. You can check feature types using group_by() and count() function.

```
 d%>%group_by(feature_type)%>%
  count()
```

```
## # A tibble: 8 x 2
## # Groups:   feature_type [8]
##   feature_type        n
##   <chr>            <int>
## 1 CDS             567862
## 2 exon            744835
## 3 gene             60603
## 4 Selenocysteine      96
## 5 start_codon      57886
## 6 stop_codon       57775
## 7 transcript      108243
## 8 UTR             159202
```

How many feature types provided in the GENCODE? And how many items stored for each feature type? Please write down the number of feature types from the dataset. Also, if you are not familiar with these types, it would be good to put one or two sentences that can describe each type).

```
#There are 8 types.
#CDS: coding sequence
#Selenocysteine: amino acid encoded by UGA codon which is actually stop codon.
#UTR: untranslated region
```

2.3. How many genes we have?

Let's count the number of genes in our genome. Since we know that the column feature_type contains rows with gene, which contains obviously annotations for genes. We might want to subset those rows from the data frame.

```
d1 <- filter(d, feature_type=="gene")
# d1 = d[d$feature_type == 'gene', ]
nrow(d1)
```

```
## [1] 60603
```

```
#60603 genes
```

2.4. Ensembl, Havana and CCDS.

Gene annotation for the human genome is provided by multiple organizations with different gene annotation methods and strategy. This means that information can be varying by resources, and users need to understand heterogeniety inherent in annotation databases.

The GENCODE project utlizes two sources of gene annotation.

1. Havana: Manual gene annotation
2. Ensembl: Automatic gene annotation

It provides the combination of Ensembl/HAVANA gene set as the default gene annotation for the human genome. In addition, they also guarantee that all transcripts from the Consensus Coding Sequence (CCDS) set are present in the GENCODE gene set. The CCDS project is a collaborative effort to identify a core set of protein coding regions that are consistently annotated and of high quality. Initial results from the Consensus CDS (CCDS) project are now available through the appropriate Ensembl gene pages and from the CCDS project page at NCBI. The CCDS set is built by consensus among Ensembl, the National Center for Biotechnology Information (NCBI), and the HUGO Gene Nomenclature Committee (HGNC) for human.

Right. Then now we count how many genes annotated with HAVANA and ENSEMBL.

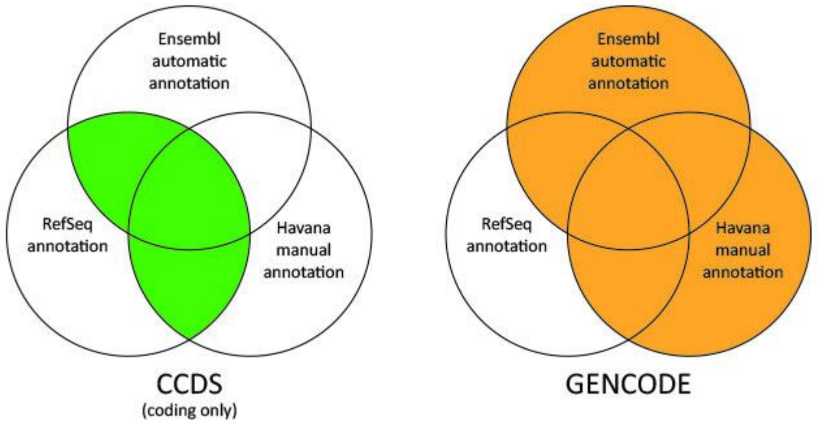

Figure 1: **Figure 2. Comparison of CCDS and Gencode**

```
d1 %>% group_by(source)%>%count()
```

```
## # A tibble: 2 x 2
## # Groups:   source [2]
##   source      n
##   <chr>   <int>
## 1 ENSEMBL  7609
## 2 HAVANA  52994
```

2.5 do.call

Since the last column info contains a long string for multiple annotations, we will need to split it to extract each annotation. For example, the first line for transcript annotation looks like this:

```
d[2, 9] %>%
    as.character()
```

```
## [1] "gene_id \"ENSG00000223972.5\"; transcript_id \"ENST00000456328.2\"; gene_type \"transcribed_unp
```

```
# chr1 HAVANA transcript 11869 14409 .  + .  gene_id
# 'ENSG00000223972.5'; transcript_id 'ENST00000456328.2'; gene_type
# 'transcribed_unprocessed_pseudogene'; gene_name 'DDX11L1';
# transcript_type 'lncRNA'; transcript_name 'DDX11L1-202'; level 2;
# transcript_support_level '1'; hgnc_id 'HGNC:37102'; tag 'basic';
# havana_gene 'OTTHUMG00000000961.2'; havana_transcript
# 'OTTHUMT00000362751.1';
```

If you would like to split transcript_support_level and create a new column, you can use strsplit function.

```
a = 'chr1    HAVANA    transcript    11869    14409    .    +    .    gene_id "ENSG00000223972.5"; tran
strsplit(a, 'transcript_support_level\\s+"')
```

```
## [[1]]
## [1] "chr1    HAVANA    transcript    11869    14409    .    +    .    gene_id \"ENSG00000223972.5\";
## [2] "1\"; hgnc_id \"HGNC:37102\"; tag \"basic\"; havana_gene \"OTTHUMG00000000961.2\"; havana_transc
```

After split the string, you can select the second item in the list ([[1]][2]).

```
strsplit(a, 'transcript_support_level\\s+"')[[1]][2]
```

```
## [1] "1\"; hgnc_id \"HGNC:37102\"; tag \"basic\"; havana_gene \"OTTHUMG00000000961.2\"; havana_transc
```

You can find the 1 in the first position, which you will need to split again.

```
b <- strsplit(a, 'transcript_support_level\\s+"')[[1]][2]
strsplit(b,'\\"')
```

```
## [[1]]
##  [1] "1"                      "; hgnc_id "           "HGNC:37102"
##  [4] "; tag "                 "basic"                "; havana_gene "
##  [7] "OTTHUMG00000000961.2" "; havana_transcript " "OTTHUMT00000362751.1"
## [10] ";"
```

From this, you will get the first item in the list ([[1]][1]). Now you would like to apply strsplit function across vectors. For this, do.call function can be easily implemented to strsplit over the vectors from one column. Let's try this.

```
head(do.call(rbind.data.frame, strsplit(a, 'transcript_support_level\\s+"'))[[2]])
```

```
## [1] "1\"; hgnc_id \"HGNC:37102\"; tag \"basic\"; havana_gene \"OTTHUMG00000000961.2\"; havana_transc
```

Now you can write two lines of codes to process two steps we discussed above.

```
# First filter transcripts and create a data frame.
d2 <- d %>% filter(feature_type == 'transcript')

# Now apply the functions.
d2$transcript_support_level <- as.character(do.call(rbind.data.frame, strsplit(d2$info, 'transcript_sup

d2$transcript_support_level <- as.character(do.call(rbind.data.frame,strsplit(d2$transcript_support_leve
```

Now you can check the strsplit works.

```
head(d2$transcript_support_level)
```

```
## [1] "1"  "NA" "NA" "NA" "5"  "5"
```

You can use the same method to extract other annotations, like gene_id, gene_name etc.

3. Exercises

Here I list the questions for your activity. Please note that it is an exercise for tidyverse functions, which you will need to use in your code. In addition, you will need to write an one-line code for each question using pipe %>%.

For questions, you should read some information thoroughly, including:

- Gene biotype.
- 0 or 1 based annotation in GTF, BED format
- Why some features have 1 bp length?
- What is the meaning of zero-length exons in GENCODE? Also fun to have a review for microexons
- Transcript support level (TSL)

3.1. Annotation of transcripts in our genome

1. Computes the number of transcripts per gene. What is the mean number of transcripts per gene? What is the quantile (25%, 50%, 75%) for these numbers? Which gene has the greatest number of transcript?

```r
d <- rtracklayer::import('gencode.v31.basic.annotation.gtf') %>%as.data.frame()
# number of transcripts per gene
geneid<-d%>%filter(type=="transcript")%>%group_by(gene_id)%>%count(gene_id)
head(geneid)
```

```
## # A tibble: 6 x 2
## # Groups:   gene_id [6]
##   gene_id                n
##   <chr>              <int>
## 1 ENSG00000000003.14     3
## 2 ENSG00000000005.6      1
## 3 ENSG00000000419.12     2
## 4 ENSG00000000457.14     3
## 5 ENSG00000000460.17     5
## 6 ENSG00000000938.13     3
```

```r
# mean number of transcripts per gene
mean(geneid$n)
```

```
## [1] 1.7861
```

```r
#quantile (25%, 50%, 75%) for these numbers
quantile(geneid$n,c(0.25,0.5,0.75))
```

```
## 25% 50% 75%
##   1   1   2
```

```r
# gene which has greatest number of transcript
geneid$gene_id[which.max(geneid$n)]
```

```
## [1] "ENSG00000109339.22"
```

2. Compute the number of transcripts per gene among gene biotypes. For example, compare the number of transcript per gene between protein-coding genes, long noncoding genes, pseudogenes.

```
# number of transcripts per gene among gene biotypes
d%>%filter(type=="transcript")%>%group_by(gene_type,gene_id)%>%count()%>%
   group_by(gene_type)%>%summarize(number_of_transcript_per_gene=mean(n))%>%
   filter(gene_type%in%c("protein_coding","lncRNA","pseudogene"))
```

```
## # A tibble: 3 x 2
##   gene_type      number_of_transcript_per_gene
##   <chr>                                  <dbl>
## 1 lncRNA                                  1.48
## 2 protein_coding                          2.90
## 3 pseudogene                              1
```

3. Final task is to compute the number of transcripts per gene per chromosome.

```
d%>%filter(type=="transcript")%>%group_by(seqnames,gene_id)%>%count()%>%
   group_by(seqnames)%>%summarize(number_of_transcript_per_chromosome=mean(n))
```

```
## # A tibble: 25 x 2
##    seqnames number_of_transcript_per_chromosome
##    <fct>                                  <dbl>
##  1 chr1                                    1.80
##  2 chr2                                    1.77
##  3 chr3                                    1.93
##  4 chr4                                    1.76
##  5 chr5                                    1.74
##  6 chr6                                    1.78
##  7 chr7                                    1.76
##  8 chr8                                    1.75
##  9 chr9                                    1.70
## 10 chr10                                   1.78
## # ... with 15 more rows
```

3.2. Gene length in the GENCODE

1. What is the average length of human genes?

```
d%>%filter(type=="gene")%>%pull(width)%>%mean()%>%round()
```

```
## [1] 32629
```

```
#32629bp
```

2. Is the distribution of gene length differed by autosomal and sex chromosomes? Please calculate the quantiles (0%, 25%, 50%, 75%, 100%) of the gene length for each group.

```
#average gene length for each chromosome
d%>%filter(type=="gene")%>%group_by(seqnames)%>%summarize(gene_length_average=mean(width))
```

```
## # A tibble: 25 x 2
##    seqnames gene_length_average
##    <fct>                  <dbl>
##  1 chr1                  29804.
##  2 chr2                  40228.
##  3 chr3                  46040.
##  4 chr4                  44331.
##  5 chr5                  41340.
##  6 chr6                  35823.
##  7 chr7                  36265.
##  8 chr8                  40032.
##  9 chr9                  32449.
## 10 chr10                 37473.
## # ... with 15 more rows
```

```
#calculate the quantiles (0%, 25%, 50%, 75%, 100%) of the gene length for autosomal and sex chromosome 
d %>% filter(type=="gene")%>%
   mutate(chromosome=case_when(seqnames%in%c("chrX","chrY")~"sex",seqnames=="chrM"~"chrM",TRUE~"autosoma
   filter(chromosome %in% c("sex","autosomal"))%>%
   group_by(chromosome)%>%summarize(quantiles=quantile(width,seq(0,1,0.25)))
```

```
## `summarise()` has grouped output by 'chromosome'. You can override using the `.groups` argument.
```

```
## # A tibble: 10 x 2
## # Groups:   chromosome [2]
##    chromosome quantiles
##    <chr>          <dbl>
##  1 autosomal          8
##  2 autosomal        565
##  3 autosomal       3779
##  4 autosomal      25813
##  5 autosomal    2473537
##  6 sex               48
##  7 sex              473
##  8 sex             1912
##  9 sex            13502
## 10 sex          2241765
```

3. Is the distribution of gene length differed by gene biotype? Please calculate the quantiles (0%, 25%, 50%, 75%, 100%) of the gene length for each group.

```
d %>% filter(type=="gene")%>%group_by(gene_type)%>%summarize(quantiles=quantile(width,seq(0,1,0.25)))
```

```
## `summarise()` has grouped output by 'gene_type'. You can override using the `.groups` argument.
```

```
## # A tibble: 200 x 2
## # Groups:   gene_type [40]
##    gene_type       quantiles
##    <chr>               <dbl>
##  1 IG_C_gene            441
##  2 IG_C_gene            477.
##  3 IG_C_gene           4590.
##  4 IG_C_gene           5479.
##  5 IG_C_gene           8914
##  6 IG_C_pseudogene      248
##  7 IG_C_pseudogene      313
##  8 IG_C_pseudogene      317
##  9 IG_C_pseudogene      734
## 10 IG_C_pseudogene     5211
## # ... with 190 more rows
```

3.3. Transcript support levels (TSL)

The GENCODE TSL provides a consistent method of evaluating the level of support that a GENCODE transcript annotation is actually expressed in humans.

1. With transcript, how many transcripts are categorized for each TSL?

```
d%>%filter(type=="transcript" & !is.na(transcript_support_level))%>%group_by(transcript_support_level)%
```

```
## # A tibble: 6 x 2
## # Groups:   transcript_support_level [6]
##   transcript_support_level     n
##   <chr>                    <int>
## 1 1                        31801
## 2 2                        13372
## 3 3                         7228
## 4 4                         2245
## 5 5                        13674
## 6 NA                       27843
```

2. From the first question, please count the number of transcript for each TSL by gene biotype.

```
d%>%filter(type=="transcript" & !is.na(transcript_support_level))%>%group_by(gene_type,transcript_suppo
```

```
## # A tibble: 76 x 3
## # Groups:   gene_type, transcript_support_level [76]
##    gene_type       transcript_support_level     n
##    <chr>           <chr>                    <int>
##  1 IG_C_gene       1                            1
##  2 IG_C_gene       5                            1
##  3 IG_C_gene       NA                           7
##  4 IG_C_pseudogene NA                           9
##  5 IG_D_gene       NA                          37
##  6 IG_J_gene       NA                          18
##  7 IG_J_pseudogene NA                           3
##  8 IG_pseudogene   NA                           1
##  9 IG_V_gene       5                            3
## 10 IG_V_gene       NA                         141
## # ... with 66 more rows
```

3. From the first question, please count the number of transcript for each TSL by source.

```
d%>%filter(type=="transcript" & !is.na(transcript_support_level))%>%group_by(source,transcript_support_l
   group_by(source)%>%
   mutate(proportion=n/sum(n))
```

```
## # A tibble: 12 x 4
## # Groups:   source [2]
##    source  transcript_support_level     n proportion
##    <fct>   <chr>                    <int>      <dbl>
##  1 HAVANA  1                        29434    0.365
##  2 HAVANA  2                        12052    0.149
##  3 HAVANA  3                         6964    0.0863
##  4 HAVANA  4                         2116    0.0262
##  5 HAVANA  5                        10157    0.126
##  6 HAVANA  NA                       19962    0.247
##  7 ENSEMBL 1                         2367    0.153
##  8 ENSEMBL 2                         1320    0.0853
##  9 ENSEMBL 3                          264    0.0171
## 10 ENSEMBL 4                          129    0.00833
## 11 ENSEMBL 5                         3517    0.227
## 12 ENSEMBL NA                        7881    0.509
```

3.4 CCDS in the GENCODE

1. With gene, please create a data frame with the columns - gene_id, gene_name, hgnc_id, gene_type, chromosome, start, end, and strand. Then, please create new columns for presence of hgnc and ccds. For example, you can put 1 in the column isHgnc, if hgnc annotation is avaiable, or 0 if not. Then, you can put 1 in the column isCCDS, if ccds annotation is avaiable, or 0 if not.

```
# I created new data frame called 'newd'
newd<- d%>%select(gene_id,gene_name,hgnc_id,ccdsid,gene_type,seqnames,start,end,strand)%>%mutate(isHgnc=
head(newd)
```

```
##              gene_id gene_name    hgnc_id ccdsid
## 1 ENSG00000223972.5   DDX11L1 HGNC:37102   <NA>
## 2 ENSG00000223972.5   DDX11L1 HGNC:37102   <NA>
## 3 ENSG00000223972.5   DDX11L1 HGNC:37102   <NA>
## 4 ENSG00000223972.5   DDX11L1 HGNC:37102   <NA>
## 5 ENSG00000223972.5   DDX11L1 HGNC:37102   <NA>
## 6 ENSG00000223972.5   DDX11L1 HGNC:37102   <NA>
##                            gene_type seqnames start   end strand isHgnc isCCDS
## 1 transcribed_unprocessed_pseudogene     chr1 11869 14409      +      1      0
## 2 transcribed_unprocessed_pseudogene     chr1 11869 14409      +      1      0
## 3 transcribed_unprocessed_pseudogene     chr1 11869 12227      +      1      0
## 4 transcribed_unprocessed_pseudogene     chr1 12613 12721      +      1      0
## 5 transcribed_unprocessed_pseudogene     chr1 13221 14409      +      1      0
## 6 transcribed_unprocessed_pseudogene     chr1 12010 13670      +      1      0
```

2. Please count the number of hgnc by gene biotypes.

```r
d %>% filter(!is.na(hgnc_id))%>%group_by(gene_type) %>% count()
```

```
## # A tibble: 36 x 2
## # Groups:   gene_type [36]
##    gene_type           n
##    <chr>           <int>
##  1 IG_C_gene         176
##  2 IG_C_pseudogene    33
##  3 IG_D_gene         152
##  4 IG_J_gene          76
##  5 IG_J_pseudogene     9
##  6 IG_V_gene        1101
##  7 IG_V_pseudogene   653
##  8 lncRNA          51105
##  9 miRNA            5568
## 10 misc_RNA         3099
## # ... with 26 more rows
```

3. Please count the number of hgnc by level. Please note that level in this question is not TSL. Please find information in this link: 1 (verified loci), 2 (manually annotated loci), 3 (automatically annotated loci).

```r
d%>%filter(!is.na(hgnc_id))%>%group_by(level)%>%count()
```

```
## # A tibble: 3 x 2
## # Groups:   level [3]
##   level       n
##   <chr>   <int>
## 1 1      107054
## 2 2     1279964
## 3 3      237265
```

3.5. Transcripts in the GENCODE

1. Which gene has the largest number of transcripts?

```r
d%>%filter(type=="transcript")%>%group_by(gene_id)%>%count()%>%pull(n)%>%which.max()%>%d$gene_id[.]
```

```
## [1] "ENSG00000248333.8"
```

2. Please calculate the quantiles (0%, 25%, 50%, 75%, 100%) of the gene length for protein coding genes and long noncoding genes.

```r
d%>%filter(type=="gene"& gene_type%in%c("protein_coding","lncRNA"))%>%group_by(gene_type)%>%summarize(q
```

```
## `summarise()` has grouped output by 'gene_type'. You can override using the `.groups` argument.
## # A tibble: 10 x 2
## # Groups:   gene_type [2]
##    gene_type      quantile_of_gene_length
##    <chr>                            <dbl>
##  1 lncRNA                              68
##  2 lncRNA                            1874.
##  3 lncRNA                            6272.
##  4 lncRNA                           24774.
##  5 lncRNA                         1375317
##  6 protein_coding                     117
##  7 protein_coding                    9632.
##  8 protein_coding                   27212
##  9 protein_coding                   70809
## 10 protein_coding                 2473537
```

3. Please count the number of transcripts by chromosomes.

```
d%>%filter(type=="transcript")%>%group_by(seqnames)%>%count()
```

```
## # A tibble: 25 x 2
## # Groups:   seqnames [25]
##    seqnames     n
##    <fct>    <int>
##  1 chr1      9827
##  2 chr2      7432
##  3 chr3      6157
##  4 chr4      4662
##  5 chr5      5203
##  6 chr6      5455
##  7 chr7      5292
##  8 chr8      4350
##  9 chr9      3949
## 10 chr10     4157
## # ... with 15 more rows
```

3.6. Autosomal vs. Sex chromosomes.

1. Please calculate the number of genes per chromosome.

```
d%>%filter(type=="gene")%>%group_by(seqnames)%>%count()
```

```
## # A tibble: 25 x 2
## # Groups:   seqnames [25]
##    seqnames     n
##    <fct>    <int>
##  1 chr1      5471
##  2 chr2      4196
##  3 chr3      3185
##  4 chr4      2651
##  5 chr5      2983
##  6 chr6      3059
##  7 chr7      3014
##  8 chr8      2482
##  9 chr9      2327
## 10 chr10     2332
## # ... with 15 more rows
```

2. Please compare the number of genes between autosomal and sex chromosome (Mean, Median).

```
d %>% filter(type=="gene")%>%
   mutate(chromosome=case_when(seqnames%in%c("chrX","chrY")~"sex",seqnames=="chrM"~"chrM",TRUE~"autosoma
   filter(chromosome %in% c("sex","autosomal"))%>%
   group_by(chromosome)%>%count()
```

```
## # A tibble: 2 x 2
## # Groups:   chromosome [2]
##   chromosome     n
##   <chr>      <int>
## 1 autosomal  57577
## 2 sex         2989
```

3. Please divide the genes into groups ‘protein coding’ and ‘long noncoding’, and then compare the number of genes in each chromosomes within groups.

```
d%>%filter(type=="gene"&gene_type%in%c("protein_coding","lncRNA"))%>%group_by(gene_type,seqnames)%>%cour
```

```
## # A tibble: 49 x 3
## # Groups:   gene_type, seqnames [49]
##    gene_type seqnames     n
##    <chr>     <fct>    <int>
##  1 lncRNA    chr1      1416
##  2 lncRNA    chr2      1241
##  3 lncRNA    chr3       861
##  4 lncRNA    chr4       790
##  5 lncRNA    chr5       950
##  6 lncRNA    chr6       826
##  7 lncRNA    chr7       720
##  8 lncRNA    chr8       831
##  9 lncRNA    chr9       555
## 10 lncRNA    chr10      695
## # ... with 39 more rows
```