Test R part

Ziang_C

2023-09-19

R Markdown

Task 1: How many unique transcripts are there?

```
trans_count_ID = length(unique(exgr_test$transcript_id))

trans_count_name =length(unique(exgr_test$transcript_name))
if (trans_count_ID == trans_count_name) {
  print(paste("The number of unique transcript is:", trans_count_name))
}
```

```
## [1] "The number of unique transcript is: 234485"
```

Task 2: How many unique exons are there?

```
exon_count = length(unique(exgr_test$exon_name))
print(paste("The number of unique exon is:", exon_count))
```

```
## [1] "The number of unique exon is: 760040"
```

#Here, number of unique exons' ID is 509 larger than the number of unique exons' name. This is because the same exons (same name) are labeled as different ID in ChrX and ChrY. The top 6 in the list of those exons are as follow:

```
new_table <- exgr_test[c("exon_id", "exon_name")]
shortened_table <- new_table %>%
  distinct(exon_id, .keep_all = TRUE)
s2 <- new_table %>%
  distinct(exon_name, .keep_all = TRUE)

difference <- anti_join(shortened_table, s2, by = "exon_id")
head(difference)
```

```
##   exon_id         exon_name
## 1  757103 ENSE00001702291.1
```

```
## 2  757104 ENSE00001655436.1
## 3  757105 ENSE00001489430.1
## 4  757111 ENSE00001306908.2
## 5  757117 ENSE00003812308.1
## 6  757121 ENSE00003812426.1
```

Task 3: what is the average length of an exon? What is the median length?

```
exon_mean = mean(exgr_test$width)
exon_med = median(exgr_test$width)
print(paste("The average length of an exon is:", exon_mean, "The median length is:",exon_med ))
```

```
## [1] "The average length of an exon is: 262.993063588563 The median length is: 130"
```

Task 4:Find the length of the introns between the exons.(length must be a positve number)

```
#Set the time count for the task
start_time <- Sys.time()
#seperate data by +/- strand, set this outside loop will reduce the running time
data1 <- subset(exgr_test, strand == '+')
trans1 <- list() #store list

#loop over the positive strand to find the length of introns
for (i in 1:nrow(data1)) {
  #if rank is not 1, then it contains intron between it's last exons
  if (data1$rank[i] != 1) {
      #record intron's information(transcript information,rank,length)
      trans1$transcript_id[i] <- data1$transcript_id[i]
      trans1$transcript_name[i] <- data1$transcript_name[i]
      trans1$strand[i] <- '+'
      trans1$introns_rank[i] <- data1$rank[i-1]
      trans1$length[i] <- data1$start[i] -  data1$end[i-1] - 1
  }
}

#loop over the negative strand to find the length of introns
data2 <- subset(exgr_test, strand == '-')
trans2 <- list()
for (i in 1:nrow(data2)) {
  #if rank is not 1, then it contains intron between its last exons
  if (data2$rank[i] != 1) {
    #record intron's information(transcript information,rank,length)
    trans2$transcript_id[i] <- data2$transcript_id[i]
    trans2$transcript_name[i] <- data2$transcript_name[i]
    trans2$strand[i] <- '-'
```

```
    trans2$introns_rank[i] <- data2$rank[i-1]
    trans2$length[i] <- data2$start[i-1] -  data2$end[i] - 1
  }
}

# combine the table of negative and positive strands
table1 <- data.frame(trans1)
table2 <- data.frame(trans2)
combined_table <- rbind(table1, table2)
introns_table <- combined_table[complete.cases(combined_table), ]

# count time cost
end_time <- Sys.time()
time_cost <- end_time - start_time
print(paste("The time cost of this task is:", time_cost))
```

```
## [1] "The time cost of this task is: 11.0353600978851"
```

```
# view the head of the table to check format
head(introns_table)
```

```
##   transcript_id   transcript_name strand introns_rank length
## 2             1 ENST00000456328.2      +            1    385
## 3             1 ENST00000456328.2      +            2    499
## 5             2 ENST00000450305.2      +            1    121
## 6             2 ENST00000450305.2      +            2    385
## 7             2 ENST00000450305.2      +            3    277
## 8             2 ENST00000450305.2      +            4    168
```

```
# Write the result table to a tab-delimited text file
write.table(introns_table, file = "Intron_length.txt", sep = "\t", row.names = FALSE)
```