ETC1010/ETC5510: Introduction to Data Analysis

Arindam Baruah (32779267)

```
# Please do not touch this R code chunk!
knitr::opts_chunk$set(
  echo = TRUE,
  eval = FALSE,
  out.width = "70%",
  fig.width = 8,
  fig.height = 6,
  fig.retina = 3)
set.seed(6)
filter <- dplyr::filter
```

Instructions to Students

This is an individual assignment and you must work on it on your own. Collaboration on the assignment constitute collusion. For more on collusion and misconduct please see this webpage. (https://connect.monash.edu/s/article/FAQ-2144)

This assignment is designed to simulate a scenario in which you are taking over someone's existing work and continuing with it to draw further insights.

You have just joined an online music streaming service as a data analyst. You've been brought on to help understand the preferences of the companies user base. To get you started on understanding the data that the company has on its users, you are to perform a short EDA on a snippet of user data taken from a single users music library. You are to communicate your findings about this user's musical tastes to the head data scientist. This is not a formal report, but rather something you are giving to your manager that describes the data with some interesting insights.

Please make sure you read the hints throughout the assignment to help guide you on the tasks.

The points allocated for each of the elements in the assignment are marked in the questions and next to the code for those questions where a code scaffolding is provided.

Marking + Grades

This assignment will be worth **10%** of your total grade. **Due on: Friday March 31st, by 5:00pm (Melbourne time)**. **Late submissions will not be accepted.**

For this assignment, you will need to upload the following into Moodle:

- The rendered html file saved as a pdf. The assignment will be only marked if the pdf is uploaded in Moodle. **The submitted assignment pdf file must have all the code and output visible.**
- To complete the assignment, you will need to fill in the blanks with appropriate R code for some questions. These sections are marked with ___ . For other questions, you will need to write the entire R code chunk. For the inline code questions, you will need to replace the uppercase "R"

portion of the inline code with a lowercase “r”. For instance, in the code `R ___ ggplot()` you will replace the “R” at the beginning with “r”.

- **At a minimum, your assignment should be able to be “knitted”** using the `Knit` button for your Rmarkdown document so that you can produce a html file that you will save as pdf file and upload it into Moodle. You will be reminded about how to save the rendered html file into pdf in the tutorials of Week 3.

If you want to view what the assignment looks like as you progress, remember that you can set the R chunk options to `eval = FALSE` like so to ensure that you can knit the file:

````
```{r this-chunk-will-not-run, eval = FALSE} `r''`
a <- 1 + 2
```
````

If you use `eval = FALSE` or `echo = FALSE`, please remember to ensure that you have set to `eval = TRUE` and `echo = TRUE` when you submit the assignment, to ensure all your R codes run.

IMPORTANT: You must use R code to answer all the questions in the report.

Due Date

This assignment is due in by close of business (5:00pm) on Friday, March 31st 2023. You will submit the knitted html file **saved as a pdf** via Moodle. Please make sure you add your name on the YAML part of the Rmd file before you knit it and save it as pdf. **Please save the pdf in the format `name_Assign1_ETC1010` if you are enrolled in ETC1010, and `name_Assign1_ETC5510` if you are enrolled in ETC5510.

How to find help from R functions?

Remember, you can look up the help file for functions by typing: `?function_name`. For example, `?mean`.

Load all the libraries that you need here

```
library(tidyverse)
library(emo)
```

Reading and preparing data

```
music <- read_csv("data/music-sub.csv")
```

Question 1: Display the first 10 rows of the data set (1pt). **Hint:** Check *?head* in your R console

```
head(music,10)
```

```
## # A tibble: 10 × 8
##    ...1           artist type       lvar  lave  lmax lfener lfreq
##    <chr>          <chr>  <chr>     <dbl> <dbl> <dbl>  <dbl> <dbl>
##  1 Dancing Queen  Abba   Rock  17600756. -90.0 29921   106.  59.6
##  2 Knowing Me     Abba   Rock   9543021. -75.8 27626   103.  58.5
##  3 Take a Chance  Abba   Rock   9049482. -98.1 26372   102. 125.
##  4 Mamma Mia      Abba   Rock   7557437. -90.5 28898   102.  48.8
##  5 Lay All You    Abba   Rock   6282286. -89.0 27940   100.  74.0
##  6 Super Trouper  Abba   Rock   4665867. -69.0 25531   100.  81.4
##  7 I Have A Dream Abba   Rock   3369670. -71.7 14699   105. 305.
##  8 The Winner     Abba   Rock   1135862  -67.8  8928   104. 278.
##  9 Money          Abba   Rock   6146943. -76.3 22962   102. 165.
## 10 SOS            Abba   Rock   3482882. -74.1 15517   104. 147.
```

Question 2: How many observations and variables does the data set *music* have (1pt)? Use inline code to complete the sentence below (2pts)

The number of observations are **62** (1pt) and the number of variables are **8** (1pt)

Question 3: What is the name of the 3rd variable in this data set (2pts)? Use R commands to answer this question.

```
colnames(music[,3])
```

```
## [1] "type"
```

Question 4: Using the *music* data set, rename the first variable to *song* and save this new data frame as *tab_music* (2pts). Display the first 4 rows corresponding to the artist “Vivaldi” for all the variables in *tab_music* (1pt).

```
tab_music<- music  %>%
  rename(song = ...1)

tab_music %>% #1pt
  dplyr::filter(artist == "Vivaldi") %>% #1pt
  head(4)                                # 1pt
```

```
## # A tibble: 4 × 8
##   song  artist  type          lvar  lave  lmax lfener lfreq
##   <chr> <chr>   <chr>        <dbl> <dbl> <dbl>  <dbl> <dbl>
## 1 V1    Vivaldi Classical 3677296.  66.7 24229   99.3  330.
## 2 V2    Vivaldi Classical  771492.  21.7  6936  104.   844.
## 3 V3    Vivaldi Classical 5227573.  88.6 17721  105.   166.
## 4 V4    Vivaldi Classical  334719.  13.8  4123  104.   294.
```

Question 5: How many songs are recorded in the *music* data frame for `type` Rock (2pts)? Hint: you can use `count` or `nrow` to complete this.

```
rock_music <- music %>%
  dplyr::filter(type=='Rock')
nrow(rock_music)
```

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

Question 6: In the dataframe `music`, select all observations corresponding to the genres `rock` and `New wave` and store this data in a new data object called *data_tab* (3pts). Print the first 4 rows of the *data_tab* data set (1pt). What is the dimension of the new data set data_tab? (2pts)

```
data_tab<- music %>%
  dplyr::filter(type %in% c('Rock','New wave'))

head(data_tab,4)
```

```
## # A tibble: 4 × 8
##   ...1          artist type        lvar  lave  lmax lfener lfreq
##   <chr>         <chr>  <chr>      <dbl> <dbl> <dbl>  <dbl> <dbl>
## 1 Dancing Queen Abba   Rock  17600756. -90.0 29921   106.  59.6
## 2 Knowing Me    Abba   Rock   9543021. -75.8 27626   103.  58.5
## 3 Take a Chance Abba   Rock   9049482. -98.1 26372   102. 125.
## 4 Mamma Mia     Abba   Rock   7557437. -90.5 28898   102.  48.8
```

```
dim(data_tab)
```

```
## [1] 35  8
```

```
write.csv(data_tab,'data/data_tab.csv')
```

The dimension of data_tab is **35** (#1pt) rows and **8** columns (#1pt).

Question 7: How many unique artists are recorded for each of the genres in *data_tab* (2pt)? Display the results using functions from the tidyverse package. **Hint:**This is equivalent to displaying the number of observations for each of the artists.

```
unique_artists = data_tab %>% group_by(type)  %>%
                    summarise(total_artists = sum(n_distinct(artist)))
unique_artists
```

```
## # A tibble: 2 × 2
##   type     total_artists
##   <chr>            <int>
## 1 New wave             1
## 2 Rock                 3
```

Question 8: What are the unique elements in the variable *artist* in the data object *data_tab* (Display the results using R code) (1pt)? How many are there (use an R command to count the number of elements) and complete the sentence below using inline R code (1pt). Hint: type `?unique` or `?length` into the R console if unsure what to do.

```
unique(data_tab$artist)
```

```
## [1] "Abba"    "Eels"    "Beatles" "Enya"
```

There are **4** different elements in the variable `artist` inside the **data_tab** dataframe.

Question 9: Using the *data_tab* data frame, calculate the average frequence (recorded in `lfreq`) for each of the rock artists in the data set. Store the results in a new variable called *avg*. (2pts). Store the new data frame in a data object called *Avg_music* and display the results (1pt). **Hint:** This new data object will need to have two columns.

```
Avg_music <- data_tab %>%
  dplyr::filter(type=="Rock") %>%  #1pt
  group_by(artist) %>%        #1pt
  summarise(Avg = mean(lfreq))  #0.5pt

Avg_music #0.5pt
```

```
## # A tibble: 3 × 2
##   artist    Avg
##   <chr>   <dbl>
## 1 Abba     135.
## 2 Beatles  147.
## 3 Eels     181.
```

Question 10: What is the within genre frequency range for each piece of music in the `music` dataset? To answer this question, use the *tab_music* data frame, and create a new variable called *range* and store the new data frame under the data object *tab_music_range*. Display the first four rows of the resulting data frame (3pts) **Hint:** To calculate the frequency range, take each genre specific value of `lfreq` and subtract from it the minimum value of

`lfreq` for that genre, and then divide this answer by the maximum value of `lfreq` for that genre.

```
tab_music_range <- tab_music %>%
                   group_by(type) %>%
                   mutate(range = (lfreq-min(lfreq))/max(lfreq))

head(tab_music_range,4)
```

```
## # A tibble: 4 × 9
## # Groups:   type [1]
##   song          artist type       lvar  lave  lmax lfener lfreq  range
##   <chr>         <chr>  <chr>     <dbl> <dbl> <dbl>  <dbl> <dbl>  <dbl>
## 1 Dancing Queen Abba   Rock  17600756. -90.0 29921   106.  59.6 0.0463
## 2 Knowing Me    Abba   Rock   9543021. -75.8 27626   103.  58.5 0.0435
## 3 Take a Chance Abba   Rock   9049482. -98.1 26372   102. 125.  0.212
## 4 Mamma Mia     Abba   Rock   7557437. -90.5 28898   102.  48.8 0.0188
```

```
write.csv(tab_music_range,'data/tab_music_range.csv')
```

Question 11: Use the data object *tab_music*, order the observations from largest to smallest according to the *range* variable and display the results in a table (1pt). Which genre type has the highest average frequence range (1pt)? Which genre type has the least variable frequency range (1pt)?

```
or_tab_music <- tab_music_range %>%
  arrange(-range) #1pt

or_tab_music
```

```
## # A tibble: 62 × 9
## # Groups:   type [3]
##    song                     artist type    lvar    lave  lmax lfener lfreq range
##    <chr>                    <chr>  <chr>  <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>
##  1 V7                       Vival… Clas… 3.64e6   9.84  21450   101.  878. 0.933
##  2 V2                       Vival… Clas… 7.71e5  21.7    6936   104.  844. 0.895
##  3 Girl                     Eels   Rock  8.85e7   0.336 32744   112.  392. 0.894
##  4 Rock Hard Times          Eels   Rock  5.47e7   1.98  32759   109.  312. 0.691
##  5 Agony                    Eels   Rock  1.63e7  -0.141 30106   104.  312. 0.690
##  6 The Memory of Trees      Enya   New … 1.14e6 -10.6    9994   102.  155. 0.684
##  7 I Have A Dream           Abba   Rock  3.37e6 -71.7   14699   105.  305. 0.672
##  8 I Want to Hold Your Hand Beatl… Rock  6.13e7  -6.03  28502   112.  295. 0.646
##  9 The Winner               Abba   Rock  1.14e6 -67.8    8928   104.  278. 0.602
## 10 B4                       Beeth… Clas… 2.35e7  -0.941 32766   106.  529. 0.536
## # … with 52 more rows
```

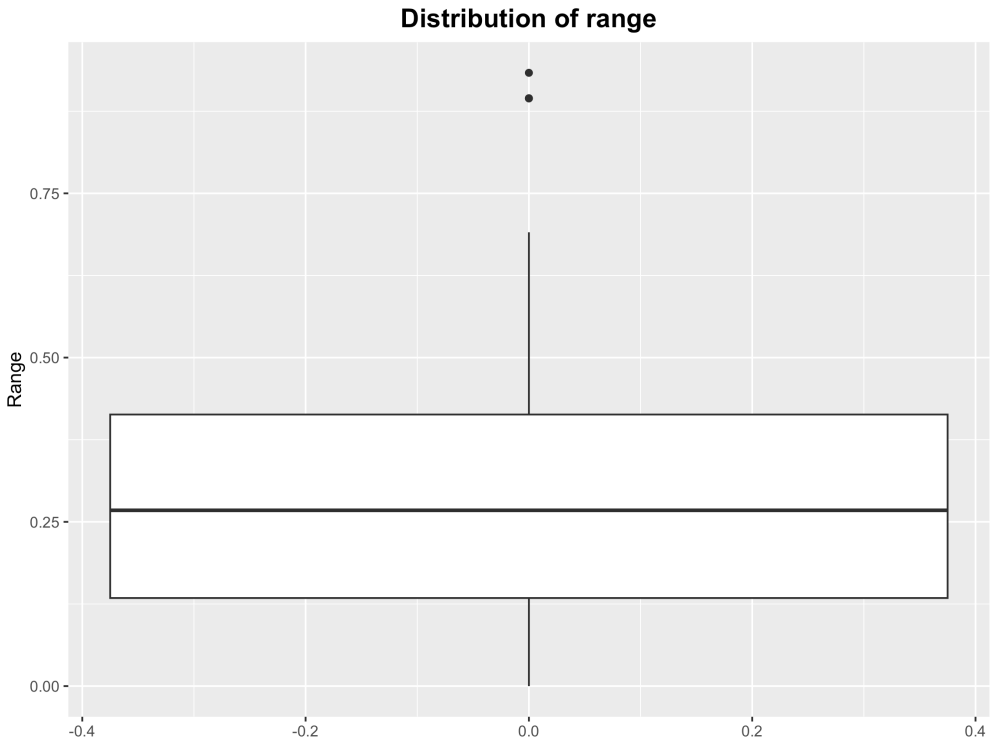
```
freqs_avg<-or_tab_music %>% group_by(type) %>% summarise(mean=mean(range)) %>% arra
nge(-mean)
freqs_var<-or_tab_music %>% group_by(type) %>% summarise(sd=sd(range)) %>% arrange(
sd)

write.csv(freqs_avg,'data/freqs_avg.csv')
write.csv(freqs_var,'data/freqs_var.csv')
```

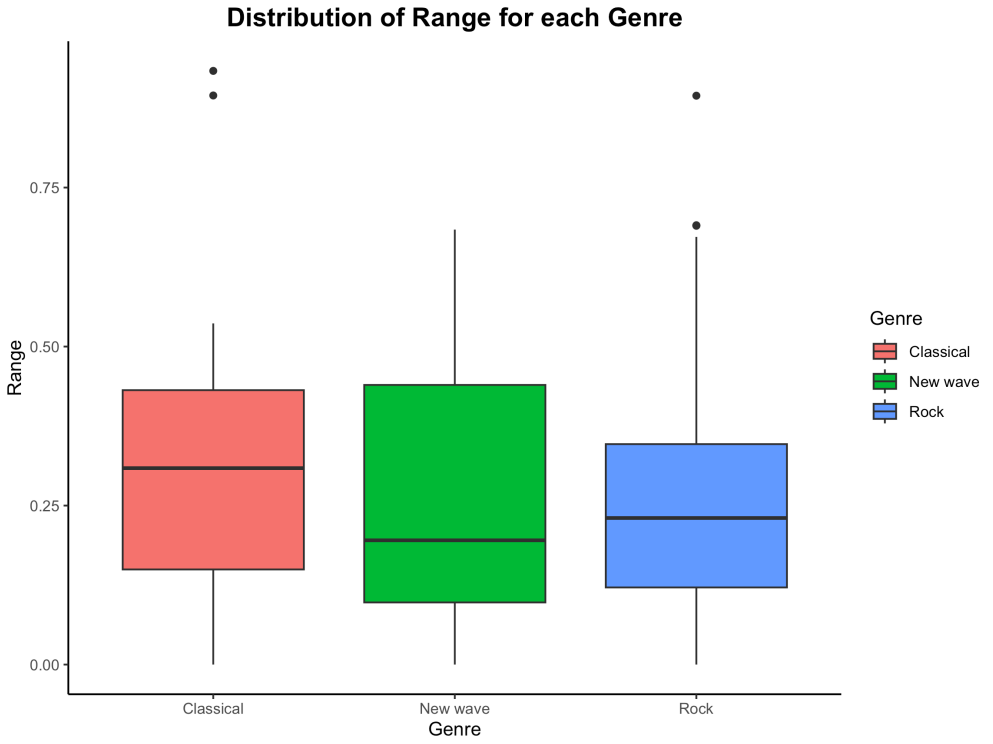
The genre type with the highest average frequency is **Classical** and with least variable is **Classical**

Question 12: Using the *tab_music_range* data object, display in a boxplot the frequency range of the music library by genre (2pts). Then, using boxplots display the frequencies (`lfreq`) by genre type (1pt). State which genre type has the highest and lowest original frequency range? (2pts).

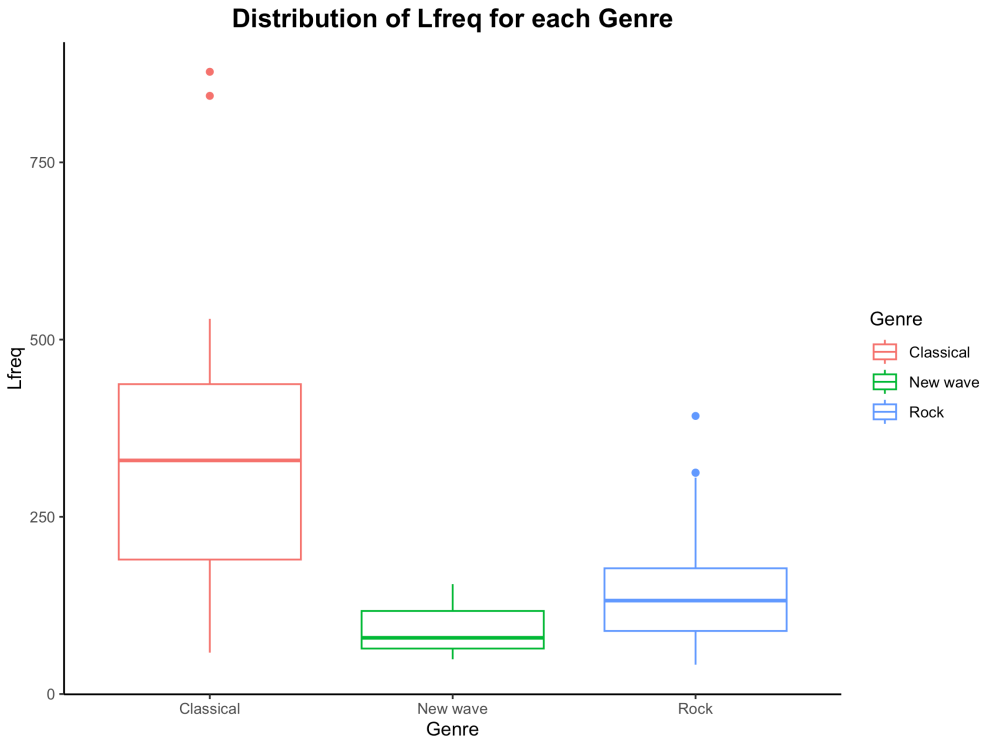
```
tab_music_range %>% ggplot(aes(y=range))+geom_boxplot() + ylab('Range') +  ggtitle(
'Distribution of range') + theme(plot.title = element_text(hjust = 0.5,size=15,face
='bold'))
```




```
tab_music_range %>% ggplot(aes(x=type,y=range,fill=type)) + geom_boxplot() + xlab('
Genre') + ylab('Range') + labs(fill='Genre') + ggtitle("Distribution of Range for e
ach Genre") +theme_classic() + theme(plot.title = element_text(hjust = 0.5,size=15,
face='bold'))
```




```
tab_music_range %>% ggplot(aes(x=type,y=lfreq,color=type))+geom_boxplot() + xlab('G
enre') + ylab('Lfreq') + labs(color='Genre') + ggtitle("Distribution of Lfreq for e
ach Genre") + theme_classic() + theme(plot.title = element_text(hjust = 0.5,size=15
,face='bold'))
```




```
original_lfreq_range <- tab_music_range %>%
  dplyr:: group_by(type) %>%
  mutate(lfreq_range = max(lfreq)-min(lfreq)) %>%
  arrange(-lfreq_range)

write.csv(original_lfreq_range,'data/original_lfreq_range.csv')
```

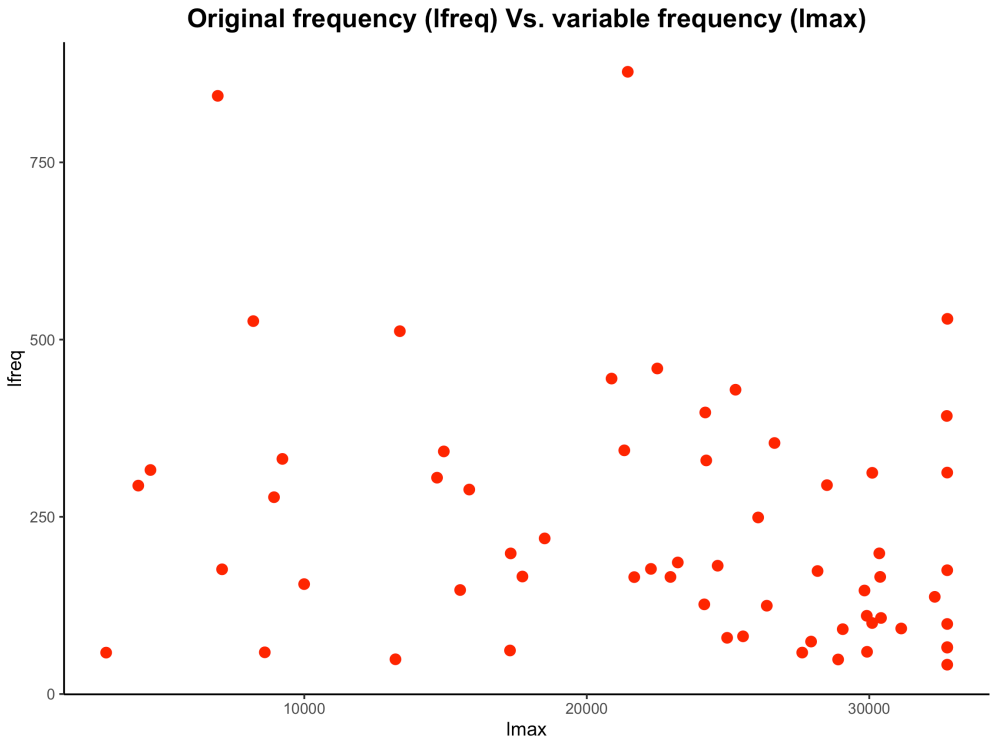
The genre type with the highest original frequency range is **Classical** and with the smallest is **New wave**.

Question 13: In one or two sentences, explain why the differences between the two sets of box plots (relative and original frequencies) is not unexpected. **Hint: Think about what the relative frequency transformation does to the data** (2pt)

Due to the mathematical transformation done on the original frequency **“lfreq”** by forming **“range”** parameter, the y-values of range in the boxplots have now compressed to values between 0 and 1. There is no longer a notable difference between the relative frequency values for Classical music when compared to New Wave and Rock music.This is due to the reason that the original frequency (lfreq) values have now been scaled appropriately according to the maximum and minimum frequency values for each individual genre, thereby compressing every value of **“lfreq”** to values within 0 and 1 irrespective of the genre and bringing them all to one common scale. This helps us understand the relatively distribution of the frequencies for each genre side by side through the boxplot.

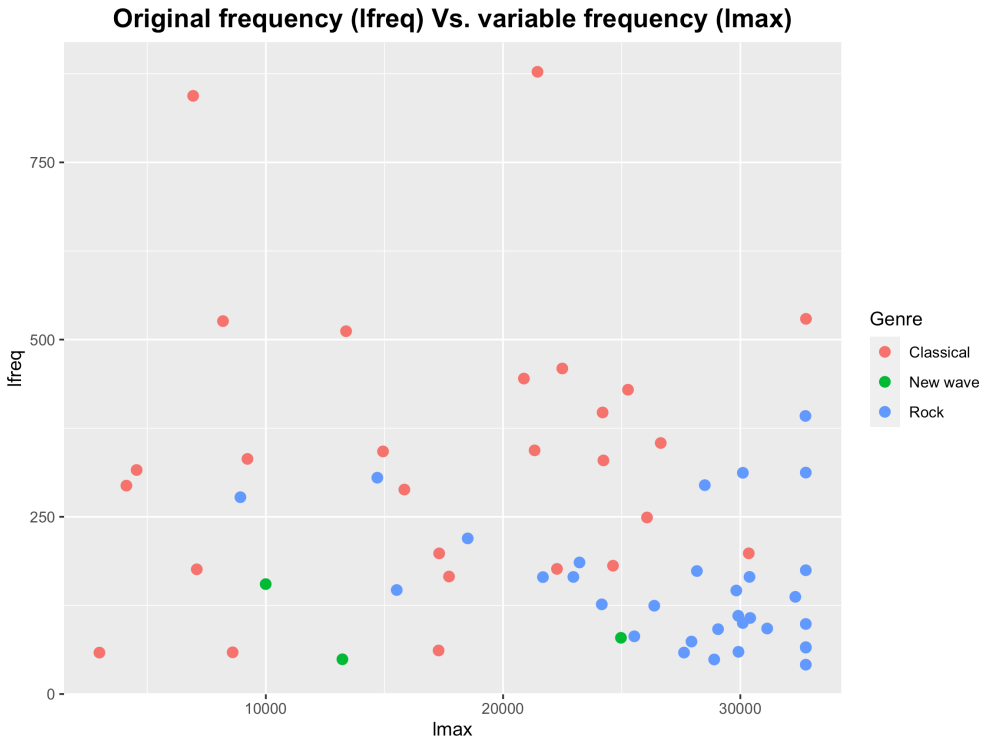
Question 14: Using the *tab_music_range* data object, plot the relationship between the orginal frequency range (`lfreq`) and the variable `lmax` (on the x-axis display `lmax` and on the y-axis `lfreq`).(2pts).

```
tab_music_range %>% ggplot(aes(y=lfreq,x=lmax)) + geom_point(size=2.5,color='red')
+ ggtitle("Original frequency (lfreq) Vs. variable frequency (lmax)") + theme_class
ic() + theme(plot.title = element_text(hjust = 0.5,size=15,face='bold'))
```



Question 15: Is the relationship between the orginal frequency range (`lfreq`) and the variable `lmax` constant across genre types? **Hint:** use color inside the asthetics. (2pts)

```
tab_music_range %>% ggplot(aes(x=lmax,y=lfreq,color=type))+geom_point(size=2.5) + g
gtitle("Original frequency (lfreq) Vs. variable frequency (lmax)") + labs(color='Ge
nre') + theme(plot.title = element_text(hjust = 0.5,size=15,face='bold'))
```



Question 16: In two sentences, what do you observe from the figure in Question 15 (2pt)?

There appears to be a stronger relation between the **lfreq** and **lmax** values of **Rock** music as majority of the scatter points are observed to be concentrated in the bottom right section of the scatter plot having **lmax** values greater than 20000 and **lfreq** values less than 250.

However, the same cannot be inferred for **Classical** music as there seems to be **no clear relation** between the lfreq and lmax values since the scatter points are observed to be scattered across the plot. The number of observations for **New Wave** music are **insufficient** for the relation to be interpreted.