Alright, seems like this is developing into a blog where I am increasingly investigating my own music listening habits.  
Recently, I’ve come across the [analyzelastfm](https://towardsdatascience.com/music-listener-statistics-last-fms-last-year-as-an-r-package-c21ac7eee102) package by Sebastian Wolf. I used it to download my complete listening history from Last.FM for the last ten years. That’s a complete dataset from 2009 to 2018 with exactly 65,356 “scrobbles” (which is the word Last.FM uses to describe one instance of a playback of a song).

**Getting scrobble data**

I’m pretty sure this can be done more efficiently, but this is how I’m getting the data. Be advised that this takes quite a long time.

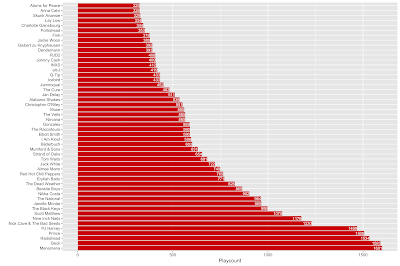
install.packages(“devtools”)  
devtools::install\_github(“zappingseb/analyze\_last\_fm”)  
library(analyzelastfm)  
library(lubridate)  
library(ggplot2)  
library(gridExtra)  
library(grid)  
library(ggrepel)  
library(scales)  
  
lkey <- “< Last.FM API key goes here >”  
  
data18 <- UserData$new(“< Last.FM user name >”, lkey, year = 2018)  
data17 <- UserData$new(“< Last.FM user name >“, lkey, year = 2017)  
data16 <- UserData$new(“< Last.FM user name >“, lkey, year = 2016)  
data15 <- UserData$new(“< Last.FM user name >“, lkey, year = 2015)  
data14 <- UserData$new(“< Last.FM user name >“, lkey, year = 2014)  
data13 <- UserData$new(“< Last.FM user name >“, lkey, year = 2013)  
data12 <- UserData$new(“< Last.FM user name >“, lkey, year = 2012)  
data11 <- UserData$new(“< Last.FM user name >“, lkey, year = 2011)  
data10 <- UserData$new(“< Last.FM user name >“, lkey, year = 2010)  
data09 <- UserData$new(“< Last.FM user name >“, lkey, year = 2009)  
  
scrobs <- rbind(data18$data\_table,  
                data17$data\_table,  
                data16$data\_table,  
                data15$data\_table,  
                data14$data\_table,  
                data13$data\_table,  
                data12$data\_table,  
                data11$data\_table,  
                data10$data\_table,  
                data09$data\_table)  
  
Before moving on, let’s parse the timestamp and order the dataset by it.  
  
scrobs$timestamp <- parse\_date\_time(scrobs$datetext, “dmYHM”)  
  
scrobs <- scrobs[order(scrobs$timestamp),]

rownames(scrobs) <- 1:nrow(scrobs)

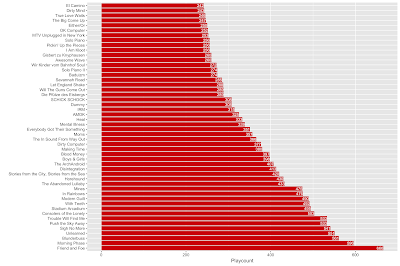
**Artist and album charts**

I have a dataframe “scrobs” with one scrobble per row now. Let’s do the obvious thing and plot 10-year overall charts of artists and albums – NB: I’m using Last.FM’s brand color here 

top.artists <- as.data.frame(sort(table(scrobs$artist), decreasing = T)[1:50])  
ggplot(top.artists, aes(x = Var1, y = Freq)) +  
  geom\_bar(stat = “identity”, fill = “#d51007”) +  
  coord\_flip() +  
  geom\_text(aes(label = Freq), hjust = “right”, size = 3, col = “white”) +  
  labs(x = “Playcount”, y = “”)  
  
top.albums <- as.data.frame(sort(table(scrobs$album), decreasing = T)[1:54])  
ggplot(top.albums, aes(x = Var1, y = Freq)) +  
  geom\_bar(stat = “identity”, fill = “#d51007”) +  
  coord\_flip() +  
  geom\_text(aes(label = Freq), hjust = “right”, size = 3, col = “white”) +  
  labs(y = “Playcount”, x = “”)

[](https://i1.wp.com/4.bp.blogspot.com/-dXpt2ynNGL8/XDjbb07xltI/AAAAAAAADcY/Jz0gX2rIlmQxlEdfYZzmGTnXZHiiQPTBwCLcBGAs/s1600/top_artists.png?ssl=1)

Top 50 artists

[](https://i2.wp.com/3.bp.blogspot.com/-aZN8d_TF-f4/XDjbb22tfJI/AAAAAAAADcU/7gtyr8rHlxwR6-IpSIzOKcMKhPQh_AhdwCEwYBhgL/s1600/top_albums.png?ssl=1)

Top 50 albums

I wonder how many artists you could name from my top 50 albums. It’s quite interesting to see how I obviously have a group of top 5 artists but only one album that stands out at the top of the charts (seriously, “Friend and Foe” by Menomena is great art!).

**Artists and albums through time**

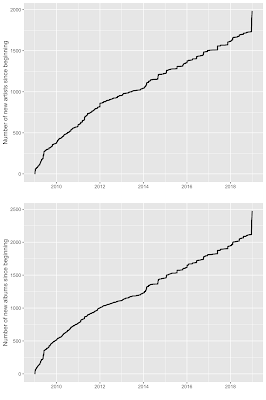
Next, I wanted to see how the total number of artists I played during all this time developed over all these years. So, what I want to do is get a cumulative view of the number of artists (and albums) over time. Whenever a new artist or album appears in the list of scrobbles, the counter is increased by 1. I did it this way (again, I think it could be done more efficiently, but this works and is quite transparent).

# New artist annotation  
  
scrobs$new.artist <- F  
for (i in 1:nrow(scrobs)) {  
  if (i %% 1000 == 0) cat(i, “\n”)  
  cur.row <- scrobs[i,]  
  prev.rows <- scrobs[1:(i-1),]  
  if (!(cur.row$artist %in% prev.rows$artist)) scrobs[i, “new.artist”] <- T   
}  
scrobs$new.artists.till.now <- cumsum(scrobs$new.artist)  
  
# New album annotation  
  
scrobs$new.album <- F  
for (i in 1:nrow(scrobs)) {  
  if (i %% 1000 == 0) cat(i, “\n”)  
  cur.row <- scrobs[i,]  
  prev.rows <- scrobs[1:(i-1),]  
  if (!(cur.row$album %in% prev.rows$album)) scrobs[i, “new.album”] <- T   
}  
scrobs$new.albums.till.now <- cumsum(scrobs$new.album)

Now that we have the information available in columns “new.artists.till.now” and “new.albums.till.now”, let’s plot them.

p1 <- ggplot(scrobs, aes(x = timestamp, y = new.artists.till.now)) +  
  geom\_line(size = 1) +  
  labs(x = “”, y = “Number of new artists since beginning”)  
  
p2 <- ggplot(scrobs, aes(x = timestamp, y = new.albums.till.now)) +  
  geom\_line(size = 1) +  
  labs(x = “”, y = “Number of new albums since beginning”)  
  
p3 <- grid.arrange(p1, p2)

This is what we get (as always: please click on the figure to see a larger version).

[](https://i2.wp.com/1.bp.blogspot.com/-ogYl_4W5n7s/XDNlCDOIXUI/AAAAAAAADb8/XYIW3jx_09Y3iuvYfd4fYQTwZs1tXFu0gCLcBGAs/s1600/artists_albums_cumulated.png?ssl=1)

It’s obvious that the lines look quite similar. New artists coincide with new albums. Then, there are a two other things worth mentioning:

* After the beginning of 2014, the curve seems to get “steppier”. Each sharp upward movement means that there have been a number of new artists in a shorter period of time. My suspicion is that, after 2013, I had increased phases where I was looking for new artists purposefully.
* Obviously, at the end of 2018, there has been a sharp increase in the number of artists. This has to do something with me getting a Spotify account. I’m curious how this develops in the future.

**Getting tag information**

Now, I want to add some information to each scrobble: the top 5 tags given to them by the community on Last.FM. I am coming back to this later. This procedure takes quite long because it involves lots of calls to the Last.FM API – just in case you want to do this with your own data.

In case you are wondering why I am looking up the tags for the “unique.tracks” object: It’s a lot faster, because the tags for one track are always the same. Whenever a track repeats within “scrobs”, the API would give the exact same result. So, I am using “unique.tracks” which is about a sixth of the size of “scrobs” (meaning that, on average, I have been listening to each track 6 times – of course, this distribution is heavily skewed, so the mean value does not tell us much here).

library(RLastFM)

unique.tracks <- unique(scrobs[, c(“track”, “artist”)])

counter <- 0

unique.tracks$track.top5.tags <- apply(unique.tracks, 1, FUN = function (row) {

  counter <<- counter + 1

  if (counter %% 500 == 0) cat(counter, “\n”)

  Sys.sleep(.1)

  paste(track.getTopTags(track = row[“track”],

                         artist = row[“artist”])$tag[1:5], collapse = “, “)

})

unique.tracks$merge.col <- paste(unique.tracks$track, unique.tracks$artist, sep = “\_\_\_”)

scrobs$merge.col <- paste(scrobs$track, scrobs$artist, sep = “\_\_\_”)

scrobs <- merge(scrobs, unique.tracks[,c(“merge.col”, “track.top5.tags”)], by = “merge.col”, all.x = T, all.y = F)

scrobs <- scrobs[order(scrobs$timestamp),]

scrobs <- scrobs[,-1] # getting rid of merge.col

Now for some clean-up of these tags. While we’re at lowering all the tags, translating some of them (different forms of “deutsch” into “german”) and unifying the different varieties of “hip-hop”/”hip hop”, and “hophop”,  I am also putting them all into a vector (“all.tags”).

all.tags <- list()

counter <- 0

scrobs$track.top5.tags <- sapply(scrobs$track.top5.tags, USE.NAMES = F, FUN = function (x) {

  counter <<- counter + 1

  if (counter %% 20000 == 0) cat(counter, “\n”)

  tags.i <- strsplit(x, “, “, fixed = T)[[1]]

  tags.i <- tolower(tags.i)

  tags.i <- gsub(“hiphop”, “hip-hop”, tags.i, fixed = T)

  tags.i <- gsub(“hip hop”, “hip-hop”, tags.i, fixed = T)

  tags.i <- gsub(“deutscher”, “german”, tags.i, fixed = T)

  tags.i <- gsub(“deutsch”, “german”, tags.i, fixed = T)

  tags.i <- gsub(“deutsche”, “german”, tags.i, fixed = T)

  all.tags[[length(all.tags) + 1]] <<- tags.i

  paste(tags.i, collapse = “, “)

})

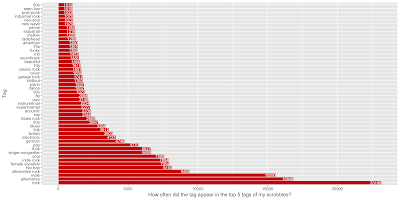
rm(counter)

all.tags <- do.call(“c”, all.tags)

**Tag chart**

With this, we can already plot a rough overview over the top 50 tags in the top 5 tags (that sounds a bit complicated) of my scrobbles in 10 years.

tag.tab <- as.data.frame(sort(table(all.tags), decreasing = T)[1:50])  
tag.tab <- tag.tab[!(tag.tab$all.tags %in% c(“null”, “na”)),]  
ggplot(tag.tab, aes(x = all.tags, y = Freq)) +  
  geom\_bar(stat = “identity”, fill = “#d51007”) +  
  labs(x = “Tag”, y = “How often did the tag appear in the top 5 tags of my scrobbles?”) +  
  geom\_text(aes(label = Freq), hjust = “right”, size = 3, col = “white”) +  
  
  coord\_flip()

[](https://i2.wp.com/4.bp.blogspot.com/-nwmlopPSJXw/XDjdUZyeCtI/AAAAAAAADco/nOGbMuRcNe8X5E1OYRqHSCperqiIeVAKQCLcBGAs/s1600/tag_count.png?ssl=1)

There are some patterns here: “rock” is by far the most frequent tag. However, I guess this tag is one of the most general on this list. There is some “hip-hop” going on and “female vocalists” are also quite prominent in the most frequent tags. Also, “funk” and “soul” appears in the higher ranks. Obviously, some people tag songs with “radiohead” (songs by Radiohead, I reckon). Well, some people…

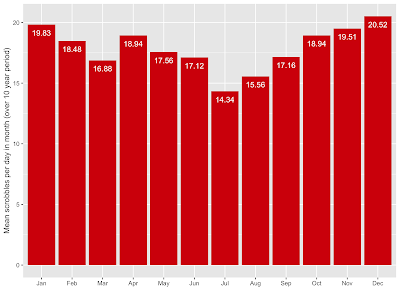
Of course, there’s a lot more one can do with these tags. But I wanted to play around with all-time scrobble visualizations first. For that, we need to code the year and month of each scrobble (we just need to get that out of the timestamp).

scrobs$month <- substr(scrobs$datetext, 4, 6)  
scrobs$month <- factor(scrobs$month, levels = c(“Jan”, “Feb”, “Mar”, “Apr”, “May”, “Jun”,  
                                                “Jul”, “Aug”, “Sep”, “Oct”, “Nov”, “Dec”))  
scrobs$month.year <- substr(scrobs$datetext, 4, 11)  
month.year.order <- paste(c(“Jan”, “Feb”, “Mar”, “Apr”, “May”, “Jun”, “Jul”, “Aug”, “Sep”, “Oct”, “Nov”, “Dec”),  
                          rep(2009:2018, each = 12))  
scrobs$month.year <- factor(scrobs$month.year, levels = month.year.order)

**Number of scrobbles per month, year, and day**

Let’s get a simple overview over the number of scrobbles per months. For this, I want to take into account how many days the months have. If there’s more time, you can listen to more music, right? For this, I am dividing the total number of scrobbles in each month by the number of days within this month multiplied by 10 (because there are ten years in the dataset). February is a little complicated because there were two leap years (in German, it’s “switch years” – Schaltjahre), 2012 and 2016. So, I am adding 2 days for those years (not that it would make much difference in the plots…).  
  
month.tab <- as.data.frame(table(scrobs$month))  
month.tab$norm.scrobs <- month.tab$Freq / c(31\*10, 28\*10+2, 31\*10, 30\*10, 31\*10, 30\*10, 31\*10, 31\*10, 30\*10, 31\*10, 30\*10, 31\*10)  
  
ggplot(month.tab, aes(x = Var1, y = norm.scrobs)) +  
  geom\_bar(stat = “identity”, fill = “#d51007”) +  
  geom\_text(aes(label = round(norm.scrobs, 2)), vjust = 2, col = “white”) +

  labs(x = “”, y = “Mean scrobbles per day in month (over 10 year period)”)

[](https://i2.wp.com/1.bp.blogspot.com/-hNL1fsvTz4Q/XDjh-rzyfYI/AAAAAAAADc0/zSjk5_9d1E0tzdngLfTOmj42PA90dkUigCLcBGAs/s1600/scrobs_per_month.png?ssl=1)

As expected, months in the summer (especially July) are months when I didn’t listen to music so frequently, mainly because I didn’t take my music collection with me on summer vacations. Also, the winter months are the months when I stay at home more. And there: music.

Of course, we can also look at the scrobbles in “instances” of months and not aggregated over years. We could do it this way (and still distinguish the years by bar color). I did not do the “length of month” normalization here.

ggplot(scrobs, aes(x = month.year, fill = factor(year))) +

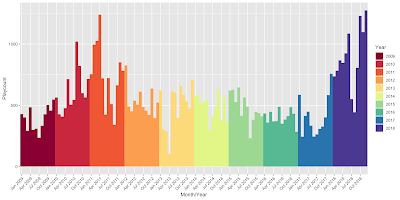
  geom\_bar(width = 1) +

  labs(x = “Month/Year”, y = “Playcount”, fill = “Year”) +

  scale\_fill\_manual(values = brewer.pal(10, “Spectral”)) +

  scale\_x\_discrete(breaks = levels(scrobs$month.year)[seq(1, nlevels(scrobs$month.year), 3)]) +

  theme(axis.text.x = element\_text(angle = 45, hjust = 1))

[](https://i0.wp.com/4.bp.blogspot.com/-1fS_x58MrJs/XDji9PtCZ_I/AAAAAAAADc8/yGPMXhNFCfU3B4gzDg2e5l0RBYLnSU3gwCLcBGAs/s1600/scrobs_per_month_year_bars.png?ssl=1)

We could even do this by day and add a smoother line to it.

plot.df <- as.data.frame(table(scrobs$date))

names(plot.df) <- c(“Date”, “Playcount”)

plot.df$year <- year(plot.df$Date)

plot.df$month <- month(plot.df$Date)

plot.df$month.year <- paste(plot.df$year, plot.df$month, sep = “-“)

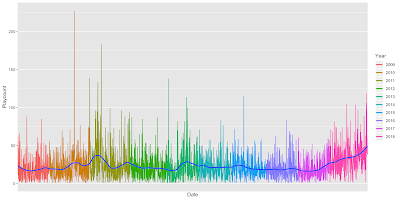
ggplot(plot.df, aes(x = Date, y = Playcount, col = factor(year), group = 1)) +

  geom\_line() +

  labs(col = “Year”) +

  scale\_x\_discrete(breaks = NULL) +

  geom\_smooth(method = “loess”, span = .1, se = F)

[](https://i1.wp.com/4.bp.blogspot.com/-MngxvpoWocc/XDjjcls6J-I/AAAAAAAADdE/sz-aig2UMzokq_T_HkzBu2aK-rXiBeB3gCLcBGAs/s1600/scrobs_per_day.png?ssl=1)

Well OK, this is mighty colorful but not entirely useful, right? The only thing that’s quite salient is the increasing number of scrobbles at the very end of 2018. We also see this in a violin plot of the years. I am showing you this in two ways: without facets with the year on the x-axis and faceted by years.

ggplot(plot.df, aes(x = factor(year), y = Playcount, fill = factor(year))) +

  geom\_violin(draw\_quantiles = c(1/4, 1/2, 3/4), col = “#00000000”) +

  geom\_jitter(height = 0, width = .2, alpha = .3) +

  geom\_point(inherit.aes = F, aes(x = factor(year), y = Playcount), stat = “summary”,

             color = “white”, size = 3) +

  scale\_fill\_manual(values = brewer.pal(10, “Spectral”)) +

  labs(x = “”) + guides(fill = F)

ggplot(plot.df, aes(x= 1, y = Playcount, fill = factor(year))) +

  geom\_violin(draw\_quantiles = c(1/4, 1/2, 3/4), col = “#00000000”) +

  geom\_jitter(height = 0, width = .2, alpha = .3) +

  geom\_point(inherit.aes = F, aes(x = 1, y = Playcount), stat = “summary”,

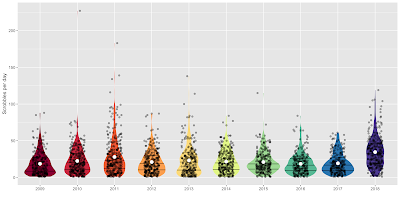
             color = “white”, size = 3) +

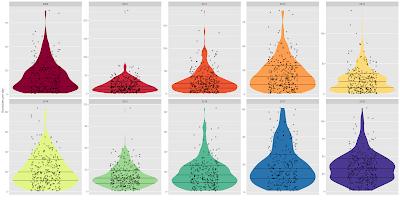
  scale\_fill\_manual(values = brewer.pal(10, “Spectral”)) +

  scale\_x\_continuous(breaks = NULL) +

  labs(x = “”) + guides(fill = F) +

  facet\_wrap(~ factor(year), scales = “free”, ncol = 5)

[](https://i0.wp.com/1.bp.blogspot.com/-9DL2YUmA59s/XDmrn3ig5FI/AAAAAAAADdo/TQ0mjnL6-Z4Nk1BX2VdcbBrsNUdUOdhhgCLcBGAs/s1600/scrobs_per_year_vio.png?ssl=1)

[](https://i0.wp.com/2.bp.blogspot.com/-AhEq1gfeXS8/XDmrtErZhxI/AAAAAAAADds/jwahCHDDrDYNeLLjqN5enP5YNDhETdazgCLcBGAs/s1600/scrobs_per_year_vio_faceted.png?ssl=1)

Aren’t these gorgeous? I really like violin plots. The horizontal lines denote quartiles, the white dot represents the mean scrobbles per day for each year. Each violin spans from the minimum to the maximum value and the outlines of the violins represent the density (“distribution”) of data points. Each day is represented by a semi-transparent black dot.

For the faceted one I allowed the y-axis to vary freely because there is one outlier in 2010 that “crushes” all the violins to the baseline. What’s going on there?

I can tell you: It’s August 21st, 2010 with 227 scrobbles. With an hypothetical mean track length of 4 minutes, that’s 227\*4/60 = 15.13 hours of music which does not sound to plausible. Here’s what happened: I’ve been on summer holidays and had my iPod nano with me (it still lies in a drawer here, still working, but the smartphone took his job). Back in 2010, I forgot the iPod in the hotel and when I returned home, I asked the staff to look for it and send it to me via mail. They found it and sent it on its way to Germany. Of course, they didn’t switch it off before mailing it – meaning that at some point during its journey (presumably very early on) – the “shake-to-play-random-songs” trigger went off. This caused the iPod to play songs till the battery went dead. The battery obviously lasted quite long.

Due to the varying y-axes in the faceted case, I left out the white dots denoting the mean number of scrobbles per day in the non-faceted case – this would just be utterly confusing when comparing the height of a white dot in one facet with another one.

**Artist dispersion**

One last thing before I will wrap this post up: I was interested in the dispersion of artists. While frequency captures the overall number of scrobbles per artist, dispersion tries to capture how the data points are spread out over the whole dataset. I have played around with a few more sophisticated dispersion measures (e.g., Gries DP from the realm of linguistic research). However, the data at hand does not seem to be fit for this kind of data because I get really low dispersion measures. That is why I opted for a very basic measure: The ratio of days, each artist has been played. An easy example: Suppose, we had 100 days in the dataset and artist x was played at 35 of these days (one scrobble is enough, which could be a weakness of this measure), artist x gets a ratio value of 35 / 100 = .35 or 35%. Let’s do this with all artists in the top 30 and the whole range of days:

topx <- sort(table(scrobs$artist), decreasing = T)[1:30]

n.days <- length(unique(scrobs$date))

day.ratio.topx <- sapply(names(topx), USE.NAMES = T, FUN = function (art) {

  n.art.days <- length(unique(scrobs[scrobs$artist %in% art, “date”]))

  n.art.days / n.days

})

normed.day.ratio.topx <- day.ratio.topx / topx # We’ll use this later

We can do the same with months.

n.months <- length(unique(scrobs$month.year))

month.ratio.topx <- sapply(names(topx), USE.NAMES = T, FUN = function (art) {

  n.art.mnths <- length(unique(scrobs[scrobs$artist %in% art, “month.year”]))

  n.art.mnths / n.months

})

normed.month.ratio.topx <- month.ratio.topx / topx # For later

I am putting all these values in a dataframe and plot everything for the top 30 artists. I am putting the total playcount on the x-axis and the new measures on the y-axis. I am putting in some smoother lines to have a quick look at the relationship between the two variables.

art.df <- data.frame(topx)

art.df$day.ratio <- day.ratio.topx

art.df$normed.day.ratio <- normed.day.ratio.topx

art.df$month.ratio <- month.ratio.topx

art.df$normed.month.ratio <- normed.month.ratio.topx

ggplot(art.df, aes(x = Freq, y = day.ratio, label = Var1)) +

  geom\_point() +

  geom\_label\_repel() +

  geom\_line(stat = “smooth”, method = loess, alpha = .3, size = 2, col = “blue”) +

  labs(x = “Playcount”, y = “Artist played on % of all days”) +

  scale\_y\_continuous(labels = percent)

ggplot(art.df, aes(x = Freq, y = month.ratio, label = Var1)) +

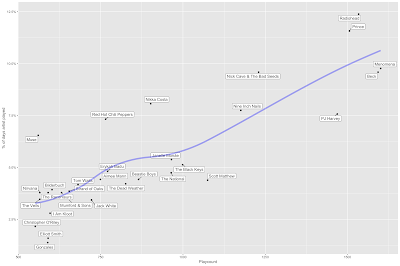
  geom\_point() +

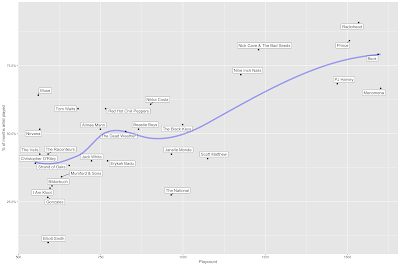
  geom\_label\_repel() +

  geom\_line(stat = “smooth”, method = loess, alpha = .3, size = 2, col = “blue”) +

  labs(x = “Playcount”, y = “Artist played in % of all months”) +

  scale\_y\_continuous(labels = percent)

[](https://i2.wp.com/2.bp.blogspot.com/-bK8m0-J9HGU/XDnMrHbv0TI/AAAAAAAADd8/pAbD2aGU2ac1hSlwhT5NweHlwrvRX142gCLcBGAs/s1600/ratio_days.png?ssl=1)

[](https://i2.wp.com/4.bp.blogspot.com/-YipKi4lQKDo/XDnMvkKQ9mI/AAAAAAAADeA/0el6OoqJXgkV44P_24SKe_tMupkgrhjYQCLcBGAs/s1600/ratio_months.png?ssl=1)

This is quite interesting: while Menomena are my top artists, Radiohead is played in more months, the same holds for Prince. That means that my scrobbles of Radiohead are more evenly “spread out” over the whole range of the data. However, there is one problem with these plots: it is no surprise that there seems to be a positive relationship between the variables with total playcount. If you listen to an artist more, the probability rises that it is played in more months, right? We can correct for this effect by norming the measures by the total playcount. The picture looks a bit different then. Also, the relationships are weakened. I already did this with the variables normed.day.ratio and normed.month.ratio. Let’s plot them.

ggplot(art.df, aes(x = Freq, y = normed.day.ratio, label = Var1)) +

  geom\_point() +

  geom\_label\_repel() +

  geom\_line(stat = “smooth”, method = loess, alpha = .3, size = 2, col = “blue”) +

  labs(x = “Playcount”, y = “Artist played on % of all days / Playcount”) +

  scale\_y\_continuous()

ggplot(art.df, aes(x = Freq, y = normed.month.ratio, label = Var1)) +

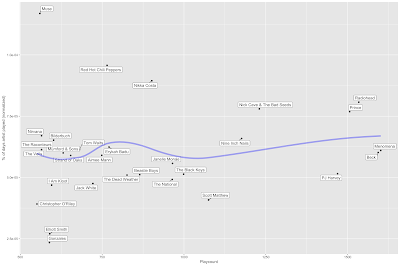
  geom\_point() +

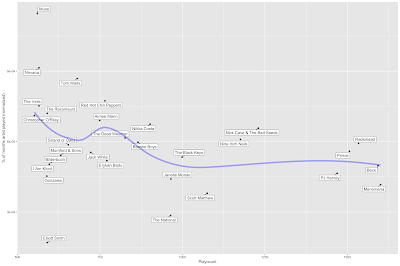
  geom\_label\_repel() +

  geom\_line(stat = “smooth”, method = loess, alpha = .3, size = 2, col = “blue”) +

  labs(x = “Playcount”, y = “Artist played in % of all months / Playcount”) +

  scale\_y\_continuous()

[](https://i1.wp.com/3.bp.blogspot.com/-08AuSbgR_B8/XDnOSsM7AMI/AAAAAAAADeQ/AaT-89T6PIog8eEUTBFekLNWSjU-AF4eACLcBGAs/s1600/ratio_days_normed.png?ssl=1)

[](https://i1.wp.com/3.bp.blogspot.com/-at33JiwdNCQ/XDnOXZ-K4CI/AAAAAAAADeU/FJgvseRSmzM_xF-CahaiG_qYzvijdeXGgCLcBGAs/s1600/ratio_months_normed.png?ssl=1)

Do you see what happened? All top 5 artists “get corrected” a lot. And especially Muse are gaining points on the normed ratio scale. That means: when I correct for the total number of plays, Muse is my top artist in terms of being played throughout all days and months. I have two tracks in mind that could have triggered this: “Panic Station” and “Supermassive Black Hole” – two tracks I come back to quite frequently. Maybe, I’ll go into more detail in a future post. This correction might be a little too strict and one could play around a lot more with this (e.g., with residualization techniques), but it gives a good first impression, I think.

**Wrapping up**

That’s it for this post. I will come back to this dataset because there is a lot more to analyze and visualize here. But for now, let’s wrap it up:

* We got 10 years of scrobble data from Last.FM.
* We made simple artist and album charts.
* We had a look at the cumulative development of artists and albums in time.
* We got tag information on all tracks from Last.FM (if there was information available).
* We created a tag chart.
* We had a look at scrobbles per months, days and years.
* We analysed the dispersion of artists over the whole period of 10 years and tried a simple norming procedure to correct for the overall playcount.