Visualizing Artist Recommendation

Romy Ratolojanahary
M2 Data Science Maths

Mélanie Salignat
M2 Data Science Maths

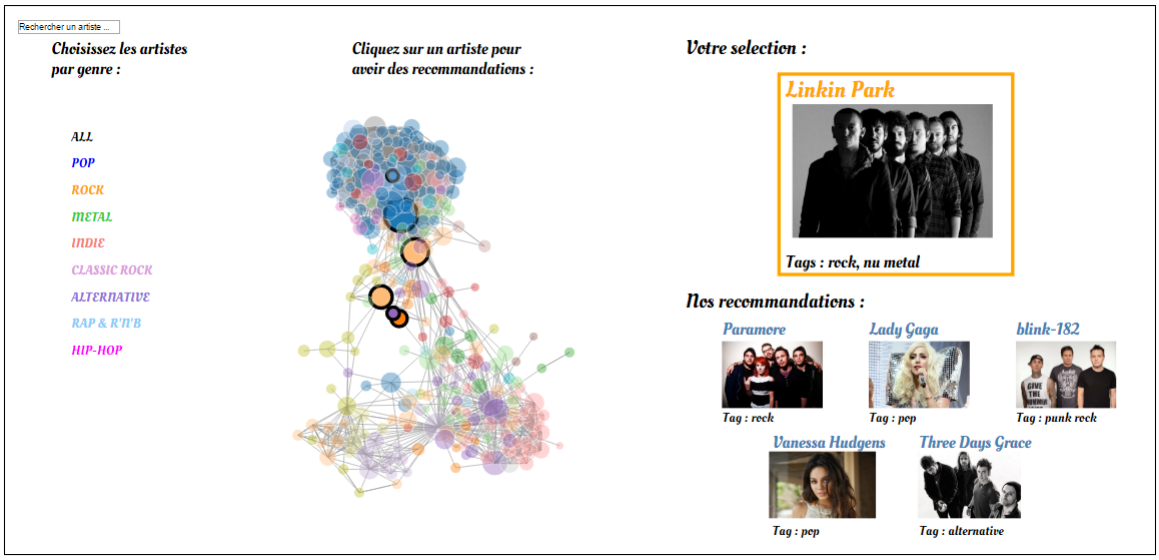

Figure 1: Artist recommendation

Abstract

We introduce a new technique for visualizing artist recommendation. Given an artist recommendation dataset, we represent it with a force-directed graph using d3.js, that the user can see entirely. To obtain recommendations, the user may click on a node, use a search bar or use the legend. We then discuss the effectiveness of the visualization. This project is as part of a Data Visualization course provided in Université Lyon 1, supervised by Nicolas Bonneel.

1 Introduction

When searching for one thing on the Internet, on Google for example, we get more than a billion results. How to choose which website to click on? Instinctively, one will click on the most highlighted, i.e. the first one.

More generally, the way the results of a query are represented has a considerable impact on their interpretation. It will tell the user whether a result is more important or relevant than another, and thus will affect their choice.

In this article, we focus on music recommendation, and particularly artist recommendation. It is an universal and fun subject, which everyone will be able to relate to.

How do people discover new music? When listening to the radio or old CDs is not enough anymore, Internet is the ideal choice. Music streaming applications like Spotify or Deezer all have their own recommendation system, and visualize them simply: as an ordered list (with images).

Some websites are specifically dedicated to recommendation (see next part: Related Work). We tried them and concluded that:

1. All the visualizations look alike.
2. We prefer the ones with images rather than a simple list.

We have two primary goals: (1) design a new effective artist recommendation , (2) keep it playful for the user.

For this purpose, we will first retrieve and prepare the dataset using KNIME Analytics Platform. Then we will draw the graph representing all the artists and their recommendations. We will complete the visualization with traditional features such as a search bar, a legend and the list of the recommendations.

In this paper we will not discuss the quality of the recommendation algorithm but will only comment on the visuals.

2 Related Work

As said before, there are already many websites for artists recommendation. Each one uses a different visualization. To create our own visualization out of these recommendations, we took as example two websites: Gnoosic and TasteKid.

2.1 *Gnoosic*

On the first page, we are asked our 3 favourite artists. Then a series of pages are displayed, each one containing an artist and 3 questions:

- I like it
- I don't like it
- I don't know it

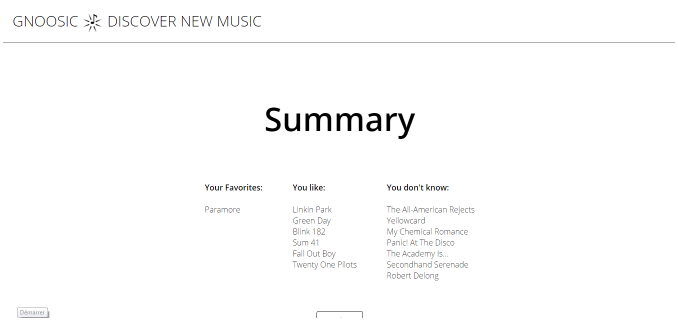

Figure 2: Visualization of Gnoosic website recommendations

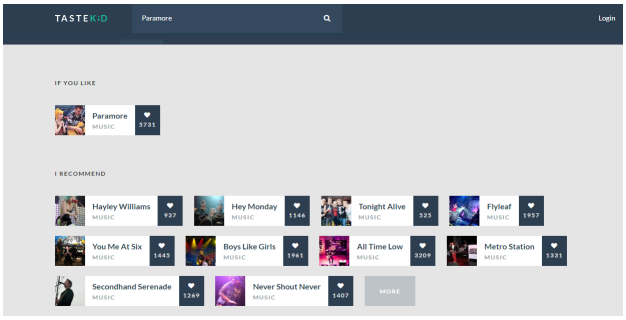

Figure 3: Visualization of TasteKid website recommendations

This site then summarizes our musical tastes by presenting the results as a list of artists arranged in several columns: those the user likes, does not like and does not know.

This representation is not very visual because it gives no information about the artist except their name: we do not know what kind of music they belong to, whether the artist is popular or not, nor what they look like.

2.2 *TasteKid*

This site gives Books, Movies, Shows, Authors and Games recommendation. It even has a "everything" section.

This site represents each recommendation in a rectangle containing a small image of the artist, as well as the number of people who like them. This time we have a picture of the artist but we still do not know the kind of music they do. This representation is nevertheless more visual than the first one: it makes users want to use their site rather than the first one.

Another interesting feature this site provides is the recommendation of any type of category from another. For example, we can type that we like the Author Jane Austen in the music section, and we will obtain a classical artist as a recommendation.

2.3 Our improvements

After analyzing and criticizing those two visualizations, we will try to improve them . We begin by representing all the artists and all the links between them in a graph, so that we can see how close two musicians are to each other.
We also represent on the graph the artists by musical genre, in order to easily access to all the artists "Pop" for example. We add, in the right, the top 5 artists in each category, with the picture of the artist, his name and his tag.

And finally, we represent the 5 best recommendations, in a rectangle, with an image of the artist as well as their musical genre.

3 Project Description

All the code was written with d3.js. A link is avalaible at the end of the article.

3.1 Recommendation Dataset

The dataset contains social networking, tagging, and music artist listening information from a set of 2K users from Last.fm online music system. It describes 17632 artists and was released in 2011. We use 4 files from that dataset:

- *artist.data*: name, picture url
- *tags.data*: tagID, tag
- *user-artists.data*: userID, artistID, weight
- *user-artists.data*: userID, artistID, tagID

One of the reasons why we chose this dataset is that it provides a picture url for each artist.

With KNIME, we join the different tables and use the node "Association Rule Learner" to generate the recommendations. It searches for the frequent set of artists usually liked by the same users. We only consider the most popular artists to lighten the output file.

Finally, we craft 2 csv files:

- *recommandations.csv*: Consequence, Antecedant, ItemSetSupport
- *artists-tags-count.csv*: artist, pictureurl, tags, count

We get 2554 recommendations for only 280 unique artists from the first file.

Recommendations are sorted by ItemSetSupport. It represents the number of users who listen to both artists.

Tags are stocked in a list and are sorted by frequency.

The users are the ones "tagging" the artists. This means these are not objective tags. We can find surprising ones such as *"excellent reason for crying"* or *"I was fighting with myself not to listen to it but I lost"*. Nevertheless, we noticed that the most used tags are genres, oftenly the correct ones. Consequently, we can use them in our genres legend.

3.2 Force-Directed Graph

We represent the two tables with a force-directed graph in d3.js. Each node is an artist, and there is a link between two nodes if one is recommended from the other. However, in the file *recommandations.csv*, this relation is directed. We omit this information for the representation, but will use it later when listing the recommendations.

The length of a link depends on the ItemSetSupport value. The larger the ItemSetSupport of two nodes is, the closer the nodes are.

The size of each node is the square root of its value *count*. Thereby the larger a node is, the more popular the artist is.

The node color depends on the artist's top tag. As the result was quite satisfying, we decided to stick to the default colors.

Figure 4: Nodes of different shape and color

3.3 Recommendation

In *recommandations.csv*, there are 141 consequences and 275 antecedants. In addition, the number of recommendations per antecedant is ranged from 1 to 29.

Consequently, if we would like all the nodes of our graph to have recommendations, we have to create a new recommendation calculation function.

For this project, we decide to set up the number of recommendations per artist to 5.

For each artist, we take the top 5 recommendations. If there are less than 5 recommended artists or if the artist is not in the Antecedant column, we browse the column Consequence and add the associated antecedants. If we still have less than 5 recommendations, we apply the recommendation function to the first element of our recommendation list, and it goes on.

3.4 Font-style

The police employed in the whole page is Oleo Script', Helvetica, sans-serif.

3.5 Tooltip

We add a tooltip so that we can display the images while hovering on a node. Infortunately, the picture urls included in our original dataset are no longer working. We had to add all the urls by hand. We tried to choose same-format images, and only contrained the height of the image in the tooltip so that it does not get deformed.

Figure 5: Tooltip

3.6 List of Recommendations

The recommandations are listed very simply.
A main highlighted orange square indicates the chosen artist, and below we place the images of the recommended artist in a natural order:

1 2 3

4 5

3.7 Legend

The legend consists of clickable text zones, labeled as different label genres. We chose 7 main music genres: Pop, Rock, Metal, Indie, Classic Rock, Alternative, Rap & R'N'B, Hip-Hop.
When we click on "Pop" for example, all the artists containing the tag "pop" in their top 5 tags are highlighted.

We obtain the legend color by picking the most frequent node color among those sharing the same tag.

3.8 Search Bar

Figure 6: Search bar

We include a search bar with suggestions to find the node easily in the graph.

Note: Infortunately, even if suggestions appear, we have to type the musician name entirely to make the search functional.

We let it in its default position, i.e top right, and reduce the number of displayed suggestions from 6 to 1 for a better visualization.

3.9 Interaction

When the user arrives on the page, he sees the colourful graph in the center of the page, a bar search at the top right, and a legend underneath.
The instructions are included on the page: click on the legend, on a node or use the search bar to get recommendations.

He immediately notices there are 3 main clusters: one for each generic music genre: Pop-rap-rnb, Rock and Metal.

It is also possible to zoom the graph in and out as much as wanted.

When the user hovers on a node, the name of the associated artist and the image appear in a tooltip. If he clicks on a node, the node and its recommendations are highlighted with a bold black contour, and the opacity of all the other nodes is reduced. In parallel, the top

Figure 7: Prominent color is orange.

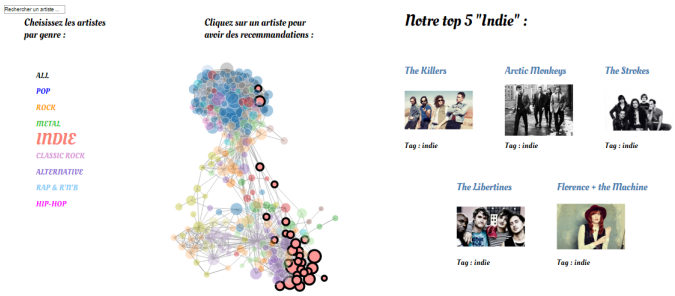

Figure 8: Top 5 Indie

5 recommendations are displayed on the right.

If the user wants a specific artist, he types their name in the search bar. If the artist is present in the database, suggestions will appear and the associated recommendations are shown.

In case the user does not have any particular artist in mind, but a favourite genre, he can click on the legend. When doing so, he will get our special recommendations: our personal top 5 artists for each genre (except for Metal, we just picked the 5 most listened).

The "ALL" category brings back the graph to its initial state, i.e with no nodes highlighted and the same opacity for all the nodes. Instead of the basic recommendations or top 5, the user will see our 5 favourite artists on the right.

4 Discussion

4.1 The benefits of our visualization

Our visualization, unlike the other music recommendation websites, (TasteKid and Gnoostic for example), is very colorful. It is more appealing to the eye of the users.

The interactive legend allows us to spot the interesting artists, those who have this particular musical register.

It is possible to see the links between the artists and to distinguish different clusters using the graph representation. This way, the user is able to judge the distance between two artists, and consequently judge the quality of the recommendation by himself.

We can also zoom in on the area we are interested in to better distinguish the artists from this cluster. This feature is as useful as fun, added to the dragabbility of the graph.

The interactivity of the graph is more visual thanks to the tooltip with the name of the artist as well as their picture while hovering on the node.

The fact that we associate an image with each artist or group makes our recommendations more visual. Users can have a first impression on the recommended artist.

The recommandations are obtained instantly, which is more satisfying for the user.

4.2 Things to improve

The biggest problem with the graph is that even with not many artists, the graph is still too dense. In the "Pop" cluster for example, the artists are very compressed.

The fact of having made a Force Directed Graph gives us Clusters easily but the problem is that, in the Cluster "Pop", we could have tried to separate these artists to make them more visible by increasing the length of the links and zoom automatically.

The search bar position should be changed and not impact the graph and legend's position. As for now, it moves them below.

As the user may notice, the suggestions made by the search bar have only one role: check if an artist is present in our database or not. The ideal would be to make the automatic completion while typing effective.

Another major improvement would be the selection of multiple elements: several genres in the legend for example with a ctrl + click.It would be possible to visualize both the "Rock" and "Pop" artists on the graph for example. We could also do the same for nodes, and compute a new recommendation function and have a new visualisation for the recommended artists.

To make the visualization even more fun, we could have made the images from the recommendation list clickable.

We represented the popularity of each artist by varying the size of each node in the graph. To make it even more visible, we could have added the information while displaying the recommendations.

5 Conclusion

Representing the dataset from which to recommend has one main advantage: the user has an exact idea of the quality of the recommendation, in addition to the classic ordered list.
The playful colors and the draggable / zoomable graph make it more enjoyable for the user.

Acknowledgments

The authors wish to thank N. Bonneel and all the fellow students who helped them.