D3.2 – Predictive analytics and recommendation framework v2

August 31st, 2019

Authors: Thomas Lidy (MMAP), Adrian Lecoutre (MMAP), Khalil Boulkenafet (MMAP), Manos Schinas (CERTH), Christos Koutlis (CERTH), Symeon Papadopoulos (CERTH)

Contributor/s: Vasiliki Gkatziaki (CERTH), Emmanouil Krasanakis (CERTH), Polychronis Charitidis (CERTH)

Deliverable Lead Beneficiary: MMAP

This project has been co-funded by the HORIZON 2020 Programme of the European Union. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use, which may be made of the information contained therein.

Deliverable number or supporting document title	D3.2 Predictive analytics and recommendation framework
Type	Report
Dissemination level	Public
Publication date	31-08-2019
Author(s)	Thomas Lidy (MMAP), Adrian Lecoutre (MMAP), Khalil Boulkenafet (MMAP), Manos Schinas (CERTH), Christos Koutlis (CERTH), Symeon Papadopoulos (CERTH)
Contributor(s)	Emmanouil Krasanakis (CERTH), Vasiliki Gkatziaki (CERTH), Polychronis Charitidis (CERTH)
Reviewer(s)	Rémi Mignot (IRCAM)
Keywords	Track popularity, artist popularity, music genre popularity, track recognition estimation, emerging artist discovery, popularity forecasting
Website	www.futurepulse.eu

CHANGE LOG

Version	Date	Description of change	Responsible
V0.1	25/06/2019	First deliverable draft version, table of contents	Thomas Lidy (MMAP)
V0.2	18/07/2019	Main contribution on track recognition estimation and artist predictive analysis	Christos Koutlis, Manos Schinas (CERTH)
V0.3	24/07/2019	Structure check	Thomas Lidy (MMAP)
V0.4	07/08/2019	Track popularity prediction structure	Adrian Lecoutre (MMAP), Khalil Boulkenafet (MMAP)
V0.5	12/08/2019	Track popularity estimation and prediction	Thomas Lidy (MMAP), Adrian Lecoutre (MMAP), Khalil Boulkenafet (MMAP)
V0.6	16/08/2019	Track popularity prediction experiments description	Thomas Lidy (MMAP), Adrian Lecoutre (MMAP), Khalil Boulkenafet (MMAP)

Funded by the European Commission

V0.7	20/08/2019	Track popularity prediction experiments finalization, Conclusions	Thomas Lidy (MMAP), Adrian Lecoutre (MMAP), Khalil Boulkenafet (MMAP)
V0.8	21/08/2019	Document review, Executive Summary, Relation to other WPs, Final conclusions, final edits	Thomas Lidy (MMAP), Manos Schinas (CERTH), Symeon Papadopoulos (CERTH)
V0.9	29/08/2019	Final formatting	Manos Schinas (CERTH)
V1.0	30/08/2019	Minor fixes, PDF generation	Christos Koutlis (CERTH), Manos Schinas (CERTH), Symeon Papadopoulos (CERTH)

Table of Contents

Funded by the European Commission

Funded by the
European Commission

1 Executive Summary

This deliverable describes the work carried out during the second period of the project regarding the development of approaches for **predictive analytics and recommendation** for the three entities of interest: **tracks, artists and music genres**. Having considered the updated version of FuturePulse requirements, the outcomes of deliverable D2.3 regarding data specifications and collection and the first version of Predictive analytics and recommendation framework (D3.1), we proceeded with the update of the methods developed during the previous period and the development of new approaches to tackle the requirements of the second period. At the same time, the work described in this document is accompanied by the appropriate update of the Application Programming Interfaces (APIs), which have been made available within the consortium in view of the platform integration activities.

We are presenting the methodologies chosen for the different predictive analytics approaches alongside a set of experiments validating their choice. We conclude which algorithms have been selected for integration into the FuturePulse platform through provision of internal APIs.

Funded by the European Commission

2 Introduction and Relation to other WPs/Tasks

This report documents the current progress of the FuturePulse consortium in the area of predictive analytics and recommendation.

As described also in D3.1, in terms of organization of the effort, two key groups of requirements have been identified, one related to popularity models for tracks, and another related to popularity models for artists and genres. These parts were assigned to two partners of the consortium, MMAP and CERTH. In particular, MMAP is responsible for modeling the popularity of tracks (Section 3.2) and CERTH for modeling the popularity of artists (Section 4) and genres (Section 5). Moreover, CERTH has also developed and tested models for the estimation of track recognition (Section 3.1).

The overall research direction and the particular problem definitions that are presented in this work were largely shaped through the analysis of user requirements and several discussions with end user partners of the consortium (WP1). Also, the work presented here has been highly dependent on the work done in the data collection Work Package (WP2), in the sense that the presented models and experiments rely on music data collected and extracted on the basis of work done within WP2. The resulting components from this work have been already integrated in the FuturePulse platform (WP4) and their pilot testing has started (WP5), while the resulting scientific advances have already been published in relevant venues (WP6). Most of the work carried out during that second year and described in this deliverable is provided as API calls. The structure and usage of these calls is described in detail in the previous version of predictive analytics and recommendations deliverable (3.1). The results provided by these calls have been updated accordingly, based on the outcomes of the work in this deliverable. When there was a need for new endpoints, we have updated the corresponding APIs. We describe these new endpoints in the implementation sections of this document.

To sum up, this deliverable marks the achievement of the following platform features:

- Estimate the current level and make short-term predictions about the popularity of music tracks (with a focus on tracks provided by PGM and SYB) based on the evolution of their music streaming (Spotify and Deezer), YouTube viewing patterns, music charts and airplay data (provided by BMAT).
- Improve the methodology that estimates a track's recognition by incorporating additional sources (Spotify and YouTube) alongside music charts and refining the way that these are combined based on a user study carried out by SYB.
- Estimate the current level and make short-term predictions about the popularity of artists by combining in a principled way multiple metric from social media and streaming platforms (Spotify, Deezer, YouTube, Twitter, Facebook, Last.fm, Soundcloud).
- Make short term predictions on the evolution of metrics collected at artist level.
- Estimate the impact of an event on individual success metrics of artists.
- Identify genres that could potentially describe artists for which that information is not available.
- Estimate the current and past level of popularity of music genres at country level, by aggregating information from diverse charts (traditional charts and charts provided by streaming platforms).

Funded by the European Commission

Table 1 presents the connection between the WP3-related requirements, the conducted work and the section where the corresponding results are presented.

Req.	Description	Work conducted	Sections
BMP_REQ#1	Recognition level of a track	Estimation of the current level of recognition of a track per market using past data of music charts and current data from Spotify and YouTube (CERTH)	3.1
BMP_REQ#2	Popularity level of a track	Estimation of the current level of popularity of a track globally using Spotify popularity, Deezer rank, Youtube and BMAT signals (MMAP)	3.2
RL_REQ#1	Predict success of tracks based on initial response		
RL_REQ#2	A combined visual timeline for streaming statistics of an artist	Artist Metrics Prediction (CERTH)	4.1
		Geometric Artist Popularity: multivariate (non-linear) score definition and evaluation (CERTH)	4.2
RL_REQ#6	Release day / Event impact on success	Implementation of a method that estimates the impact of events on success metrics (CERTH)	4.3
LM_REQ#5	Artist popularity in a given genre	Identify artist popularity within a specific community by considering artist-venue relationships (CERTH)	5.4
LM_REQ#8	Top upcoming artists per genre	Identify upcoming artists by using Beatport charts as a source (CERTH)	4.4
RL_REQ#5	Genres trending for each market	Identify genre associations to overcome data sparsity in small markets (CERTH) Predict genres of artists (CERTH) Estimation of genre popularity per market using past chart data (CERTH)	5.1, 5.2, 5.3
LM_REQ#9	Genre popularity		
BMP_REQ#15	Genre popularity for each market		

Table 1 List of FuturePulse requirements, work conducted and section in this document

Funded by the European Commission

3 Predictive Analysis for Tracks

3.1 Track Recognition Model: Data, Methodology and Results

In the first iteration of this deliverable, to support BMP_REQ#1 (Recognition level of a track), we presented a track recognition model based on charts and forgetting curve dynamics, while herein we present an improved version of it, called T-REC, that led to an accepted conference paper[1]. The main conducted improvements include:

- making the recognition retention percentage variable across tracks instead of constant, depending on the number of weeks each track remained in the charts;
- incorporating YouTube views and Spotify popularity in the model.

More detailed description of the changes upon the previously proposed model and the rationale behind them is presented in Section 3.1.5.

For estimating the recognition levels of music tracks, our starting point was a list of tracks provided by Soundtrack Your Brand. The list consists of 39,466 tracks from 21,450 artists in 75 countries. We also made use of data from 211 charts, 198 track charts and 13 singles charts, that span long periods of time (in some cases from the 60's until today) from 62 countries around the globe including Sweden. We also use the Spotify API to annotate chart entries with the Spotify id and International Standard Recording Codes (ISRC) of each of the songs. Since our user study focused on a sample of the Swedish population, we present the monitored charts for Sweden along with the corresponding monitored periods in Table 2.

chart name	since	until
Spotify Daily Chart	2017-01-01*	2018-03-06
Spotify Weekly Chart	2016-12-23	2018-03-01
Veckolista Svenskt Topp-20	2015-01-17	2018-06-15
Veckolista Singlar	1988-01-16	2018-06-15
Veckolista Heatseeker	2015-01-10	2018-06-15
Veckolista Svenska Singlar	2015-01-10	2015-01-16
SINGLES TOP 100	1975-11-08	2018-06-01
Sweden Top 20	2001-06-12	2018-07-07
Sweden Singles Top 100	2017-12-29	2018-07-05

Table 2 The list of Swedish charts we used in this study. The first column presents the chart name, the second and third columns present the start and end dates of monitoring respectively. *All dates are in YY-MM-DD format.

The vast majority of songs do not make it in the charts; therefore, we additionally employed YouTube views and Spotify popularity of tracks as proxies of their current popularity. Knowing the Spotify id of the tracks and the id of an associated official video

[1] C. Koutlis, M. Schinas, V. Gkatziaki, S. Papadopoulos, Y. Kompatsiaris. Data-driven song recognition estimation using collective memory dynamics models. In Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, 2019.

Funded by the European Commission

in YouTube[2] we retrieved these two signals using the public APIs offered by Spotify and YouTube respectively. The intuition behind the use of these two metrics, is that they reflect the exposure of a song in two widely used platforms. Number of video views on YouTube is a direct measure of how many people heard a song. On the other hand, although Spotify popularity is a score generated internally by Spotify and the exact formula is not known, that score reflects the actual number of streams a song received recently. Therefore, we can safely assume that a song having a high popularity score is currently listened more than songs with a lower score.

3.1.1 T-REC Methodology

The proposed song recognition model builds on three components: a) the *recognition growth* that represents the level of recognition a track reaches during its initial prosperity time (when it is placed in charts), b) the *recognition decay* that represents the collective memory decay process (i.e. the mechanism of collective forgetting of songs) and c) the *recognition proxy-based adjustment* that adjusts the recognition level of tracks, which is especially useful for tracks with no chart information.

Having annotated chart entries with the corresponding ISRC, we were able to retrieve the positions of tracks in the Swedish charts of Table 2. These are then used to estimate their *recognition growth* (in Sweden) according to Equation 1:

$$g(t) = \begin{cases} 100 \cdot \dfrac{c_K + 1 - r_K(t)}{c_K} \cdot \sigma_1, if\ track\ in\ chart\ K\ at\ time\ t \\ 0, otherwise \end{cases} \quad (1)$$

where c_K is the number of tracks in chart K, $r_K(t)$ is the rank of the track in chart K at time $t \in [t_0, t_{today}]$ and $\sigma_1 = \sigma_1(\theta_0, \theta_1, x) = \left(1 + e^{-\theta_1 \cdot x + \theta_0}\right)^{-1}$ adjusts the rank's importance using an S-shaped learning curve with $x \in (0, +\infty)$ and $\theta_0, \theta_1 \in R$. The logistic part of the model is incorporated to control the importance of a chart position given the number of weeks x the track has remained in the charts. If a track remained in the charts for only one week, its rank's importance would be a lot lower (54.9%) compared to it remaining for 20 weeks (98.2%). Therefore, in the first case the decay process will begin from a much lower point. The value of $g(t)$ is assigned to all $g(t_\iota)$ with $t_\iota \in [t - n + 1, t]$ according to the chart's frequency; e.g. if it is weekly $n = 7$. If a track gains multiple values at a single date, the maximum value is used.

Towards formally defining the *recognition decay*, we build on findings from research literature in the area of human memory, and more precisely on the concept of forgetting curves. A forgetting curve is the rule by which the memory regarding a specific learned item is reduced. In our case we consider as learned items the music tracks. Hence, we aim at estimating the function that describes the forgetting procedure that has been proposed to be exponentially decreasing in many studies [Loftus 1985; Murre and Dros 2015]. We also opt for the exponentially decreasing forgetting curve for song recognition but at a less steep rate. It is natural to consider that the level of *recognition decay* is impossible to be higher than the level of *recognition growth* at its peak for a particular track. Also, each time the track re-emerges in the charts the forgetting procedure restarts

[2] We used the Soundiiz (www.soundiiz.com) service, which supports playlist conversion between platforms.

Funded by the European Commission

from a new higher point of recognition. Additionally, we consider a variable decay rate as a reasonable consideration would be that not all songs' recognition decays with the same velocity. The *recognition decay* is defined in Equation 2:

$$d(t) = \begin{cases} g(t), & if\ d(t-1) \leq g(t) \\ \sigma_2 \cdot d(t-1) + (1-\sigma_2) \cdot g(t), & otherwise \end{cases} \quad (2)$$

where $\sigma_2 = \sigma_2(\varphi_0, \varphi_1, x) = (1 + e^{-\varphi_1 \cdot x + \varphi_0})^{-1}$ is the recognition retention percentage with x being the number of weeks the track has remained in the charts and $g(t)$ is the previously defined in Equation 1 *recognition growth*. If a track has remained for a long time in the charts its retention percentage would be considerably higher and its forgetting process would be rather slow, while if a track has remained in the charts for only few weeks its retention percentage would be low and its forgetting process fast. The first logistic function σ_1 controls the initial recognition level from which the decreasing trajectory begins and the second logistic function σ_2 controls the velocity of recognition decay, both individually per track.

To model the *recognition proxy-based adjustment*, we consider a multiple linear regression model with input the track's current Spotify popularity index (P_S) and the log-transformed YouTube views (P_{YT}) as in Equation 3:

$$s(t_{today}) = \alpha_0 + \alpha_1 \cdot log(P_{YT}) + \alpha_2 \cdot P_S \quad (3)$$

The composite T-REC model is defined as a linear combination of *recognition decay* and *recognition proxy-based adjustment* at t_{today}:

$$T - REC = w_0 + w_1 \cdot d(t_{today}) + w_2 \cdot s(t_{today}) \quad (4)$$

All the model parameters (*θ, φ, α, w*) are optimized as described in Section 3.1.3.

3.1.2 Competitive Models

For a comparative study, four competitive models were employed for the task of song recognition estimation:

1. Spotify popularity index (P_S)
2. Multiple Linear Regression (MLR: Equation 3)
3. Random Forest (RF)
4. Log-normal decay model (LOGN) [Wang et al. 2013]

3.1.3 Optimization and Evaluation

We consider a holdout strategy (70% training, 30% test) for the models' evaluation as described in [Cerqueira et al. 2019]. The optimization of all models' parameters was performed on the training set by the truncated Newton algorithm as implemented by the SciPy package. We used as objective function the mean absolute error between measured (by the user study) and computed (by each model) song recognition.

Funded by the European Commission

The model performance was then evaluated on the test set by the Mean Absolute Error (MAE) between the measured and the computed recognition as in Equation 5:

$$MAE = \frac{\sum_{i=1}^{k} |x_i - y_i|}{k} \quad (5)$$

where k is the number of tracks, x_i the measured recognition for track i and y_i the computed recognition for track i. A perfectly accurate model would lead to a MAE value of 0.

3.1.4 User Study

To proceed with the user study, we employed an initial and much simpler version of the recognition score (the model presented in D3.1, i.e. v1 of this deliverable). This initial version had a constant decay rate across all tracks and for the tracks with no chart data the average recognition score of the closest, in terms of YouTube views and Spotify popularity, tracks was considered as their recognition score.[3]

After the assignment of the initial recognition score (corresponding to the time the survey was conducted) to each of the 39,466 tracks, we formed two lists. One list containing the 600 most recognized tracks in Sweden and a second containing the 600 least recognized tracks in Sweden.[4] Consequently, 50 tracks were randomly chosen out of each of these two lists as representative of high and low recognition tracks.

A study was then conducted in order to obtain the actual recognition percentages for each of these 100 songs among a test population of 1041 annotators in Sweden.[5] We divided the initial list of 100 songs in 10 groups of 10 songs (5 of low and 5 of high recognition level in a randomized order), then each participant listened to 30-second samples of all the songs of one group and for each song he/she indicated whether he/she recognized it or not. We had ~100 respondents per song[6] so we got a score 0-100 based on the percentage of respondents who responded positively.

	class	#	fraction	Sweden
gender	male	521	50.05%	50.24%
	female	520	49.95%	49.76%
age	18-24	54	5.19%	18.15%
	25-34	422	40.54%	22.46%
	35-44	277	26.61%	20.01%
	45-54	244	23.44%	21.12%
	55-65	44	4.22%	18.26%

Table 3 Demographics of the test population and Sweden (normalized within the group of people between 15 and 65 years old). *This figure refers to 15-24 age group.

[3] The rationale behind the alterations on this model that led to T-REC is illustrated in Section 3.1.5.

[4] Given that recognition estimation is the result of a sampling process, we expect measurements in the extremes (i.e. least and most recognized songs) to be less noisy than in intermediate recognition levels. This motivated our choice to perform the initial song selection out of two distinct sets (high, low).

[5] The study was performed through the Cint survey platform (https://www.cint.com/).

[6] Some variability was due to the fact that not all respondents completed the process successfully.

Funded by the European Commission

We consider the recorded responses as ground truth for our experiments and we evaluate our model as well as the competitive models on this basis. Demographics of the test and Swedish population[7] are illustrated in Table 3. We observe divergent age demographics, yet almost identical gender demographics between the test and the actual population. As the selection of annotators was carried out by Cint, we could not better approximate the Swedish population distribution.

Despite the over-representation of some age groups and under-representation of others, T-REC is still a sound methodology; given a different population sample to learn from, the model tuning would lead to a slightly different recognition estimation model.

3.1.5 Results

The analysis of the survey data shows that the initial recognition score classified the tracks effectively with 50/50 (100%) correctly labelled as low and 37/50 (74%) correctly labelled as high recognition (measured recognition <50% is considered as low and >50% as high). The 13 songs that were falsely classified as high recognition obtain a smaller recognition score than the rest (on average 6 units lower). Despite the promising classification performance, the measured recognition was in many cases far from the computed score especially in cases of tracks with no chart data. This was the reason for developing an improved version of the recognition model (T-REC).

θ_0	θ_1	φ_0	φ_1	α_0
0.233	0.043	0.847	0.029	1.299
α_1	α_2	w_0	w_1	w_2
0.999	-0.093	22.586	0.452	0.928

Table 4 Parameter values for T-REC after fitting.

Table 4 presents the parameter values of T-REC after optimization, using measured recognition of the 100 user study tracks as ground truth. The model gives significant weights on both *recognition decay* (w_1) and *recognition proxy-based adjustment* (w_2) components, but considers YouTube views (α_1) as more important than Spotify popularity (α_2) for the under study problem. The impact of the rest of the parameters on the final model, namely the shape of the two logistic functions that control the *recognition growth* (θ_0, θ_1) and *recognition decay* (φ_0, φ_1) components is illustrated in Figure 1. The logistic part of *recognition growth* (rank's importance) is less steep than the logistic part of *recognition decay* (retention percentage), indicating that a music track will need almost 7 weeks in the charts to achieve a very slow rate towards oblivion, but at least 25 weeks to achieve its highest contemporary recognition.

[7] Sources: statista.com/statistics/521717/sweden-population-by-age/, statista.com/statistics/521540/sweden-population-by-gender/

Funded by the European Commission

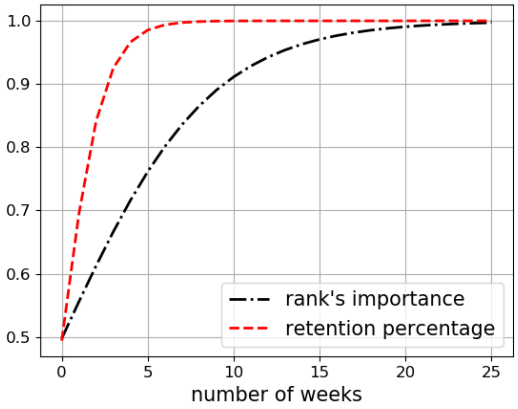


Figure 1 The logistic parts of recognition growth (rank's importance) and recognition decay (retention percentage) components as formed after the model fitting.

Figure 2 illustrates two examples of how T-REC models the mechanism of song recognition decay. The song «Rude Boy» by Rihanna stayed in Swedish charts for 19 weeks, and according to the *recognition decay* component it maintained 99.9% of its initial recognition (Figure 2a). The *recognition proxy-based adjustment* input adjusts T-REC very close to the measured recognition (error=3.11). A different example presented shows that Mariah Carey's "All I Want for Christmas Is You" was initially not a big hit in Sweden, remaining for only three weeks in the charts back in 1995 (Figure 2b). Afterwards, its recognition exhibited a significant decrease during the next decade, but after 2007 when the song kept re-emerging in the charts every year its recognition decay rate slowed down and both its *recognition growth* and *decay* components grew larger. The *recognition proxy-based adjustment* component adjusts T-REC a little lower. Although we lack ground truth for this song to compare it to T-REC's estimation (as it was not in the survey's lists), we believe that 80.99% recognition is closer to the real recognition rate[8] than the 98.99% computed by the *recognition decay* component, which is obviously too high even for a massive hit such as this.

Figure 3 illustrates the performance of T-REC on estimating the actual current recognition level of songs in Sweden. Most of the points are concentrated close to the identity line except for some tracks of intermediate recognition levels, which are overestimated. Table 5 compares the performance of T-REC with the competitive models in terms of average MAE. All models (except for Spotify popularity index) are trained using Monte Carlo cross validation, i.e. we used 100 different training sets, each containing 70 randomly selected tracks out of the initial set of 100 tracks and then their MAE is measured on the 100 corresponding test sets, each containing the remaining 30 tracks. T-REC exhibits the best performance among all models with a very high statistical significance level as indicated by the p-value=10^{-17}, according to the Wilcoxon signed rank paired test.[9]

[8] Or only slightly underestimating it given that the top-3 measured recognition percentages of our survey are 89.42, 85.57 and 84.61.

[9] This is the maximum p-value among all four comparisons.

Funded by the European Commission

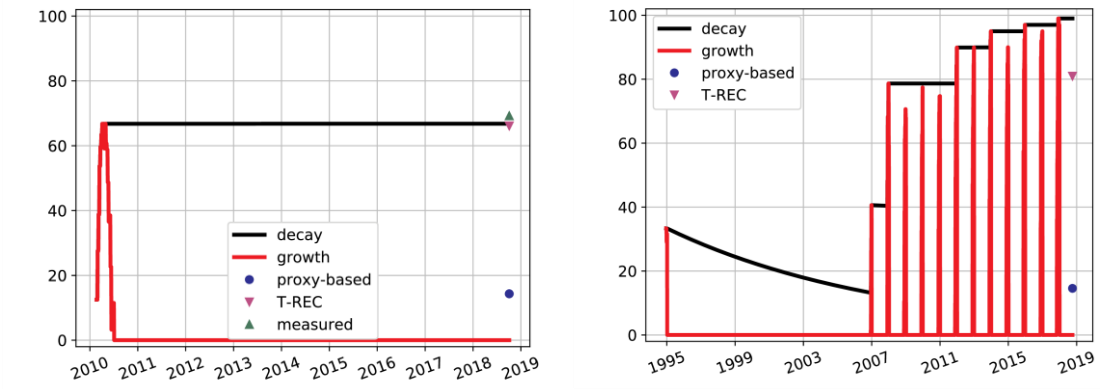


Figure 2 T-REC components (recognition growth, decay and proxy-based adjustment) for two highly recognized songs

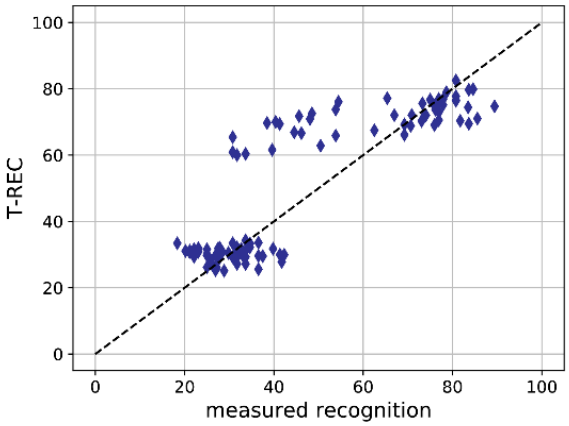


Figure 3 Scatter plot with the T-REC score on the y axis and measured recognition on the x axis.

Model	AMAE
P_S	20.63
MLR (Equation 3)	11.30
RF	10.27
LOGN (Wang et al. 2013)	22.00
T-REC	**8.50**

Table 5 Average MAE over 100 randomly selected test sets for T-REC, Spotify popularity ( ), Multiple Linear Regression (MLR), Random Forest (RF) and log-normal (LOGN) models. For Spotify popularity we computed once the mean absolute error over all 100 tracks.

The diverging behavior of the songs with intermediate recognition level in Figure 3 is also apparent in their YouTube and Spotify metrics as shown in Figure 4. One possible explanation for this behavior is that 15 out of these 17 songs have been released during the last three years and they still are in their initial popularity phase. Thus, there has not passed a considerable amount of time for these tracks to experience significant recognition decay, which T-REC would likely capture. As exemplified in Figure 5 the more recent the track the bigger the error our model produces, which is a limitation of the

Funded by the European Commission

proposed model, even though the average errors in all periods are small (the maximum is 9.5) and less than any other compared model.

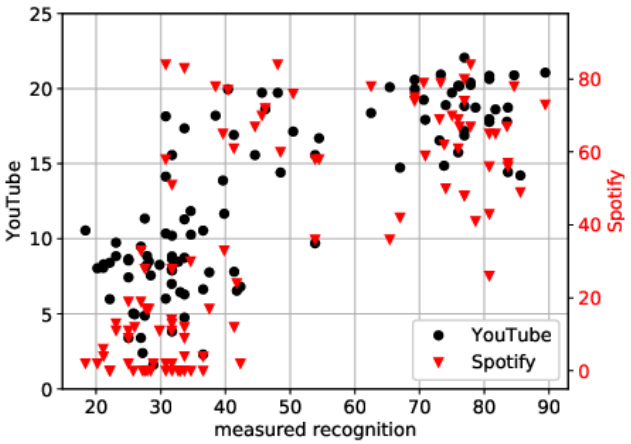


Figure 4 Scatter plot - y axes: YouTube views (log) and Spotify popularity, x axis: measured recognition.

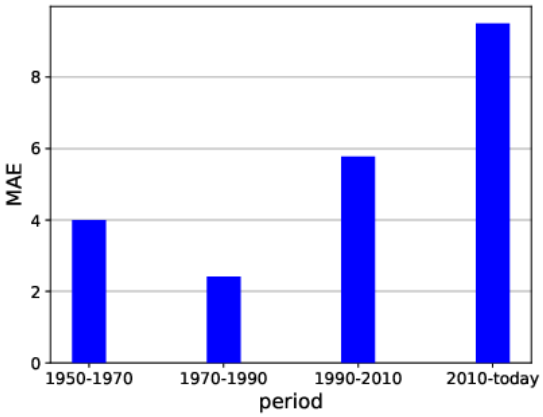


Figure 5 Mean absolute error of T-REC on tracks released in different periods of time.

Also, we would like to elaborate on the rationale behind the refinements that we performed on our model in order to take its final form (Equation 4). In Figure 4 a linear interaction is observed between 1) the log-transformed YouTube views and Spotify popularity and 2) the measured recognition, with Pearson correlation coefficients 0.79 and 0.71 respectively. For that reason, we incorporated the multiple linear regression model with the corresponding input quantities as *recognition proxy-based adjustment* component in the final T-REC formula. The consideration of a constant decay rate in the formula of *recognition decay* is not plausible, since it would further lead to a zero rate as the best choice after model fitting, which is highly unrealistic, as the model would degenerate into the *recognition growth* component. As a result, the final form of T-REC includes a variable decay rate across music tracks that depends on the number of weeks the track has remained in the charts. This refinement resulted in significantly lower errors showcasing the major role of the number-of-weeks feature in song recognition estimation. More specifically, the initial recognition score achieved a MAE of 12.32, while T-REC a much lower MAE of 8.50 as presented in Table 5.

Funded by the European Commission

Table 6 and Table 7 present the 10 most recognized songs (according to T-REC) in Sweden and USA respectively. Also, Table 8 and Table 9 present the corresponding most recognized artists, in terms of maximum entries and maximum score. In the Appendix A we present longer lists with the top-100 songs for USA and Sweden. Especially for the case of Sweden the "Top-100 songs" list is overwhelmed by non-Swedish artists and songs (mainly American and English), which is partially explained by the fact that our dataset contains only 1,082 Swedish songs out of a total of 39,466 songs. Notwithstanding, the Top-1 recognized artist in Sweden, according to Table 8, is "Avicii", a Swedish artist with international impact.

For comparison purposes we used Billboard's "The Hot 100's All-Time Top 100 Songs" list[10] where the Top-100 songs of all time, according to Billboard's "The hot 100" chart, are illustrated. Many of these songs are not included in our songs dataset, thus we compared T-REC's Top-N list (for N=100, 1000, 2000,...) for USA, with the intersection of Billboard's list and our songs dataset, namely 58 common songs.

1	More Than You Know	Axwell /\ Ingrosso	84.4
2	rockstar	Post Malone	84.2
3	Never Be Like You (feat. Kai)	Flume	83.4
4	Havana	Camila Cabello	83.3
5	Despacito (Featuring Daddy Yankee)	Daddy Yankee	82.9
6	Despacito - Remix	Daddy Yankee	82.7
7	Thunder	Imagine Dragons	82.5
8	Mambo No. 5 (A Little Bit of...)	Lou Bega	82.4
9	Last Christmas	Various Artists	81.9
10	Shape of You	Ed Sheeran	81.4

Table 6 The Top-10 recognized songs in Sweden according to T-REC.

1	Escápate Conmigo	Wisin	86.1
2	Ginza	J Balvin	86.1
3	I Knew I Loved You	Various Artists	86
4	Thunder	Imagine Dragons	86
5	Believer	Imagine Dragons	85.9
6	Whatever It Takes	Imagine Dragons	85.8
7	I Get The Bag (feat. Migos)	Gucci Mane	85.5
8	Stay With Me	Sam Smith	85.5
9	No Roots	Alice Merton	85.3
10	Stayin' Alive	Bee Gees	84.9

Table 7 The Top-10 recognized songs in USA according to T-REC.

[10] https://www.billboard.com/articles/news/hot-100-turns-60/8468142/hot-100-all-time-biggest-hits-songs-list

Funded by the European Commission

1	Avicii	4	1	Axwell /\ Ingrosso	84.4
2	The Chainsmokers	3	2	Post Malone	84.2
3	Lady Gaga	3	3	Flume	83.4
4	Ed Sheeran	3	4	Camila Cabello	83.3
5	Katy Perry	2	5	Daddy Yankee	82.9
6	Post Malone	2	6	Imagine Dragons	82.5
7	Jonas Blue	2	7	Lou Bega	82.4
8	Justin Bieber	2	8	Ed Sheeran	81.4
9	Imagine Dragons	2	9	Sam Smith	81.3
10	Major Lazer	2	10	Katy Perry	81.2

Table 8 Top-10 artists in terms of maximum entries (left) and maximum score (right) in the list of Top-100 recognized songs in Sweden.

1	Taylor Swift	4	1	Wisin	86.1
2	Charlie Puth	3	2	J Balvin	86.1
3	J Balvin	3	3	Imagine Dragons	86
4	Imagine Dragons	3	4	Gucci Mane	85.5
5	Maluma	3	5	Sam Smith	85.5
6	Daddy Yankee	3	6	Alice Merton	85.3
7	Christina Aguilera	3	7	Bee Gees	84.9
8	Player	2	8	Taylor Swift	84.9
9	Flume	2	9	Player	84.8
10	David Guetta	2	10	Post Malone	84.8

Table 9 Top-10 artists in terms of maximum entries (left) and maximum score (right) in the list of Top-100 recognized songs in USA.

Figure 6a presents the number of top Billboard songs included in T-REC's Top-N lists for USA, while Figure 6b presents the number of top Billboard songs included in T-REC's Top-N lists for USA, where N is the number of songs with T-REC score greater than a threshold specified by the x coordinate.

As expected, T-REC's values are significantly higher and more concentrated when “The Hot 100” is included in the computations (see Figure 7 for more details). This fact indicates that the criteria for song ranking may vary a lot among different charts. Also, it is remarkable that half of Billboard's top songs are ranked by T-REC[11] lower than the

[11] In this calculation we excluded Billboard's “The Hot 100” chart.

Funded by the European Commission

1450th position, which occurs primarily due to lack of “The Hot 100” information and secondarily duo to ISRC annotation mismatches in our database.[12]

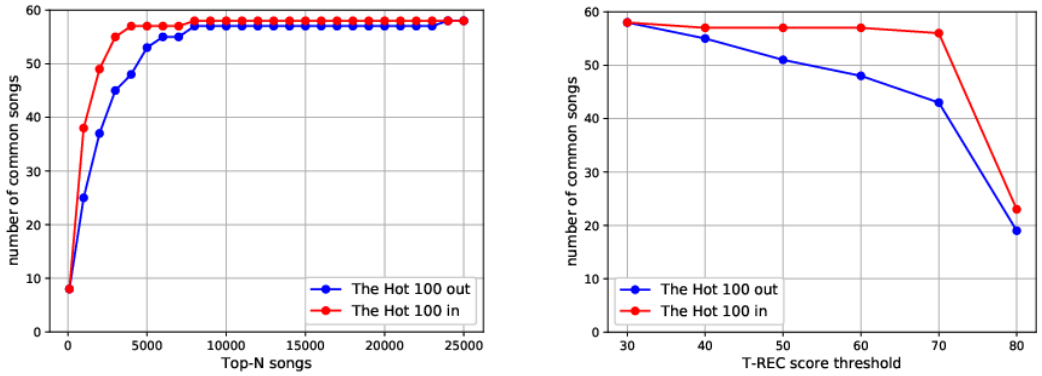


Figure 6 Number of common songs in T-REC's Top-N lists for USA and Billboard's ``The Hot 100'' Top-100 songs of all time. Chart data from “The Hot 100” were discarded during the computation of T-REC to form the blue line in contrast with the red line for which all charts were included. In (a) the number of listed songs N is specified by the x coordinate, while in (b) the number of listed songs N is the number of songs with T-REC greater than a threshold specified by x.

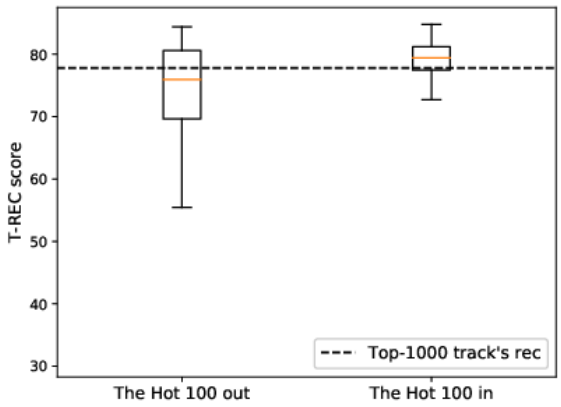


Figure 7 Distribution of T-REC's scores for Billboard's top songs with chart data from ``The Hot 100'' not included (left boxplot) and included (right boxplot) in the computation. The dashed line indicates the T-REC level of the Top-1000 song when "The Hot 100" is not included in the computation.

3.1.6 Implementation and integration of results

In the first version of track recognition module, we calculated the monthly recognition values for about 80k tracks, related to PGM and SYB. These values are available through the appropriate API endpoint[13]. In other words, through the API, the user can get the evolution of a track’s recognition or can sort tracks by recognition at specific dates and countries. In the current version of the platform, the endpoint, the parameters that can

[12] The main factor for these mismatches is that the same song may have multiple ISRCs, as it can be included in different releases and different countries. In cases where we fail to merge these ISRCs, the estimated T-REC of a song is much lower than the actual, as it does not include all the chart appearances of the song in the calculation.

[13] The track recognition values for a specific track, can be obtained by calling the /tracks/<:track_id>/recognition endpoint. To sort tracks by recognition the sort parameter can be used in the /tracks endpoint. For more details, please check D3.1.

Funded by the European Commission

be used to refine the results (e.g. define a specific country or time range for which recognition is requested), and the structure of the response have remained unchanged. However, in the back end we had to revise the way this module is executed, as the updated version takes into account not only historic data from music charts but also the recent values of Spotify popularity and YouTube views. More precisely, to calculate recognition of a track at a specific date, we need to incorporate in the calculation the number of YouTube views and Spotify popularity at that specific date. The results presented in the previous sections have been produced by using the values of these additional signals had in October 2018. From then on, we have used updated values of these metrics, alongside updated charts data to produce periodically the recognition score of tracks imported in FuturePulse platform[14].

3.2 Track Popularity Estimation and Prediction

This section is about the estimation of a global indicator of track popularity, as well as the prediction of a track's future popularity, to support BMP_REQ#2 (Popularity level of a track) and partially RL_REQ#1 (Predict success of tracks based on initial response).

3.2.1 Track Popularity Data Sources

We used different sources of track popularity. The Spotify Charts data (dataset A) was collected for early, preliminary analysis of prediction algorithms. For the actual implemented track popularity estimation and prediction, currently four data sources are used: Spotify popularity, Deezer rank, YouTube and BMAT Vericast airplays. These are collected for the set of tracks provided by SYB (dataset B).

A) Spotify Charts Streaming data

In order to carry preliminary prediction experiments with highly popular tracks, we collected **Spotify charts** streaming data. The Spotify Charts website[15] allows to download the platform's daily and weekly charts as csv files. For each track in a chart, we get its position and stream count. Thus, we retrieved daily **US top 200 charts from January 1st, 2017 to June 12th, 2019** (only 3 days within this time frame were missing). An example sample is provided in Table 10.

The resulting dataset **(Dataset A)** contains streaming information for **4,542 tracks**, although each track stream count is available only when and while it is in the chart — which leads to gaps in the dataset for some tracks (see Figure 8).

B) Daily Crawled Popularity Data

The following data indicators and sources are used for the final implementations of track popularity estimation and prediction:

- ***Spotify popularity***[16]: metric computed by *Spotify* that provides a current evaluation of the popularity of each track. The value is an integer between 0 and 100. As specified in the API documentation, the metric is based mostly on the

[14] As described in D2.3 we are currently collecting Spotify and YouTube metrics at track level. We plan to use the corresponding endpoint that provides these metrics, to update systematically the monthly recognition scores of tracks.
[15] https://spotifycharts.com/
[16] https://developer.spotify.com/documentation/web-api/reference/tracks/get-track/

Funded by the European Commission

number of plays that a track has had on recent days. Note that this value can suffer from some delay by a few days.

- ***Deezer rank***[17]: global indicator of a song's popularity from 0 to 1 million computed by Deezer. Similarly to Spotify popularity, it is based on the track's recent stream count on the platform.
- ***YouTube views, likes, dislikes* and *comments***: 4 cumulative metrics provided by YouTube's API[18] on each YouTube link.
- ***BMAT Vericast***[19]: BMAT monitors 5,000 radio stations and 1,500 television channels in 134 countries, and over 1,000 clubs worldwide. BMAT's Vericast audio recognition and music identification platform uses audio fingerprinting technologies to monitor up to 72 million tracks and provides airplay data for the tracks played in any of those outlets through Vericast API. We collected *global* airplay data for the months of June and July 2019.

Data acquisition ("crawling") was started from Spotify and YouTube in February 2018. In June 2019 BMAT and Deezer were added. Data acquisition is performed **daily**, as the values change daily.

The daily crawling of each signals has been done on **39,466 tracks references provided by SYB**. Each data source reference for a track is aligned and then defined regarding the track's ISRC. Note that not all tracks are available from all sources. For the experiments where all sources are needed or compared, the effective number of tracks was **35,388 tracks** available from all 4 sources. Therefore, our tests and experimentation focused on 3 different datasets, defined by their date ranges:

A) **Dataset B1**: 503 days, with dates from 28-02-2018 to 15-07-2019, only on *Spotify Popularity* and the four *YouTube* signals.
B) **Dataset B2**: 60 days, with dates from 01-06-2019 to 31-07-2019, for all source signals.

Date	Position	Title	Artist	Spotify ID	Stream count
2017-01-01	1	Bad and Boujee (feat. Lil Uzi Vert)	Migos	4Km5HrUvYTaSUfiSGPJeQR	1371493
2017-01-01	2	Fake Love	Drake	343YBumqHu19cGoGARUTsd	1180074
2017-01-01	3	Starboy	The Weeknd	5aAx2yezTd8zXrkmtKl66Z	1064351
2017-01-01	4	Closer	The Chainsmokers	7BKLCZ1jbUBVqRi2FVlTVw	1010492
2017-01-01	5	Black Beatles	Rae Sremmurd	6fujklziTHa8uoM5OQSfIo	874289

Table 10 Spotify US top 200 chart sample

[17] https://developers.deezer.com/api
[18] https://www.youtube.com/intl/fr/yt/dev/api-resources/
[19] https://www.bmat.com/en/api-vericast/

Funded by the European Commission

Data Pre-processing

Several reformatting and transformation processes have been applied to the signals from different data sources. From each raw signal $X'_s(t)$ we perform a series of transformations to get a final usable signal $X_s(t) = f_s(X'_s(t))$. The transformations $f_s()$ depend on the nature of the original signal, but in general:

- Knowing that collected data can suffer from data gaps (due to various reasons: internal technical issues, API updates, requests limits, etc.), a **gap filling** process based on simple linear interpolation is applied;
- Cumulative signals (i.e *YouTube*'s signals) have been **differentiated** on periods of X days in order to have a signal representing the current trend on each day (in our experiments, X = 5 days);
- Moreover, we **log-transformed** the cumulative signals (e.g. from YouTube) to approach a normal distribution (see Figure 9).
- Finally, for all signals, the processed time series are **rescaled to a range between 0 and 1** (with a global minimum and maximum for all the tracks, per signal);

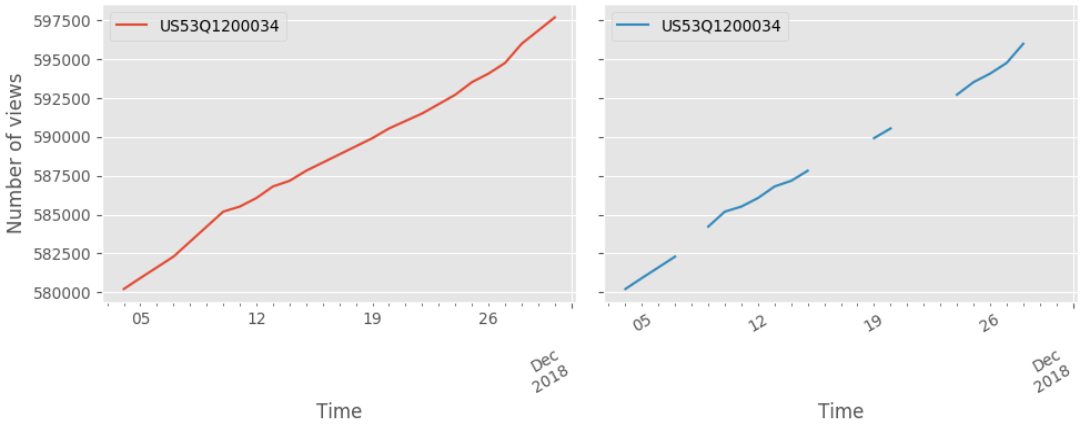


Figure 8 Gap-filled signal vs. original signal data from web data collection.

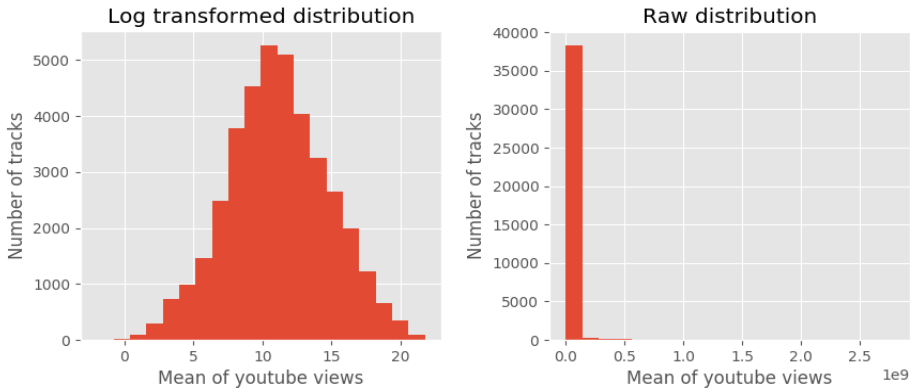


Figure 9 Distribution of YouTube views: differentiated (log transformed) vs. differentiated (before log transform).

Funded by the European Commission

Data Splitting and Chunking

The data from one signal *s* is first split into train and test set, with a train/test set ratio of 80%:20% of the tracks. Completely avoiding overlap between the training and test set allows us to effectively evaluate the performance of a prediction of a model on unseen data. From each set, a defined number of chunks is extracted regarding the number of days define for history n_h and prediction n_p. All training and learning steps are performed on the chunks from the training set, whereas the evaluation is applied on the test set. Chunks inside a set can be overlapping or not and can be of various lengths. A visual explanation is presented in Figure 10.

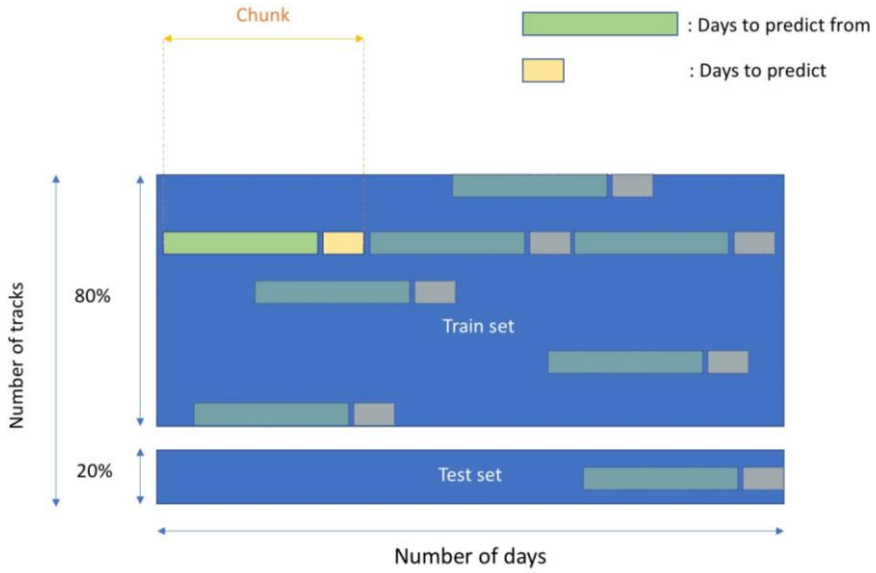


Figure 10 Dataset splitting and chunking.

To avoid redundancy and for computational reasons, we decided to use all chunks available for each track with an interval of 7 days (chunks will then overlap if $n_h+n_p > 7$).

For instance, using a dataset of 35388 tracks x 60 days (e.g. Dataset B2) and a chunk size of 28+21 = 39 days, we end up with 56220 chunks for training and 14156 for testing.

Pre-Analysis of Spotify Charts Dataset (A)

Number of days a track is in the charts

On average, a track appears in this chart for 40 days (not necessarily consecutive), the longest and shortest appearances being 889 days and 2 days, respectively. Figure 11 presents the distribution of days in chart for the Spotify's top 200 US chart. Figure 12 presents the daily streams of the track having the longest appearance in the charts (889 days). Figure 13 gives an example where there are gaps in the charts data.

Funded by the European Commission

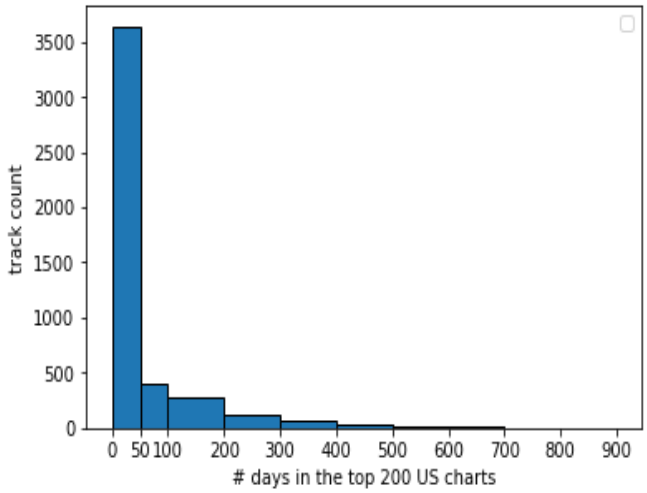


Figure 11 - Number of days tracks have been in Spotify US top 200 charts.

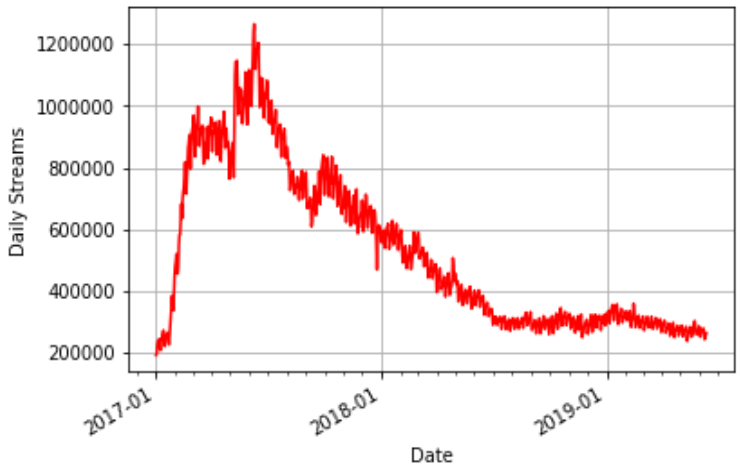


Figure 12 Example of complete charts data: 'Congratulations' by Post Malone (889 days in the charts).

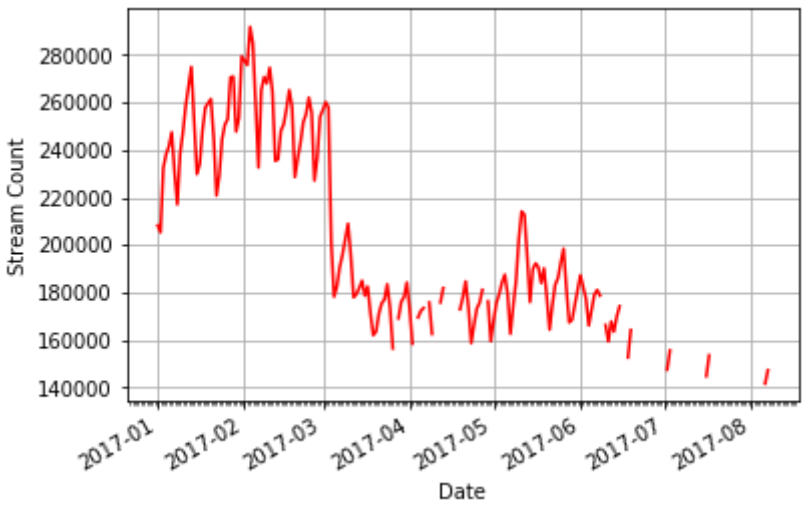


Figure 13 Example of gaps in the charts data: 'Erase Your Social' by Lil Uzi Vert (160 days in the charts, on and off).

Funded by the European Commission

Number of Streams

Debut

We looked at the number of streams these tracks reached on their debut in the chart (Figure 14). A track has got around 478 k streams on its first day on average, whereas the minimum and maximum debut streams are around 122 k and 5.75 M respectively.

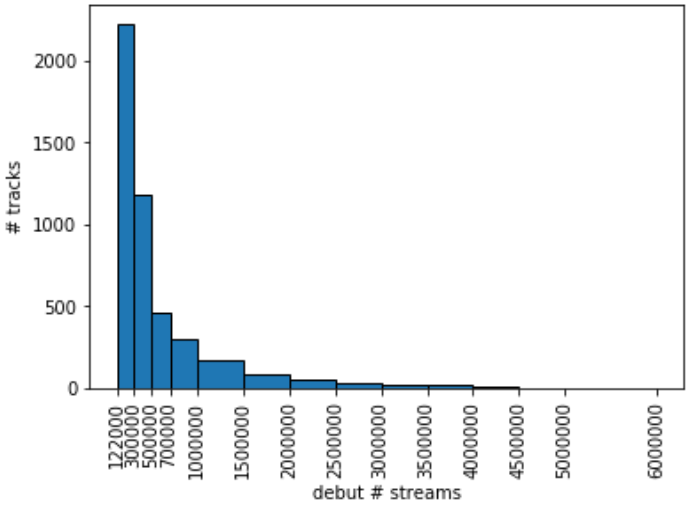


Figure 14 Debut number of streams for a track

Trend & Seasonality

By looking at the evolution of stream counts, one can observe common trends for most tracks. Indeed, the daily stream count is likely to rise at first, then decrease over time — when the track has been in the chart for long enough. Some tracks keep decreasing though.

In addition, one can clearly identify a weekly seasonality. The stream count appears to decrease on Sundays, as shown on Figure 15.

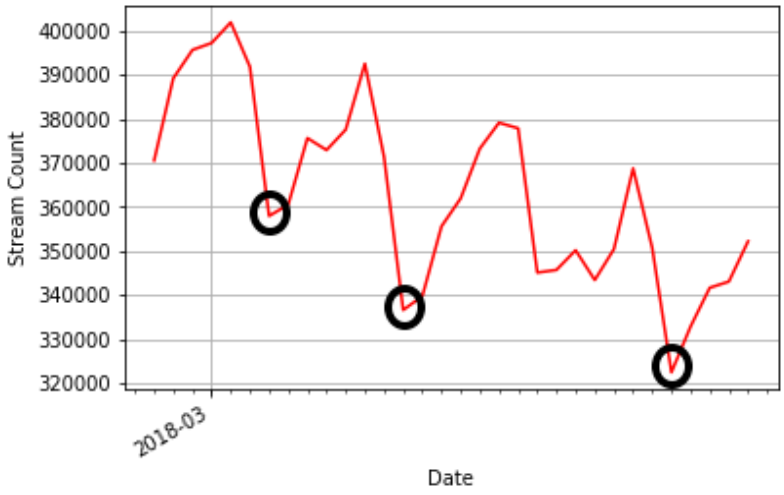


Figure 15 Seasonality in Spotify streams: 'Goosebumps' by Travis Scott (from Monday 2018-02-26 to Thursday 2018-03-29)

Funded by the European Commission

Pre-Analysis of Daily Crawled Data for SYB Dataset (B):

General Trend

The **general trend** classifies a curve regarding its general start to finish difference (that can be increasing, decreasing or flat). The general trend of a time-series of size *n* is simply defined by equation 6.

$$gtrend(X_s) = \begin{cases} 1 \; if \; X_s(t_n) > X_s(t_1) + e \\ -1 \; if \; X_s(t_n) < X_s(t_1) - e \\ 0 \; otherwise \end{cases} \quad (6)$$

(With *e* being the minimum variation value, usually set to 1% of the maximum of all signals values.). Figure 16 and 17 show the general trend distribution of all tracks on datasets B1 and B3, respectively.

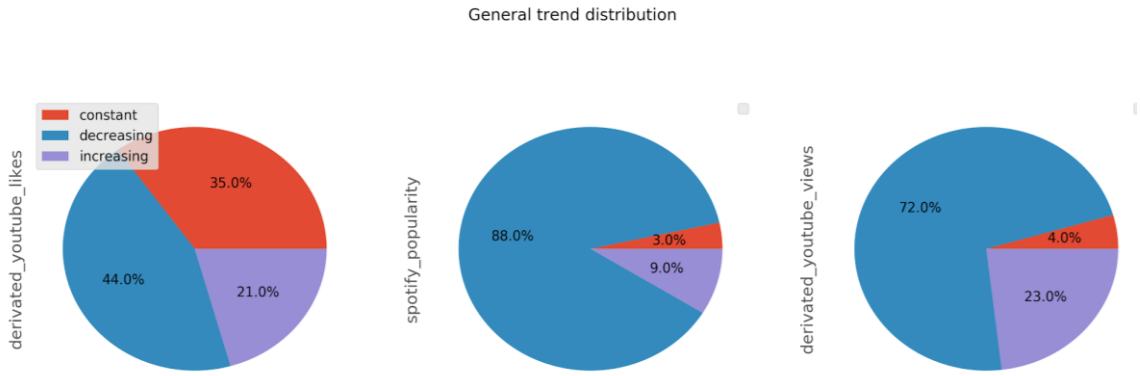


Figure 16 General trend distribution of dataset B1

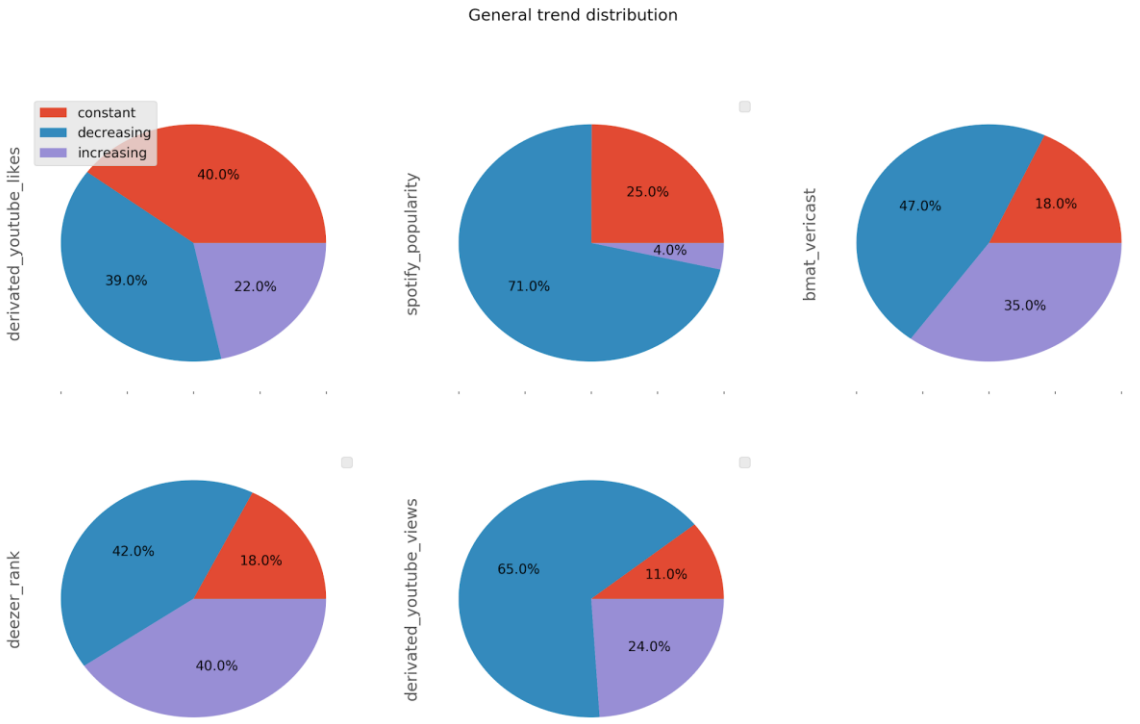


Figure 17 General trend distribution of Dataset B2

Funded by the European Commission

Maximum variation

The maximum variation of a time-series is defined by the difference between its maximum and minimum value. A curve with a flat general trend can still have high variations and valuable information to learn.

Figure 18 and Figure 19 present a histogram with the number of tracks having a maximum variation above a certain threshold. The threshold value is a percentage of the overall possible maximum variation.

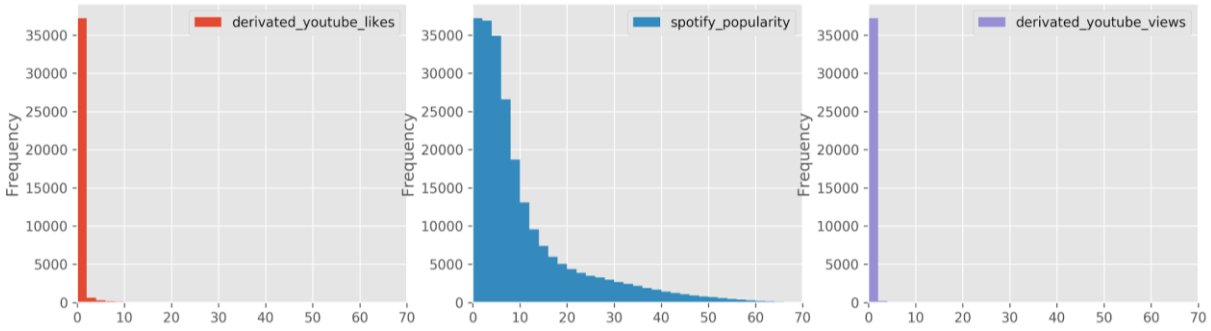


Figure 18 Maximum variation distribution of dataset B1 (in percentage of the maximum)

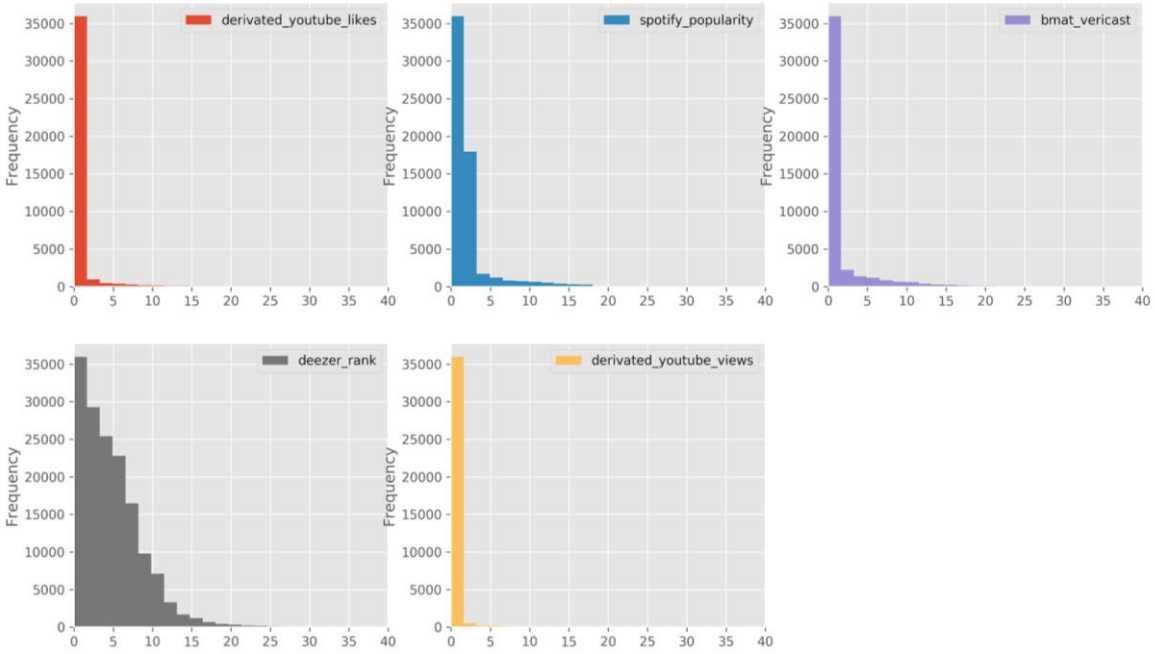


Figure 19 Distribution of maximum variation distribution of Dataset B2 (in percentage of the maximum)

3.2.2 Track Popularity Aggregation and Estimation

The different signals / metrics were aggregated into one general popularity metric, representing the general track popularity at a defined moment in time.

Funded by the European Commission

Regarding the different formatted signals $X_s(t)$ presented in section 3.2.1 the popularity metric is the result of the weighted average of those different signals as defined in equation 7.

$$P(t) = \frac{\sum_S X_s(t) \times W_s(t)}{\sum_S W_s(t)} \quad (7)$$

For a signal *s*, the weight can be a defined constant C_s, or can be time dependent to any other formatted metric *s'*

$$W_s(t) = C_s \ \ or \ \ C_s \times f(X_{s'}(t)) \quad (8)$$

For instance, the current implementation of the popularity aggregation defines the weighting of the formatted *Spotify Popularity* and *Deezer rank* as constant, considering the fact that those metrics have been internally normalized and adjusted by the providers. On the other hand, a proportional weighting regarding the *scaled YouTube views* is used for weighting the derivated *YouTube views*, with the aim to assign more importance to YouTube views variations when the number of views is high.

For the general popularity metric as an outcome of aggregation, no evaluation has been performed at this point, as due to lack of ground truth data, this cannot be performed in an automated way (by contrast to track popularity prediction - see next subsection). A user-based evaluation with the help of music industry representatives (e.g. from PGM) is planned for the next period.

3.2.3 Track Popularity Prediction

While the previous subsection described the estimation of an aggregated general *current* track popularity metric, this subsection deals with the prediction of *future* track popularity (trends).

By splitting the available data into "past" and "future" to predict, an automatic evaluation has been performed. Therefore, we present some evaluation metrics on which the performance measurement of the algorithms to be presented next has been done.

Evaluation metrics

The evaluation results of each prediction model were performed and analysed regarding different metrics:

- **Mean Absolute Error**, or **MAE**: Average of the absolute differences between the predicted value and the real value at each time step.

$$MAE = \frac{1}{n}\sum_i |y_i - \hat{y}_i| \quad (9)$$

- **Mean Squared Error**, or **MSE**: Average of the **squared** differences between the predicted value and the real value at each time step.

$$MSE = \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2$$

Funded by the European Commission

(10)

- **Mean Trend Score**, or **MTS**: Proportion of the cases were the predicted *general trend* is equal to the actual *general trend*.

$$MTS = \frac{1}{n}\sum_{i}\{1 \; if \; gtrend(y_i) = \; gtrend(\hat{y}_i); \;\; 0 \;\; otherwise\}$$

(11)

Note that the error metrics need to be minimized, whereas the Mean Trend Score metric needs to be maximized. Each of these metrics has been computed on the full test set (i.e. 20% of the used dataset). Since we found that most of the chunks on the datasets are relatively flat, each metric has also been computed on a **top_1% test set**, which corresponds to the first 1% tracks with highest general variation in the real values to be predicted. This score retrieves more accurately the ability of a model to perform on highly trending data (or highly untrending).

Note that for brevity of result presentation, we mostly report MAE and top 1% MAE in the following subsections, while all three measures have been computed in the experiments.

Prediction models

The aim of each prediction model is to accurately output a predicted time series of n_p days $\hat{y}$ from the same or all signals from the previous n_h history days.

Baseline

For a given signal s, the baseline model simply predicts $\hat{y}$ as a repetition of the last day in the provided history.

k-Nearest Neighbors (kNN)

In order to predict a signal *s*, the kNN approach uses a Nearest Neighbors model trained on the train-dataset of *s*. Then, the *k* nearest neighbors of the history regarding a given distance metrics are retrieved, and their n_p following days are averaged. The average is performed uniformly or weighted regarding the distance to the query signal. Besides the parameters n_h and n_p, the new parameters *k*, distance metric and weighted method are tested in our further experiments. The Python *scikit-learn* implementation has been used for *k-Nearest Neighbors*[20]***.***

LSTM

The Long Short-Term Memory [Hochreiter et al. 1997] or LSTM approach is a Deep-Learning based predictor, used to predict sequences of n-dimensional data. If correctly set up and trained, it can theoretically learn in more detail the patterns and trends of time series data.

We used the Python Deep Learning framework Keras for the LSTM experiments[21]. We evaluated different LSTM model architectures, from simple networks with one LSTM layer and one Dense output layer of size n_p, to more complex models with a stack of multiple LSTM layers. Figure 20, Figure 21 and Figure 22 depict the LSTM model

[20] https://dl.acm.org/citation.cfm?id=1246450
[21] https://keras.io/layers/recurrent/

Funded by the European Commission

definitions in Keras notation. Later experiment sections refer to the model names indicated in those figures:

- LSTM 32
- LSTM 256
- LSTM complex

All models use ReLU activations and a Dropout value of 0.2 on the LSTM layers. As optimizer the Adam approach was chosen, using MAE as the loss function. In univariate experiments, X_train is the signal history of 1 source signal (either Spotify popularity, or YouTube views, or YouTube likes, etc.). In a multivariate setup, multiple (or all) signals are used together as an input to the network for training. For the prediction, we have the choice of predicting univariate or multivariate output from a multivariate input.

```
def LSTM_simple_32(X_train, y_train, activation = 'relu', dropout = 0.2):
    model = Sequential()

    model.add(LSTM(32, activation=activation,
                       input_shape=(X_train.shape[1], 1)))
    model.add(Dropout(dropout))
    model.add(Dense(y_train.shape[1]))
    model.compile(optimizer='adam', loss='mae')
    return model
```

Figure 20 Definition of "LSTM 32" model in Keras notation.

```
def LSTM_simple_256(X_train, y_train, activation = 'relu', dropout = 0.2):
    model = Sequential()

    model.add(LSTM(256,activation=activation,input_shape=X_train.shape[1:]))
    model.add(Dropout(dropout))
    model.add(Dense(y_train.shape[1]))
    model.compile(optimizer='adam', loss='mae')
    return model
```

Figure 21 Definition of "LSTM 256" model in Keras notation.

```
def LSTM_complex(X_train, y_train, activation = 'relu', dropout = 0.2):
    model = Sequential()

    model.add(LSTM(256, activation=activation, return_sequences=True,
                        input_shape=X_train.shape[1:]))
    model.add(Dropout(dropout))

    model.add(LSTM(128, activation=activation, return_sequences=True))
    model.add(Dropout(dropout))

    model.add(LSTM(64, activation=activation, return_sequences=True))
    model.add(Dropout(dropout))

    model.add(LSTM(32, activation=activation))
    model.add(Dropout(dropout))

    model.add(Dense(32, activation=activation))
    model.add(Dense(y_train.shape[1]))

    model.compile(optimizer='adam', loss='mae')
    return model
```

Figure 22 Definition of "LSTM complex" model in Keras notation.

Preliminary Experiment on Spotify Charts (Dataset A)

On Dataset A (Spotify Charts Data) we carried out preliminary experiments in order to get a first insight into the performance of the k-Nearest Neighbors (kNN) vs. LSTM approaches.
Given a track's streams sequence, the goal was to forecast its number of streams for the 7 next days using the previous algorithms. The underlying assumption of this approach is that a relatively small history is enough to predict the future. Therefore we tried different input sequence lengths (n_h = {14, 28, 35}) and different models for prediction: a kNN model (k = 10) and 3 different LSTM models (complex LSTM and simple LSTM with 256 and 32 units, respectively) were trained with 50 epochs.

Data preparation

A track-wise min-max scaling to the range of 0 and 1 was performed first. Then the original data was cut into the input and target sequences. Each model is fed with a set of streaming sequences of the same length (14, 28 or 35 days) as input. For each of these sequences, the corresponding target is made of the **next 7 days** of data. As presented in section on "*Data splitting and chunking*", the track time-series are split into multiple chunks. Note that the number of chunks that results from the time range available in the source data decreases when increasing number of days of training history, as it can be seen in Table 11. Then, the data was split into a training and a test set, using Group K-Fold to ensure that none of the tracks had sequences in both training and test sets.

n_h	**14 days**	**28 days**	**35 days**
Train sequences	90,950	75,000	68,892
Test sequences	22,738	18,751	17,223

Table 11 Number of sequences ("chunks") available in training and test set for 3 different input lengths

Table 10:

Results

The results of the experiments are presented in Table 12. We used the MAE to compare the predictions.

n_h	**14 days**	**28 days**	**35 days**
kNN (k = 10)	0.04546	0.03768	0.03873
LSTM (32 units)	0.04309	0.03342	0.03293
LSTM (256 units)	**0.03961**	**0.03062**	**0.02933**
LSTM (complex)	0.05189	0.04912	0.04048

Table 12 Comparison of k-NN and LSTM models for different nh and np = 7 days (MAE)

Funded by the European Commission

From the above results, the simple LSTM models (with 256 LSTM units) seem to be the best approaches: the LSTM with 256 units performs best followed by LSTM with 32 units; kNN follows next with the complex LSTM performing the worst.

The LSTM models seem to predict the shape of the curve and capture the seasonality quite well. It looks that they have the same overall behaviour regardless of the number of days considered for history. Although the kNN algorithm appears to be slightly less accurate, it looks consistent. Figure 23 and Figure 24 show some predictions for two tracks of dataset A.

The prediction seems more difficult for some particular examples where the signal has a significant unexpected variation within the prediction time range (Figure 25). This seems to be independent of the model used.

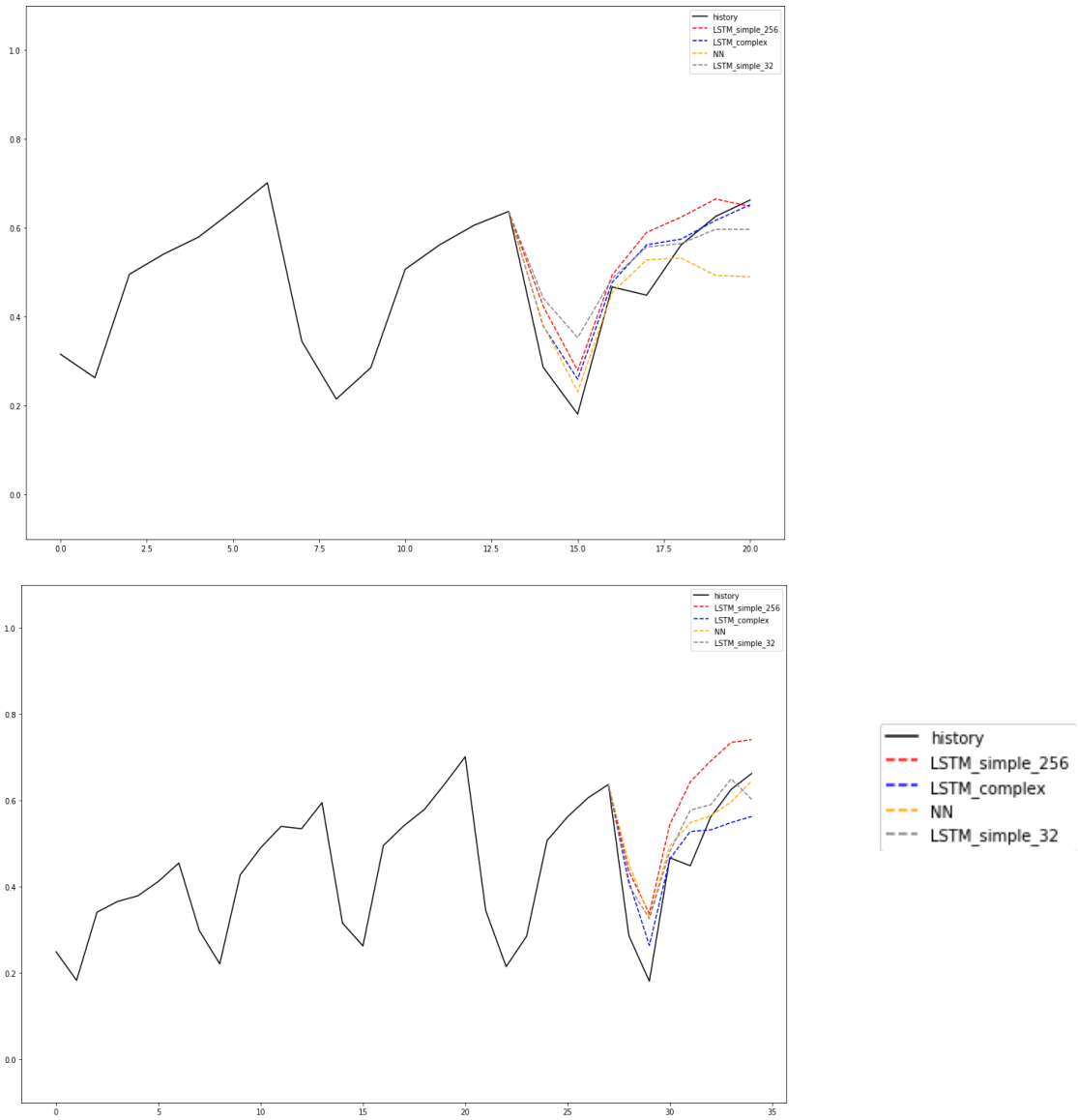


Figure 23 Juice WRLD's "Black & White" Spotify streams predictions
Top : n_h=14, n_p=7
Bottom : n_h=28, n_p=7

Funded by the European Commission

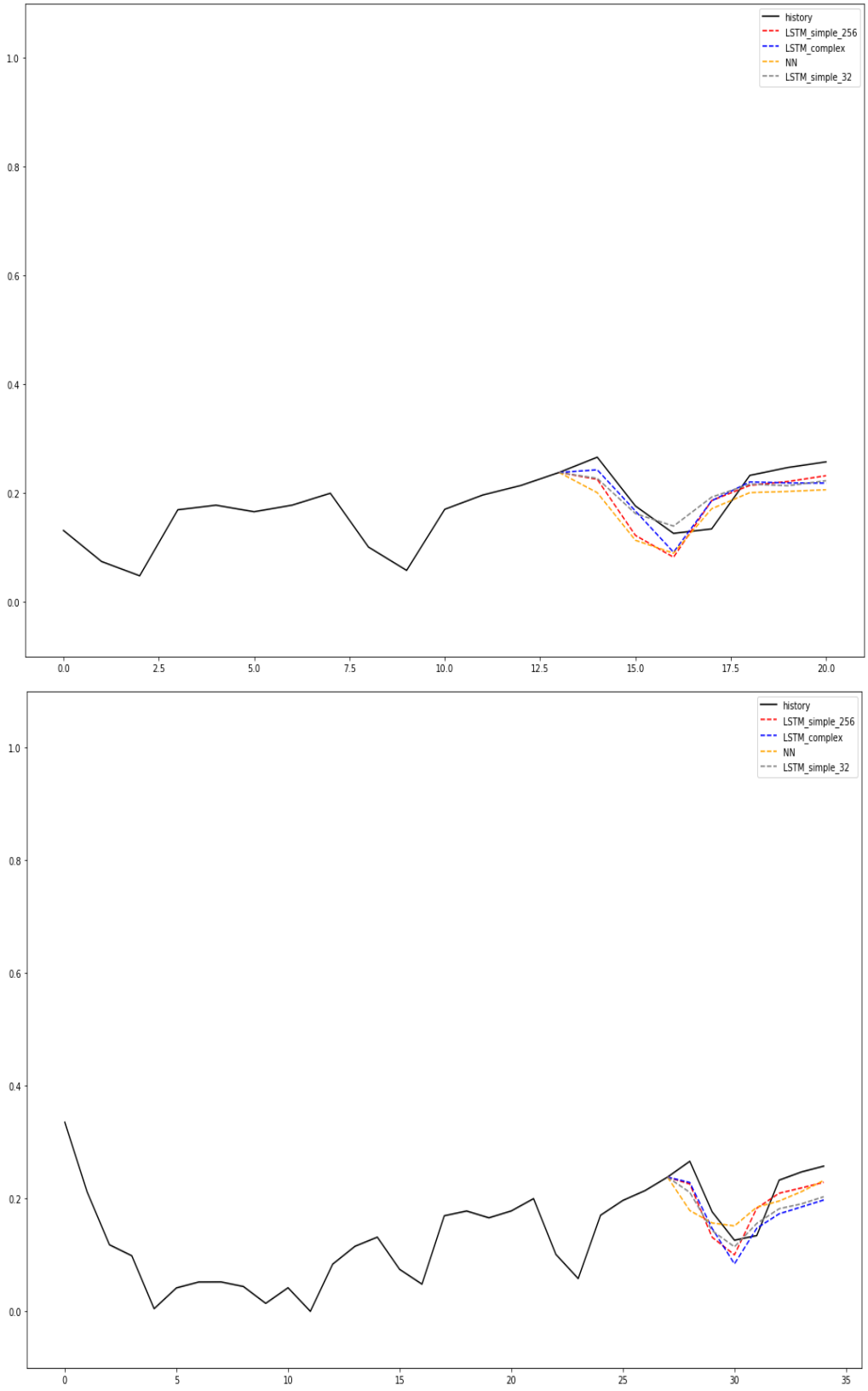


Figure 24 A Boogie Wit da Hoodie's "Swervin (ft. 6ix9ine)" Spotify streams predictions
Top : $n_h = 14$, $n_p = 7$
Bottom : $n_h = 28$, $n_p = 7$

Funded by the European Commission

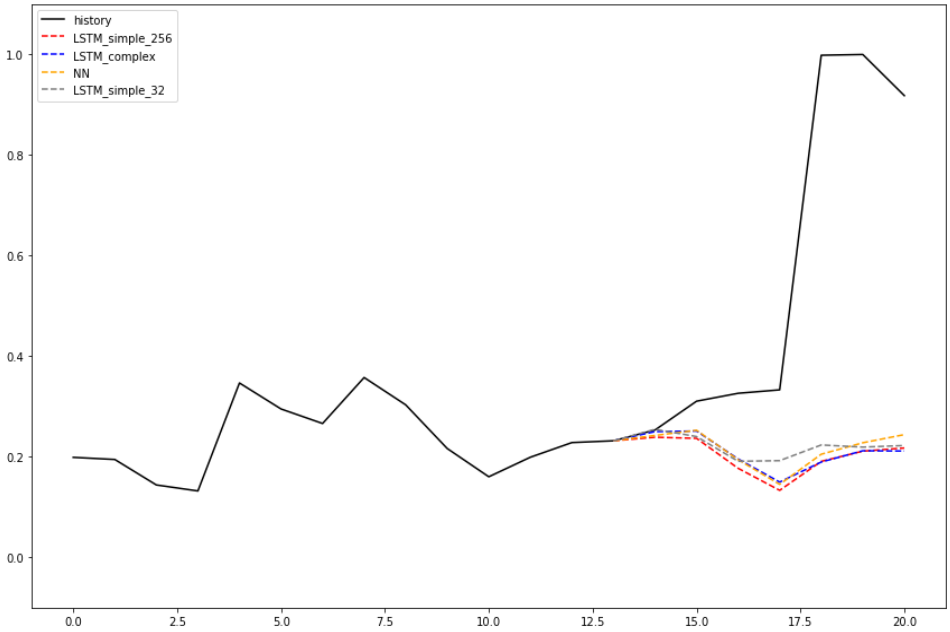


Figure 25 Lady Gaga & Bradley Cooper's "Shallow" Spotify streams prediction (nh = 14, np = 7)

Conclusions

This preliminary experiment helped us get some insights into song popularity prediction thanks to the comparison between kNN and LSTM approaches. It also helped with figuring out and agreeing on the data preparation for such a problem.

We obtained **slightly better results with LSTM** models, namely with the 256 units model. We could observe that LSTM models learned recurring patterns pretty well (weekly seasonality), and that the greater number of input days the better MAE, which is not necessarily true for the kNN model. The Nearest Neighbours approach shows interesting results too, as it seems to capture the seasonality as well and performs better than our complex LSTM architecture.

These simple approaches had limitations regarding the prediction of unusual variations which we could have tried to handle by taking events in consideration[22].

Although we've obtained encouraging results, one has to keep in mind that dataset A is very different from datasets B. Indeed, the former contains only popular songs and raw data (Spotify stream count), whereas datasets B include more complex quantities (e.g. Spotify popularity).

Experiments on Crawled Signals for SYB (Dataset B)

In the following subsections we describe experiments carried out on the crawled signals for the tracks of the SYB dataset (datasets B1 and B2).

Comparison of kNN and LSTM

The first experiment is a comparison of the kNN approach with an LSTM approach.

[22] Data about important events, e.g. a marketing campaign or a concert, could be provided by the labels, but would need to be added manually to the data sets.

Dataset B2 with 60 days of data for five different signals was used in this experiment: **Deezer rank, Spotify popularity, YouTube views and likes** (log derivative) and **BMAT daily airplay count.**

For this experiment we performed 7 days prediction based on 14 days history sequences **(n_h = 14, n_p = 7)** with a step of 7 day**)**. This results in 169,860 training sequences and 42,468 test sequences for 35,988 different tracks. For all signals a global min/max normalization to the (0,1) range was performed.

Comparably to the early experiments on Spotify charts data, we kept the kNN approach along with LSTM for comparison. For kNN, k = 10 neighbours were chosen. For LSTM, a simple model with 256 LSTM units and a Dense output layer of n_p = 7 days was chosen and trained for 50 epochs. Although we performed the training for 50 epochs, we used early stopping (halting the training if the model didn't improve within 5 epochs) and kept the best model (i.e. the one with the smallest validation MAE) for each signal.

Performance comparison

Legend for the following tables:

grey: not better than baseline

bold: better result comparing the different approaches

MAE scores

	Baseline		kNN (k = 10)		LSTM simple (256 units)	
Signal	MAE	MAE (top 1%)	MAE	MAE (top 1%)	MAE	MAE (top 1%)
Deezer Rank	0.012504	0.053378	0.014333	**0.052096**	**0.012342**	0.053060
Spotify Popularity	0.002766	0.007532	0.002790	0.011058	0.003419	0.008628
Youtube Likes	0.023461	0.125824	0.022155	**0.085542**	**0.019526**	0.088806
Youtube Views	0.012801	0.052095	0.011572	**0.043852**	**0.011414**	0.045984
BMAT Airplays	0.001332	0.025355	**0.001098**	**0.022492**	0.001124	0.023063

Table 13 Comparison of kNN and LSTM MAE scores on SYB dataset for different signals (bold = best performance, grey = not better than baseline)

Note that for Spotify Popularity, none of the approaches has led to better MAE than the baseline prediction.

From these results (Table 13) one can see that LSTM approach seem to lead to slightly better predictions. Indeed, it has the best MAE score for 3 signals out of 4. In addition, for all signals, the MAE scores over the top 1% tracks (i.e. the 1% of tracks with biggest variations over the prediction time range) are much higher than the overall MAE scores. As in the preliminary experiment, it seems difficult to predict unexpected variations in the

signals. However, kNN algorithm appeared to be the best approach to predict this kind of variations.

MSE scores

	Baseline		kNN (k = 10)		LSTM simple (256 units)	
Signal	MSE	MSE (top 1%)	MSE	MSE (top 1%)	MSE	MSE (top 1%)
Deezer Rank	0.000560	0.005742	0.000537	**0.005246**	**0.000478**	0.005675
Spotify Popularity	0.000034	0.000397	**0.0000292**	0.000909	0.0000293	0.000473
Youtube Likes	0.002080	0.027575	0.001479	**0.011835**	**0.001391**	0.014992
Youtube Views	0.000472	0.004786	0.000392	**0.003537**	**0.000353**	0.003752
BMAT Airplays	0.000054	0.001634	**0.000038**	**0.001504**	0.000039	0.001543

Table 14 Comparison of kNN and LSTM MSE scores on SYB dataset for different signals (bold = best performance, grey = not better than baseline).

This comparison of the MSE scores presented in Table 14 seems to confirm the previous results. The LSTM approach offers better results for the majority of the signals and kNN performs better with sequences with high increases within the prediction range. Therefore, our LSTM model looks rather good at capturing some usual behaviours of the signals (as seen with the preliminary experiment on Spotify Charts data) whereas less predictable curves are better approached with a simple nearest neighbours average.

With BMAT air play count, the kNN algorithm is better considering both MAE and MSE scores. It has also a better MSE score with Spotify Popularity. However, one has to consider that the data for each of these signals contains more than 50% of completely flat sequences. Figure 26 and Figure 27 show YouTube Views and Deezer Rank prediction examples.

Funded by the European Commission

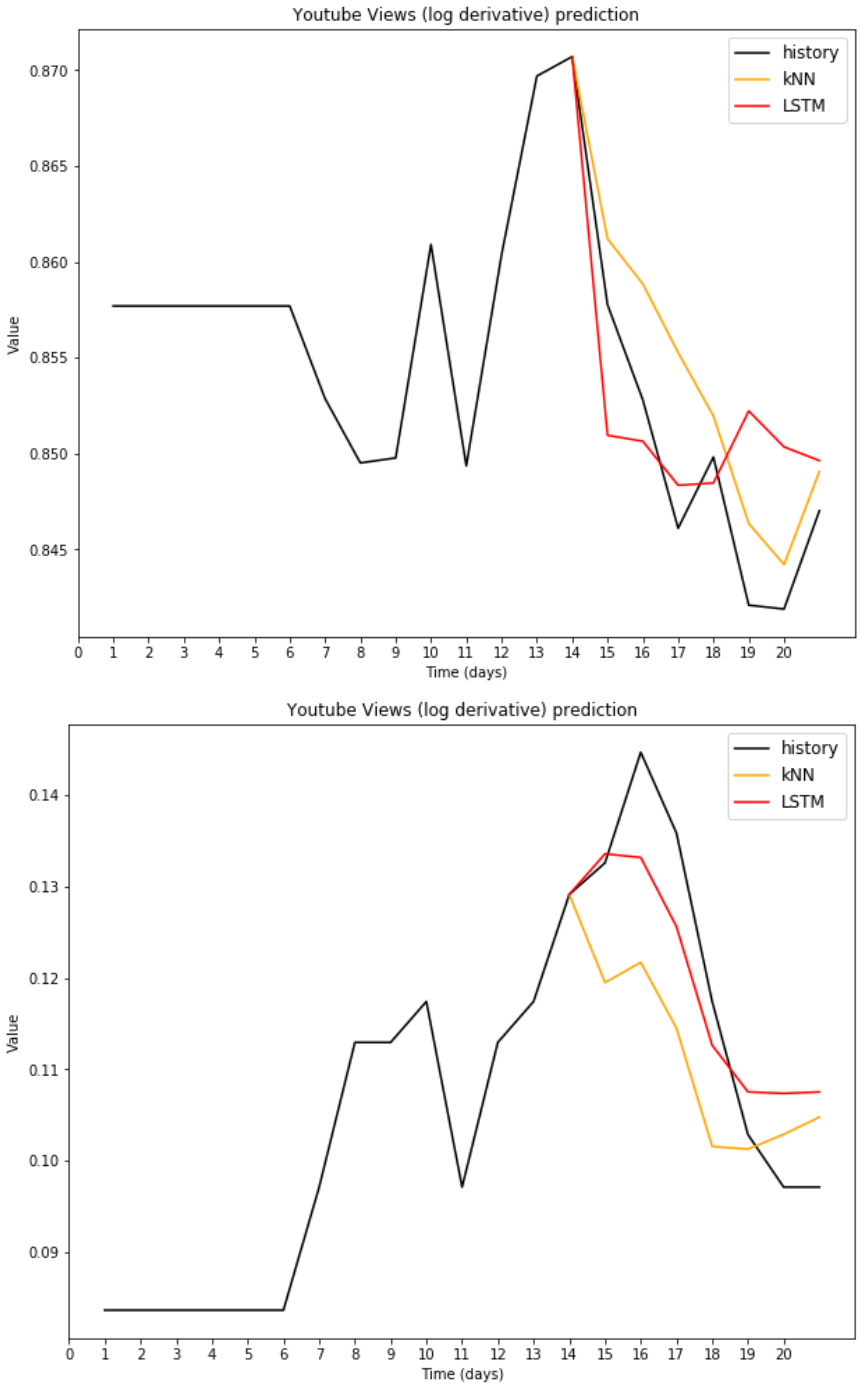


Figure 26 YouTube Views predictions for 2 sequences (nh = 14, np = 7)

Funded by the European Commission

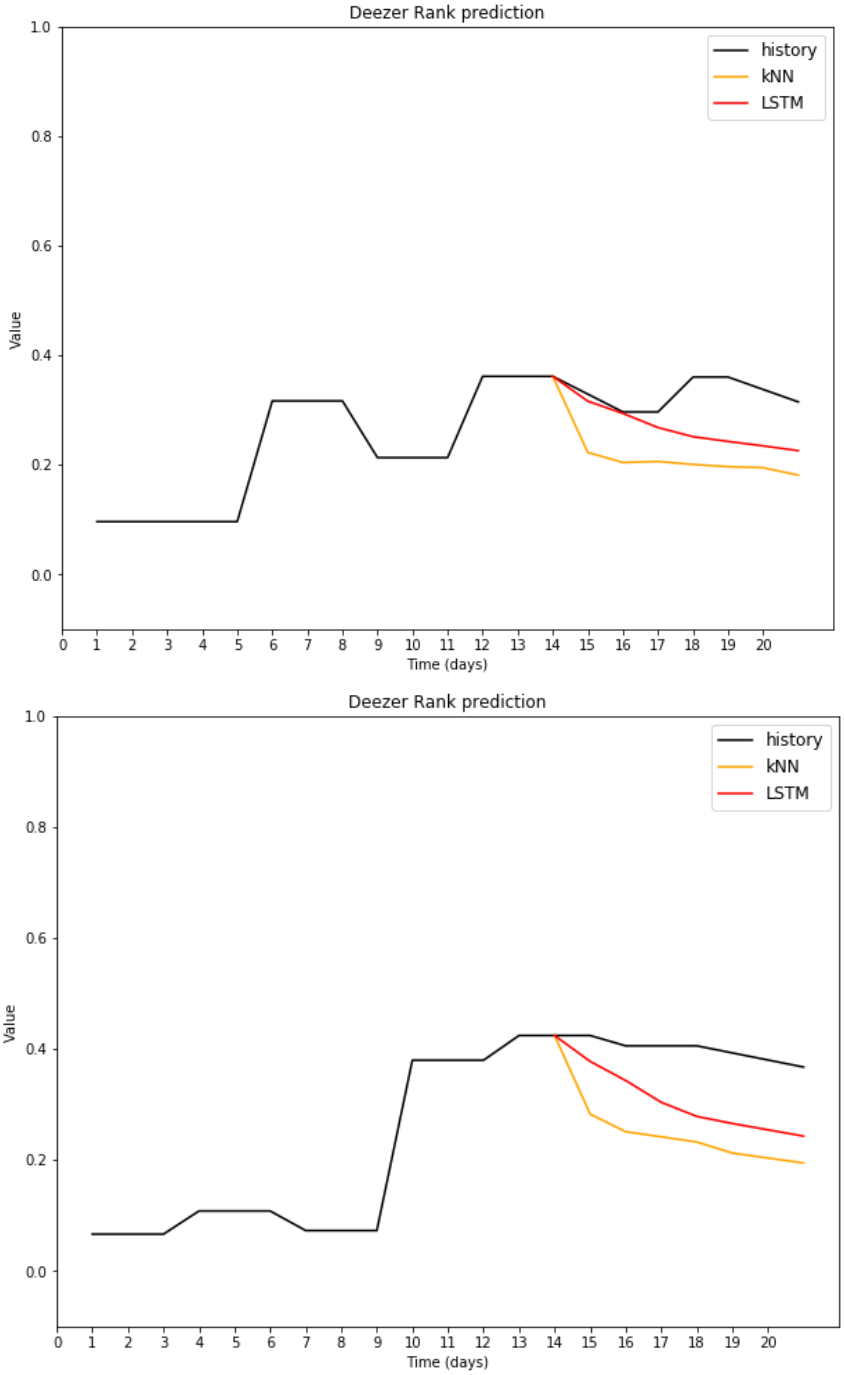


Figure 27 - Deezer Rank predictions for 2 sequences (nh = 14, np = 7)

Computational comparison

For each signal we measured the computation time of each algorithm, the results are shown in Table 15 below. The computation time varies heavily between the different input signals for the kNN approaches, despite the same amount of data used. This is assumed to be related to different sparsity in the data (e.g. in BMAT airplays 86% of the sequences have all 0s). The LSTM time measurement was taken for the best model, whenever it was reached within 50 epochs.

Funded by the European Commission

	Computation time (minutes)	
Signal	**kNN (*k* = 10)**	**LSTM**
Deezer Rank	0.28	50.35
Spotify Popularity	0.19	43.33
Youtube Likes	1.55	60.82
Youtube Views	0.28	64.6
BMAT Airplays	3.06	6.23

Table 15 Computation time comparison (in minutes).

On average, our **LSTM** model is more than **135 times slower** than the kNN algorithm, the lowest ratios being about 2 (BMAT) and the highest ratios reaching up to 230.

Conclusions

As we have seen from the results, LSTM seems slightly better than the kNN algorithm. Although the prediction is satisfying from some signals, our models still have difficulty dealing with some cases. Actually, the data contained a lot of completely flat sequences (e.g. 50% for Spotify popularity, 86% for BMAT air playcount). Hence the training of the LSTM models might not have been optimal. We plan to remove the excess of flat sequences for a better-balanced dataset in the future.

From a computational point of view, one can observe that the LSTM approach is much slower than the kNN algorithm. Since the ratio of computation time between the two approaches can reach up to 230, we might have to consider a computation time-accuracy trade-off.

Univariate vs. Multivariate LSTM

From the previous experiment we saw that LSTM models are capable of learning the time series prediction well in some cases. So far, we have done experiments that predict each signal *individually*, from its past to its future. It seems natural to combine the various input signals to train a model that is informed by all available sources. Music streaming data is supposed to be correlated to some extent, even when originating from different sources. E.g. when a track rises heavily on YouTube, also its rank on Deezer or popularity on Spotify are likely to rise.[23]

Approaches

While a multivariate setup is more difficult to achieve with k-Nearest Neighbors it is easily possible to set up the input to an LSTM network with a multivariate time series. Thus, in

[23] We intended to perform a time lag analysis, e.g. to observe with how much delay another source would respond to an increase or decrease on one source. The available data, however, did not include sufficient data to perform this analysis properly. It is planned for the future.

Funded by the European Commission

the following set of LSTM experiments the training data was set up to combine the following five different signals into a single training set:

- Spotify popularity
- Deezer rank
- BMAT airplay counts
- YouTube views (log derivative)
- YouTube likes (log derivative)

We used the 60 days data (**Dataset B2**) to train with n_h = 14 days of history to predict n_p = 7 days. The chunking approach was used to increase the number of training samples. A step of 7 days was chosen, except for the day by day experiments, where the step size of 3 days was used to create chunks for training and 7 days was kept for the test data set (in order to create a larger number of training samples for single-day training). In addition, an 80:20 % track-wise training/test set split was applied (so that no training chunks appear in the test set). See Table 16 for the training/test set sizes in terms of number of chunks.

Approach	X_train	X_test	y_train	y_test
Univariate 7-days	(169860x14x1)	(42468x14x1)	(169860x7)	(42468x7)
Univariate day-by-day	(452960x14x1)	(42468x14x1)	(452960x1)	(42468x7)
Multivariate single signal prediction 7-days	(169860x14x5)	(42468x14x5)	(169860x7)	(42468x7)
Multivariate multi signal prediction day-by-day	(452960x14x5)	(42468x14x5)	(452960x5)	(42468x7x5)

Table 16 Train and test set sizes for X (input) and y (output): (number of chunks x number of days x signals)

We compare the following setups:

- Univariate training and 7-day prediction (each signal individually, as in the section before)
- Univariate training and day-by-day prediction
- Multivariate training with a single signal prediction
- Multivariate training with a multivariate day-by-day time series prediction

The **LSTM 256 model** presented in the models subsection was used. All models were trained using 50 epochs as the main setting for training; however, an early stopping approach was applied, stopping the training if the validation loss did not improve after 5 epochs anymore (using the model with best validation loss).

The univariate approach (same as used in the previous subsection) has been included as a baseline to compare the multivariate approach with it. In addition, we compare the 7-day prediction to a day-by-day prediction. The difference is as follows:

The regular LSTM approach as presented in the previous subsection applies a dense layer as an output layer with *n* number of units for *n* days to predict. In the day-by-day approach, this output layer is defined with just one output unit, to predict the next day

Funded by the European Commission

performance. In this approach, a prediction is performed by predicting always one day, adding the prediction iteratively to *X_test* to predict the next day. The evaluation was performed on 7 days, as with the other setups. However, any number of days *n* can be predicted with this approach. In the Multivariate day-by-day training, the output dimension is 1 day x 5 signals, and all output signals are predicted at the same time.

Results

We are measuring MAE and MSE as described before, for the full data and the top 1% of most highly increasing tracks in the prediction timeframe (7 last days) and are applying a baseline experiment predicting the last popularity value flat. For brevity, we present MAE and top 1% MAE only. In the following tables (Table 17 & Table 18), a grey box marks a result that is not better than baseline, and therefore not to be considered. Comparing the different approaches, we mark the best result in bold font.

	univariate: 7-day prediction		univariate: day by day prediction	
Predicted signal	**MAE**	**MAE top1%**	**MAE**	**MAE top1%**
BMAT airplays	0.0011654	**0.0231272**	**0.0011304**	0.0244511
Deezer rank	**0.0124502**	0.0533098	0.0151471	**0.0524089**
Spotify popularity	0.0032984	0.0095173	0.0039045	0.0080893
YouTube views	**0.0113745**	**0.0456901**	0.0124742	0.0510137
YouTube likes	**0.0197632**	**0.0832827**	0.0272401	0.1362009

Table 17 Univariate LSTM: comparison of Dense 7-day prediction vs. Dense 1 day-by-day prediction.

	univariate: 7-day prediction		multivariate: 7-day single signal prediction		multivariate: day-by-day multi signal prediction	
Predicted signal	**MAE**	**MAE top1%**	**MAE**	**MAE top1%**	**MAE**	**MAE top1%**
BMAT airplays	0.0011654	**0.0231272**	**0.0011040**	0.0231762	0.0014932	0.0305338
Deezer rank	**0.0124502**	0.0533098	**0.0124501**	**0.0528097**	0.0225435	0.0629664
Spotify popularity	0.0032984	0.0095173	0.0037954	0.0095236	0.0093602	0.0169521
Youtube views	**0.0113745**	0.0456901	0.0116221	0.0468568	0.0203874	**0.0426316**
Youtube likes	0.0197632	0.0832827	**0.0190840**	**0.0764448**	0.0249124	0.1063755

Table 18 Univariate vs. Multivariate LSTM, with single signal (7 day) and multi-signal prediction (day-by-day).

In the univariate LSTM comparison of 7 day vs. iterative day-by-day prediction, the direct 7-day prediction approach performed better in most cases. Only for the top 1% of tracks by Deezer rank the prediction performed better day-wise. Note that the Spotify popularity could not be predicted reliably at all, with none of the approaches.

Funded by the European Commission

Conclusions

Comparing the multivariate LSTM approach to the univariate one, we see that informing the neural network with all sources at the same time brought slight gains for the 7-day prediction.

Here we also tried the day-by-day prediction as well, which would have two huge advantages: *Any* number of days could be predicted, and *all* signals would be predicted at once. However, this approach did not perform better than baseline, with the exception of YouTube views on the top 1% of tracks. Note that again the Spotify popularity could not be predicted reliably at all, with none of the approaches.

In the multivariate 7-day prediction approach, one LSTM model is trained for each output signal to be predicted separately by each model (i.e. 5 models in total, however, all of them are fed with the same multivariate input). This seems the most promising approach for now but demands quite large computational resources.

Comparison of kNN parameters

Focusing on the faster approach of k-Nearest Neighbour again, we carried out a range of experiments to optimize some parameters with kNN models on **Dataset B2**.

Different time periods analysis (number of days of history and prediction)

First, we looked at the impact on the MAE when variating the **number of days** used in the history n_h and the number of days used for predictions n_p, on each signal individually. Grey cells indicate results not better than the baseline, while bold results are the best result for n days predicted (per column)

	n days predicted					
n days history	**1**	**3**	**7**	**14**	**21**	**28**
1	0.00134	0.00131	0.00134	0.00140	0.00176	0.00211
3	0.00112	0.00114	0.00115	0.00121	0.00134	0.00159
7	0.00108	0.00108	0.00112	0.00120	0.00126	0.00137
14	0.00108	0.00108	0.00113	0.00119	0.00129	0.00129
21	0.00109	0.00107	0.00109	0.00111	0.00112	0.00116
28	**0.00109**	**0.00105**	**0.00106**	**0.00109**	**0.00110**	**0.00113**

Table 19 Number of days in kNN - MAE for BMAT Airplay Count.

Funded by the European Commission

	n days predicted					
n days history	**1**	**3**	**7**	**14**	**21**	**28**
1	0.00668	0.01147	0.01623	0.02011	0.02206	0.02330
3	**0.00542**	**0.01023**	0.01516	0.01898	0.02081	0.02193
7	0.00610	0.01046	**0.01464**	0.01793	0.01963	0.02084
14	0.00771	0.01140	0.01488	0.01758	0.01913	0.02012
21	0.00892	0.01203	0.01511	0.01754	0.01891	0.01990
28	0.00969	0.01247	0.01520	**0.01743**	**0.01881**	**0.01977**

Table 20 Number of days in kNN - MAE for Deezer Rank.

	n days predicted					
n days history	**1**	**3**	**7**	**14**	**21**	**28**
1	0.00126	0.00201	0.00323	0.00522	0.00686	0.00684
3	**0.00107**	0.00218	0.00351	0.00541	0.00722	0.00698
7	0.00125	0.00195	0.00300	0.00472	0.00629	0.00631
14	0.00135	**0.00189**	**0.00283**	**0.00460**	0.00620	0.00590
21	0.00148	0.00203	0.00300	0.00469	0.00610	**0.00530**
28	0.00171	0.00221	0.00320	0.00456	**0.00532**	0.00454

Table 21 Number of days in kNN - MAE for Spotify popularity

Funded by the European Commission

	n days predicted					
n days history	**1**	**3**	**7**	**14**	**21**	**28**
1	0.00640	0.00961	0.01316	0.01620	0.01742	0.02071
3	0.00670	0.00874	0.01249	0.01581	0.01651	0.01963
7	**0.00526**	**0.00849**	0.01209	0.01423	0.01574	0.01779
14	0.00640	0.00882	**0.01190**	**0.01394**	0.01569	0.01636
21	0.00751	0.00966	0.01221	0.01395	0.01506	**0.01584**
28	0.00814	0.01015	0.01237	0.01424	**0.01498**	0.01589

Table 22 Number of days in kNN: - MAE for log- derivated YouTube view count.

	n days predicted					
n days history	**1**	**3**	**7**	**14**	**21**	**28**
1	0.01261	0.01763	0.02597	0.03472	0.03700	0.04012
3	0.00973	**0.01470**	0.02576	0.03536	0.03349	0.04150
7	**0.00894**	0.01481	0.02253	0.03905	0.03255	0.03582
14	0.01065	0.01580	0.02279	0.02568	0.02789	0.03006
21	0.01194	0.01614	0.02099	0.02501	0.02513	**0.02671**
28	0.01255	0.01623	**0.02083**	**0.02404**	**0.02440**	0.02728

Table 23 Number of days in kNN - MAE for log-derivated youtube like count.

Funded by the European Commission

From Table 19 to Table 23, we can note that each signal presents a different behaviour. However, we can underline some general tendency:

- When **predicting a low number of days**, it seems to be better to use **only a low number of days of history.** Using too much history might induce too much noise or create some distance artifact due to the high number of dimensions.
- Using **28 days to predict 21 days seems to be performing well** in general, and better than the baseline (except for *Deezer rank,* where all the kNN predictions are worse than the baseline).

Number of neighbors (k)

One of the most important parameters in the *k-Nearest-Neighbors approach* is the number of neighbors to be averaged for the prediction (*k*). Using n_h=28 and n_p=1, we tested varying the number of neighbors to be averaged for each metric (see Figure 28).

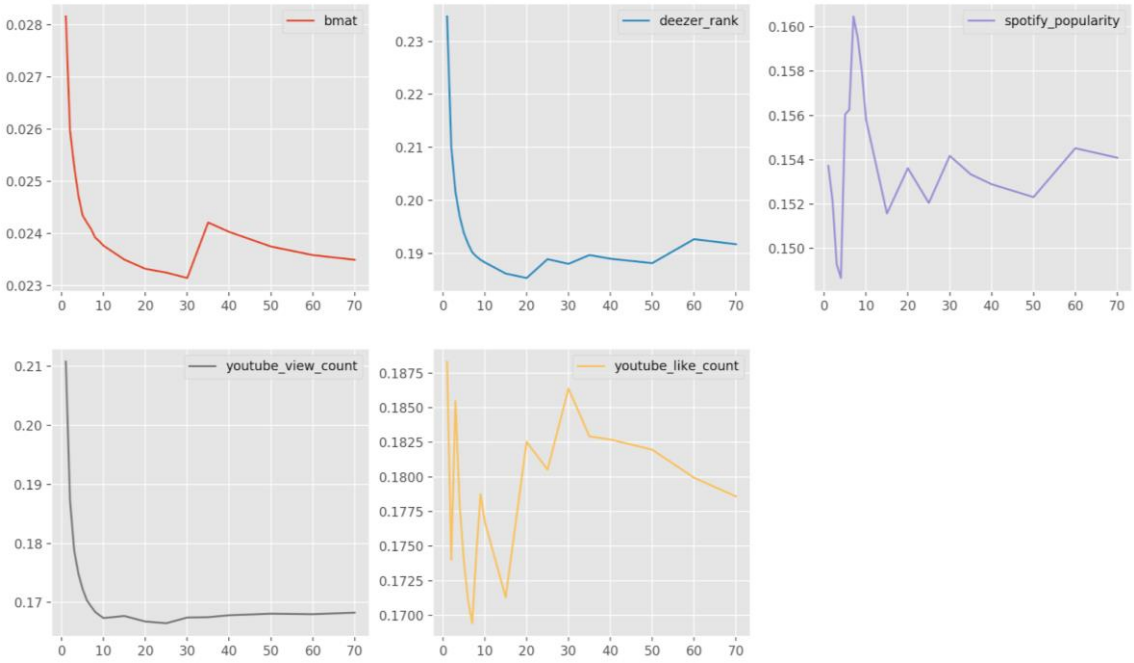


Figure 28 MAE regarding the number of neighbours averaged.

We then can then retrieve the number of neighbours giving us the best *MAE* for each metric (see Table 24).

Signal	Best *k* value
BMAT airplays	34
Deezer rank	21
Spotify popularity	4
Youtube views	27
Youtube likes	7

Table 24 Number of neighbours giving the minimum average.

Funded by the European Commission

The range of the variations that we can graphically see for *youtube_like_count* and *spotify_popularity* being relatively small, it is likely that the numbers of neighbours does not influence significantly the model performance for those signals. Nevertheless, having a number of neighbours between 20 and 30 seems to be a good fit overall. We decided to keep a general number of neighbours *k*=25 for our further implementations.

Comparison of distance metric

Another important parameter in the *k-Nearest-Neighbours* approach is the distance metric used to compute the neighbors. We looked at the performance effect on the MAE for each signal when using the distance metrics *cosine, Euclidean, cityblock and correlation* (see Figure 29).

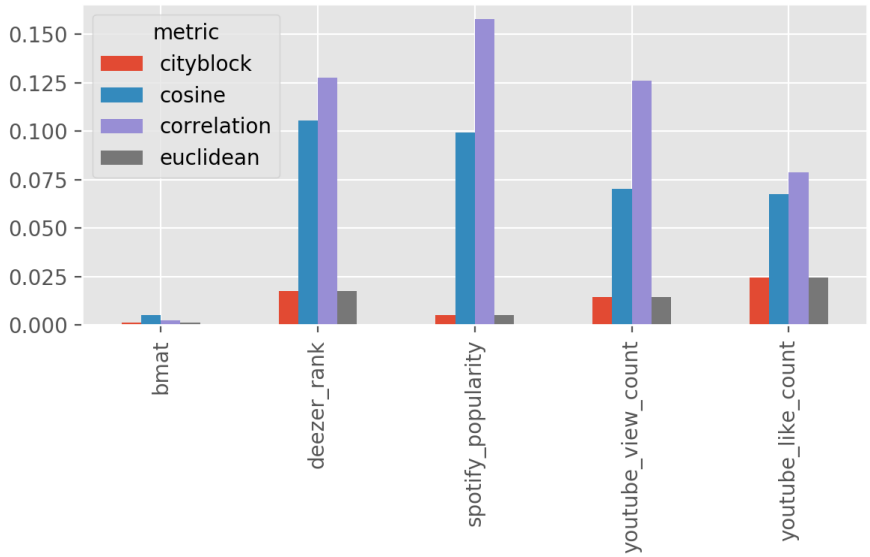


Figure 29 Test MAE regarding the distance metric used for kNN.

Since the **cityblock** distance gave in general a better MAE, we decided to keep this metric for further implementations. We also tested the effect of a linear average, or a weighted average regarding the actual distance given by the kNN algorithm, but the changes were not significant.

Conclusions for Track Popularity Prediction and Implementation

Based on the results presented in the previous sections, we conclude that LSTM has a slightly better performance in some cases, but k-Nearest Neighbours performs almost as good as LSTM in most of the cases (sometimes better) (as measured by MAE and MAE on the top 1% most increasing tracks).

The multivariate LSTM approach, incorporating multiple sources of popularity indicators into one model, shows potential, but is also only slightly better than individual LSTM models per source, and only in some cases.

We noted that Spotify popularity at this point is not possible to predict, not even with a multivariate LSTM model. This is likely due to Spotify popularity index (0-100) being very stable most of the time, leading to more or less flat curves in the data.

Funded by the European Commission

Given the drastically heavier computation needs of LSTM models (100-200 times slower than kNN), we **decided to go forward with implementing the *k- Nearest-Neighbors* (kNN) approach** for the realization of track popularity prediction in FuturePulse, for the time being.

Also, from the training/validation loss curves of the LSTM models we observe that these models do not learn stable and well enough yet. When more data (e.g. more tracks, more history, more sources) become available, LSTM models will be reconsidered.

In order to optimize the kNN models, we performed a detailed evaluation of the ideal number of *k* neighbours, and the time period of history to be used and ideal time period of number of days to be predicted. This was presented in the last subsection. Using **28 days to predict 21 days** seems to be performing well in general; that is, we are currently able to **predict three weeks into the future**, with the kNN approach.

The FuturePulse use case partners emphasized there is strong interest in getting each source signal likely performance in the future; therefore we will perform the prediction per available source signal, and then aggregate both the history and the predicted signals to a global popularity indicator, as presented in the section on Track Popularity Aggregation and Estimation.

Track popularity estimation and prediction is available as a service on a FuturePulse internal API by the end of M24 and will be next integrated in the FuturePulse platform.

Funded by the European Commission

4 Artist Popularity Estimation and Prediction

4.1 Artist Metrics Prediction

To support part of the RL_REQ#2 (A combined visual timeline for streaming statistics of an artist), we track web-based popularity metrics for 2349 PGM artists. More precisely, we monitor the metrics presented in Table 25 for each artist from May 2018 until today (May 2019). The aim of the work presented in this section, is to display a prediction trajectory for the next few days (1-2 weeks) along with the timeline of preceded values for each metric.

Source	Metric
Deezer	artist fans
Facebook	fan count
	mentions
Last.fm	artist listener count
	artist play count (the last 30 days)
Soundcloud	artist followers count
Spotify	followers
	popularity
Twitter	user followers
	user listed count
YouTube	channel subscribers
	channel views (the last 30 days)

Table 25 Sources and metrics for artist popularity.

In order to select the optimal forecasting model with regards to our dataset we conducted a comparative study among the following models:

- Long-short term memory network (LSTM)
- Decision tree
- Random forest
- Gaussian process
- Support vector machine for regression (SVR)
- Gradient boosting
- K nearest neighbors

The models are trained and evaluated on every single metric per artist. We used the first differences (derivative) of the time series in order to achieve stationarity and normalized it in [-1, 1]. We used 21 days as input features and one day ahead as target value. The evaluation of the models is conducted on the test set, namely the last 20% of each time series, for their performance in multistep prediction 7 days ahead, using the mean absolute error score.

Funded by the European Commission

Model	Details
Long-short term memory network	Input (21 units) → LSTM (100 units, elu activation) → batch normalization → Dense (1 unit, tanh activation) 20% validation split, 400 epochs, model checkpoint for best validation loss
Decision tree	At least 5 samples per leaf
Random forest	300 estimators, at least 5 samples per leaf
Gaussian process	RBF kernel
SVM	RBF kernel non-linearity
Gradient boosting	default sklearn params[24]
K nearest neighbors	default sklearn params[25]

Table 26 Predictive models' hyperparameter values and structure.

Details regarding the models' parameters are provided in Table 26 and the corresponding results are presented in Table 27 and Table 28. The training of the LSTM proved to be computationally costly to a non-affordable degree having results for only 131 artists in 4 days. In Table 27 we present the ranking of the rest 6 models on the whole dataset, while in Table 28. we present the ranking of all models on the first 131 artists. We also present the performance of k nearest neighbours in one step ahead prediction in Figure 30.

In both tables (Table 27 & Table 28) the supremacy of k nearest neighbors is apparent, hence we opt for this model for the metrics predictions shown in the user interface. The most promising of all, being the LSTM, is not the best in this admittedly small fraction of the dataset and has non-acceptable computational cost. We also tried training one LSTM model for all artists per metric, but it failed to accurately predict future values on account of the dataset's heterogeneity and the sparsity of many artists. Another option for future evaluation would be to train one LSTM model for each artist with all metrics' past as input and all metrics' current value as target. Such a model would be a lot faster in training, considering predictions for the whole dataset.

Model	Average MAE
1. k nearest neighbors	MAE=0.0031, std=0.0110
2. random forest	MAE=0.0036, std=0.0149
3. decision tree	MAE=0.0039, std=0.0169
4. Gaussian process	MAE=0.0042, std=0.0096
5. gradient boosting	MAE=0.0055, std=0.0520
6. SVR	MAE=0.0114, std=0.0379

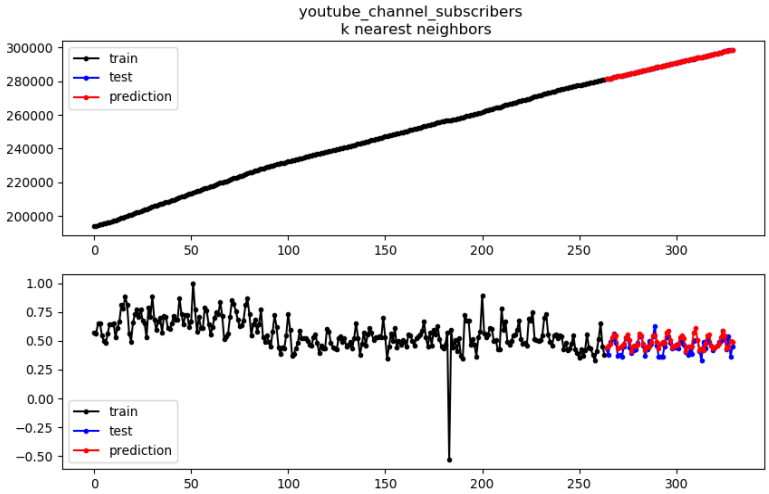
Table 27 Ranking of 6 predictive models, in terms of average mean absolute error, with regard to the whole dataset.

[24] scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html
[25] scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html

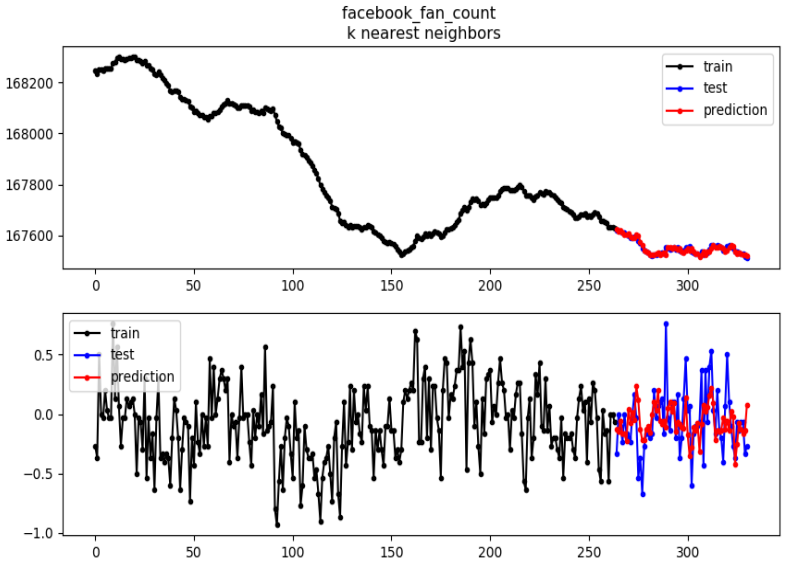
Funded by the European Commission

Model	Average MAE
1. k nearest neighbors	MAE=0.0015, std=0.0033
2. gradient boosting	MAE=0.0015, std=0.0035
3. random forest	MAE=0.0015, std=0.0038
4. LSTM	MAE=0.0016, std=0.0051
5. decision tree	MAE=0.0017, std=0.0047
6. gaussian process	MAE=0.0028, std=0.0050
7. SVR	MAE=0.0048, std=0.0150

Table 28 Ranking of all 7 predictive models, in terms of average mean absolute error, with regard to 131 artists.



(a)



(b)

Figure 30 K-Nearest Neighbors forecasting performance. (a) Upper: YouTube channel subscribers, lower: first differences. (b) Upper: number of Facebook fans, lower: first differences.

Funded by the European Commission

4.2 Geometric Artist Popularity

4.2.1 Background

There have been many attempts, in academic literature, to determine artist popularity or music popularity in general, through online popularity metrics or the traditional chart rankings [Grace et al. 2008; Koenigstein and Shavitt 2009; Schedl et al. 2010; Schedl 2011; Bellogin et al. 2013; Kim et al. 2014; Mesnage et al. 2015; Ren et al. 2016]. However, the determination of an evaluation method for such popularity scores remains a challenge as no general agreement regarding an acceptable ground truth has been established. This leads researchers to evaluation through comparison with several other existing popularity metrics such as Spotify popularity, page counts and the charts. In Table 29 we present the evaluation methods (and ground truth) followed by research papers for their proposed popularity scores.

Paper	Evaluation method / ground truth (popularity score)
Grace et al. 2008	One popularity proxy evaluated by user study (sentiment of comments on artists' pages in MySpace)
Koenigstein and Shavitt 2009	One popularity proxy compared with billboard hot 100 (P2P search queries from Gnutella)
Schedl et al. 2010	Four popularity proxies compared pairwise (page counts Google-Exalead, Twitter posts, shared folders in Gnutella P2P, Last.fm playcounts)
Schedl 2011	One popularity proxy compared with Last.fm's charts (number of tweets with regards to an artist)
Bellogin et al. 2013	Four popularity proxies compared pairwise (EchoNest score, Spotify popularity, number of Last.fm playcounts, number of clicks related to an artist from Bit.ly)
Kim et al. 2014	One popularity proxy used to predict Billboard ranks (number of Tweets)

Table 29 Ground truth and evaluation methods for artist popularity proposed in research literature.

Moreover, according to our knowledge all popularity scores that have been proposed until today are univariate, while the score that we propose herein is the first to combine several diverse sources and metrics of artist popularity in order to summarize the whole picture for a certain artist. Although the most natural choice for metric aggregation is a simple average, the handling of many different sources is clearly not obvious and might be useful to evaluate and compare other non-linear methods as well. Our method leverages the area of geometrical shapes formed by each artist's metric values in a non-linear manner, thus we name it Geometric Artist Popularity (GAP_0 and $GAP_1$are two variations of the same concept). We also evaluate and compare two other methods for metric aggregation:

- TOPSIS [Yoon 1987]
- Preference Ranking Organization method for enrichment evaluation (PRO) [Brans and Vincke 1985]

Funded by the European Commission

As evaluation indices we use the following:

- Spearman's correlation (r_S)
- Pearson's correlation (r_P)
- Mutual information (*MI*)
- Overall rank overlap (*ORO*) [Schedl et al. 2010]
- Spearman's footrule distance (*F*) [Dwork et al. 2001]
- Kendall's tau (r_K)
- Kendall's tau distance (*K*) [Dwork et al. 2001]

F and *K* are distance measures hence the smaller value the better, yet all other indices are similarity measures hence the higher value the better.

4.2.2 Definition of Artist Popularity Score

Here we propose a composite artist popularity score that leverages multi-source web-based information in order to assess the level of an artist's current popularity. More specifically in Table 25 we have already presented the sources and metrics that we use as input to our artist popularity model. For each artist we monitor some or all of these 12 metrics since May 2018 and thus we can compute the corresponding artist popularity timelines.

In order to determine the popularity of artist a at time *t*, first we normalize the respective metric values $v_{a,t,i}$ for $i = 1, \dots, n$ (where *n* is the number of monitored metrics for the under study artist) to [0, 1] using a power transformation as in Equation 12:

$$m_{a,i,t} = \frac{{v_{a,i,t}}^p}{T} \tag{12}$$

where T is the chosen maximum power transformed value, cf. below, $v_{a,t,i}$ is the initial metric value, $p = \frac{log(T)}{log(V_{t,i})}$ with $V_{t,i}$ the maximum value of $v_{a,t,i}$ over all artists a, and $m_{a,i,t}$ is the normalized metric value. We did not opt for a simple "divide by maximum" normalization because there are metrics with huge variation such as YouTube views that in some cases reach billions and thus artists with millions of views would seem unimportant. Also, we did not opt for a log transform because there are metrics with small ranges such as Spotify popularity where after the transformation all normalized values would be close and high. The power transform alleviates both issues with a relatively high $T = 100$, which could be optimized if one considers appropriate reference data.

After the normalization we consider the unit circle and *n* equidistant points k_i on it. On each radius from k_i to the center we select the point l_i with distance $m_{a,i,t}$ from k_i. Geometric Artist Popularity (GAP_0) is then defined as:

$$\frac{E_{out} - E_{in}}{E_{out}} \cdot 100 \tag{13}$$

where E_{out} is the area of the outer regular n-sided polygon determined by k_i and E_{in} is the area of the inner polygon determined by l_i. If an artist performs best on all metrics the inner polygon would coincide with the circle's center and the geometric artist popularity would be 100, while if an artist performs worst on all metrics the inner polygon

Funded by the European Commission

would coincide with the outer regular polygon and the geometric artist popularity would be 0. All other cases result in intermediate values. Of course, different orders of the metrics result in different popularity scores, thus for consistency we first sort the metric values and then apply the computations on the sorted sequence of metrics. In Figure 31 a random example for the computation of Geometric Artist Popularity is exemplified.

A second approach on Geometric Artist Popularity (GAP_1) is to represent the metrics by the sides of the polygon and not by the vertices. Thus, the inner polygon in this case is the aggregate of *n* isosceles triangles with side length equal to $m_{a,i,t}$ as depicted in Figure 32. The popularity is then calculated as in the first approach (Equation 13).

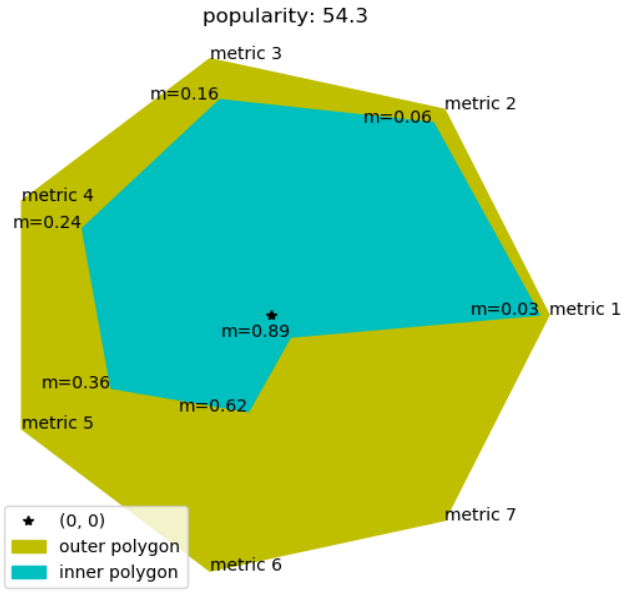


Figure 31 Example for the computation of Geometric Artist Popularity (GAP0).

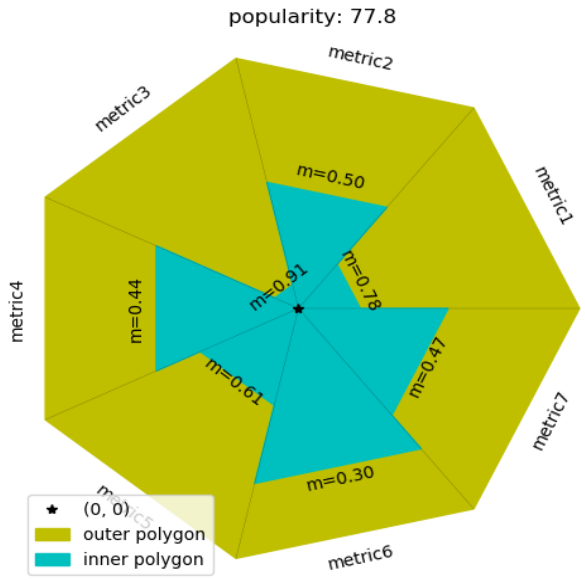


Figure 32 Example for the computation of Geometric Artist Popularity (GAP1).

Funded by the European Commission

1.1.1 Geometric Artist Popularity

The calculation of $GAP_0(m)$ and $GAP_1(m)$, while not straightforward, is actually simple given the vector of normalized metric values $m = \{i = 1, \dots, n\}$, for artist *a* at time *t*:

$$GAP_0(m) = \frac{1}{n}\sum_{i=1}^{n}\left(m_{a,i,t} + m_{a,i+1,t}\left(1 - m_{a,i,t}\right)\right) \tag{14}$$

$$GAP_1(m) = \frac{1}{n}\sum_{i=1}^{n}\left(2 \cdot m_{a,i,t} - {m_{a,i,t}}^2\right) \tag{15}$$

where $m_{a,n+1,t} = m_{a,1,t}$. The proof of these two equations (proof 1) can be found in the Appendix B. Also, considering the most natural choice for composite popularity i.e. the average normalized metric values (Average Artist Popularity):

$$AAP(m) = \frac{1}{n}\sum_{i=1}^{n} m_{a,i,t} \tag{16}$$

it is remarkable that:

$$AAP(m) \leq GAP_1(m) \leq GAP_0(m)$$

for all sorted m, with $m_{a,i,t} \in [0,1]$. The proof of this can be also found in the Appendix B (proof 2).

1.1.2 Evaluation

In the background section we cited many studies that consider already existing popularity metrics as ground truth in order to evaluate other popularity scores. We accordingly opt for Last.fm play counts and YouTube channel views (summed streams over the last 30 days) as ground truth for evaluation purposes. We chose these metrics because we believe that streaming activity reflects artist popularity more accurately than fan count (followers not always committed to the artist), social media mentions (not always related to music) or proprietary popularity scores (with not known algorithm e.g. Spotify popularity). The five composite artist popularity proxies AAP, GAP_0, GAP_1, TOPSIS and PRO are compared, and the results are presented here.

In Figure 33 we compare, with regard to a certain date, the values of composite scores with Last.fm play counts and YouTube channel views and in Table 30 we present the corresponding evaluation scores: Pearson correlation (r_P) and Mutual information (MI) for linear and non-linear interrelationship between composite scores and target. It is apparent that all composite scores are correlated with Last.fm play counts in a much higher degree than with YouTube views, thus we chose Last.fm play counts as ground truth for our experiments. In Table 31 the average performance of the composite scores across time (from 01-07-2018 until 31-05-2019) is exemplified in terms of linear/non-linear correlation and rank correlation/distance.

The results show that GAP_1 exhibits the best performance in 4/7 evaluation indices, while GAP_0, AAP and PRO in 1/7 each and TOPSIS in 0/7. Thus, we chose GAP_1 for

Funded by the European Commission

FuturePulse composite artist popularity index. In Figure 34 we present the composite artist popularity scores' timelines for 10 popular artists with the highest discrepancy among the monitored popularity metrics. It is observed that all of them retain high composite popularity values despite the low level of popularity in some metrics. Also, a more stable trajectory is exhibited by GAP_0, GAP_1 and AAP comparing with TOPSIS and PRO that are more volatile which maybe explains their worse performance.

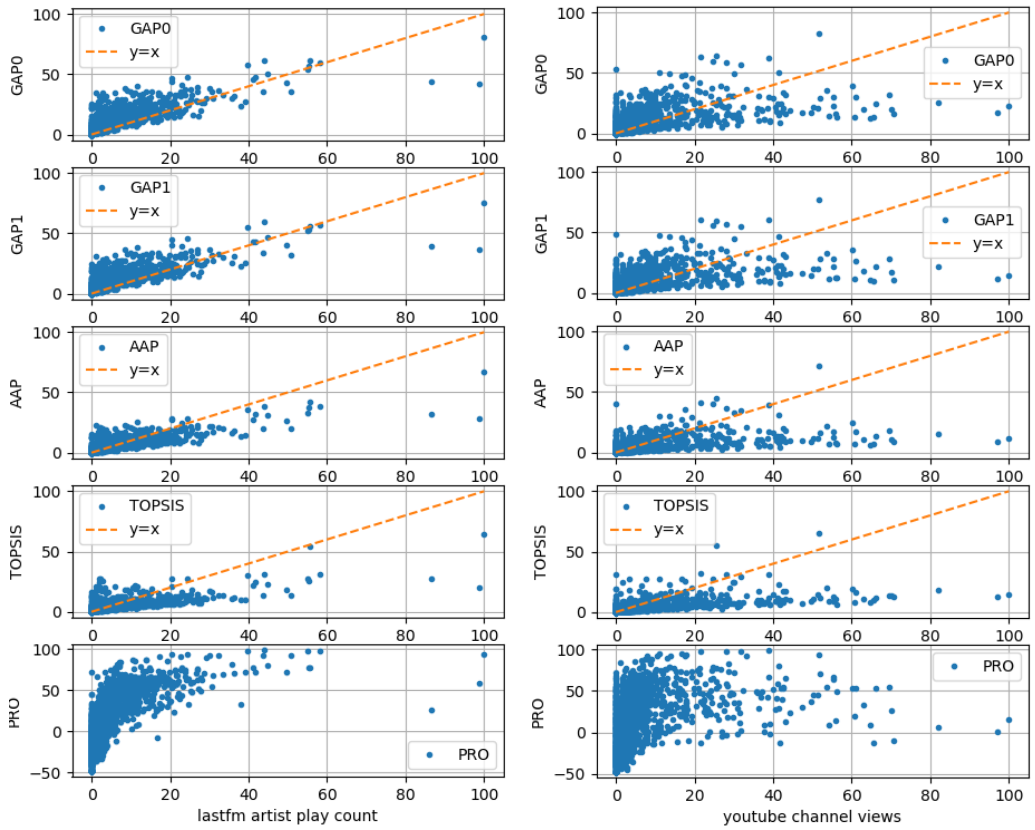


Figure 33 Scatter plots of AAP, GAP_0, GAP_1, TOPSIS and PRO vs. Last.fm play counts (left) and YouTube channel views (right). Each dot represents an artist.

	Last.fm		YouTube	
	r_P	MI	r_P	MI
GAP_0	0.8086	0.7150	0.5655	**0.5084**
GAP_1	0.8073	**0.7164**	0.5496	0.4990
AAP	**0.8287**	0.7098	0.5539	0.5046
TOPSIS	0.7531	0.6381	**0.5908**	0.4912
PRO	0.6325	0.6629	0.4262	0.4484

Table 30 Evaluation scores for AAP, GAP_0, GAP_1, TOPSIS and PRO.

	r_S	r_P	*MI*	*ORO*	*F*	r_K	*K*
GAP_0	0.8609	0.8086	0.7150	**0.8195**	0.1526	0.7791	0.1105
GAP_1	0.8624	0.8073	**0.7164**	0.8194	**0.1523**	**0.7799**	**0.1101**
AAP	0.8605	**0.8287**	0.7098	0.8195	0.1526	0.7790	0.1105
TOPSIS	0.8315	0.7531	0.6381	0.7938	0.1722	0.7513	0.1244
PRO	**0.8656**	0.6325	0.6629	0.8069	0.1583	0.7743	0.1128

Table 31 Average performance of composite scores across time. With bold we denote the best performance per evaluation index.

Funded by the European Commission

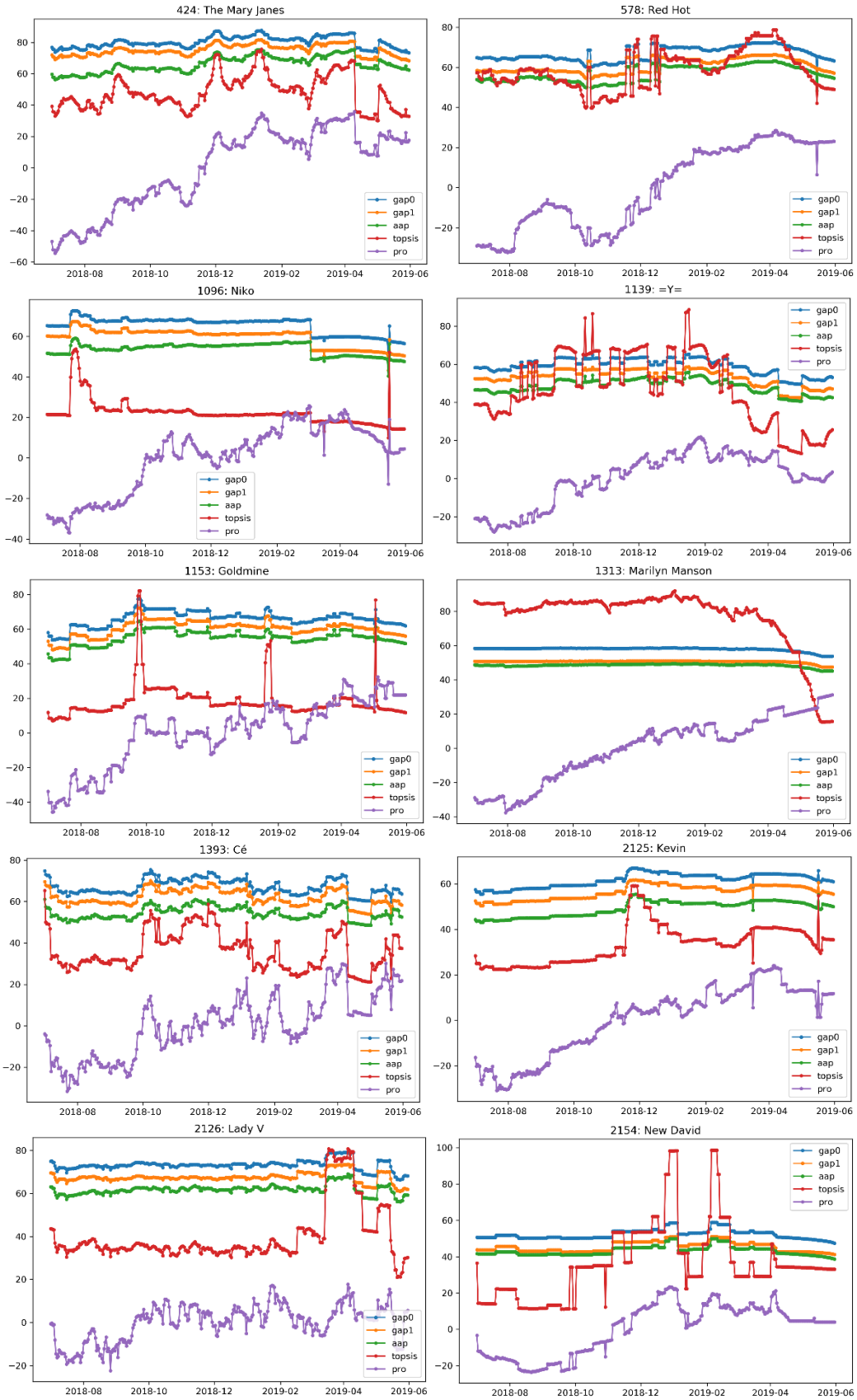


Figure 34 Timelines of composite artist popularity for 10 artists.

Funded by the European Commission

4.3 Artist Discovery

Live Music and Record Label use cases include requirements for the discovery of upcoming and emerging artists. For example, for the RL_REQ#7, we need to discover trending tracks/artists in platforms such as Spotify by tracking playlists and discovering artists (or tracks) that tend to appear more in them in a specific time period compared to the past. Additionally, the live music use case (LM_REQ#8 - Top upcoming artists per genre and LM_REQ#6 - Growth of artist popularity), which focuses on electronic music artists, is in need of being informed by additional sources such as Resident Advisor (RA) and Beatport.

In the first version of this deliverable (D3.1), we devised VenueRank [Krasanakis et al. 2018] on RA data to estimate artist popularity and popularity growth. To be more precise, VenueRank is an unsupervised graph ranking algorithm that assigns ranks on nodes based on their importance within an artists-venues bipartite graph structure. By applying VenueRank on successive time windows, we were able to estimate the growth of artists popularity as required by LM_REQ#6. However, RA on its own is far from being a complete source of artists. Beatport is an additional valuable data source, especially in the case of electronic music artists. Our main goal for the work presented in this section was to discover trending artists on the Beatport platform, to support the related Live Music requirements such as LM_REQ#8 (Top upcoming artists per genre). The results produced by this module can then be merged with results produced by VenueRank, or the tracking of platforms such as Spotify and Soundcloud. Having lists of emerging/upcoming artists for a specific time period (e.g. weekly or monthly) in several sources, we can then communicate them to FuturePulse end users, who can in turn decide to start tracking these artists in order to acquire more detailed data and more fine-grained popularity estimations and predictions as described in the previous sections.

To discover artists from Beatport we devised a trend analysis process on Beatport charts. Trend analysis calculates the change regarding an activity of an item over a defined period of time. A first approach is to calculate the percentage of change (PoC) on the activity of Artists, over two successive time periods. We defined as activity the number of times tracks of the specified artist appear on Beatport charts in a specific time period:

$$PoC = \frac{a_i - a_{i-1}}{a_{i-1}}, \tag{17}$$

with, a_i the number of times tracks of the specified artist appear on Beatport charts in a specific time window, and $a_{i-1}$the number of times tracks of the specified artist appear on Beatport charts in the previous time window.

Nevertheless, the percentage of change is not considered a good metric for revealing trending artists since it does not take into consideration their long-term historic activity. The percentage of change compares the current activity with the activity observed only in the previous time frame. To this end, we decided to define as trending score the standard score or **z-score**, which considers the historic average and the standard deviation of the artist's activity, and is calculated as in the following equation:

$$z = \frac{a_i - \mu}{\sigma} \tag{18}$$

Funded by the European Commission

with μ the mean (historic average of artist's activity), σ the standard deviation (of that activity) and a_i the raw activity of the artist in the time period for which we calculate z. Items that get positive values, obtain a score above the mean and hence are labelled as positively trending. On the other hand, items that get negative values, obtain a score below the mean and are labelled as negatively trending.

In total, for this study we made use of 402,364 charts from Beatport that refer to 1,189,711 different tracks of 183,182 different artists. The time period covered by the extracted charts is from 17/12/2004 until 26/08/2019. To calculate the change regarding artists' activity on Beatport charts, we used two-months time span as time window and as history the recorded activity of each artist during the last 12 months (6 different time periods). Eventually, we calculated both PoC and standard scores of artists on the resulted 4 time periods of 2019.

Table 32 and Table 33 present the top 10 trending artists on Beatport during periods 3 and 4 of 2019 (May-June, July-August) based on the percentage of change between the current and the previous period of time. As we have already noted, the percentage of change does not take into consideration the previous activity of artists on Beatport. On the other hand, in Table 34 and Table 35 the top 10 trending artists on Beatport during periods 3 and 4 of 2019 (May-June, July-August) based on the standard score is presented. It is noticeable that only four artists appear in the top 10 as trending in both metrics. The noticeable rise on the activity of these artists most of the times is linked with a track or album release that made a significant impression. Samim finds himself on the top of beatport charts due to a remix release of an old song of his as discussed in several articles[26]. IVA also finds herself on the top artists because of a release of a track that she is featuring[27]. Finally, Lokkhi Terra[28] scores high in standard scores because of the release of a new album in collaboration with other Afrobeat artists.

Figure 35 and Figure 36 present the timelines of the previous activity on Beatport charts of the top 10 artists of the 4th period of 2019 as emerged based on the standard score and PoC respectively. As it is exhibited in Figure 36, standard score tends to score higher artists that showcase sudden outbursts in their activity since it quantifies the distance of the current activity of the artist from its average activity. PoC does not take into consideration the long-term activity of the artist and tends to score higher artists that present significant change in their activity only with comparison to the previous time period.

[26] https://bit.ly/2ZvwWTT , https://bit.ly/2ZsQnBK
[27] https://bit.ly/2Uk157K
[28] https://bit.ly/30OlhBa

Funded by the European Commission

	Artist	Appears	PoC
1	Samim	92	9100
2	Niko Zografos	44	4300
3	Alex Dimou	42	4200
4	IVA	40	3900
5	The Vision	36	3500
6	BOHO	36	3500
7	Dyzen	35	3400
8	Los Suruba	34	3300
9	Dewitt Sound	31	3100
10	Rafa Barrios	32	3100

Table 32 Trending artists based on the percentage of change, over the 4th period of 2019 (July - August 2019)

	Artist	Appears	Trending Score
1	Samim	92	137.00
2	IVA	40	126.93
3	The Vision	36	114.20
4	Slarta John	49	73.00
5	Lokkhi Terra	23	72.83
6	Denise	25	59.60
7	Blindsmyth	25	52.33
8	Butterjack	16	50.56
9	White Perception	15	47.38
10	Novecento	14	44.19

Table 33 Trending artists based on standard score, over the 4th period of 2019 (July-August 2019)

	Artist	Appears	PoC
1	S.A.M.	57	5700
2	Eli Nissan	53	5300
3	DJ Assault	46	4600
4	Oliver Schories	45	4400
5	Ben Hemsley	43	4300
6	Cubicolor	41	4000
7	The YellowHeads	41	4000
8	Goodboys	39	3900
9	Marco Madia	39	3900
10	Meduza	39	3800

Table 34 Trending artists based on the percentage of change, over the 3rd period of 2019 (May-June 2019)

	Artist	Appears	Trending Score
1	S.A.M.	57	171.97
2	Baggi	31	93.36
3	Meduza	39	89.49
4	Marco Madia	39	89.49
5	TM.Park	29	87.31
6	M.I.G.	28	84.29
7	Jessie J	23	69.17
8	Sunar	22	66.14
9	Basstreque	22	66.14
10	Pete Lazonby	44	66.14

Table 35 Trending artists based on the standard score, over the 3rd period of 2019 (May-June 2019)

Funded by the European Commission

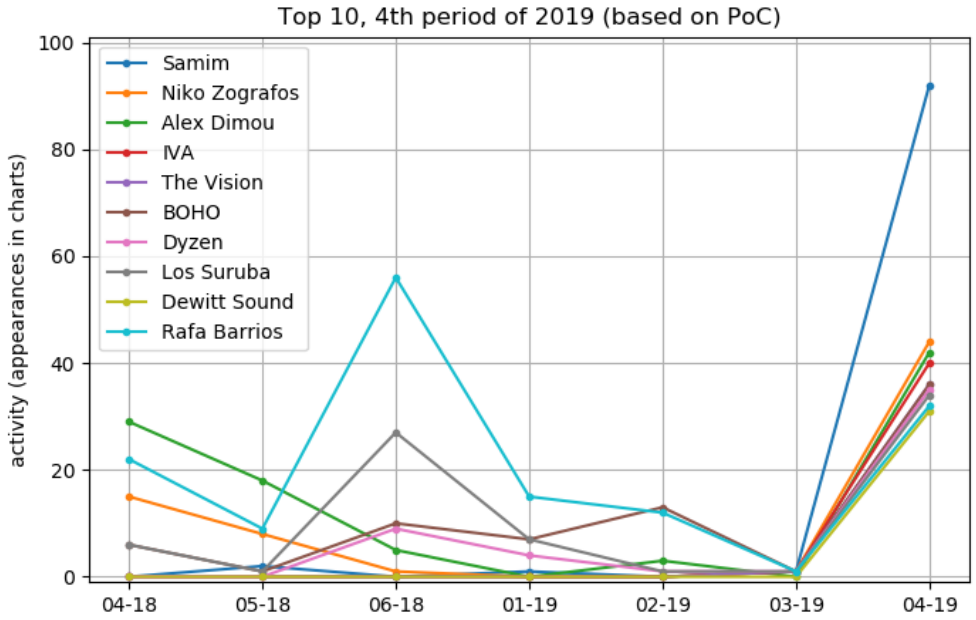


Figure 35 Activity timeline of trending artists on Beatport based on the PoC, over the 4th period of 2019 (May-June 2019).

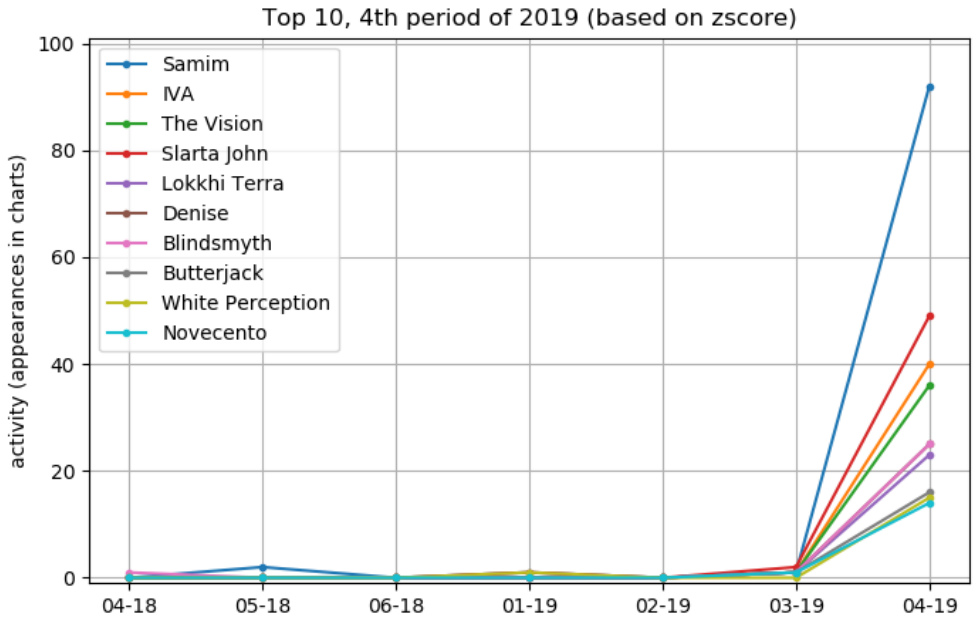


Figure 36 Activity timeline of trending artists on Beatport based on the standard score, over the 4th period of 2019 (May-June 2019).

Funded by the European Commission

4.4 Event Impact on Artist Success

In the context of the Record Label Use Case, it is important to estimate the impact of events on several metrics of artist popularity. For example, it is important to understand how an interview, the start of an ad campaign, an addition of a song to a playlist, or other events, influence streaming quantities and social media metrics (e.g. likes and mentions on Facebook, followers and streams on Spotify, subscribers and views on YouTube). This task is related to FuturePulse requirement RL_REQ#6 – "Release day / Event impact on success" and here we present our first approach towards addressing the problem. To this end, we review the existing academic literature on the topic of event impact estimation, experiment with different methods for estimation of event impact and present the corresponding results for six PGM artists where event data was available at the time of writing the deliverable. This is admittedly a small fraction of the overall dataset, but sufficient enough to gain useful insights on what is working and what should be further explored. In the next iteration we plan more comprehensive evaluation of event impact analysis methods in order to refine our approach.

4.4.1 Background

The estimation of an event's impact on the evolution of a time series (e.g. metric of success, KPI) is an interesting topic of research as it might give beneficial insights regarding the types of events that could lead to positive effects on important indicators. Researchers have employed several methodologies to address this problem, including the following:

- statistical tests for identification of significant changes before and after the event [Casella and Berger, 2002; Mbugua et al., 1995; Moses et al., 1992]
- segmented linear regression for identification of trend changes before and after the event [Lagarde, 2011]
- intervention analysis using ARIMA models before and after the event [Box and Tiao, 1975; Koski et al., 2007; Murry et al., 1993]
- change-point detection in order to determine timepoints that a phase transition took place [Jaruskova, 1997; Adams and MacKay, 2007; Guralnik and Srivastava, 1999]

Here, we perform event impact estimation analysis making use of a methodology based on segmented linear regression and another one based on statistical tests. Additionally, we use a change-point detection algorithm in order to determine timepoints of phase transition in streaming activity and relate them to certain events.

4.4.2 Data

As a starting point, PGM has provided us access to detailed lists of events per artist in which the time and type of each event is mentioned. The following types of events for impact estimation are considered therein:

- Interview
- Record Store Day
- Phoner
- Album review
- Concert announcement

Funded by the European Commission

- Concert review
- Video premiere
- Track of the day
- Featured in newsletter
- Q&A
- Concert recommendation
- Single announcement
- Album announcement
- Played on radio
- News item
- Concert report
- Documentary announcement
- Documentary feature
- Grammis nomination announcement
- Guldbaggen nomination announcement
- TV performance announcement
- Added to playlist
- Video recommendation
- Live performance
- Tour announcement

The impact of the events on the collected time series data per artists is then estimated. The time series data used in this study comprise timelines of artist popularity metrics related to play-counts (YouTube views and Last.fm artist play counts). Also, streaming activity timelines are considered from Spotify, Deezer and iTunes streaming platforms. The metric timelines are monitored from 05-2018 until today and the steaming activity timelines start from varying dates, across artists, between 2016 and 2018 and end today. In Table 36 we present the start and end date of our data per artist. The artists that we have event information for and are used in this analysis are Larz-Kristerz, Slowgold, Smith & Thell, The Holy, The Rasmus and Uno Svenningsson.

Artist	YouTube views	Last.fm play counts	Spotify streams	Deezer streams	iTunes streams
Larz-Kristerz	no data	no data	28/04/2018 22/06/2019	26/5/2018 22/6/2019	28/4/2018 22/6/2019
Slowgold	24/05/2018 15/07/2019	20/06/2018 15/07/2019	12/11/2016 22/6/2019	12/5/2018 22/6/2019	12/11/2016 22/6/2019
Smith & Thell	25/05/2018 15/07/2019	20/06/2018 15/07/2019	30/4/2016 22/6/2019	12/5/2018 22/6/2019	2/1/2016 22/6/2019
The Holy	06/06/2019 15/07/2019	21/06/2018 05/07/2019	14/5/2016 22/6/2019	12/5/2018 22/6/2019	25/6/2016 22/6/2019
The Rasmus	25/05/2018 04/07/2019	21/06/2018 04/07/2019	30/4/2016 22/6/2019	12/5/2018 22/6/2019	18/7/2015 22/6/2019
Uno Svenningsson	31/05/2019 04/07/2019	21/06/2018 15/07/2019	30/4/2016 22/6/2019	12/5/2018 22/6/2019	25/6/2016 22/6/2019

Table 36 Start and end dates for the available streaming data per artist.

Funded by the European Commission

4.4.3 Methods for Impact Estimation

For the task of estimating an event's impact on artist success we use two methods, one based on Segmented Linear Regression (SLR) [Lagarde, 2011] and one based on Wilcoxon signed-rank test (WIL) [Wilcoxon, 1945] for identification of significant changes before and after an event. The first method estimates the timeline's trend difference while the second method estimates the timeline's distribution change before and after the event.

For SLR we consider a timeline representing the success of an artist and a certain event date. Then we apply linear regression in the time period 5 weeks prior to the event date and separate linear regression modelling in the time period 5 weeks after the event date. This approach is based on the idea that an important event might lead to a period of increasing streaming activity (or steeper increase), which is compared with the previous period's trend, which may be decreasing, increasing or flat. Then we calculate the trend of the artist's success before and after the event as shown below:

$$trend_{pre} = atan\,(s_{pre})$$
$$trend_{post} = atan\,(s_{post}) \quad (19)$$

where s_{pre} and s_{post} are the slopes of the linear regression models before and after the event respectively. The trend is the angle of the fitted lines versus the x axis, where positive values indicate increasing success level, negative values indicate decreasing success level and zero value indicates flat success level. In order to quantify the change of trend as a measure of event impact on success we take the difference between the angles and normalize it from -100% to 100% with the following formula:

$$SLR = \frac{trend_{post} - trend_{pre}}{\pi} \cdot 100\% \quad (20)$$

where π is the maximal difference between trends.

The second method, WIL, is based on the idea that an important event would cause statistically significant increase on the timeline just after the event and thus the distribution of values would be different. So, we perform the Wilcoxon signed rank test for the pre-event/post event periods and use the p-value as an indicator of the event's importance. The smaller the p-value, the more important the event is and we consider only events that resulted in higher activity after the event on average.

Finally, for the change-point detection analysis we employ the algorithm implemented in the ruptures python package[29].

4.4.4 Results

In Figure 37 the YouTube views timeline along with its derivative and event dates are illustrated for Slowgold. In Figure 38 the Last.fm play counts timeline, its derivative and the corresponding event dates are exemplified for the same artist and in Figure 39 the Spotify, Deezer and iTunes streams are presented along with the event dates. In the appendix we present the corresponding figures (YouTube views, Last.fm playcounts,

[29] ruptures: change point detection in Python (https://arxiv.org/pdf/1801.00826.pdf)

Funded by the European Commission

Spotify, iTunes, Deezer streams and event dates) for the remaining 5 artists. In Appendix C we have included additional examples, depicting events and timelines for several artists related to PGM. From these Figures, it is apparent that the events do not have an observable impact on the metrics timelines (YouTube and Last.fm) but they do in some cases on the streaming activity timelines (Spotify, Deezer, iTunes). Hence, we compute the impact of events on Spotify streams using both methods (SLR, WIL) for all artists and we present the 10 most important events in Table 37. Additionally, in Figure 40 & Figure 41 the corresponding streaming activity before and after these events is illustrated (we illustrate the first 9 events for better visualization).

SLR		**WIL**	
artist/date	**type**	**artist/date**	**type**
Larz-Kristerz / 2018-12-14	documentary announcement	Smith & Thell / 2019-04-15	concert announcement
The Holy / 2018-09-12	single reviews, video premier, featured in newsletter	Slowgold / 2018-01-29	single recommendation, featured on playlist
The Holy / 2018-10-25	review	Slowgold / 2018-02-01	interview, news item
Slowgold / 2018-02-09	news item	Slowgold / 2018-02-07	concert announcement
Slowgold / 2018-02-07	concert announcement	Slowgold / 2018-02-08	concert announcement
Slowgold / 2018-02-08	concert announcement	Slowgold / 2018-02-09	news item
Slowgold / 2018-02-12	album review, featured on playlist	Slowgold / 2018-02-12	album review, featured on playlist
Smith & Thell / 2019-07-08	concert review	Slowgold / 2018-02-19	single and tour announcement
Slowgold / 2018-04-06	album recommendation, concert recommendation	Slowgold / 2018-02-21	interview, album review
The Rasmus / 2018-10-15	interview, concert announcement	Slowgold / 2018-02-23	album review

Table 37 The Top 10 events according to SLR and WIL.

As we observe in Table 37, the most important events are of certain types, the most frequent of which are “concert announcement” and “album review”. Also, the two different methods of event impact estimation indicate four common events (artist/date) as

Funded by the European Commission

important out of the Top-10 events. This fact shows that both methods are capable of indicating events that lead to streams' increase and generally capture similar patterns, but SLR has also the ability to indicate events that stop a decreasing streaming trajectory as one can see in Figure 40 (Slowgold 2018-04-06). In addition, WIL is more prone to indicate all events that happen at approximately the same time as important and responsible for an upcoming streams' increase than SLR. The latter in most cases selects the event that leads to maximal trend difference among the neighboring ones.

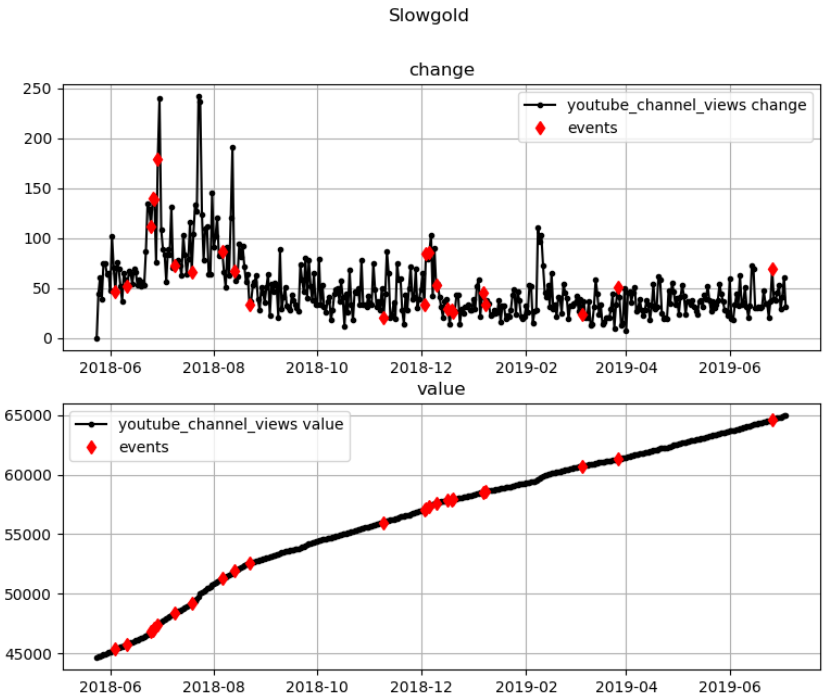


Figure 37 Timeline of YouTube channel views (value) and its derivative (change) along with event dates for Slowgold.

Funded by the European Commission

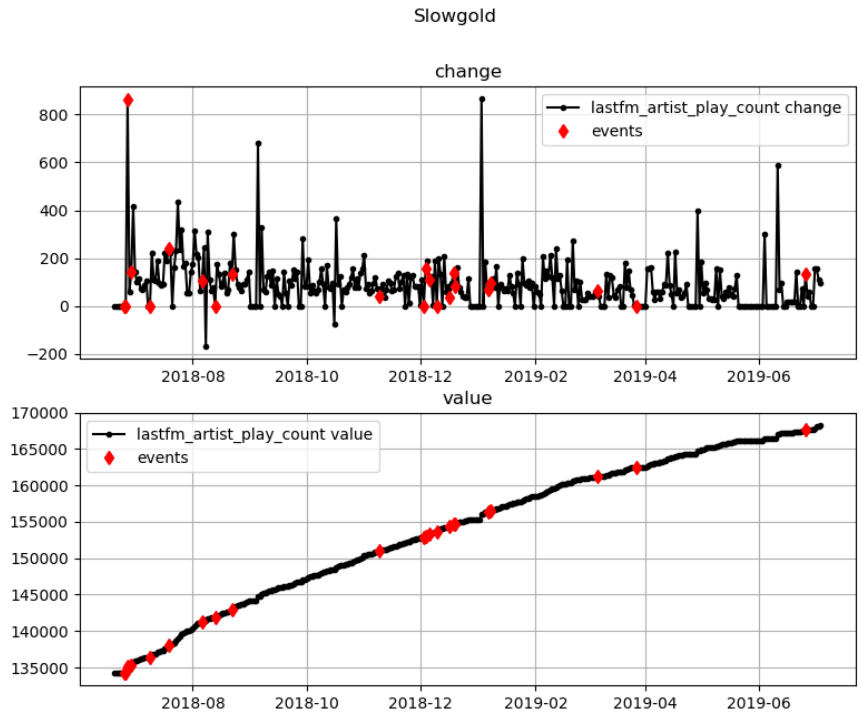


Figure 38 Timeline of Last.fm artist play counts (value) and its derivative (change) along with event dates for Slowgold.

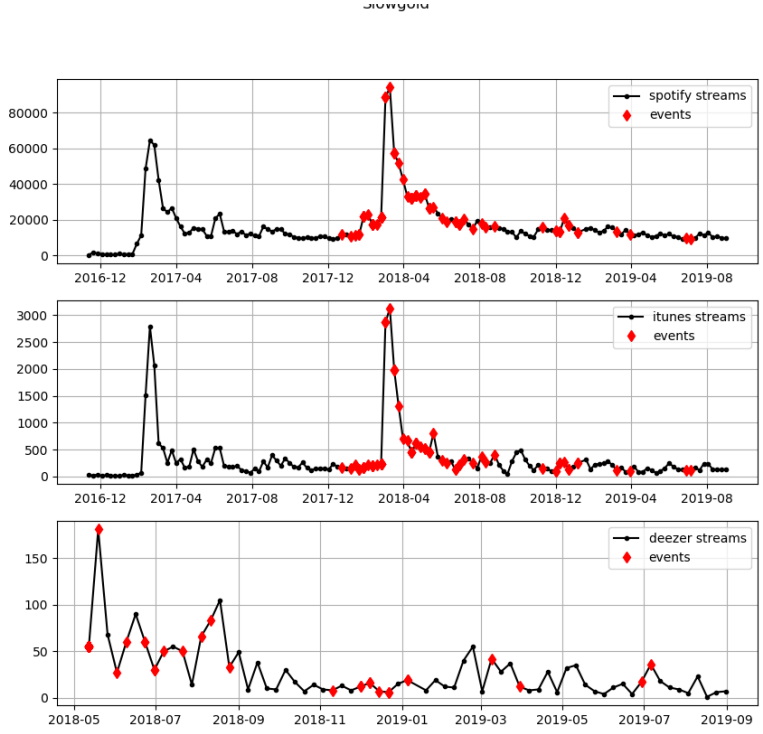


Figure 39 Spotify, iTunes and Deezer streams timelines and event dates for Slowgold. Spotify and iTunes streams are shown for the same period of time, while Deezer streams are shown for a shorter one.

Funded by the European Commission

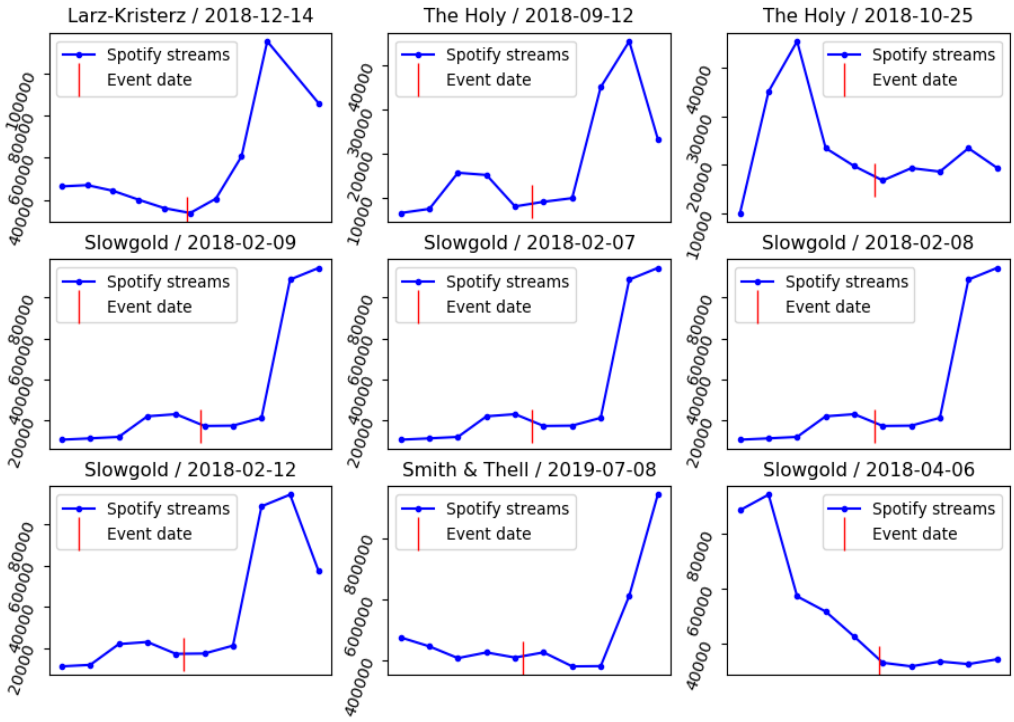


Figure 40 Spotify streams timeline segments before and after the 9 most important events according to SLR.

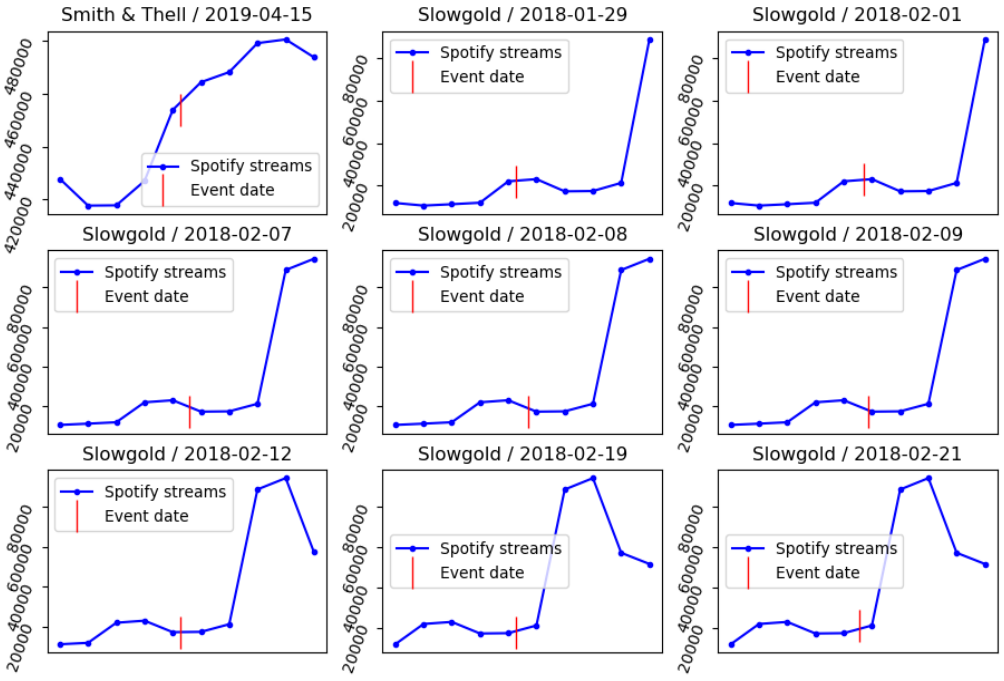


Figure 41 Spotify streams timeline segments before and after the 9 most important events according to WIL.

Funded by the European Commission

Artist/Date	Related event(s)
Larz-Kristerz / 2018-06-02	single announcement on 2018-05-21
Larz-Kristerz / 2018-12-29	documentary feature on 2018-12-19 (Aftonbladet: Biggest daily tabloid in Sweden)
Larz-Kristerz / 2019-06-29	concert announcement on 2019-06-28
Slowgold / 2017-12-30	three album announcements on 2017-12-20
Smith & Thell / 2018-11-24	Interview on 2018-11-19
The Holy / 2018-09-29	multiple events on 2018-09-28 including interviews, video premieres and track of the day
The Rasmus / 2018-09-15	three interviews on 2018-09-14
The Rasmus / 2018-10-20	multiple concert announcements in United Kingdom on 2018-10-15
Uno / 2018-06-02	after this change point the streams decrease, thus we do not correlate it with any event

Table 38 Change-points on Spotify streams timelines. Other change points are also indicated by the algorithm but on dates before the oldest or after the newest event in PGM's lists.

.

From the change-point detection analysis on Spotify streaming timelines per artist we obtain pairs of artist/date related to certain events exemplified in Table 38. Also, in Figure 42 one can see all the change-points (red dots) in the timelines, per artist. For the pairs of artist and date, if there are close events previous to the change-points that may be responsible for the transition to increased streams, we mention the event type(s) in the second column of Table 38. As we see in Figure 42 some change-points are related to streams increase and some others are related to streams decrease. There are also a few change-points for which the detected transition is not observable (for instance Slowgold / 2019-03-02).

Funded by the European Commission

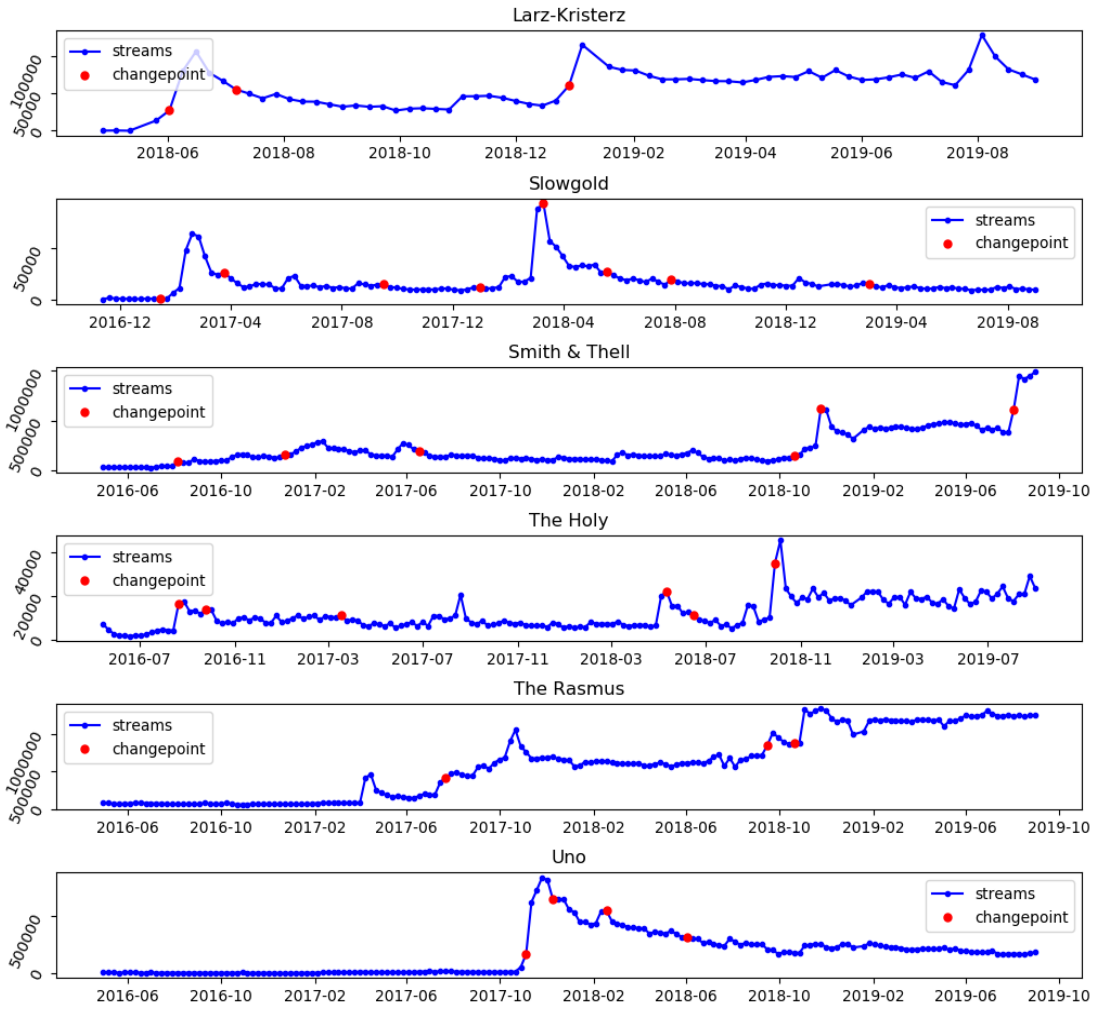


Figure 42 Change-points on Spotify streams timelines per artist.

Finally, in Figure 43 we illustrate one remarkable example in which 44 events of type "concert announcement" conducted in Sweden from 2019-03-15 until 2019-07-08 do not significantly affect the global streaming activity of Smith & Thell but influence local streaming activity (in Sweden) to a large degree. This example indicates that demographic groups (including different markets) may exhibit totally different streaming patterns and that searching for important events should not be limited to global activity. Additionally, we observe an abrupt increase in global streams right after the period of concert announcements. Unfortunately, we do not have event information for that period in order to relate an event or a series of events to that behavior of global streams timeline but the existence of an important event (e.g. an addition to a popular "global" Spotify playlist) prior to the global streams increase seems like a plausible hypothesis.

Funded by the European Commission

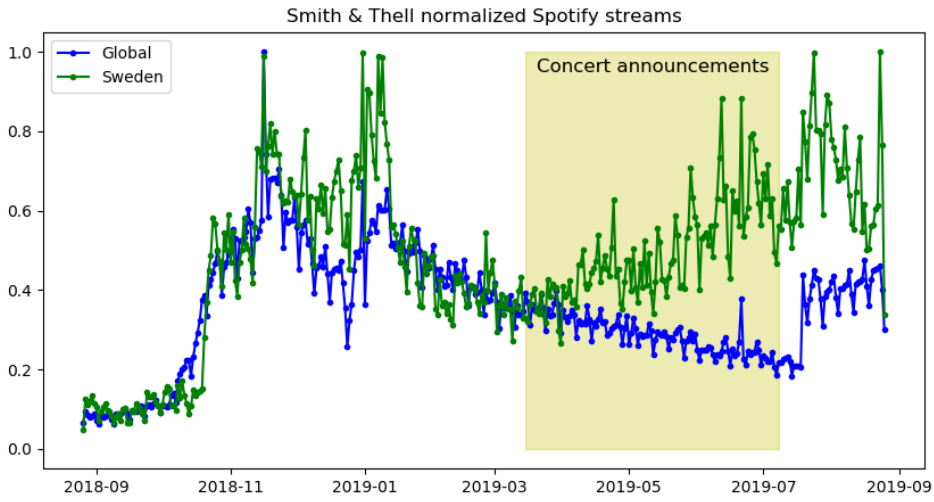


Figure 43 Normalized Spotify streams timelines (total streams and streams in Sweden) for Smith & Thell.

4.5 Implementation and integration of results related to artist popularity

Based on the conclusions drawn from the experiments described in this section, we opted to update or integrate three features in the platform:

- the three artist popularity indices (GAP_0, GAP_1, AAP)
- the predictions of individual metrics
- the event impact estimation on each individual metric of an artist

Regarding artist popularity, we update daily the three popularity indices as we also collect metrics on a daily basis. Therefore, by requesting the popularity of a specific artist, we get a list of values per day. These values can be refined by setting specific since/until dates. However, we should note that we produce these composite scores only for a small portion of artists which are under tracking (more details on the collection are given in deliverable D2.3). In order to get the artists in our database ordered by popularity, we can use the calculated popularity scores as a sorting parameter in the corresponding artists endpoints. As our analysis indicates that GAP1 reflects slightly better the overall popularity of an artist we use this popularity index for sorting purposes. The usage and structure of the API endpoint for the provision of artist popularity or artist sorting by popularity have remained unchanged. More details for the endpoint are provided in the previous version of deliverable (D3.1).

To provide predictions of individual metrics we have updated the corresponding API endpoint that provides the raw metrics of an artist with an additional parameter that indicates that the response should contain also an additional field with the predictions for that metrics. To get metrics per platform:

GET */sources/<:source_type>/<:source_id>/<:metric_name>*

For example:

GET */sources/spotify_artist/<spotify_artist_id>/spotify_artist_followers*

Funded by the European Commission

To include predictions of that metric the parameter *predict=true* has to be appended in the request. Optionally we can also set the number of input values in days to be used in the model (with *lag=28* being default value), and the number of days ahead we want to predict (with *step=14* being the default value).
To note that as calculating predictions is usually a fast operation, we execute this feature on demand ("lazy computation"), each time a user requests for it.

The event impact is integrated in the *events* endpoint. Given a specific event associated with an artist, the impact values for all the metrics collected for that artist can be obtained by calling:

GET */artists/<:artist_id>/events/<:event_id>*

The response of that request, apart from event details includes also a field named `impact_on_metrics` with the impact on the 11 metrics collected by the FuturePulse platform.

The artist discovery module has been executed for Beatport data from January 2005 until March 2019. All the beatport artists discovered during that period have been imported to the FuturePulse platform, annotated with additional sources such as Spotify and Resident Advisor. We plan to integrate this module as an offline procedure into the platform that runs in parallel with VenueRank in Resident Advisor and automatically imports newly discovered artists in our backend database. These artists can then be further monitored on demand, and more detailed predictions can then be produced.

Funded by the European Commission

5 Genre Popularity Estimation and Prediction

This section presents the analysis we performed on music genres in order to facilitate genre popularity estimation. More specifically, during the second year we refined the models developed during the first year of the project, and we also developed several approaches to solve genre-related issues we encountered during the same period.

At this point we should note that, as described in WP2 deliverables (D2.1 and 2.3), we have defined a taxonomy of genres for the Record Label and Background Music Provider (RL_BMP) use cases, and a more fine-grained taxonomy of electronic music genres to be used in the Live Music (LM) use case. However, as described in the previous version of predictive analytics and recommendations deliverable, we use Spotify genres as a reference taxonomy for the popularity estimation tasks, as Spotify is the most convenient way to annotate the data, we use such as charts, artists, etc. To provide insights for the genres in the FuturePulse taxonomies we have manually created a mapping between the Spotify, RL_BMP and LM taxonomies.

However, that mapping is far from complete. Also, Spotify genres taxonomy is much longer and more complex as it contains many subgenres. These two issues make difficult to accurately estimate popularity, causality or correlation between markets. For example, due to sparsity in the data we collect, it is difficult to estimate genre popularity per country (requirements RL_REQ#5, LM_REQ#9 and BMP_REQ#15) in cases of less known genres and small countries. To overcome this limitation, we leveraged subgenre associations and country-specific genres by finding genre associations (section 5.1). In that way, we improve the results of the genre popularity estimation process (section 5.2), and any genre-specific analysis we plan to perform in the next period.

Another issue we encounter is the fact that it is difficult to acquire genre information for less known artists. For example, even in the case of Spotify, only a small portion of artists are annotated with genres. Given that knowing the genres of artists is crucial for many requirements, we developed a graph-based approach to annotate artists with relevant genre tags (section 5.3). For example, LM_REQ#5 requires estimating artist popularity in a given genre, and LM_REQ#8 requires identifying top upcoming artists per genre. In other words, within different genre-based communities, the same set of artists could have a totally different popularity ranking. Therefore, we need to know the degree of association of an artist with such a community and use this information to adapt its popularity (section 5.4).

5.1 Music Genre Association Mining

Music genres is a conventional but meaningful way to classify music, used traditionally by the music industry. More precisely, a music genre characterizes music based on cultural traits, style, sound, etc. However, given the growing diversity of music offered in streaming platforms, the boundaries separating genres have become even more vague. In other words, as music can be divided into different genres in many different ways, genres typically overlap and different associations between them emerge. For example, there are genres that are sub-genres of another genre, therefore although there is no direct connection between them, they share the same ancestor. In some cases, this linkage can be long, giving complex associations between genres. For example, genres that have emerged by mixing multiple other parent genres.

Funded by the European Commission

Identifying these associations between genres could be crucial for a variety of tasks, such as contextualizing artist popularity within a specific genre or estimating genre popularity in a specific market by leveraging sub-genres in that specific country. However, that could be a challenging task, as shown in Figure 44, which depicts a co-occurrence subgraph between two popular and generic genres, *pop* and *rock*. This graph produced by using co-occurrences of genres in Spotify artists. From this figure, we can see that there is a strong link between pop and rock, as well as between genres *modern rock* and *indie rock*. There are also connections to other subgenres. However, it is difficult to draw conclusions about how similar two genres are by inspecting this visual representation.

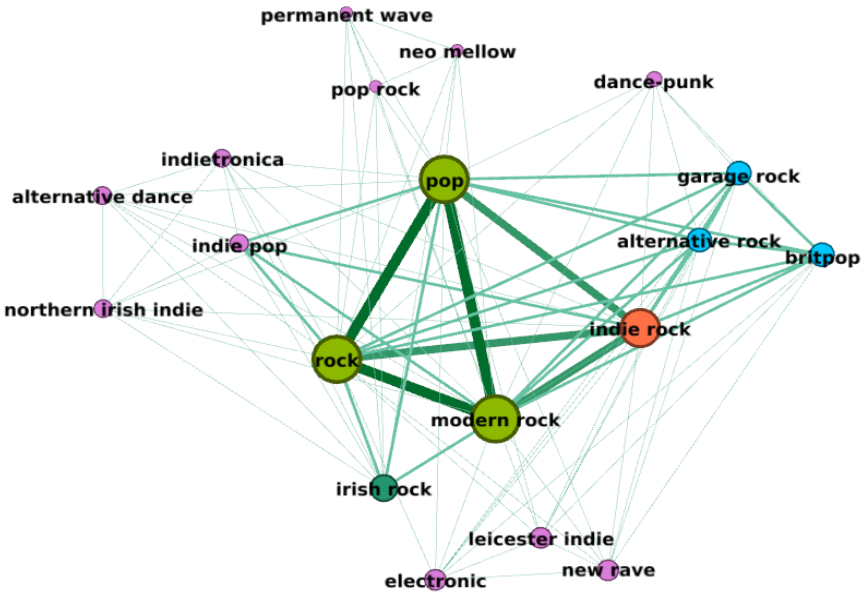


Figure 44 Genre co-occurrences subgraph for “pop” and “rock”.

A data-driven way to identify associations between genres is to exploit genre co-occurrences. In our case we used Spotify artists and their genre annotations respectively to perform genre association mining. One possible approach is to use association rule learning. Association rule learning [Agrawal et al. 1993] is a machine learning method for discovering interesting relations by detecting frequent patterns in *transactions*. Initially that type of approaches was used to solve the “market basket analysis” problem, i.e. to identify regularities in customer behaviour that could be then used as a basis for business decisions. In our case, the genre annotations of a specific artist (i.e. set of genre labels associated with the artist) are considered as a transaction. For example, Merilyn Manson is annotated with the following tuple:

{rock, alternative metal, post-grunge, industrial, industrial metal, industrial rock, nu metal, rap rock, wrestling}

To discover genre association rules, we need to calculate *Support* and *Confidence* for all co-occurring pairs of genres. Support refers to the default popularity of an item e.g. genres, and can be calculated by finding the number of transactions containing a particular item divided by the total number of transactions:

Funded by the European Commission

$$Support(i) = \frac{Transactions\ containing\ i}{Total\ Transactions} \tag{21}$$

Confidence refers to the likelihood that an item j also appears if item i appears.

$$Confidence(i \rightarrow j) = \frac{Transactions\ containing\ both\ i\ and\ j}{Transactions\ containing\ i} \tag{22}$$

By defining appropriate support and confidence thresholds, we can discover rules corresponding to relationships such as the sub-genres relation. However, two main factors have a negative effect on the performance of this approach. First, to generate meaningful association rules, support and confidence levels must be defined manually. Finding the most appropriate levels is a challenging task and may lead to the inclusion of insignificant rules or the rejection of important ones. Another reason that makes this approach less effective is that it considers only the direct associations between genres. However, although the number of co-occurrences between two genres might be low, leading to no association rule between them, their second order proximity might be high. In other words, two genres may have limited direct associations but share many common neighbours e.g. as in case of sub-genres of the same genre.

To this end we investigated two novel approaches to identify associations between genres, both based on a genres co-occurrences graph. The first approach attempts to cluster genres into groups of similar ones using an unsupervised graph clustering procedure, while the second maps genres into a latent vector space by leveraging a graph embeddings technique.

The first approach works by constructing a graph of genre tags, in which tags are linked to those co-occurring with them (at artist level) more than 30% of the time and partitioning this graph into non-overlapping groups of closely linked genres. To accommodate groups arising from higher-level proximities, we try to discover those that have a high number of links between their members and also link to few other genre tags. A quantification of these qualities is the measure of conductance, which is defined for a group G of graph nodes that comprise less than half of the graph edges as:

$$\varphi(G) = \frac{|G\ links\ to\ other\ nodes|}{|links\ between\ nodes\ of\ G|} \tag{23}$$

Discovering minimum conductance groups of graph nodes is an NP-hard problem, but there exist fast algorithms that find groups of small conductance around given nodes, such as the normalized sweep algorithm [Andersen et. al, 2006], [Andersen et. al, 2007]. With this in mind, we devise the following process for partitioning music genres into non-overlapping groups of similar sizes and small conductances:

1. Rank the importances of all genre tags in the genre co-occurrence graph C through the PageRank algorithm [Page et al. 1999]
2. Process a new genre cluster
    a. Select the genre g with the maximal importance that is not in any group

Funded by the European Commission

b. $C' \leftarrow copy\ C, G \leftarrow nodes\ of\ C'$
c. While $|G| \geq \sqrt{nodes\ in\ C'}$
    i. Perform normalized sweep to find a group of genres G that minimizes $\varphi(G)$ inC' and contains g
    ii. $C' \leftarrow subgraph\ of\ C'$ with nodes of G
d. Remove all nodes from the group that belong to a previous group
e. Set g as the group label

3. Repeat from 2 until no genres remain

The second approach we used to identify genre associations is based on node2vec [Grover & Leskovec, 2016], a node embeddings technique, that preserves both first and second order proximity. Doing so, we are able to represent each genre as a latent feature vector, with the cosine similarity between them preserving the local and global structure of the genres co-occurrences graph. More precisely, node embeddings are a mapping function $f: V \rightarrow R^d$ which maps a node $u \in V$ to a d-dimensional feature vector $f(u)$. In other words, f is a matrix $|V| \times d$, with each row corresponding to the feature representation of a specific node. Node2vec estimates f by optimizing the following objective:

$$max \sum_{u \in V} \log \Pr\big(N_S(u) \big| f(u)\big) \tag{24}$$

with $N_S(u) \subset V$ being the neighborhood of node u generated through a sampling strategy S. Node2vec uses a sampling strategy based on random walks, that combines Breadth-first Search (BFS) and Depth-first Search (DFS), in order to capture both the local and global structure of the graph. In case that only BFS is used to sample the neighbourhood of a node, the nodes that are highly interconnected or belong to similar network communities tend to be mapped close in the resulting feature space. On the contrary, DFS tends to favour structural similarity, i.e. the nodes that have a similar structural role in the graph tend to be embedded closer. Node2vec controls BFS and DFS by using two parameters p and q. A low p parameter keeps the random walk starting at node *u* local i.e. close to the starting node. On the other hand, if q > 1 the random walk is biased towards nodes close to the starting node, while a value of q less than 1, leads to visiting nodes further away (global structure).

5.1.1 Results

To generate clusters of similar Spotify genres and estimate their feature vector representations, we used a set of 733,043 Spotify artists annotated with 3,310 genres. To create that set, we started with the 128,000 artists that are currently imported in the FuturePulse platform. Out of those, a small portion (~2,600 artists) has been provided by PGM for the Record Label use case, while the rest were identified in sources such as Resident Advisor and Beatport as described in the artist discovery section and then were mapped to Spotify (if there is a related account). To further expand that initial set, we used the related artists endpoint provided by Spotify[30], that returns artists similar to a given one, based on co-listening patterns. More precisely, for each of the 128,000 artists in the FuturePulse platform, we obtained the 20 most similar Spotify artists based on

[30] https://developer.spotify.com/documentation/web-api/reference/artists/get-related-artists/

Funded by the European Commission

analysis of the Spotify community's listening history. We then acquired the Spotify genres of these artists.

By running the graph clustering technique described in the previous sub-section, we generated 2,421 clusters of similar genres. However, only 99 clusters consist of more than one genre, having 10 genres on average. The rest are trivial instances of only a single genre which failed to be grouped with similar ones.

Clusters of similar genres
latin rock, spanish indie pop, ska argentino, mexican rock, rock nacional, argentine rock, latin alternative, spanish modern rock, spanish noise pop, rock en espanol, mexican indie
hip hop, trap music, southern soul, east coast hip hop, viral pop, old school hip hop, post-disco, hyphy, gangster rap, deep pop r&b, electropop, pop rap, memphis hip hop, urban contemporary, funk, motown, new jack swing, boom bap, alternative hip hop, electro, crunk, soul, metropopolis, swedish electropop, southern hip hop, minneapolis sound, west coast rap, r&b, g funk, hardcore hip hop, chicago soul, dance pop, conscious hip hop, classic soul, pop, rap, philly soul, turntablism, dirty south rap, brit funk, cali rap, disco, atl hip hop, houston rap, post-teen pop, hip hop, deep southern trap, west coast trap, boy band, quiet storm, hi-nrg, memphis soul, neo soul, girl group
atmospheric black metal, blackgaze, atmospheric black metal, doom metal, avantgarde metal, post-black metal, funeral doom, celtic metal, retro metal, melodic black metal, epic doom, chaotic black metal, swedish death metal, pagan black metal, folk metal, cosmic black metal, stoner rock, instrumental stoner rock, neo-trad doom metal, avant-garde black metal, sludge metal, voidgaze, psychedelic doom, norwegian black metal, depressive black metal, viking metal, swedish metal, black metal, norwegian metal, post-metal, drone metal, usbm, space rock, stoner metal, post-doom metal, dark black metal, autonomous black metal, symphonic black metal, technical black metal, swedish black metal
folk rock, doo-wop, glam metal, modern blues, rock, cosmic american, folk, album rock, merseybeat, classic girl group, appalachian folk, adult standards, brill building pop, singer-songwriter, southern rock, blues, american folk revival, ectofolk, metal, funk rock, country rock, rockabilly, british invasion, freakbeat, british blues, experimental, soft rock, blues-rock, folk rock, alternative country, classic garage rock, traditional folk, permanent wave, electric blues, yacht rock, bubblegum pop, rock-and-roll, roots rock, lilith, zolo, outlaw country, psychedelic rock, hard rock, post-grunge, heartland rock, nwobhm, christmas, classic rock, country, mellow gold, protopunk, art rock, rhythm and blues
edm, trance, tribal house, uplifting trance, zapstep, hip house, tracestep, deep groove house, big room, brostep, moombahton, eurodance, complextro, europop, traprun, deep big room, bubblegum dance, deep house, progressive electro house, progressive uplifting trance, edm, deep tropical house, vocal house, pop edm, disco house, bass trap, diva house, progressive house, sky room, electronic trap, tropical house, house, chicago house, deep uplifting trance, progressive trance, filthstep, uk dance, electro house, acid house, catstep

Table 39 Examples of clusters obtained using graph partitioning.

To generate genre embeddings with node2vec, we used the same initial genres graph, without applying any edge pruning, leading ultimately to a genres graph with 3310 nodes and 49,900 edges between them. Although edges having a low weight might be false associations between genres, the sampling of edges in node2vec takes into account that weight; therefore, it is unlikely that such an association will be included many times in the learning instances.

Funded by the European Commission

Also, we modified the resulting graph in order to boost the weight of edges between genres that even though do not co-occur often, their names indicate that there is an apparent association. For example, “german hip hop” and “deep german hip hop” have very few edges but the latter is clearly a subgenre of the former. To boost such an edge, we set its weight to the average weight of the edges of the adjacent nodes/genres. Note that, although it is easy to infer associations between pairs of that type, i.e. pairs for which one genre is just the other plus a prefix (e.g “deep”), their inclusion in the graph can be beneficial for the node embeddings procedure, as boosting of their weights moves the corresponding genres closer in the vector space, and that affects the position of genres associated with those two genres.

We used parameters p=1 and q=1.5 to favour local search and create a representation of genres that reflects the communities of genres existing in the co-occurrences graph. However, we keep also q close to 1, to include random walks consisting of nodes/genres of which the participation in the same genre community is controversial. Regarding the rest of the parameters, we set feature vectors’ dimension to $d = 256$ and we generated 32 random walks per node with 100 steps each.

In Table 40 we present the top similar genres of a given genre, both by using directly the edges of the augmented / modified co-occurrence graph and by selecting the closest nodes in the embedding space. Although in the first case, the selected genres seem reasonable, we observe that with the use of embeddings the selected nodes tend to belong in the local community of sub-genres. For example, although “hip hop” is related to “rap”, “trap” and “r&b”, it is easy to understand that “hardcore hip hop”, “memphis hip hop” or “southern hip hop” are more relevant. In other words, the embeddings approach boosts sub-genres by suppressing more generic genres. Also in some cases, not obvious associations are identified as in the case of “indie rock”, where “noise pop”, and “la indie” are identified among the most relevant although the number of direct co-occurrences is low. Of course, embeddings also preserve the first-order proximity i.e. the direct edges therefore in many cases there is a significant overlap between the two lists (e.g. in the edm example).

Genre	Associated genres using co-occurrences graph	Similar genres using embeddings
hip hop	rap, pop rap, gangster rap, trap music, r&b, hip pop	hardcore hip hop, detroit hip hop, conscious hip hop, rap, memphis hip hop, southern hip hop
indie rock	modern rock, indie pop, alternative rock, lo-fi, indie folk	small room, british indie rock, noise pop, la indie, brooklyn indie
latin rock	rock en espanol, mexican rock, argentine rock, mexican indie, spanish indie pop	deep latin alternative, argentine indie, monterrey indie, argentine reggae, spanish modern rock
edm	electro house, big room, pop edm, progressive house, progressive electro house	dutch house, pop edm, electro house, deep tropical house, canadian electronic

Table 40 Examples of similar genres using node2vec embeddings.

Funded by the European Commission

5.2 Genre Popularity Estimation

Genre popularity estimation is a core requirement for the FuturePulse platform, as it is needed for all the three use cases (RL_REQ#5 - Genres trending for each market, LM_REQ#9 - Genre popularity, BMP_REQ#15 - Genre popularity for each market).

As described in the first version of the predictive analytics and recommendations deliverable (D3.1) for the computation of genre popularity on a global level we used chart data. Each chart entry (referring to tracks, albums or artists) is associated with multiple Spotify genres and we aggregated the total entries associated with each genre for a certain time period. More precisely, we pre-calculated total counts per genre[31] on a monthly, quarterly and yearly basis and we computed the genre popularity score for each time period as the percentage of each genre's counts compared to all genres:

$$p(g) = \frac{\mathrm{C_g}}{\sum_g \mathrm{C_g}} \tag{25}$$

where C_g is the number of chart entries for genre g. To calculate genre popularity at a market level, we refined the score by including only the charts of a specific market in the calculation of the counts.

However, an issue that arises for less popular genres at small markets is that the count number is usually low or even zero for many short periods of time. For example, there are cases that on a monthly basis the number of chart entries having a specific genre is zero. By using the formula above, the calculated popularity is also zero. To tackle this problem, we refined the previous approach by incorporating the subgenres counts for the calculation of the popularity of a specific genre. However, instead of adding the counts of subgenres directly in the counts of the genre, we used them as a smoothing term in the above equation. To get the subgenres of a given genre we use its feature representation as estimated by node2vec, and we get the top-n closest genres having a lower degree centrality. Intuitively, a first-level genre has a higher degree centrality from a second-level genre as it is connected to more nodes and as it is more generic can be also connected with other less related genres. As a first approach to calculate the score, we adapted the formula above as follows:

$$p(g) = (1-\lambda)\frac{C_g}{\sum_g C_g} + \lambda \frac{\sum_{g' \in G(g)} C_{g'}}{\sum_g \sum_{g' \in G(g)} C_{g'})} \tag{26}$$

where λ is the smoothing parameter and $G(g)$ is a list of subgenres for genre g, generated as described in section 5.1.

[31] During the first year, we did that for the genres in the taxonomies defined in the contact of FuturePulse (RL_BMP and LM) but we counted appearances based on Spotify genres, by using the manual mapping between taxonomies.

Funded by the European Commission

5.2.1 Results

We used the chart data already imported in the platform to have a preliminary evaluation of the modification presented in the previous method. We considered a 3 months period from 2018-01-01 to 2018-03-31 and we got the number of counts for the 3310 Spotify genres mentioned in section 5.1 in every country for which we have available charts. We estimated popularity by using either single genres or subgenres groups and then we inspected the results trying to identify cases that validate the results produced by the proposed smoothing approach.

In the case of Italy, there are four charts, having 188 non-zero genres with an average of 334 and a maximum of 10,708 counts per genre. In case of hip hop, which is evidently a popular music genre at a global scale, we get only 1232 counts, leading to a very low popularity score. If we consider other subgenres such as underground hip hop (107), alternative hip hop (35), southern hip hop (341) and latin hip hop (316) we can increase that estimation. Interestingly, the genre with the most counts is italian hip hop (10,708) making hip hop the most popular type of music in Italy. Taking into account that italian hip hop is among the top 20 most similar genres to the initial hip hop genre, we can assume that if we include a number of most similar genres, the second estimation reflects better its real-world popularity in Italy. Similar observations can be made for other genres and country-specific estimations.

However, among the top 20 associated genres is also rap with 1704 counts, which is a related music style but not a subgenre. Therefore, the estimated popularity is also affected by that negative associations. In other words, the overall accuracy of the popularity estimations is determined by the number of subgenres considered in the calculation of the score, and also by the λ parameter, used to merge the original and the updated score. In the preliminary experiments we used a value of 0.5 that balances the two components and gives reasonable results. However, we should notice that as we do not have ground truth for the prevalence of music genres in different territories it is difficult to fine tune the model.

Another important issue that arises using the approaches described above is that the normalization of the counts of a genre (with or without the inclusion of subgenres) is performed by dividing with the sum over all the other genres. In other words, we assume that all genres compete with each other over people's attention, as the increase of one's counts leads to the decrease of another's popularity (as the sum over all genres, i.e. the denominator in the above equations increases). Although that assumption might be valid when we have a list of genres at the same level in a taxonomy, when we consider more complex hierarchies that assumption is weakened. For example, *hiphop*, *southern hiphop* and *detroit hiphop* are not necessarily antagonistic genres but the growth of popularity for one impacts positively the popularity of the other. Also, as stated before, not only sub-genres but other relevant genres are considered in the calculation of genre popularity. Although that might be problematic in many cases, we should notice that some genres may have a long-term impact in other closely related but different genres. Therefore, as there is a need to model that interplay between relevant, antagonistic or not genres in a principled way, we plan to move towards that direction in the forthcoming period.

Funded by the European Commission

5.3 Estimating Artist Relevance to Genres

An issue that makes difficult the estimation of genre popularity (RL_REQ#5, LM_REQ#9 and BMP_REQ#15) or estimation of artist popularity within a specific genre community (LM_REQ#5 - Artist popularity in a given genre) is the fact that genre annotations are missing for many less known artists. The same holds true for less popular and marginal genres. Therefore, if there are artists of such genres in a chart but we miss this annotation, it is not feasible to assess the popularity of the genre by using the number of counts, as described in the previous section. Hence, it is crucial to have a method that can estimate the relevance of artists to particular genres.

To do this, we developed a graph-based recommendation system that ranks the relevance of artists to genres of interest. In this system, we start with the few known artist genre tags. For these tags we mark respective artists as fully (i.e. 100%) relevant to respective genres and then iteratively spread these relevances to artists they have co-performed with. If we create a matrix

$$M[i,j] = \begin{cases} 1 & \text{if artists } i,j \text{ have co} - \text{performed} \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

that corresponds to a graph with edges that represent any kind of association between artists (e.g. co-performances in the same venue or co-listening patterns in Spotify), we can express the propagation $\pi(i,g)$ of how much artists i are relevant to genres g through a symmetric random walk with restart scheme [Tong et al. 2006] that is more suited to modeling the relevances of symmetric relations compared to the previously employed PageRank. This scheme satisfies:

$$\pi(i,g) = (1-a)\pi_0(i,g) + a\sum_j \frac{M[i,j]}{\sqrt{deg\,(i)\,deg(j)}}\pi(j,g) \tag{28}$$

where $deg(i) = \sum_j M[i,j]$, $1-a$ is called the restart probability of the scheme and $\pi_0(i,g) = \{1$ if artist i has a prior genre tag g, 0 otherwise$\}$ indicates the priorly known genre tags, called seeds. Using the genre relevances calculated by this scheme we rank artists in each genre so that the artist who is most relevant is ranked first.

Although this random walk with restart scheme is well-established bibliographically, we found that the small number of artist tags constrained propagation mostly to artists residing a few hops away from those of known tags. To address this problem, we devised a seed oversampling methodology [Krasanakis et al. 2019] that thresholds the calculated relevances to obtain more known artists for each genre. Using the new set of known artists, our methodology then repeats the symmetric random walk with restart scheme.

As a final refinement to accommodate lesser known genre tags, we tune the restart probability to maximize the AUC (see below) evaluated on 50% of known genre tags when using the other 50% to calculate relevances.

5.3.1 Results

To evaluate the proposed method, we used two different datasets with two different types of associations between artists.

Funded by the European Commission

The first dataset utilizes events and artists, extracted from Resident Advisor and Spotify respectively. Each Resident Advisor event comprises its venue and artists who participated. The artists are matched to those extracted from Spotify, for whom we retain their name, annotated genres and Spotify's calculated popularity (which takes values 0-100). This data model is presented in Figure 45.

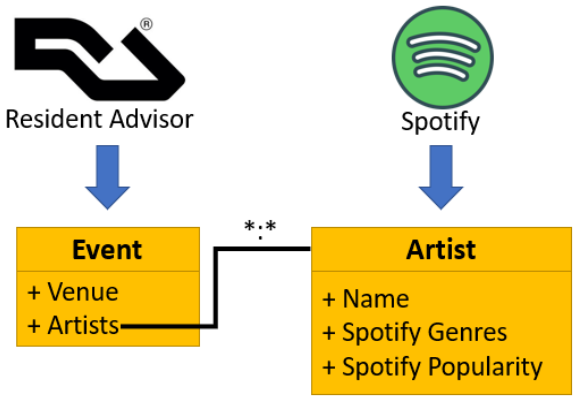


Figure 45 Extracted Data from Resident Advisor and Spotify.

Extracted data include 29,456 events that took place in 16,174 venues across six European countries between September 2002 and March 2019 and involve a total of 23,181 artists.

As we already have a mapping between Spotify's genres and the genres defined within the context of FuturePulse by Soundtrack Your Brand and Playground (see Figure 46), each artist's Spotify genres are mapped to the latter genres taxonomy. However, genres were missing for over two thirds of Spotify artists and the available mapping was not available for all Spotify genres. As a result, genre tags ended up missing from most artists (20,934). The devised system aims to complement and augment these tags with genre-related ranks.

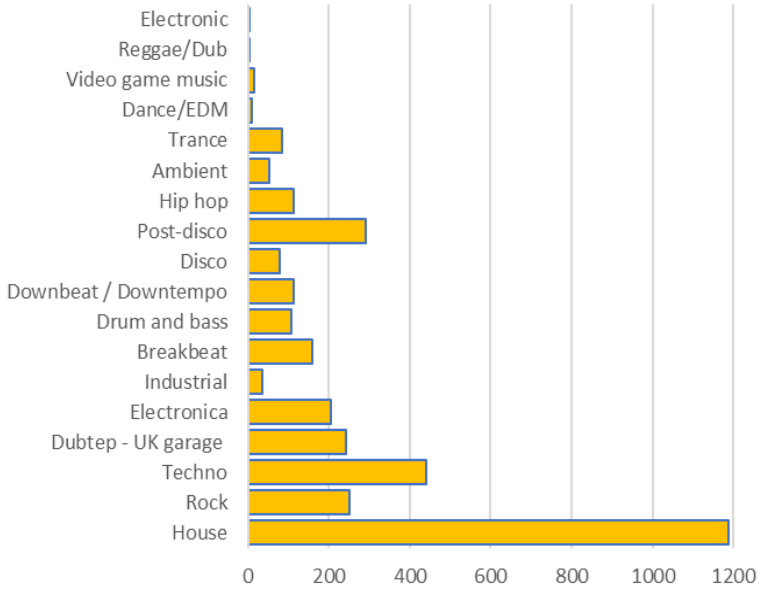


Figure 46 Soundtrack Your Brand and Playground genre distribution (number of artists tagged with each genre – each artist may belong to multiple genres).

Funded by the European Commission

We also experiment on a much larger dataset that comprises the 733,043 Spotify artists used in genre association mining, out of which only 213,029 had genre tags. As described in the related subsection, these artists were discovered through the "related artists" feature of Spotify and we used this type of association to create links between them. The resulting graph comprises 5,883,675 links and includes 3,310 types of Spotify genre tags, many of which could not be mapped through the previously described genre map. As a result, we employ the genre clustering described in the previous subsection to map genre tags to larger clusters (see Figure 47).

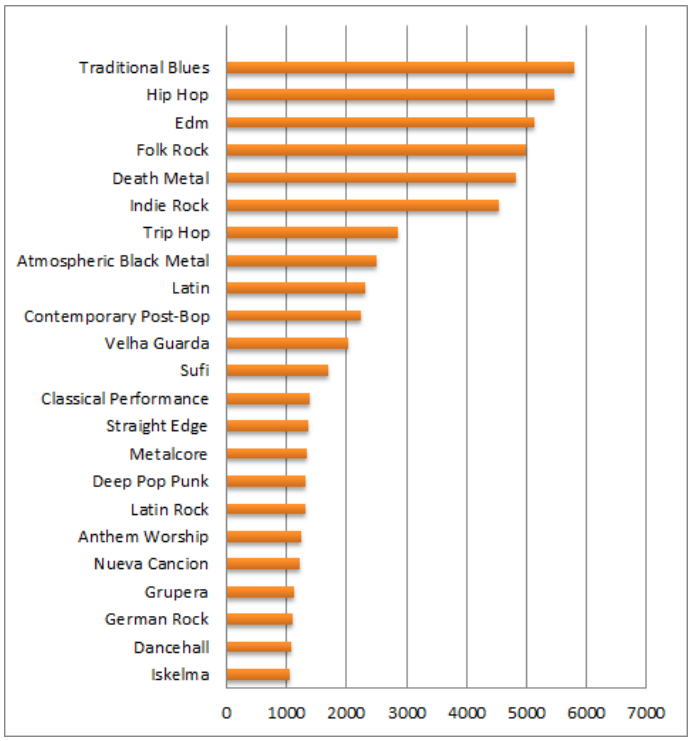


Figure 47 Clustered genre distribution for the genre with more than 1,000 artists (number of artists tagged with each genre – each artist may belong to multiple genres).

To assess the efficacy of the proposed system in ranking the relevance of artists to genres, we set up experiments in which we evaluate its ability to place higher relevance on genre artists for the genres presented in Figures 38 and 39. In particular, for each considered genre, we randomly withhold 80% of its known genre tags and use the other 20% to calculate the relevance of all graph artists to it. Then, we evaluate the quality of artist relevance values by considering the withheld artists as the positive targets of prediction and artists with no such genre tag as the negative targets of prediction. Evaluation is performed with the Area Under Curve (AUC) measure of the Receiver Operating Characteristics (ROC) curve, which is a non-parametric measure not influenced by the disproportionately few positive targets. AUC measures whether more importance is placed on positive predictions and assumes values near 100% for high rank quality and values near 50% for random ranks.

For each considered genre, we average its AUC score over 5 repetitions of the experiment. In Figures 40 and 41 we present the AUC of artist ranks for the genres of the smaller and larger datasets respectively. We can see that, for the smaller dataset,

Funded by the European Commission

artists do not match very well to genres, achieving on average 65% AUC across genres. This can be attributed to the small size of the dataset affecting the information that would be encapsulated in a more complete graph structure, as well as the few number of available genre tags introducing a lot of false negatives in the prediction (i.e. we expect that a lot of negative genre tags are in reality positive ones, but this information is not available in the ground truth).

On the other hand, the calculated relations of artists to genre groups in the second dataset are of high quality, yielding on average 99.8% AUC across genres. We attribute this success to the "related artists" suggesting artists of similar genres and the ability of our mechanism to re-construct this information. To assert this, we experimented with calculating artist relevance to a genre as the mean between their related artists residing in the non-withheld 20% of known tags and this already achieves on average 96.6% AUC across genres. Still, our system improves this evaluation by 3.2%.

Based on our findings, we assess the developed mechanism as reasonably successful in discovering high quality relevance scores of how artists relate to genre tags.

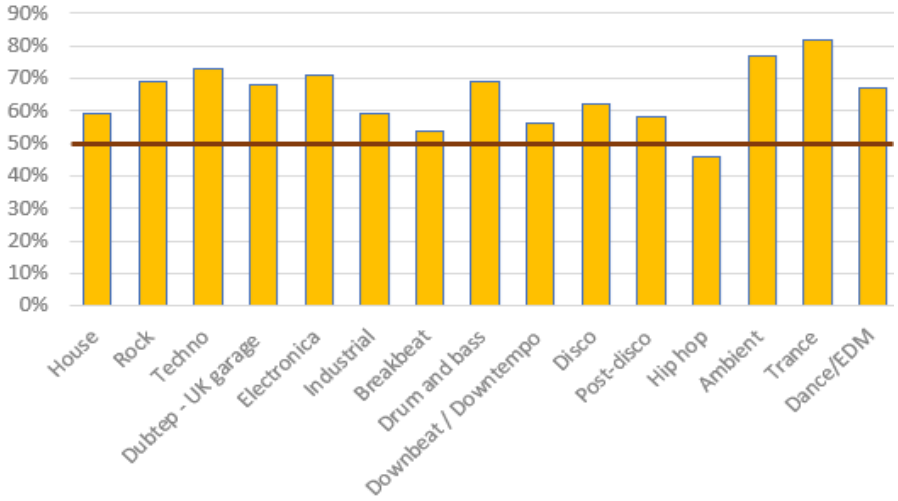


Figure 48 Genre rank AUC in the smaller dataset - average AUC 65%

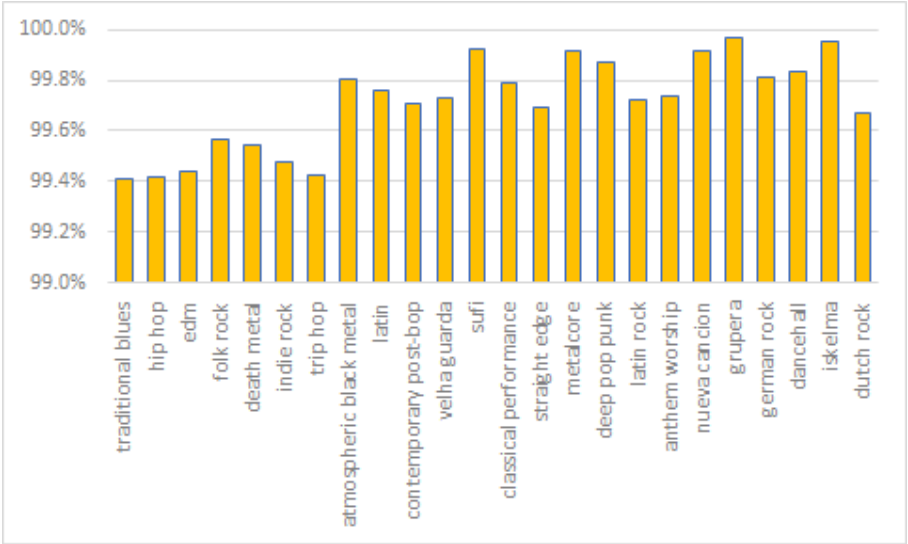


Figure 49 Genre rank AUC in the smaller dataset - average AUC 99.8%

Funded by the European Commission

5.4 Artist Popularity within Genre Communities

To discover artist popularity within a specific community (LM_REQ#5 - Artist popularity in a given genre), we consider the artist-venue relationships. In particular, we organize the data described in the previous task into a bipartite graph in which artists link to the venues they have performed in (see Figure 50). There are 290,811 such links in total.

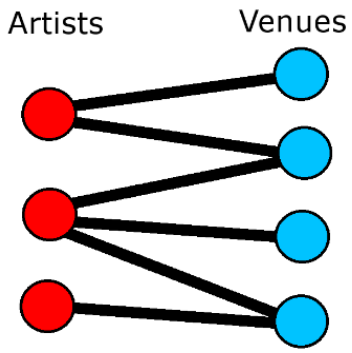


Figure 50 Artist-Venue network.

Our previous research [Krasanakis et al. 2018] shows that applying symmetric random walk with restart on artist-venue networks can help determine which venues contribute the most towards artist popularity when there is an informed initial estimation of some popular artists or when restart probability approaches zero. In this setting, algorithms rank higher the venues linked to popular artists and consider as more popular the artists linked to higher ranked venues. Therefore, the structure of artist-venue graphs can help discover popular nodes.

Motivated by this theorization, we again apply the methodology described in the previous section to calculate the relevance scores of artists towards genre tags. Since, this time, the graph's structure also promotes the discovery of more prominent artists, we expect the artist relevances calculated this way to indicate a combination of popularity and relevance to the genre and call them genre-based popularities. For example, an artist slightly less relevant to the genre but who is very popular could be ranked equally high to an unpopular artist very relevant to the genre, but both would be ranked more highly than artists who are less popular or less relevant to the genre than them.

To present values of genre popularities in a way that is intuitively understandable, we convert them to the range 0-100, whose extremes correspond to the artists with the lowest and highest genre popularity respectively.

5.4.1 Results

To find the importance of artists within the Soundtrack Your Brand and Playground genres, we applied the methodology described in this section to the artist-venue graph of the respective dataset. Contrary to the many initially missing genre tags, our approach assigns importance and relevance scores in at least one genre to all artists. Table 41 presents some examples of the genre-based relevance and the way artist popularity adapted to reflect popularity inside the genre community.

Funded by the European Commission

Artist	Spotify Popularity	Genre Relevance	Genre-Based Popularity
Electro House			
Marcus Schossow	45%	44%	75%
Vato Gonzalez	52%	100%	83%
Christian Sol	2%	7%	18%
New Rave			
Say Clap	50%	70%	100%
Royksopp	61%	73%	83%
Boys Noize	50%	92%	26%

Table 41 Genre-based artist popularity and relevance

5.5 Implementations and integrations of results

The associations detected by using the two approaches described in section 5.1 have been imported in the back-end database that is keeping the genre related information, alongside the other music entities of interest. We have also developed the appropriate endpoint in the API to get this information. More precisely, given a genre, all the associated genres can be retrieved:

GET */genres/<:genre_id>/associated_genres*

The type of association can also be defined by using the *type* parameter. In the current implementation we provide two types: *embedings_similarity* and *same_cluster*[32].

In the current version of the platform, we are still using the popularity scores generated with the initial formula, without taking into account subgenres counts[33]. To expose genre-level popularity we have pre-calculated popularity values at monthly, quarterly and annual basis since 2015-01-01. More precisely, we used 22 worldwide, 112 US and 53 UK charts of album, artist, single and track type, coming from traditional charts and streaming platforms (e.g. Spotify Top-200 charts and Spotify Viral Charts). We have also calculated the percentage change over the previous time period i.e. the previous year in case of annual popularity, the previous months in case of monthly estimation, etc. The API endpoint has remained unchanged[34].

[32] We also have a third type named same_genre that has been produced manually, and is used to associate genres between taxonomies. In the current implementation we have two taxonomies one defined by Future Pulse, and one coming from Spotify.
[33] We plan to update the produced popularities by using more principled and sophisticated approaches during the next period of the project.
[34] /genres/<:genre_id>/popularity

Funded by the European Commission

6 Summary and Conclusions

This deliverable presented the work conducted during the second year of the project to meet the requirements related to predictive analytics and recommendations. As in deliverable D3.1, the work carried out aims to produce popularity-oriented results for different music entities of interest, such as artists, tracks and genres. In most cases, the methodologies presented in this document extends the work presented in the first version of predictive analytics deliverable, based on the feedback we received from use case partners.

Regarding **song recognition** estimation, we studied collective memory dynamics and proposed a model for the approximation of the corresponding decreasing trajectory. Our recognition model comprises three main components: a) growth, b) decay, and c) proxy-based adjustment and it leverages chart data, YouTube views, Spotify popularity and forgetting curve dynamics. Also, our method considers different recognition decay rates and initial recognition levels per song, according to the number of weeks the song has remained in the charts.

We compared our model to other state-of-the art and baseline models on the task of accurately estimating the current recognition level of songs. To this end, we conducted a study in Sweden in order to measure the recognition level of 100 songs, which we then used as ground truth for the models' evaluation. The experimental results showed that our method exhibits great performance on this task, much better than the competitive models with a high statistical significance level.

We reached two remarkable conclusions:

- according to our model's parameters, a song needs almost 7 weeks in the charts to achieve a very slow velocity towards oblivion and at least 25 weeks to achieve its highest contemporary recognition;
- the role of the number-of-weeks feature incorporated in our model through the logistic functions is found to be of utmost importance for the accurate estimation of a song's recognition level.

Regarding a **track's popularity** estimation and prediction, the described approaches have shown promising results. Overall track popularity is estimated from the following sources: Deezer rank, Spotify popularity, views and likes on Youtube, and global airplay counts from BMAT.

We compared k-Nearest Neighbors (kNN) against Long-Short-Term-Memory (LSTM) deep learning models. LSTM resulted slightly better, but at a dramatically higher computational cost; therefore we decided to use the kNN implementation. In both cases, since they are based on machine learning models trained on historical crawled signals, those prediction models will theoretically become more and more accurate over time. In general, our different analyses have shown that a simple trend or seasonality can be easily predicted for a relatively small time prediction interval. However, the difficulty can rise quite sharply when the source signal is almost static (such as in the case of Spotify popularity), the predicted target goes farther in the future, or when an unexpected trend appears without any detectable pattern in the previous days of history in the data. We

Funded by the European Commission

have studied that a multivariate LSTM approach can potentially overcome these issues when more data becomes available. In the current implementation the k-Nearest Neighbors approach is used to predict a track's popularity up to 21 days into the future, based on up to 28 days of history.

For **artist popularity** we propose a non-linear aggregation method in order to combine diverse sources of popularity information e.g. Spotify followers, YouTube views, Last.fm playcounts etc. This method leverages geometrical shapes formed by the normalized metric values obtained for each artist and combines them by computing a fraction where the numerator corresponds to the artist under study and the denominator corresponds to the best possible case i.e. the most popular possible artist. The results showed that this method outperforms the most natural choice being a simple average and also it outperforms other non-linear metric aggregation methods in terms of correlation, rank correlation and rank distance with the ground truth.

The **impact of events**, such as album release, TV show appearance or interview, on an artist's popularity level is also studied herein. It is remarkable that no significant changes are observed on popularity metrics such as YouTube views/subscribers and Last.fm playcounts after the events but changes are observed on streaming activity (Spotify, iTunes, Deezer streams). We compare two different methods that estimate the level of impact an event has on future popularity values/streaming activity. The segmented linear regression method shows good performance identifying accurately the upcoming changes after an event.

For the estimation of **genre popularity** and growth we conducted a preliminary work to tackle the problem of data sparsity. We analysed genre occurrences in Spotify artists using a graph embedding technique in order to identify sub-genre associations between genres. That information is then used to count genre appearances in music charts. However, although the first results are promising, we need to further evaluate the identified associations as well as the popularity scores generated when these associations are considered.

Future activities for the coming period have the following goals:

- Analyze playlists and develop a methodology to detect similar playlists based on co-listening patterns, content similarity and music genres
- Update tracks popularity estimation and prediction by adding more sources e.g. country-wise airplays, charts, playlists, Spotify analytics data etc.
- Investigate multivariate approaches (e.g. LSTM) again with more data available
- Update and evaluate genre popularity estimation, by using genre associations
- Combine in a principled way the several artist popularity estimations developed separately for each of the use cases.
- Co-inform track popularity estimation and artist popularity estimation mutually

Funded by the European Commission

7 References

R. P. Adams, D. J. C. MacKay. Bayesian Online Changepoint Detection, *arXiv:0710.3742, 2007*.

R. Agrawal, T. Imieliński, A. Swami. Mining association rules between sets of items in large databases. *In Proceedings of the 1993 ACM SIGMOD international conference on Management of data,* 22(2):207-216, 1993.

R. Andersen, F. Chung, K. Lang. Local graph partitioning using pagerank vectors. *In 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06),* pp. 475-486, 2006.

R. Andersen, F. Chung, K. Lang. Local partitioning for directed graphs using PageRank. *In International Workshop on Algorithms and Models for the Web-Graph*, pp. 166-178, 2007.

A. Bellogin, A. P. de Vries, J. He. Artist popularity: Do web and social music services agree? *In Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media*, 2013.

G. E. Box and G. C. Tiao. Intervention Analysis with Applications to Economic and Environmental Problems. *Journal of the American Statistical Association*, 70(349): 70-79, 1975.

J. P. Brans and Ph. Vincke. A Preference Ranking Organisation Method: (The PROMETHEE Method for Multiple Criteria Decision-Making). *Management Science*, 31(6):647-656, 1985.

C. Candia, C. Jara-Figueroa, C. Rodriguez-Sickert, A. L. Barabasi and C. A. Hidalgo. The universal decay of collective memory and attention. *Nature Human Behaviour*, 3(1):82-91, 2019.

G. Casella and R. L. Berger. Statistical inference. Duxbury Pacific Grove, CA, vol. 2, 2002.

V. Cerqueira, L. Torgo, I. Mozetic. Evaluating time series forecasting models: An empirical study on performance estimation methods. *arXiv:1905.11744*, 2019.

C. Dwork, R. Kumar, M. Naor and D. Sivakumar. Ranking aggregation methods for the web. *In Proceedings of the 10th international conference on World Wide Web*, pp. 613-622, 2001.

J. Grace, D. Gruhl, K. Haas, M. Nagarajan, C. Robson, and N. Sahoo. Artist ranking through analysis of online community comments. *In Proceedings of the 17th International World Wide Web Conference*, 2008.

A. Grover, J. Leskovec. node2vec: Scalable feature learning for networks. *In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining,* pp. 855-864, 2016.

V. Guralnik and J. Srivastava. Event detection from time series data. *In Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '99)*, pp. 33-42, 1999.

S. Hochreiter, J. Schmidhuber. Long short-term memory. *Neural computation,* 9 (8): 1735-1780, 1997.

D. Jarušková. Some Problems with Application of Change-Point Detection Methods to Environmental Data. *Environmetrics*, 8: 469-483, 1997.

Funded by the European Commission

N. Koenigstein and Y. Shavitt. Song ranking based on piracy in peer-to-peer networks. *In Proceedings of the 10th International Society for Music Information Retrieval Conference*, pp. 633–638, 2009.

Y. Kim, B. Suh, and K. Lee. #nowplaying the future billboard: Mining music listening behaviors of twitter users for hit song prediction. *In Proceedings of the First International Workshop on Social Media Retrieval and Analysis*, pp. 51–56, 2014.

A. Koski, R. Siren, E. Vuori, and K. Poikolainen. Alcohol tax cuts and increase in alcohol-positive sudden deaths: a time-series intervention analysis. *Addiction*, 102(3): 362-368, 2007.

E. Krasanakis, E. Schinas, S. Papadopoulos, Y. Kompatsiaris, P. Mitkas. VenueRank: Identifying Venues that Contribute to Artist Popularity. *In Proceedings of the 19th International Society for Music Information Retrieval Conference*, pp. 702-708, 2018.

E. Krasanakis, E. Schinas, S. Papadopoulos, Y. Kompatsiaris, A. Symeonidis. Boosted seed oversampling for local community ranking. *Information Processing & Management*, 102053, 2019.

M. Lagarde. How to do (or not to do) ... Assessing the impact of a policy change with routine longitudinal data. *Health Policy Plan*, 27(1): 76-83, 2011.

G. R. Loftus. Evaluating Forgetting Curves. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 11(2): 397-406, 1985.

J. K. Mbugua, G. H. Bloom, M. M. Segall. Impact of user charges on vulnerable groups: The case of Kibwezi in rural Kenya, *Social Science & Medicine*, 41(6): 829-835, 1995.

C. Mesnage, R. Santos-Rodriguez, M. McVicar, and T. De Bie. Trend extraction on twitter time series for music discovery. *In Workshop on Machine Learning for Music Discovery, 32nd International Conference on Machine Learning*, 2015.

S. Moses, F. Plummer, F. Manji, J. Bradley, N. Nagelkerke, and M. Malisa, Impact of user fees on attendance at a referral centre for sexually transmitted diseases in Kenya, *The Lancet*, 340(8817): 463-466, 1992.

J. M. J. Murre, and J. Dros. Replication and Analysis of Ebbinghaus' Forgetting Curve. *PLoS ONE*, *10*(7), e0120644, 2015.

J. P. Murry, A. Stam, and J. Lastovicka. Evaluating an Anti-Drinking and Driving Advertising Campaign with a Sample Survey and Time Series Intervention Analysis. *Journal of the American Statistical Association*, 88(421): 50–56, 1993.

L. Page, S. Brin, R. Motwani, T. Winograd. The PageRank citation ranking: Bringing order to the web. *Stanford InfoLab*, 1999.

J. Ren, J. Shen, and R. J. Kauffman. What makes a music track popular in online social networks? *In Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 95–96, 2016.

M. Schedl, T. Pohle, N. Koenigstein, and P. Knees. What's hot? estimating country-specific artist popularity. *In Proceedings of the 11th International Society for Music Information Retrieval Conference*, pp. 117–122. ISMIR, 2010.

M. Schedl. Analyzing the potential of microblogs for spatio-temporal popularity estimation of music artists. *In Proceedings of the IJCAI*, 2011.

T., Hanghang, C. Faloutsos, and J. Y. Pan. Fast random walk with restart and its applications. *In 6th International Conference on Data Mining (ICDM'06)*. IEEE, 2006.

D. Wang, C. Song and A. L. Barabasi. Quantifying Long-Term Scientific Impact. *Science*, 342(6154):127-132, 2013.

F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6), 80-83, 1945.

K. Yoon. A Reconciliation among Discrete Compromise Solutions. *Journal of the Operational Research Society*, 38 (3): 277–286, 1987.

Appendix A

This appendix contains the top-100 lists of most recognized artists in Sweden and US, as calculated with the T-REC approach presented in Section 3.1.1.

Table 42 The Top-100 recognized songs in Sweden according to T-REC.

1	More Than You Know	Axwell /\ Ingrosso	84.4
2	rockstar	Post Malone	84.2
3	Never Be Like You (feat. Kai)	Flume	83.4
4	Havana	Camila Cabello	83.3
5	Despacito (Featuring Daddy Yankee)	Daddy Yankee	82.9
6	Despacito - Remix	Daddy Yankee	82.7
7	Thunder	Imagine Dragons	82.5
8	Mambo No. 5 (A Little Bit of...)	Lou Bega	82.4
9	Last Christmas	Various Artists	81.9
10	Shape of You	Ed Sheeran	81.4
11	Stay With Me	Sam Smith	81.3
12	Hot N Cold	Katy Perry	81.2
13	Poker Face	Lady Gaga	81.1
14	All I Want for Christmas Is You	Mariah Carey	81
15	Hymn For The Weekend	Coldplay	80.9
16	Perfect	Ed Sheeran	80.9
17	Dancing On My Own	Calum Scott	80.8
18	All of Me	John Legend	80.8
19	Un-Break My Heart	Toni Braxton	80.7
20	Bad Romance	Lady Gaga	80.6
21	Lean On	Major Lazer	80.6
22	Cheerleader - Felix Jaehn Remix Radio Edit	OMI	80.5
23	Titanium (feat. Sia)	David Guetta	80.3
24	Something Just Like This	The Chainsmokers	80.2
25	Timber	Pitbull	80
26	El Perdon	Nicky Jam	79.9
27	Someone Like You	Adele	79.8
28	When Love Takes Over (feat. Kelly Rowland)	David Guetta	79.7

Funded by the European Commission

29	Thinking Out Loud	Ed Sheeran	79.6
30	Grenade	Bruno Mars	79.6
31	Rather Be (feat. Jess Glynne)	Jess Glynne	79.6
32	Attention	Charlie Puth	79.5
33	Symphony (feat. Zara Larsson)	Clean Bandit	79.4
34	I'm Yours	Jason Mraz	79.3
35	I Gotta Feeling	The Black Eyed Peas	79.3
36	Human	The Killers	79.2
37	New Rules	Dua Lipa	79.2
38	TiK ToK	Kesha	79
39	I Love It (feat. Charli XCX)	Icona Pop	79
40	My Heart Will Go On - Love Theme from "Titanic"	Celine Dion	78.9
41	Waiting For Love	Avicii	78.9
42	Closer	The Chainsmokers	78.9
43	1-800-273-8255	Logic	78.8
44	Meant to Be (feat. Florida Georgia Line)	Bebe Rexha	78.7
45	The Ocean	Mike Perry	78.7
46	Give Me Everything	Pitbull	78.6
47	Cheap Thrills	Sia	78.5
48	Unforgettable	French Montana	78.5
49	Say You Won't Let Go	James Arthur	78.5
50	Coco Jamboo	Mr. President	78.5
51	We Found Love	Rihanna	78.4
52	Faded	Alan Walker	78.4
53	Let It Go	James Bay	78.4
54	What Lovers Do (feat. SZA)	Maroon 5	78.4
55	Mercy	Duffy	78.2
56	Bailando	Paradisio	78.1
57	Congratulations	Post Malone	78.1
58	I Need Your Love	Calvin Harris	78.1
59	Hymn For The Weekend - Seeb Remix	Coldplay	78.1
60	Do They Know It's Christmas? - 1984 Version	Band Aid 20	78

Funded by the European Commission

61	Don't Let Me Down	The Chainsmokers	77.9
62	Believer	Imagine Dragons	77.9
63	What Do You Mean?	Justin Bieber	77.9
64	Just Dance	Lady Gaga	77.8
65	Children	Robert Miles	77.8
66	Lush Life	Zara Larsson	77.8
67	This Is The Life	Amy Macdonald	77.8
68	You're My Heart You're My Soul	Modern Talking	77.8
69	I Kissed a Girl	Katy Perry	77.7
70	Whenever Wherever	Shakira	77.7
71	Sorry	Justin Bieber	77.7
72	Swalla (feat. Nicki Minaj & Ty Dolla $ign)	Nicki Minaj	77.7
73	SUBEME LA RADIO	Enrique Iglesias	77.6
74	The Nights	Avicii	77.6
75	Sunset Lover	Petit Biscuit	77.5
76	Hey Brother	Avicii	77.5
77	Cold Water	Major Lazer	77.5
78	Rockabye (feat. Sean Paul & Anne-Marie)	Clean Bandit	77.4
79	Too Good At Goodbyes	Sam Smith	77.3
80	CAN'T STOP THE FEELING! (Original Song from DreamWorks Animation's "TROLLS")	Various Artists	77.3
81	Levels - Radio Edit	Avicii	77.3
82	I'm the One	DJ Khaled	77.3
83	Fran och med Du	Oskar Linnros	77.3
84	It Feels So Good	Sonique	77.2
85	Dilemma	Nelly	77.1
86	Mama	Jonas Blue	77.1
87	I'll Be Missing You (feat. 112)	Various Artists	77.1
88	Fireflies	Owl City	77.1
89	I Don't Feel Like Dancin'	Scissor Sisters	77.1
90	Can't Feel My Face	The Weeknd	77
91	Umbrella	Rihanna	77
92	Ain't Nobody (Loves Me Better)	Felix Jaehn	77

Funded by the European Commission

93	How Far I'll Go - From "Moana"	Alessia Cara	77
94	Boom Boom Boom Boom!!	Vengaboys	76.9
95	Perfect Strangers	Jonas Blue	76.9
96	See You Again (feat. Charlie Puth)	Charlie Puth	76.9
97	Truly Madly Deeply	Savage Garden	76.8
98	Halo	Beyonce	76.8
99	Cotton Eye Joe	Rednex	76.8
100	If I Were a Boy	Beyonce	76.8

Table 43 The Top-100 recognized songs in USA according to T-REC.

1	Escápate Conmigo	Wisin	86.1
2	Ginza	J Balvin	86.1
3	I Knew I Loved You	Various Artists	86
4	Thunder	Imagine Dragons	86
5	Believer	Imagine Dragons	85.9
6	Whatever It Takes	Imagine Dragons	85.8
7	I Get The Bag (feat. Migos)	Gucci Mane	85.5
8	Stay With Me	Sam Smith	85.5
9	No Roots	Alice Merton	85.3
10	Stayin' Alive	Bee Gees	84.9
11	Look What You Made Me Do	Taylor Swift	84.9
12	Baby Come Back	Player	84.8
13	rockstar	Post Malone	84.8
14	Glad You Came	The Wanted	84.8
15	Havana	Camila Cabello	84.7
16	Leave Get Out	JoJo	84.7
17	Never Be Like You (feat. Kai)	Flume	84.6
18	Hold Me (feat. Tobymac)	Jamie Grace	84.4
19	Felices los 4	Maluma	84.4
20	Baby Come Back	Player	84.3
21	Attention	Charlie Puth	84.3
22	You And Me	Lifehouse	84.3

Funded by the European Commission

23	Bottoms Up	Brantley Gilbert	84.3
24	It Won't Stop (feat. Chris Brown)	Sevyn Streeter	84.3
25	Greatest Love Story	LANCO	84.3
26	Dancing On My Own	Calum Scott	84.1
27	Genie in a Bottle	Christina Aguilera	84.1
28	Dilemma	Nelly	84.1
29	MotorSport	Cardi B	84
30	Permission	Ro James	84
31	Try Again	Aaliyah	83.8
32	Beautiful Day	Jamie Grace	83.8
33	Vacaciones	Wisin	83.8
34	My Story	Big Daddy Weave	83.8
35	Say It	Tory Lanez	83.6
36	...Ready For It?	Taylor Swift	83.5
37	Rather Be (feat. Jess Glynne)	Jess Glynne	83.4
38	Home	Phillip Phillips	83.2
39	Whatever She's Got	David Nail	83.1
40	Waiting for a Star to Fall	Boy Meets Girl	83.1
41	Firework	Katy Perry	83.1
42	Party Rock Anthem	LMFAO	83
43	Despacito (Featuring Daddy Yankee)	Daddy Yankee	82.9
44	My Way (feat. Monty)	Monty	82.9
45	Say It (feat. Tove Lo)	Flume	82.9
46	All About That Bass	Meghan Trainor	82.9
47	Come on over Baby (All I Want Is You) - Radio Version	Christina Aguilera	82.8
48	Hey Mama (feat. Nicki Minaj Bebe Rexha & Afrojack)	David Guetta	82.8
49	Tell Your Heart to Beat Again	Danny Gokey	82.8
50	Mambo No. 5 (A Little Bit of...)	Lou Bega	82.8
51	Shake It Off	Taylor Swift	82.8
52	It's Not Over Yet	for KING & COUNTRY	82.7
53	Girls Just Want to Have Fun	Various Artists	82.7
54	Despacito - Remix	Daddy Yankee	82.7

Funded by the European Commission

55	Body Party	Ciara	82.7
56	We Don't Talk Anymore (feat. Selena Gomez)	Charlie Puth	82.6
57	Lean On	Major Lazer	82.5
58	Wishing Well	Terence Trent D'Arby	82.5
59	The Power	SNAP!	82.5
60	Ay Mi Dios	IAmChino	82.4
61	Ex-Factor	Ms. Lauryn Hill	82.4
62	The House of the Rising Sun	The Animals	82.4
63	All Night Long (All Night) - Single Version	Various Artists	82.3
64	Dangerous Woman	Ariana Grande	82.3
65	Wonderwall	Oasis	82.3
66	All On Me	Devin Dawson	82.3
67	Bad Romance	Lady Gaga	82.3
68	679 (feat. Remy Boyz)	Fetty Wap	82.3
69	Rolling in the Deep	Adele	82.3
70	TiK ToK	Kesha	82.2
71	El Mismo Sol	Alvaro Soler	82.2
72	Cheerleader - Felix Jaehn Remix Radio Edit	OMI	82.2
73	What a Girl Wants	Christina Aguilera	82.2
74	No Sleeep (feat. J. Cole)	Janet Jackson	82.2
75	The Ketchup Song (Asereje) - Spanish Version	Las Ketchup	82.2
76	Mi Gente	J Balvin	82.2
77	We Are Never Ever Getting Back Together	Taylor Swift	82.2
78	Hey DJ	CNCO	82.2
79	Hope in Front of Me	Danny Gokey	82.1
80	Single Ladies (Put a Ring on It)	Beyonce	82.1
81	OOOUUU	Young M.A	82.1
82	Loco	Enrique Iglesias	82.1
83	I Love It (feat. Charli XCX)	Icona Pop	82
84	See You Again (feat. Charlie Puth)	Charlie Puth	82

Funded by the European Commission

85	Hasta el Amanecer	Nicky Jam	82
86	Sin Contrato	Maluma	82
87	El Perdedor	Maluma	82
88	She Drives Me Crazy	Fine Young Cannibals	82
89	Say Something	A Great Big World	82
90	Cheap Thrills	Sia	81.9
91	Thriller	Michael Jackson	81.9
92	La Bicicleta	Shakira	81.9
93	Total Eclipse of the Heart	Various Artists	81.9
94	Titanium (feat. Sia)	David Guetta	81.8
95	Shape of You	Ed Sheeran	81.8
96	Dile Que Tu Me Quieres	Ozuna	81.8
97	E.T.	Katy Perry	81.8
98	6:00 AM	J Balvin	81.8
99	Shaky Shaky	Daddy Yankee	81.8
100	Don't Speak	No Doubt	81.8

Funded by the European Commission

Appendix B

This appendix contains the two proofs referred to section 4.2. Namely, we prove that GAP0 and GAP1 can be calculated by simple formulas (Proof 1) and that

$$AAP(m) \leq GAP_1(m) \leq GAP_0(m)$$

for all m (Proof 2).

Proof 1

For $GAP0(m)$ the inner polygon's area is the sum of *n* triangles' areas:

$$\sum_{i=1}^{n} \frac{1}{2}\left(1 - m_{a,i,t}\right) \cdot \left(1 - m_{a,i+1,t}\right) \cdot sin\ sin\ \theta$$

where $\theta = \frac{2\pi}{n}$.

The outer polygon's area is the sum of *n* equal triangles' areas:

$$n \cdot \left(\frac{1}{2} \cdot 1 \cdot 1 \cdot sin\ sin\ \theta\right) = \frac{n \cdot sin\ sin\ \theta}{2}$$

Hence,

$$GAP0(m) = \frac{\frac{n \cdot sin\ sin\ \theta}{2} - \sum \frac{1}{2}\left(1 - m_{a,i,t}\right) \cdot \left(1 - m_{a,i+1,t}\right) \cdot sin\ sin\ \theta}{\frac{n \cdot sin\ sin\ \theta}{2}} =$$

$$1 - \frac{1}{n}\sum \left(1 - m_{a,i,t}\right) \cdot \left(1 - m_{a,i+1,t}\right) = \frac{\sum 1 - \left(1 - m_{a,i,t}\right) \cdot \left(1 - m_{a,i+1,t}\right)}{n} =$$

$$\frac{1}{n}\sum_{i=1}^{n} \left(m_{a,i,t} + m_{a,i+1,t}\left(1 - m_{a,i,t}\right)\right)$$

For $GAP1(m)$ the inner polygon's area is the sum of *n* isosceles triangles' areas:

$$\sum_{i=1}^{n} \frac{1}{2}\left(1 - m_{a,i,t}\right)^2 \cdot sin\ sin\ \theta$$

The outer polygon's area is the same as before:

$$\frac{n \cdot sin\ sin\ \theta}{2}$$

Hence,

$$GAP1(m) = \frac{\frac{n \cdot sin\ sin\ \theta}{2} - \sum \frac{1}{2}\left(1 - m_{a,i,t}\right)^2 \cdot sin\ sin\ \theta}{\frac{n \cdot sin\ sin\ \theta}{2}} = 1 - \frac{1}{n}\sum \left(1 - m_{a,i,t}\right)^2 =$$

$$\frac{1}{n}\sum_{i=1}^{n} \left(2 \cdot m_{a,i,t} - {m_{a,i,t}}^2\right) \qquad \blacksquare$$

Funded by the European Commission

Proof 2

The first part of the inequality is straightforward:

$$m_{a,i,t} \leq 1 \Rightarrow {m_{a,i,t}}^2 \leq m_{a,i,t} \Rightarrow 0 \leq m_{a,i,t} - {m_{a,i,t}}^2 \Rightarrow m_{a,i,t} \leq 2 \cdot m_{a,i,t} - {m_{a,i,t}}^2 \Rightarrow$$

$$\frac{1}{n}\sum_{i=1}^{n} m_{a,i,t} \leq \frac{1}{n}\sum_{i=1}^{n}\left(2 \cdot m_{a,i,t} - {m_{a,i,t}}^2\right) \Rightarrow AAP(m) \leq GAP1(m)$$

For the second part of the inequality we begin with the assumption that m is sorted:

$$m_{a,i,t} \leq m_{a,i+1,t}\ \forall i = 1, \dots, n-1$$

The difference D_i between the models $GAP1$ and $GAP0$ per metric i is:

$$D_i = \left(2 \cdot m_{a,i,t} - {m_{a,i,t}}^2\right) - \left(m_{a,i,t} + m_{a,i+1,t}\left(1 - m_{a,i,t}\right)\right) =$$

$$m_{a,i,t} - {m_{a,i,t}}^2 - m_{a,i+1,t} + m_{a,i+1,t} \cdot m_{a,i,t} =$$

$$m_{a,i,t}\left(1 - m_{a,i,t}\right) - m_{a,i+1,t}\left(1 - m_{a,i,t}\right) =$$

$$\left(1 - m_{a,i,t}\right)\left(m_{a,i,t} - m_{a,i+1,t}\right) \leq 0, \qquad \forall i = 1, \dots, n-1$$

and the corresponding difference for $i = n$ is $D_n = \left(1 - m_{a,n,t}\right)\left(m_{a,n,t} - m_{a,1,t}\right) \geq 0$.

The total difference between the two models then is:

$$GAP1(m) - GAP0(m) = \frac{1}{n}\sum_{i=1}^{n}\left(2 \cdot m_{a,i,t} - {m_{a,i,t}}^2\right) - \frac{1}{n}\sum_{i=1}^{n}\left(m_{a,i,t} + m_{a,i+1,t}\left(1 - m_{a,i,t}\right)\right) =$$

$$\frac{1}{n}\sum_{i=1}^{n}\left\{\left(2 \cdot m_{a,i,t} - {m_{a,i,t}}^2\right) - \left(m_{a,i,t} + m_{a,i+1,t}\left(1 - m_{a,i,t}\right)\right)\right\} = \frac{1}{n}\sum_{i=1}^{n} D_i =$$

$$\frac{1}{n}\left(\left(1 - m_{a,n,t}\right)\left(m_{a,n,t} - m_{a,1,t}\right) + \sum_{i=1}^{n-1}\left(1 - m_{a,i,t}\right)\left(m_{a,i,t} - m_{a,i+1,t}\right)\right) =$$

$$\frac{1}{n}\left(m_{a,n,t} - m_{a,1,t} - {m_{a,n,t}}^2 + m_{a,1,t} \cdot m_{a,n,t} + \sum_{i=1}^{n-1}\left(m_{a,i,t} - m_{a,i+1,t} - {m_{a,i,t}}^2 + m_{a,i+1,t} \cdot m_{a,i,t}\right)\right) =$$

$$\frac{1}{n}\left(m_{a,n,t} - m_{a,1,t} - {m_{a,n,t}}^2 + m_{a,1,t} \cdot m_{a,n,t} + \sum_{i=1}^{n-1} m_{a,i,t} - \sum_{i=1}^{n-1} m_{a,i+1,t} - \sum_{i=1}^{n-1} {m_{a,i,t}}^2 + \sum_{i=1}^{n-1} m_{a,i+1,t} \cdot m_{a,i,t}\right) =$$

Funded by the European Commission

$$\frac{1}{n}\left(m_{a,n,t} - m_{a,1,t} - {m_{a,n,t}}^2 + m_{a,1,t} \cdot m_{a,n,t} + m_{a,1,t} - m_{a,n,t} - \sum_{i=1}^{n-1} {m_{a,i,t}}^2 + \sum_{i=1}^{n-1} m_{a,i+1,t} \cdot m_{a,i,t}\right) = \frac{1}{n}(m^T m_r - m^T m)$$

where $m_r = [m_{a,2,t}, m_{a,3,t}, \dots, m_{a,n,t}, m_{a,1,t}]$ is m rolled by -1.

According to Cauchy-Schwarz inequality:

$$|\langle m, m_r \rangle|^2 \leq \langle m, m \rangle \langle m_r, m_r \rangle \Rightarrow$$

$$(m^T m_r)^2 \leq (m^T m) \cdot (m_r^T m_r) = (m^T m)^2 \; m_{a,i,t} \geq 0 \Rightarrow$$

$$m^T m_r - m^T m \leq 0 \Rightarrow$$

$$\frac{1}{n}(m^T m_r - m^T m) \leq 0 \Rightarrow$$

$$GAP1(m) - GAP0(m) \leq 0 \Rightarrow$$

$$GAP1(m) \leq GAP0(m)$$

■

Funded by the European Commission

Appendix C

This appendix contains examples of events, compared to timelines of metrics, for several PGM artists.

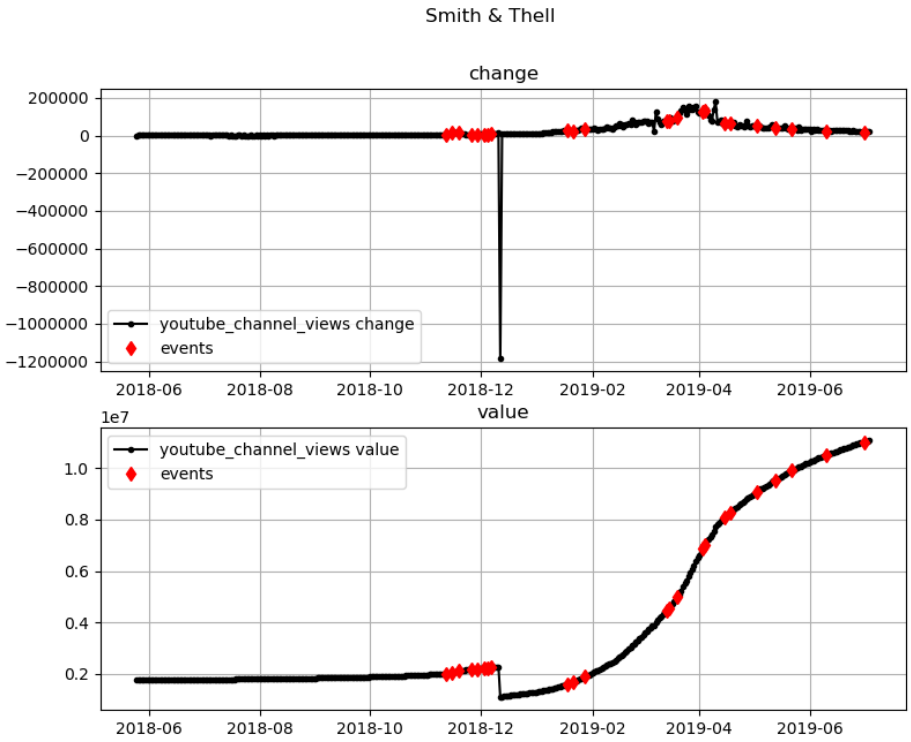


Figure 51 Timeline of YouTube channel views (value) and its derivative (change) along with event dates for Smith & Thell

Funded by the European Commission

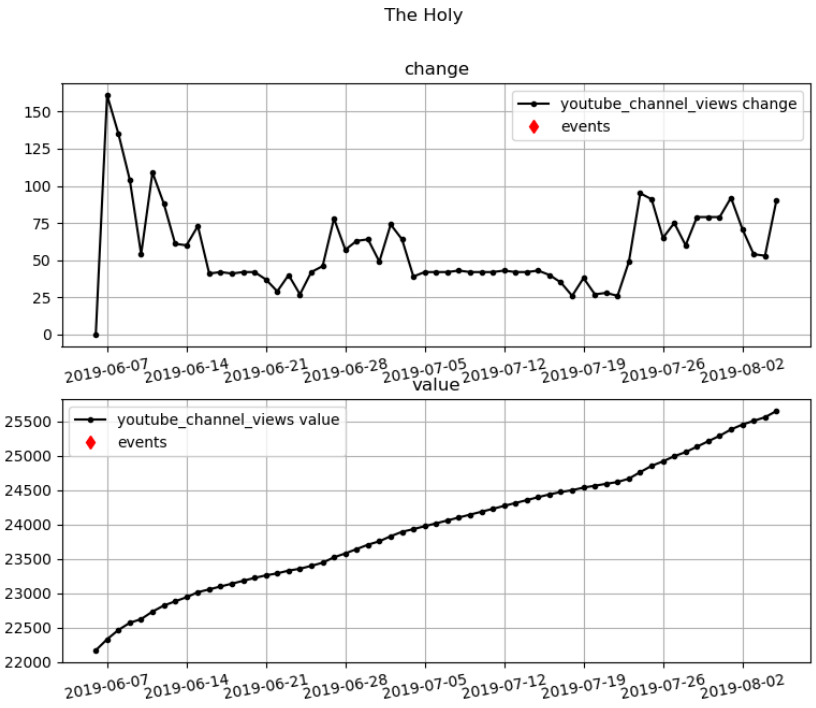


Figure 52 Timeline of YouTube channel views (value) and its derivative (change) along with event dates for The Holy

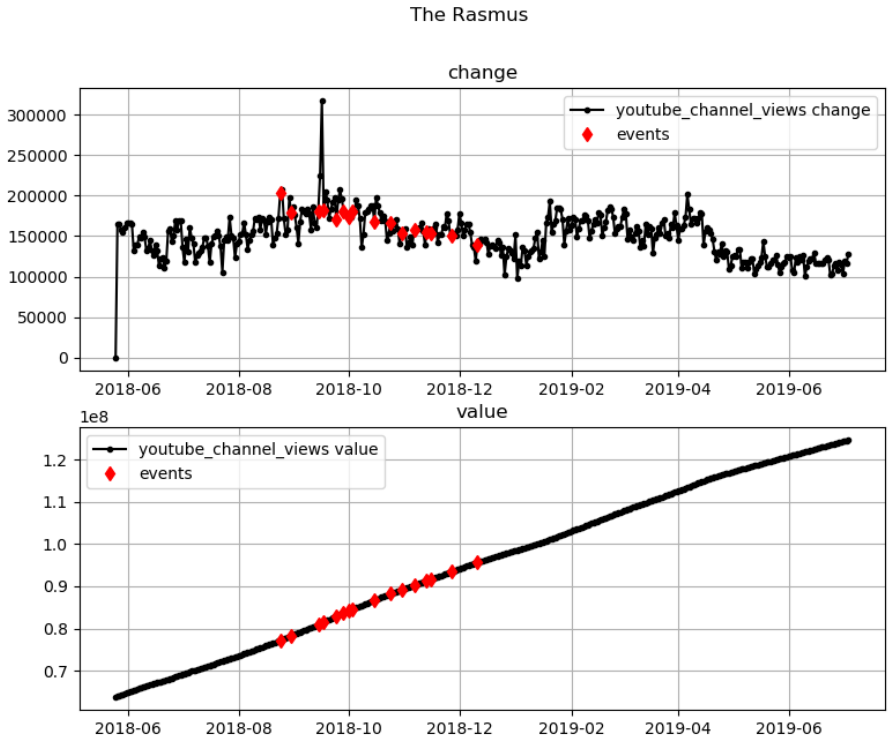


Figure 53 Timeline of YouTube channel views (value) and its derivative (change) along with event dates for The Rasmus

Funded by the European Commission

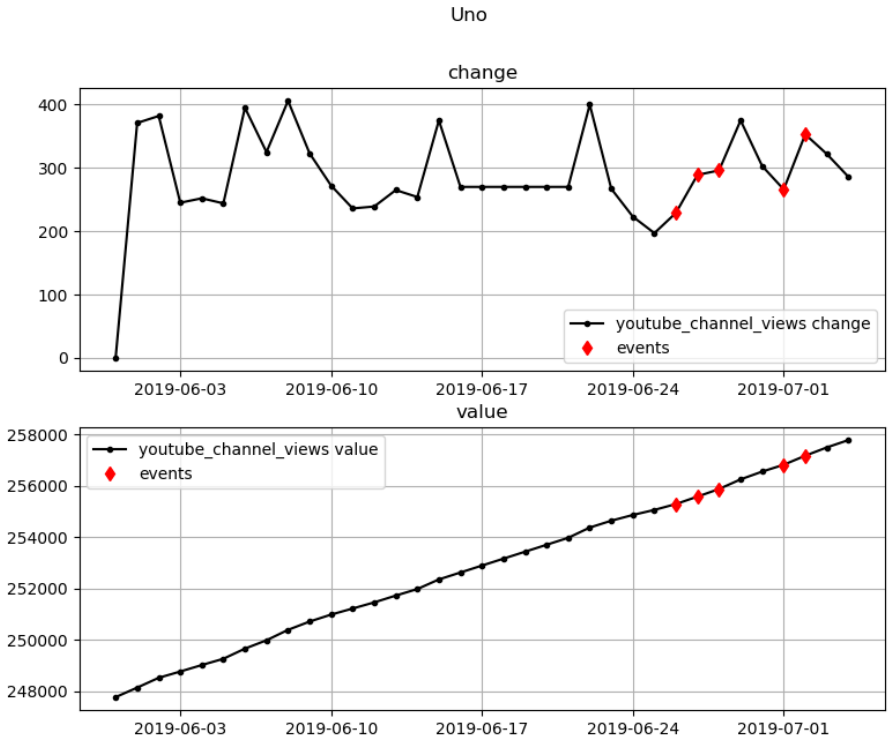


Figure 54 Timeline of YouTube channel views (value) and its derivative (change) along with event dates for Uno

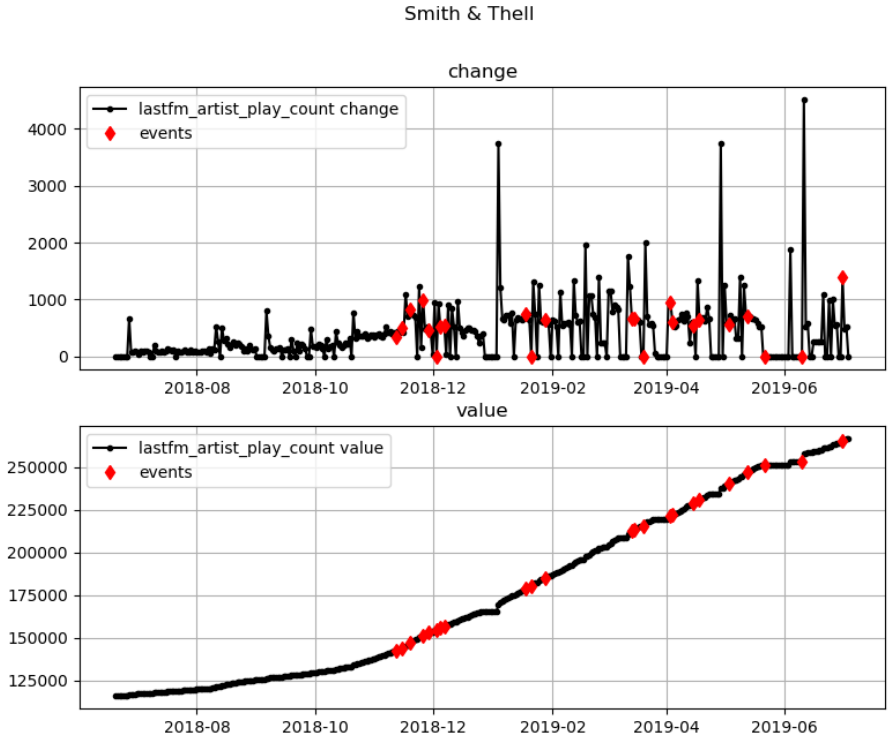


Figure 55 Timeline of Last.fm artist play counts (value) and its derivative (change) along with event dates for Smith & Thell

Funded by the European Commission

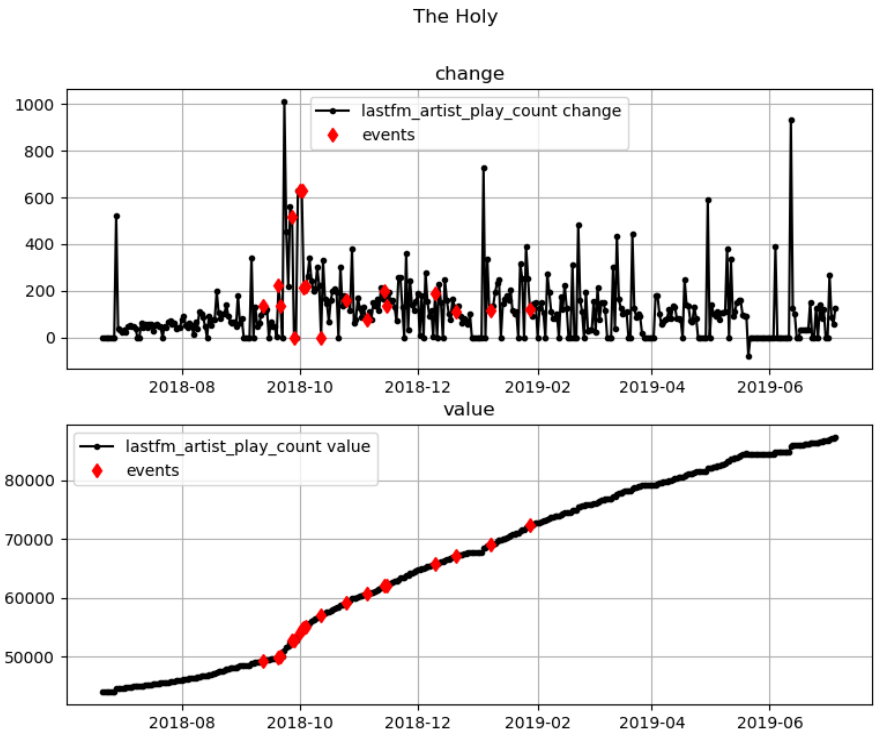


Figure 56 Timeline of Last.fm artist play counts (value) and its derivative (change) along with event dates for The Holy

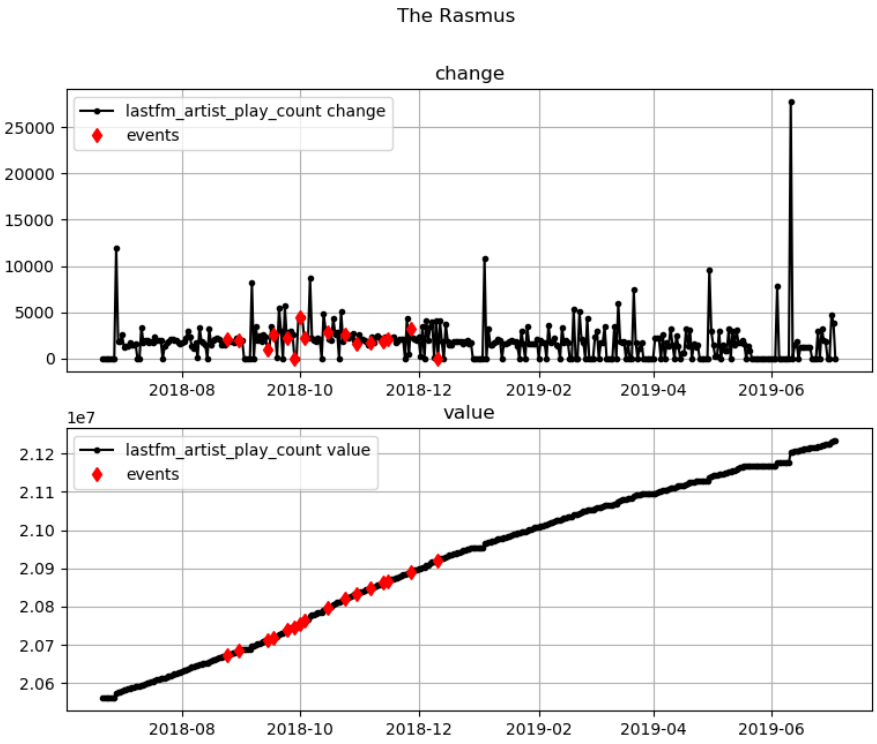


Figure 57 Timeline of Last.fm artist play counts (value) and its derivative (change) along with event dates for The Rasmus

Funded by the European Commission

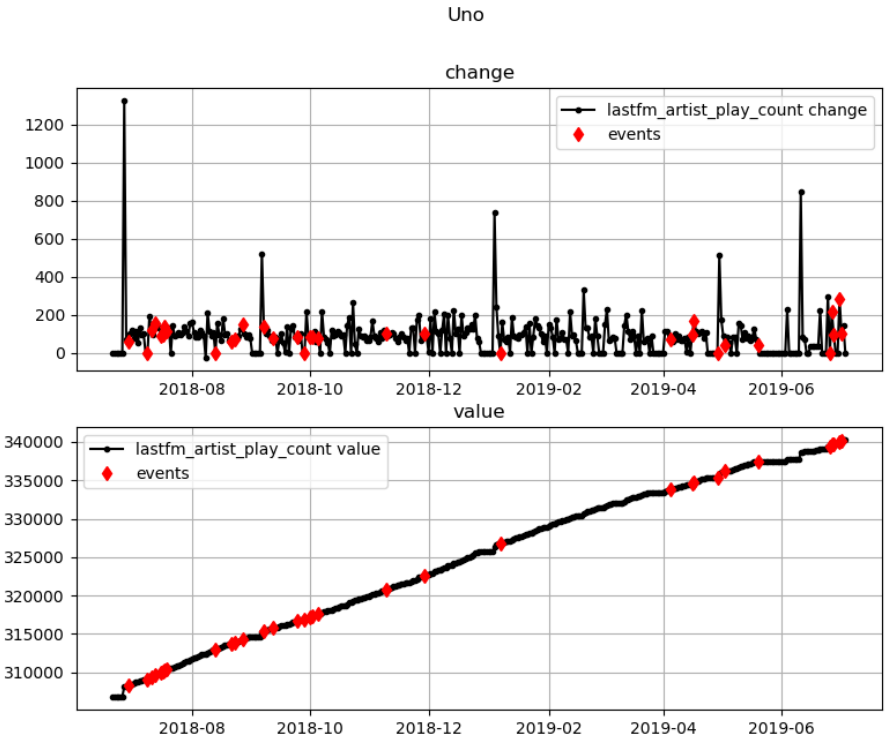


Figure 58 Timeline of Last.fm artist play counts (value) and its derivative (change) along with event dates for Uno

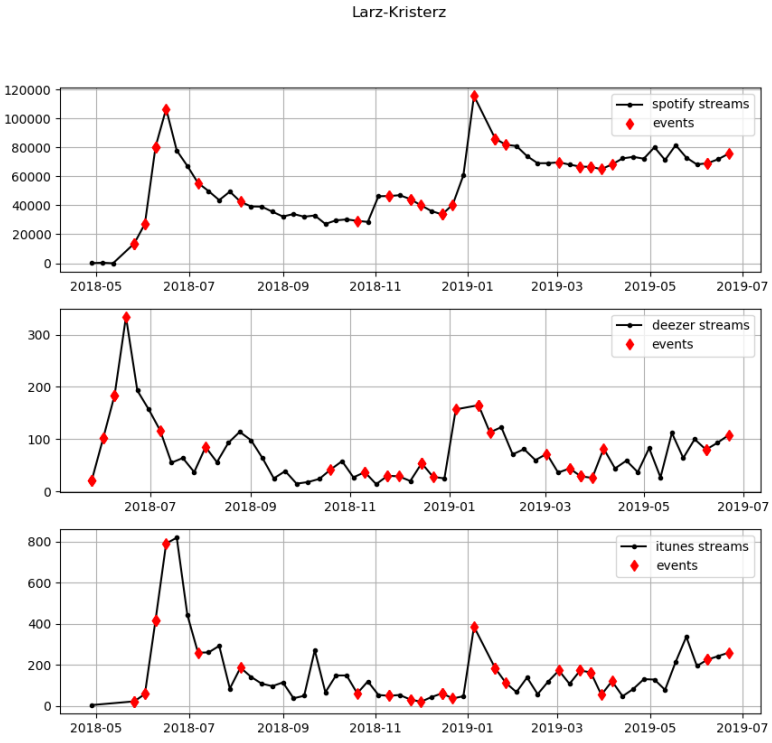


Figure 59 Spotify, Deezer and iTunes streams timelines and event dates for Larz-Kristerz

Funded by the European Commission

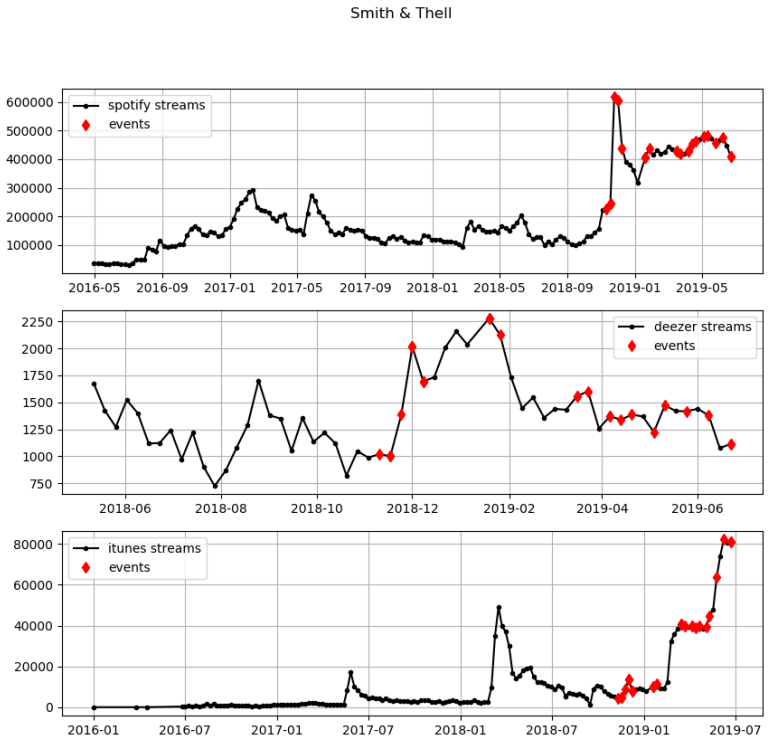


Figure 60 Spotify, Deezer and iTunes streams timelines and event dates for Smith & Thell

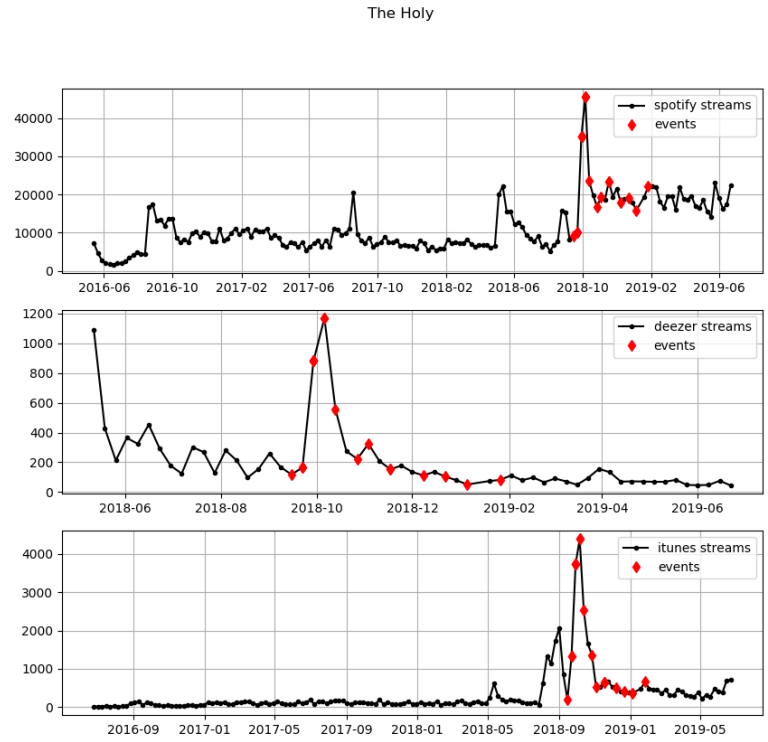


Figure 61 Spotify, Deezer and iTunes streams timelines and event dates for The Holy

Funded by the European Commission

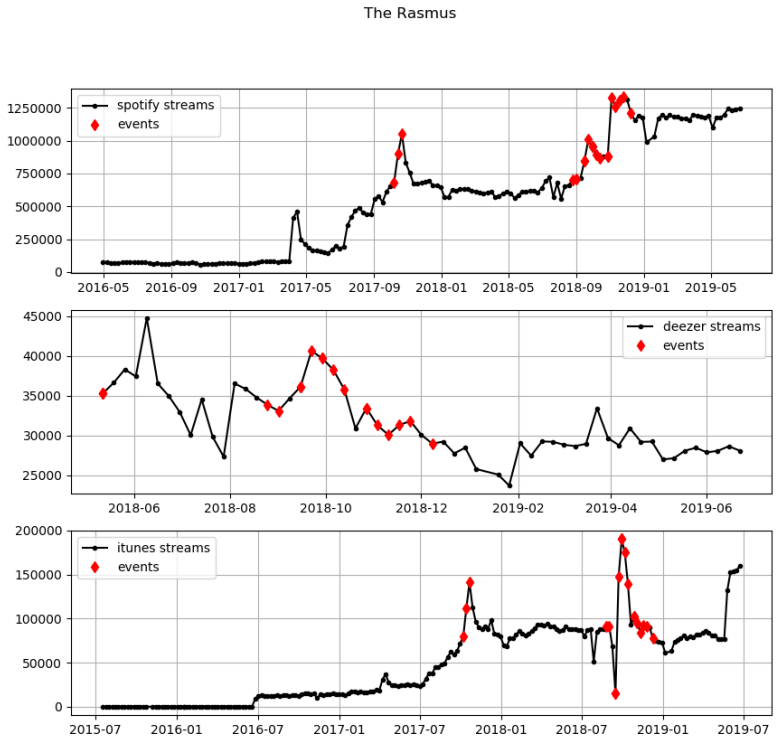


Figure 62 Spotify, Deezer and iTunes streams timelines and event dates for The Rasmus

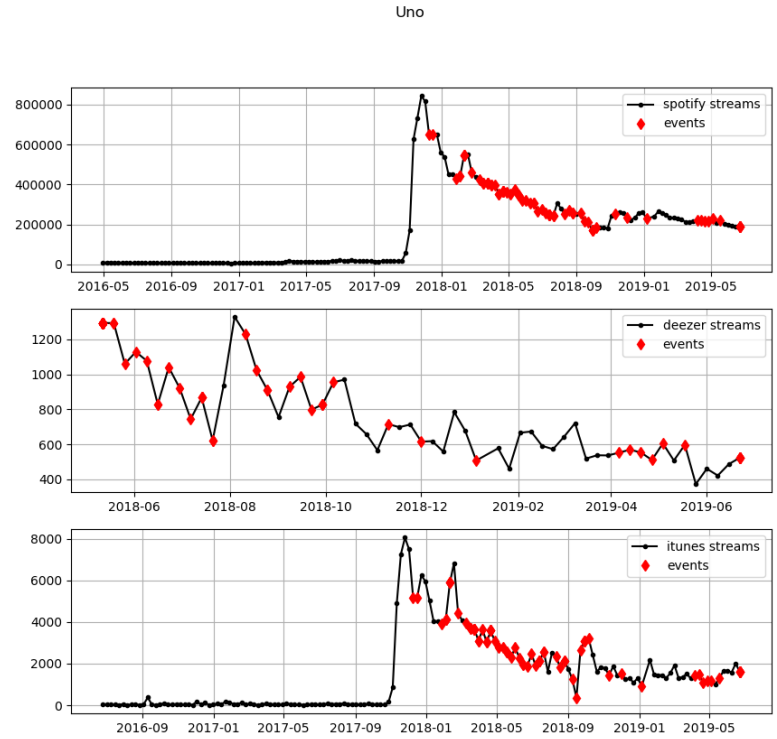


Figure 63 Spotify, Deezer and iTunes streams timelines and event dates for Uno

Funded by the European Commission