Played time and lyrics analysis with Spotify data

Author: Lorenzo Leskovar Ustarroz & Sergio-Yersi Villegas Pelegrín
Barcelona School of Economics, Universitat Pompeu Fabra,
*Carrer de Ramon Trias Fargas 25-27, 08005 Barcelona, Spain.**

Course: Statistical Modelling and Inference

Abstract: For the past years, Spotify's most relevant business goal has been to translate music into a quantitative and measurable language, in order to improve their algorithms and generate the highest possible revenue. Through the use of both Spotify and Genius API, we will extract all the relevant data regarding both of our last year's streaming histories and the lyrics of our songs, respectively, and perform different studies for each part of the course. First, just with one of our streaming histories, we will use LASSO and Bayesian regressions to predict the total amount of played time of each song and how each feature affects the output. Then, with both of our streaming histories put together, we will perform Structural Topic Modelling on the lyrics of our songs, in order to check the most common topics of our songs and check the popularity of the topics during the last 60 years. After concluding that the Bayesian Model Averaging model was the best choice due to all the used criteria, we found that listening time of the user is mostly driven by the artists and not so much from the intrinsic music features. For our Structural Topic Modelling implementation, we have obtained 4 major topics, from most to least frequent: "Love", "Nostalgia", "Journey" and "Life", together with its popularity evolution with time.

I. INTRODUCTION

Quantifying and parameterizing music has been a subject of interest for the past years due to the constantly improving quality and availability of data. Among all companies that belong to the music market, Spotify is the top-1 in contributing to the development of this field of study, where they have great harmony between their data engineering teams and their product and sales teams; by feeding off and onto each other, they drive the constant company's growth. Furthermore, Spotify allows its users to obtain their personal last year's data for them to dive into it, explore it and get to know the quantitative sense of music. Hence, we thought it could be both interesting and challenging to study it accordingly to each part of the project.

Beginning with the first part of the course (in section IV), since the private user data that Spotify provides to its users is completely unique and unrelated to others' data, our input will be a dataset with one of our streaming histories for the last year, together with all relevant musical features that Spotify provides through its API, as we will see in section III. Then, we will use and compare a LASSO regression and a Bayesian regression in order to both predict the played time of each track in the dataset (our target variable) and see how each of the musical features affects the value of it. Hence, we will interpret these two goals providing different reasoning for each one of them. Due to the probability distribution issues regarding our target variable, which we will later see in both sections III and IV, our model will compute predictions by regressing the logarithm of the played time on the musical features of our data.

Subsequently, for the second part of the course in section V, the input here will consist on both our streaming histories put together, with the aforementioned relevant musical features and most importantly, the lyrics of each song. Hence, we will implement topic modelling with the *Structural Topic Model* (STM) package into the lyrics, in order to see the most common topics of our songs and check the evolution of popularity, for the past 60 years, of each topic.

Finally, in section VI, we will briefly sum up our results, provide the most relevant conclusions of our study and comment further expansions that we would implement in the future.

II. RELATED WORK

Thanks to the size and scale of Spotify's data warehouses, which allows the company to improve their systems and algorithms in order to generate more revenue for themselves and their stakeholders, its study has been a matter of interest for many different researchers and individuals. After doing some research, we have observed several popular approaches to the problem of music success prediction, where we remark three general strategies:

- The first one uses social network data to assess current public perception and extrapolate how successful a song or album will be. [1], [2]
- The second one, in which our work could fit in, relies on acoustic information of previously successful

*Electronic address: sergioyersi.villegas@bse.eu\\\lorenzo.leskovar@bse.eu

songs to predict the success of other songs. [3], [4], [5]

- Finally, the third one uses past data from charts to predict whether a song will be featured in that same chart in the future. [6], [7]

It is worth noting that the played time of songs has not been subject of interest in the most relevant works we have found on this field. Hence, this project provides a study on music from a different perspective, in terms of the target output to obtain; the reason we said our study could correctly fit the second aforementioned category of related work is because of using musical features as a way to predict an outcome. Furthermore, the uniqueness of our private data makes this project a very specific study in comparison to other works. However, because of this, our data-set is not as extensive as the ones used in the seen related work.

In relation to the Unsupervised Learning part, through our research we found two papers that explore this area. On one hand, [8] applies latent semantic analysis (LSA) in order to detect artist similarity. On the other hand, [9] applies Pachinko allocation (PAM), a natural language processing algorithm which models correlations between topics with the objective of finding the ground truth of the topic each song has and, by using human-annotated data, allows to measure the precision of the results. In comparison to this work, our project is lacking data, since our data-set was taken from our Spotify libraries. However, our objective is different, since we use structured topic modelling (STM) to relate topics found in the corpus to the year of the songs.

III. DATASET

The dataset used for this project was obtained through the *Spotify API* and the *Genius API*. Firstly, by requesting both of our user's private streaming history of the last year *(https://support.spotify.com/us/article /data-rights-and-privacy-settings/)*, we obtained:

- Title, artist, date of streaming and played time for each song.

Then, thanks to the availability of Spotify's data through its API *(https://developer.spotify.com/ documentation/ web-api/)*, we obtained the following features regarding either the song or the artist:

- Album, release date, explicit, music markets, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration and time signature, which are musical attributes of each song.
- Popularity and followers corresponding to each artist.

Finally, using the *Genius API (https://docs.genius.com/)*, we obtained the lyrics of each song appearing in our streaming history. The final shapes of our DataFrames for each part of the project, as we already stated in section I, are: 1168x24 and 1747x26, respectively. The whole detailed codes to extract the data from the APIs and clean them are attached as a separate *.ipynb* file (see Appendices).

With our complete data, we performed several preprocessing steps after analyzing it:

1. One-hot-encoding for the categorical features of our DataFrame, which we stored as *factor* variables so that, when computing the Bayesian approach regarding the first part of the course, these dummy variables were not affected by the automatic scaling performed by the built-in R functions.
2. Dimensionality augmentation by adding interaction features between all numerical features of songs. Hence, the final number of features is 392.
3. Cleaning of the lyrics that will be used for the second part of the course. More particularly: putting all words to lowercase and removing a repeated, non-relevant text that came from the *Genius API*; punctuation, symbols and numbers removal; stopwords removal (for meaningless words that may be frequently repeated); infrequent terms removal; filtering of lyrics with a number of words between 20 and 600 and finally, just using the ones that were in English (since translation from other languages may not be accurate).

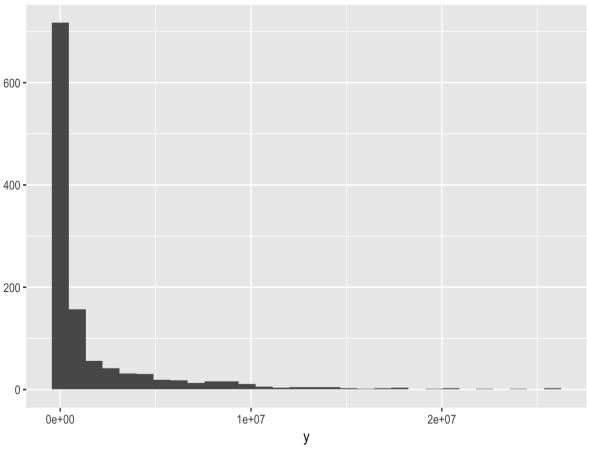
Finally, before moving on to the substance of the project, we analyzed our target variable for the first part of the course. In our case, it was the played time of each song in milliseconds (ms), which obviously always took a non-negative value. In the upcoming section IV, we will discuss the consequences of this distribution related to our model and then, see the relevant mathematical notation in order to accurately describe our study.

IV. REGRESSION

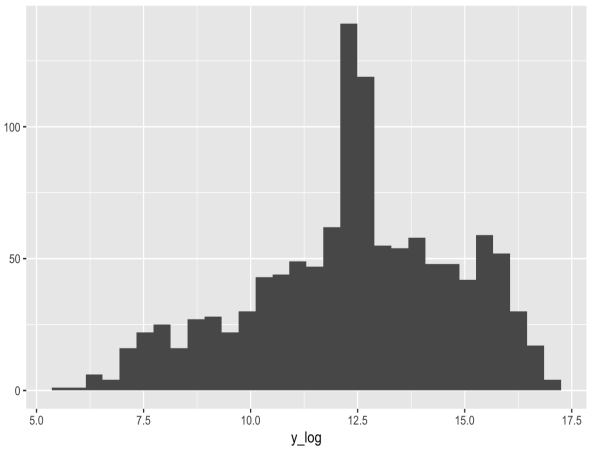
For the first part of the project, we are going to be testing different regression models for predicting the users' playing time of a song depending on the features specified before, which relate to the song, the album or the artist of each entry.

A. Methods

After taking a look at its distribution in Figure 1a, we computed and graphed its logarithm in Figure 1b and noticed an approximated normal distribution.



(a) Y



(b) Log Y

FIG. 1: Comparison between the target variable distribution and its logarithm.

Therefore, we could initially think about describing our dependent variable with a *log-normal distribution*, with the following probability density function (PDF):

$$f_X(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp -\frac{(lnx - \mu)^2}{2\sigma^2} \tag{1}$$

However, this probability distribution does not belong to the exponential family and therefore, we found it to be more convenient to define our dependent variable as $\tilde{y} \equiv \log(y)$, so its PDF will follow a normal distribution as stated below:

$$f_{\tilde{Y}}(\tilde{y}) = \frac{1}{\sigma\sqrt{2\pi}} \exp -\frac{(\tilde{y} - \mu)^2}{2\sigma^2} \tag{2}$$

This will have an impact in the way we interpret our coefficients, by doing a small transformation we can get an intuitive approach for reading them:

$$\tilde{\beta} = (exp\{\beta\} - 1) * 100 \tag{3}$$

Now, $\tilde{\beta}_j$ gives the percent increase or decrease in the dependent variable y for every one-unit increase in our independent variable x_j. Also, for obtaining point estimates of y we can also take the exponential of our predicted output to transform it. So this should not be a problem either.

Now, with the well-defined distribution of our dependent variable, we can begin to implement our models.

1. *LASSO Regression*

In our first approach, we are going to minimize a log-loss function of $\tilde{y}$ and add an L_1 penalty (LASSO). The formulation for this penalized regression model is as follows:

$$\min_{\beta} -\log p\left(\tilde{y}|\beta\right) + \lambda\|\beta\|_1 \tag{4}$$

$$\min_{\beta} \frac{1}{2N} \sum_{i=1}^{N} \left(\tilde{y}_i - x_i^T\beta\right)^2 + \lambda \sum_{j=1}^{d} \|\beta\|_j \tag{5}$$

Equation 5 is the one that the *glmnet* package in R will fit when specifying the α parameter as 1, since it originally fits an elastic-net model weighted by α, which combines both the L_1 and L_2 norms. The function minimizes the expression and obtains estimates for the betas by using a coordinate descent method.

Now, focusing on the value of λ, it will be set in two different ways. The first one, by a 10-fold cross validation which we perform using the *cv.glmnet()* function; in the results, it considered 100 different values for λ, so we will be choosing the one that minimizes the mean squared error (MSE):

$$MSE = E\left[(\tilde{y} - \hat{y})^2\right] \tag{6}$$

The second one, is through the Bayesian Information Criteria (BIC):

$$BIC = -2\log p(\tilde{y}|\beta) + \log n\|\beta\|_0 \tag{7}$$

Here, we fit the model with different values for λ (this is done automatically by the *glmnet()* function), we calculate the BIC of the model for each value and then, we choose the one with the lowest score.

To evaluate the models, we compute the R^2 coefficient as:

$$R^2 = Cor(\tilde{y}, \hat{y})^2 \tag{8}$$

Where $\hat{y}$ are the predictions obtained through a 10-fold cross validation. We will also include the MSE and the BIC, which were specified above, and the number of variables chosen by each model; that is, the ones with a non-zero coefficient. In addition we will include these results for the MLE estimate (obtained thorugh the *lm()* function) without penalization.

Model	R^2	MSE	BIC	$\lvert\beta_0\rvert \neq 0$
MLE	0.2909	4.4435	7825.3496	392
LASSO-CV	0.4228	3.2853	5756.3271	171
LASSO-BIC	0.38	3.7016	4998.6683	33

TABLE I: Results comparison between MLE and LASSO models.

We can notice that the *MLE* has the worst scores among the three models and that the *lasso-cv* produces higher prediction metrics than the *lasso-bic* but using almost 7 times more features and giving a worst BIC score (which is expected considering that *lasso-bic* chooses the value of lambda for it). Here, we can enter a discussion in which is the real purpose of our model, is it to predict or to do model selection? If we would like to predict then, by the given scores, we would use the *lasso-cv* model. But if we aim to discover the true model of our problem, then the *lasso-bic* could be a more appropriate choice.

2. Bayesian Model Averaging

The second approach we took is through Bayesian Model Selection (BMS) and Averaging (BMA). Here, we used the *mombf* package to fit our model. Let us start by doing a recap on BMS and explaining how we set our model.

In the BMS, we take all the features of our data matrix and we calculate Posterior marginal inclusion probability of them. To represent this, we define γ_j as a binary coefficient which represents if the feature j is included in the model $\gamma_j = 1$ or not $\gamma_j = 0$. Thus, the Posterior Marginal Inclusion Probability is noted as: $P(\gamma_j = 1|y)$, which can be derived using the bayes formula:

$$p(\gamma|y) = \frac{p(y|\gamma)p(\gamma)}{p(y)} \propto p(y|\gamma)p(\gamma) \tag{9}$$

- $P(\gamma)$ is the model prior probabilities which in our model are set as a Beta-Binomial(1,1) which is a default choice that sets a prior on the model size.

$$p(\gamma) = \frac{1}{d+1}\frac{1}{\binom{d}{|\gamma_j|_0}} \tag{10}$$

Once the model prior probabilities are set, we also define a prior on the chosen β-coefficients, in our model we set β with a zellner's prior:

$$p(\beta_\gamma|\gamma) \sim \mathcal{N}\left(\boldsymbol{\beta_\gamma}; \mathbf{0}, \rho g n \left(X_\gamma^T X_\gamma\right)^{-1}\right) \tag{11}$$

- **g** is a prior dispersion parameter that, because we have a Zellner's prior, we set to 1 so as to obtain a unit information prior.

- $p(y|\gamma)$ is the marginal likelihood and can be expressed as:

$$p(y|\gamma) = \int p(y|\beta_\gamma, \rho)p(\beta_\gamma, \rho)\, d\beta_\gamma\, d\rho \tag{12}$$

- We also set prior on ρ as an Inverse-Gamma distribution which is a common choice for Gaussian Bayesian regressions with parameters α and λ equal to 0.01 so that the prior is minimally informative.

$$p(\rho) \sim \mathcal{IG}(\rho; 0.01/2,\ 0.01/2) \tag{13}$$

The *modelSelection()* function calculates the marginal posterior inclusion probabilities $P(\gamma|y)$ and also the Bayesian Model Averaging posteriors given by:

$$p(\beta|y) = \sum p(\beta, \gamma|y) = \sum p(\beta|\gamma, y)p(\gamma|y) \tag{14}$$

This corresponds to the weighted average posterior distribution of the coefficients given the data over the marginal posterior inclusion probabilities.

In order to calculate both, $p(\gamma|y)$ and $p(\beta|y)$ the function uses gibbs sampling which is a variation of the Metropolis Hastings algorithm for Markov Chain Monte Carlo (MCMC).

Our model with the highest posterior probability (4.9202%) includes only 18 coefficients. These can be seen in table II.

However for our model we are going to keep the BMA estimates. Because the BMA averages with the marginal posterior inclusion probability, it may happen that not many coefficients will be exactly zero but close to it, unlike lasso, which draws coefficients to exactly 0. For example we can see how our BMA coefficients compare to our *lasso-bic* model:

loudness	artist-Ariel Posen	artist-Conociendo Rusia	artist-Eric Clapton	artist-Gustavo Cerati	artist-John Mayer
artist-Luis Alberto Spinetta	artist-Luis Miguel	artist-Pedro Aznar	artist-Pink Floyd	artist-Scorpion	artist-Soda Stereo
artist-The Black Keys	artist-Tom Misch	artist-Trent Reznor and Atticus Ross	release_date	popularity : duration_ms	loudness : liveness

TABLE II: Selected variables.

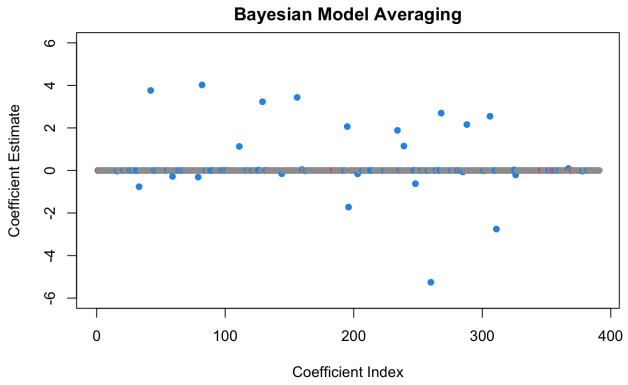


FIG. 2: BMA Coefficients.

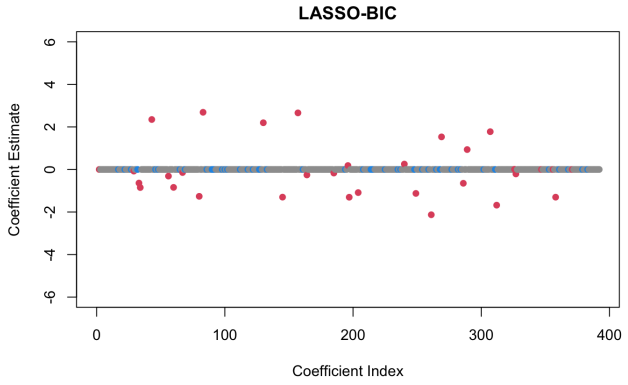


FIG. 3: LASSO-BIC Coefficients.

In the above graphs, all the grey points are coefficients set to 0, all the blue points are the coefficients chose by *BMA* and all the red points, the coefficients chose by *lasso-bic*. When one of the graphs shows dots from the other models color, it means that those variables were chosen by the other model but not by that one.

As we can see, the *BMA* has many coefficients close to zero but not strictly zero, in comparison to the *lasso-bic*. In fact, *BMA* chooses 72 coefficients, while *lasso-bic* chooses 33.

3. Model Evaluation

In order to evaluate the model and compare it with the Lasso results we obtain predictions through a 10-fold cross validation and calculate the same metrics as before: R^2, MSE, BIC.

Model	R^2	**MSE**	**BIC**	$\lvert\beta_0\rvert \neq 0$
MLE	0.2909	4.4435	7825.3496	392
LASSO-CV	0.4228	3.2853	5756.3271	171
LASSO-BIC	0.38	3.7016	4998.6683	33
BMA	0.4169	3.3035	5635.6452	72

TABLE III: Results comparison table.

As we can see, the *BMA* model attains a greater R^2 and MSE score than the *lasso-bic*, and it achieves a greater BIC score than the *lasso-cv* model with almost a 100 features less. Because the BMA seems to have a nice balance between prediction results (from the MSE and R^2) and model selection (BIC), we are going to choose it as our model to study the results and draw conclusions.

B. Results

When looking at the estimates of our model we can see that in the top 20 coefficients with the greater effect (either positive or negative) 19 of them are artists. In fact, from the 72 non-zero variables chosen by BMA, only 10 are not artists. This can be seen in table IV.

All of them have a low effect and only two of them have a high (> 0.5) marginal inclusion posterior probability. Moreover, many of them include 0 in their confidence interval, which indicates that they may not have any effect at all.

What we can conclude is that almost all of the effect captured by our model is driven by the artists and not by intrinsic music features, like the danceability or the instrumentalness, among others. This may relate to several reasons, for example that the user is drawn to his favorite artists only and he does not care for the music itself, or it may be that some of this audio features measured by Spotify are not measured right and they cannot capture the true essence of the music.

Variable	estimate	2.5%	97.5%	margpp
loudness	-0.031	-0.095	0.000	0.3248
liveness	-0.003	0.000	0.000	0.0727
release date	0.026	0.018	0.032	0.9950
explicit True	-0.211	-1.280	0.000	0.2527
danceability : liveness	-0.001	0.000	0.000	0.0136
energy : speechiness	-0.001	0.000	0.000	0.0101
energy : liveness	-0.001	0.000	0.000	0.0081
loudness : liveness	0.093	0.000	0.145	0.6086
acousticness : liveness	-0.032	0.000	0.000	0.0573
instrumentalness : liveness	-0.001	0.000	0.000	0.0039

TABLE IV: Variable Estimates.

V. UNSUPERVISED LEARNING

Now, for the second part of the project, we are going to implement Structural Topic Modelling on the lyrics from both of our streaming histories through the *STM* package of R. This type of topic modelling, which is an extension of existing topic models, allows the use of several covariates' information, which can improve inference, qualitative interpretability and also, affect topic prevalence, topic content or both. The central idea is to specify the priors as generalized linear models, through which we can condition on arbitrary observed data. This allows us to directly estimate the quantities of interest in applied problems. The model allows users to incorporate the specific structure of their corpus without developing new models from scratch.

For the purpose of this project, we will be analyzing the lyrics of our songs in order to see the most common topics that describe our music and then, show how the results vary according to an specific variable.

A. Methods

We will firstly describe the notation of the STM model, so that we can then explain the ordered process in which all variables are computed and finally, completely display the mathematical formulation of it. Here, we will describe the STM in a general way and then, when we display our results, specify the corresponding variables of interest for our project.

1. Variables

For: n being one word, d being one document, k being one topic, y being one covariate and v one *unique* word, we define the following variables:

- $\omega_{d,n} \equiv$ word n in document d.
- $\beta^k_{d,v} \equiv$ word probabilities for a single topic k in document d.
- $\theta_d \equiv 1xK$ vector that corresponds to how much of that document d is made up of a given topic k.
- $X_d \equiv$ metadata vector for a specific document d.
- $\gamma_k \equiv$ column of matrix τ
- $\tau_v \equiv$ matrix of weights.
- $z \equiv$ topics per word.
- $m \equiv 1xv$ vector that represents the initial word probabilities.
- $\kappa_v \equiv$ probability deviations matrix that includes terms depending on each document and each topic.

2. Process

1. To begin with, for topic prevalence, our goal here is to get θ_d. In order to do that, we need to go from X_d to θ_d by multiplying X_d and τ (which will act as the mean for the logistic normal distribution that ultimately computes θ). During the inference steps, we will learn the actual values of the variance of γ_k, where the Gamma prior on these parameters will be introduced in order to deal with the sparsity of our data, since there may be part of the metadata that is not relevant to the topics.

2. Now, once we have obtained θ_d, it is the same as the the Latent Dirichlet Allocation (LDA) model, where the next main objective is to obtain β^k_d. First, we obtain z, for every n word in document d. Now, in order to obtain β^k_d, we will add the κ deviations to m for each topic k, each document d and, finally, for the topic-metadata interactions of the model (therefore improving the original LDA model). When adding all the terms together, we will finally obtain β^k_d.

3. With θ_d and β^k_d already obtained, we have completed our STM, where the summarized graphical schema looks like this:

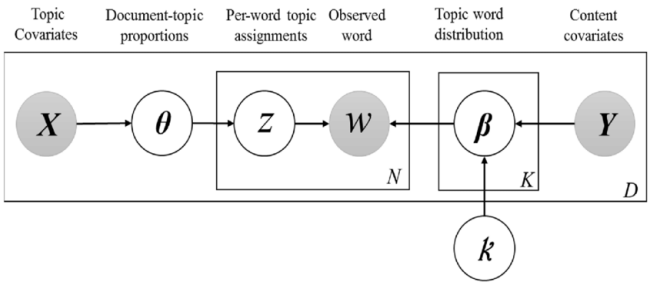


FIG. 4: STM schema where, in our case, Y=X as, in our case, the metadata used to generate topic contents is the same as the metadata used to generate topic prevalence, respectively.

3. *Mathematical model*

Hence, the mathematical formulation of the model in the *STM* package is as follows:

Topic prevalence:

$$\mu_{d,k} = X_{d\gamma_k} \tag{15}$$

$$\gamma_k \sim N(0, \sigma_k^2) \tag{16}$$

$$\sigma_k^2 \sim Gamma(s^\gamma, r^\gamma) \tag{17}$$

Language model:

$$\theta_d \sim LogNormal(\mu_d, \sigma) \tag{18}$$

$$z_{d,n} \sim Mult(\theta_d) \tag{19}$$

$$\omega_{d,n} \sim Mult(\beta_d^{k=z_{d,n}}) \tag{20}$$

Topic content:

$$\beta_{d,v}^k \sim exp(m_v + \kappa_v^{\cdot,k} + \kappa_v^{y,\cdot} + \kappa_v^{y,k}) \tag{21}$$

$$\kappa_v^{y,k} \sim Laplace(0, \tau_v^{y,k}) \tag{22}$$

$$\tau_v^{y,k} \sim Gamma(s^k, r^k) \tag{23}$$

B. Results

Now, let's see what we obtained in the analysis of our songs' lyrics. First, after all the preprocessing steps mentioned in section III, we can begin by displaying the following obtained word-cloud with the most frequent words appearing in our lyrics.

FIG. 5: Word cloud coming from our songs' lyrics.

As it could be expected, the word that excels in proportion appearance over the rest is "Love", therefore reflecting the major impact this word has in artists when composing their songs.

Then, when implementing the previously explained STM, we have chosen the release date of the songs as the prevalence covariate of our model, so we can later evaluate the effect of this covariate on the topics that we obtained independently of time. Therefore, it is worth noting that this is not Dynamic Topic Modelling, since the generated topic proportions and word frequencies of our model do not depend on time; in our model, the content of the topics remains constant with time. What we will be evaluating is how each topic is more or less popular throughout the years. After computing the model, we can display the following results:

- The obtained topics with their most relevant, associated words, which we display below:

FIG. 6: Word cloud for the first major topic, defined as "Love".

FIG. 7: Word cloud for the second major topic, defined as "Nostalgia".

FIG. 8: Word cloud for the third major topic, defined as "Journey".

FIG. 9: Word cloud for the fourth major topic, defined as "Life".

- The variation of popularity of each topic throughout the years (as it is our prevalence covariate), in order to see how each topic has been more used by song-writers during the last 60 years. This issue will be furtherly commented in sectionVI, as it is one of the facets that could be expanded in further projects related to musical topic modelling.

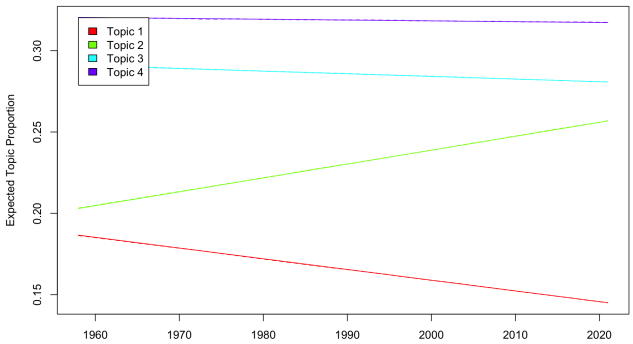


FIG. 10: Topic popularity variation for the past 60 years.

VI. DISCUSSION

Finally, let's sum up the obtained results for each part of the course and see how we could expand this project in the future:

1. Regression: by implementing lasso and BMA, we compared their results and decided to use BMA for the analysis of our data. We observed very little effect by other variables than the artist and thus we concluded that the intrinsic music features are not capturing the effect of our user's listening time. For future work, instead of taking the logarithm of the y, we could model the target variable as a Poisson distribution by taking the number of times

a song has been listened to in the year. In addition, the amount of data collected for this project was of just 1 year and for 1 user, and then is completely biased. Adding more data from different users and in a bigger time frame should improve our model.

2. Unsupervised learning: by implementing an STM to our songs' lyrics, we obtained 4 major topics of our songs, from most to least frequent, which we defined as "Love", "Nostalgia", "Journey" and "Life" after checking their word content. Then, we displayed how each of these topics has been more or less trending for the past 60 years. As we said, we have not done Dynamic Topic Modelling, since the content of each topic does not vary with time. Therefore, a way to expand this project with more time and more knowledge on the matter would be to take into consideration the variation of each topic content with time; that is, implement Dynamic Topic Modelling and see, for example, how the concept of "Love" in music evolves throughout the years by checking how its word content varies. Furthermore, without the current computational limitations, we would use a much more heavy data, so that we could try and see a general pattern of our results, and not just limited to our personal streaming histories.

[1] V. Dhar and E. A. Chang, "Does chatter matter? the impact of user-generated content on music sales," Journal of Interactive Marketing, vol. 23, no. 4, pp. 300 – 307, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1094996809000723

[2] B. Shulman, A. Sharma, and D. Cosley, "Predictability of popularity: Gaps between prediction and understanding," in Tenth International AAAI Conference on Web and Social Media, 2016, pp. 348–357.

[3] Sciandra, M., "A Model Based Approach to Spotify Data Analysis: A Beta GLMM", & Spera, I. 2020, SSRN Electronic Journal

[4] J. Lee and J. Lee, "Music popularity: Metrics, characteristics, and audio- based prediction," IEEE Transactions on Multimedia, vol. 20, no. 11, pp. 3173–3182, Nov 2018.

[5] M. Interiano, K. Kazemi, L. Wang, J. Yang, Z. Yu, and N. L. Komarova, "Musical trends and predictability of success in contemporary songs in and out of the top charts," Royal Society Open Science, vol. 5, no. 5, p. 171274, 2018. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsos.171274

[6] D. Herremans, D. Martens, and K. Sorensen, "Dance hit song prediction," Journal of New Music Research, vol. 43, no. 3, pp. 291–302, 2014. [Online]. Available: https://doi.org/10.1080/09298215. 2014.881888

[7] M. Reiman and P. Ornell, "Predicting hit songs with machine learning", 2018.

[8] Logan, B., et al., "Semantic Analysis of Song Lyrics", HPL-2004-66, 2004.

[9] Lukic, A., "A Comparison of Topic Modeling Approaches for a Comprehensive Corpus of Song Lyrics", Carnegie Mellon University, Pittsburgh PA 15213, 2018.

Appendices

Finally, we will detail the content of the attached coding files of the task, which include both Python Notebook and RMarkdown files:

- *Tidy data.ipynb*, with all the process used to get our streaming histories and clean it.
- *Final_Notebook.Rmd*, with the commented R code that has been used to obtain our results.
- *Final_Notebook.nb.html*, with the knitted version of the R code.
- *routines_final_project.R*, with the functions used to run the CV models for the first part.