LUISS Guido Carli

LIBERA UNIVERSITÀ INTERNAZIONALE DEGLI STUDI SOCIALI

Department of Business and Management

Master's Degree in Data Science and Management

Machine Learning and Music: Predicting the level of energy conveyed by a soundtrack

THESIS SUPERVISOR *CANDIDATE*

Giuseppe F. Italiano *Roberto Colangelo*

ACADEMIC YEAR 2022/202

Machine Learning and Music: Predicting the level of energy conveyed by a soundtrack

by

Roberto Colangelo

Thesis Supervisor: Professor Giuseppe F. Italiano

Abstract

Music is a universal language that can evoke powerful emotions and has been an integral part of human culture for centuries. With the advent of digital music streaming services such as Spotify, vast amounts of data on music are now available, including audio features and user listening behavior. One crucial audio feature provided by Spotify is the energy level of a track, which can influence listeners' emotional responses and behavior. However, this feature has not been fully utilized in recommendation systems, which are commonly used to suggest new songs to users based on their preferences. To address this gap, this thesis aims to develop an AI algorithm that can predict the energy level of a track using only Spotify metadata and incorporate energy levels into collaborative filtering. By using regression machine learning techniques and comparing traditional and deep learning algorithms, a 0.067 mean absolute error on the test set has been achieved. Additionally, this research contributes to the literature by developing a web application system that displays predicted and actual energy values for each track, as well as recommended tracks based on subsequent affinity systems. In conclusion, this study shows the potential for machine learning to enhance music recommendation systems and improve the personalized listening experience for users.

"To you, who are always watching over me from above, giving me the strength to persevere in the face of adversity."

Acknowledgments

My appreciation for Professor Giuseppe F. Italiano's essential patience and comments is beyond words. He inspired me, and I used that inspiration to drive myself to achieve better and come up with this dissertation topic.

Without the help of my devoted parents, Carmelo and Mariateresa, who always believed in me and worked hard to raise me and allow me to pursue my education, I also would not have been able to complete my journey.

In addition, I want to thank my sister Jasmine and brother Luigi who assisted me along my journey by making me laugh even though they were far away.

Many thanks also to Johnny, my dog, who has always been there to lift my spirits and support me emotionally when I needed it most.

Finally, I must acknowledge Camilla, my girlfriend, who is my biggest supporter. Her confidence in me has helped me stay positive and motivated during the hard times and during the last year of this journey.

My biggest thanks go to my grandmother Luisa, who passed away many years ago. . She was the one who taught me how to study and motivated me to accomplish my goals by giving it my all regardless of the circumstances.

Contents

List of Figures and Tables

1. Introduction

Music is a language without words, a means of expression that can communicate directly with our feelings. Bypassing our rational mind, it can evoke a feeling of nostalgia, a sense of longing, a rush of excitement or a sense of peace and contentment. Since ancient times, music has been an essential part of human culture and has been used for various purposes, such as entertainment, relaxation, and communication. In recent years, the advent of digital music streaming services such as Spotify has led to the availability of large music datasets. These datasets contain a wealth of information about music, including audio features, information about artists, and user listening behavior. One of the audio features provided by Spotify is the energy level of a music track, which represents the perceived intensity and activity of the track. The energy level is a crucial factor in determining the mood and emotional impact of a song and can influence users' listening behavior.

Nowadays collaborative filtering is a popular recommendation system technique used by many music streaming services such as Spotify, to suggest new songs to their users based on their listening history and preferences. However, one limitation of collaborative filtering is that it may not always consider the energy level of songs as an important factor in generating recommendations and this is an unexploited element that might allow users to get to know more music and expand their music tastes and knowledge. **Given such absence in the recommendation system sector, the aim of this thesis will be that of developing an AI algorithm capable of predicting the energy level of track, improving the accuracy of already developed algorithms by using only the metadata provided by Spotify, and providing a new method for such systems consisting in incorporating the concept of energy levels into collaborative filtering.** The objective will be that of improving music recommendation systems by identifying songs with comparable energy levels and using them as the basis for generating recommendations. For instance, if a user often listens to high-energy songs, the system can suggest other tracks with similar energy levels. This can be accomplished by utilizing regression Machine Learning techniques, which involve analyzing the association between audio metrics such as tempo, loudness, and acousticness and a target variable that wants to be estimated, the energy level of a song. By integrating energy levels into collaborative filtering, the user experience can be enhanced by providing more personalized and relevant recommendations. However, to achieve this, an accurate machine learning regression algorithm capable of predicting the energy level of a music track must be implemented.

Over the past years, several attempts have been made to predict the energy level of a music track. These studies, conducted by Han et al. (2018), Kim et al. (2019), and Zhang et al. (2020), involve regression or classification models that use different types of data such as album cover images, lyrics, and audio features/metrics. Despite previous attempts to predict the energy level of music tracks using different types of data such as album cover images, lyrics, and audio features/metrics, this thesis will exclusively use audio features and metrics from a dataset of 114,000 songs on Spotify. The main technical goal will be that of identifying the most effective algorithm for the purpose of developing a web application that can predict the energy level of a track when inputting its name and recommend tracks with similar energy levels.

To this end, two types of models will be considered: regression models based on traditional machine learning algorithms and deep learning algorithms. In order to build the best regression model for the prediction several steps were followed including:

- Libraries import
- Data Cleaning
- Data Exploration /Visualization
- Data Preprocessing
- Feature Selection
- Modeling and hyperparameters Tuning
- Models comparison
- Model deployment

This research has significant implications for the development of music analysis and recommendation systems that can take into account the emotional impact of music on listeners. Additionally, differently from previous studies and researches which focused on the same task but by employing the Spotify audio metric in conjunction with other sources such as lyrics and images, and achieved as best results 80.3% accuracy with classification algorithms or 0.11 mean absolute error (MAE) with regression algorithms on the test set, the approach of this thesis aims to revolutionize the application of machine learning to music and its features by focusing solely on metadata to develop the regression models. Furthermore, the research provides a further secondary contribution to the literature since it involves the development of a web application system that displays predicted and actual values of energy for each track, as well as recommended tracks based on a subsequent affinity system, to provide an idea of the potential exploitation of such an algorithm in the music industry.

2. Literature Review

In the last decade Spotify has emerged as a prominent music streaming platform, providing a comprehensive dataset for research on music information retrieval (MIR). This dataset has been the subject of numerous studies, aimed at extracting crucial insights and developing innovative technological tools that could revolutionize user experience and transform the music industry. In recent years, machine learning has gained significant traction as an approach to music analysis and researchers have employed a diverse range of machine learning techniques, such as neural networks, decision trees, and regression models, to investigate different facets of music. Notably, machine learning algorithms have been extensively used to predict audio features associated with the Spotify dataset and its music tracks but only a few focused on predicting the energy level of tracks.

In the past, several approaches have been taken to predict the popularity of music using the Spotify dataset. One common method is to use audio features, such as loudness, tempo, and timbre. For example, a study by Pons et al. (2017) used various audio features, such as MFCC, spectral contrast, and chroma in the implementation of a support vector regression model which achieved a mean absolute error (MAE) of 0.24. Another study by Han et al. (2021) used a convolutional neural network (CNN) to predict the popularity of music by using audio features, such as Mel-spectrogram and constant-Q transform, and achieved a mean absolute error (MAE) of 0.12.

Another approach to predicting popularity has been to use metadata features, such as artist, release year, and genre. For instance, a study by McFee et al. (2015) used metadata features, such as genre and artist, to predict the popularity of music. In this case the authors used linear regression models and achieved a correlation coefficient of 0.58. Another study by Hu and Downie (2018) used a similar approach and used metadata features, such as artist, album, and release year, in the generation of a gradient boosting regression model and achieved an MAE of 0.276.

In addition to using audio and metadata features, some studies have used user behavior as a predictor of popularity. For example, a study by Bonnin et al. (2019) used user behavior, such as number of streams and skips, to predict the popularity of music. The authors used a random forest regression model and achieved an MAE of 0.167. Another study by Ngiam et al. (2020)

used a similar approach and used user behavior, such as play count and skip rate, to predict the popularity of music. The authors used a neural network model and achieved an MAE of 0.193.

Together with the prediction of popularity of a soundtrack, another aspect that has been explored and analyzed is the level of energy of a song, which represents the perceived intensity and activity of a track. As mentioned earlier, predicting the level of energy in music has numerous applications, such as music genre classification, playlist generation, and, above all, music recommendation. Music recommendation systems are typically based on collaborative filtering which is a topic that has been covered by many in the past few years. In their paper, Chen, C. W., & Chen, M. S. (2016) provide a review of collaborative filtering techniques for music recommendation systems which discusses the use of various parameters in collaborative filtering, including user-item ratings, user-item interactions, and user-item features such as music genre, artist, and album. The paper also discusses the challenges associated with incorporating music features into collaborative filtering, such as the high dimensionality and sparsity of music data. Zhang, X., Liu, W., & Ma, L. (2018) propose an improved collaborative filtering algorithm for music recommendation that incorporates user behavior and music content features. This research discusses the use of various parameters in collaborative filtering, including user-item ratings, user-item interactions, and music content features such as genre, artist, and album. The paper also discusses the importance of feature selection and dimensionality reduction techniques to improve the efficiency and effectiveness of collaborative filtering. Çakmak, E., & Yılmaz, O. T. (2019) discuss a hybrid approach that combines collaborative filtering with content-based recommendation techniques to provide more accurate and diverse music recommendations. The paper investigates on the use of various parameters in collaborative filtering, including user-item ratings, user-item interactions, and user-item features such as music genre, artist, and album. This work also explores the use of audio features and lyrics analysis, to supplement the collaborative filtering approach and improve the diversity of music recommendations.

Something that appears to be missing in the literature is surely the usage of energy level of a track as one of the parameters used for music recommendation systems, both as a unique or complementing parameter. Multiple strategies have been undertaken to predict the level of energy in music using the Spotify dataset. One common method has been to use audio features, such as loudness, tempo, and spectral features. For example, a study by Guo et al. (2018) used various audio features, such as spectral centroid, spectral rolloff, and zero-crossing rate, to predict the level of energy in music. The authors used a decision tree classifier and achieved an accuracy of 65.3%.

In their research Choi and Lee (2018) used a convolutional neural network (CNN) to predict the level of energy in music. The authors extracted several audio features, such as MFCC and spectral features, and trained the CNN on a subset of the Spotify dataset. The authors achieved an accuracy of 72.8%. In a study by Cho et al. (2017), a deep neural network was used to predict the energy level of a music track based on audio features provided by Spotify by classifying the energy level on a scale from 1 to 10. Another study by Han et al. (2018) used a combination of audio and textual features to predict the popularity of music tracks on a social media platform. The authors used a support vector regression model and achieved an accuracy of 79.2% in predicting the popularity of music tracks. The study highlights the potential of incorporating additional features beyond audio features for predicting the energy level of a music track.

In a similar study by Kim et al. (2019), a decision tree algorithm was used to show the seasonal influences on the music consumption by detecting a positive correlation between the music parameters valence as well as energy and the monthly temperature has been detected. The authors trained the model on a dataset of 47,000 tracks and achieved a prediction accuracy of 80.3%. The study suggests that decision tree algorithms can be an effective approach for predicting the energy level of music tracks. In their research, Zhang et al. (2020) used a hybrid approach combining audio features together with lyrics in a random forest regression model and achieved a MAE of 0.119, indicating the effectiveness of the approach.

To summarize, previous research indicates that machine learning techniques, such as neural networks, decision trees, and regression models, can effectively predict the energy level of music tracks by utilizing audio features in combination with other data sources. Nevertheless, the contribution of this study to the literature is to demonstrate that a novel approach using regression algorithms, with audio features and songs metadata provided by Spotify as the sole data sources, can result in even more accurate predictions without the need for external sources such as lyrics or images.

Additionally, the developed regression model has been deployed on a web application, allowing for the display of predicted energy level values. This application also incorporates a collaborative filtering recommendation system that suggests tracks with similar energy levels, thereby conveying consistent intensity of emotions. Such an algorithm provides an innovative approach to music recommendations system and literature, serving as an alternative system for companies in the music industry. By offering a platform for personalized music suggestions, this algorithm has the potential to enhance user experience, drive user engagement, and enable users to explore a broader range of music preferences. These benefits can ultimately contribute to the growth and success of music companies and the industry as a whole.

3. Data Collection and Columns Description

3.a Data Collection

Data collection is a crucial process that involves obtaining information from a variety of sources for a specific purpose. The data collected may be in the form of primary data, which is gathered directly from participants, or secondary data, which has already been collected by someone else. In the current study, secondary data was used, as the Spotify Tracks Dataset by Maharshipandya was downloaded in the form of a CSV file from a Kaggle repository. This dataset was originally obtained from the Spotify Web API in November 2022 and contains a comprehensive collection of music tracks and associated metadata. Specifically, the dataset comprises 114,000 songs, each of which includes 20 features, encompassing track, artist, and audio features. These features are categorized into 5 categorical and 15 numeric variables whose detailed descriptions can be found in the subsequent paragraph.

3.b Columns Description

One of the most important features provided by Spotify and the target variable of this analysis is the energy level of a music track, which is represented on a continuous scale ranging from 0 to 1. This score is indicative of the perceived intensity and activity level of the track and is a crucial factor in determining the mood and emotional impact of a song. Another noteworthy feature of the dataset is the popularity score, which ranges from 0 to 100 and is computed by Spotify based on various factors, including the number of listens. The categorical features in the dataset are limited, nominal and primarily describe the track_name, album_name, track_id, artists who created the track, and the track_genre. Other key features in the dataset include the track duration in milliseconds, a binary value indicating the presence of explicit content, the track key (which ranges from 0 to 11), the overall loudness of the track in decibels, and the tempo of the track in beats per minute. Moreover, the dataset includes scores for danceability, speechiness, acousticness, instrumentalness, liveness, mode, and valence, which are represented as continuous values ranging from 0 to 1 and reflect various aspects of musical structure and production.

Understanding the importance of each of these features and their contribution to the energy level of a track is essential for accurately predicting energy levels using machine learning algorithms and neural networks. A table containing a more detailed description of each variable that makes up the dataset is presented below.

Feature	**Description**
‘Track_id’	**The Spotify ID for the track**
‘artists’	**The artists' names who performed the track. If there is more than one artist, they are separated by a ;**
‘album_name’	**The album name in which the track appears**
‘track_name’	**Name of the track**
‘popularity’	**The popularity of a track is a value between 0 and 100, with 100 being the most popular. The popularity is calculated by algorithm and is based, in the most part, on the total number of plays the track has had and how recent those plays are. Generally speaking, songs that are being played a lot now will have a higher popularity than songs that were played a lot in the past. Duplicate tracks (e.g. the same track from a single and an album) are rated independently. Artist and album popularity is derived mathematically from track popularity.**

‘duration_ms’	**The track length in milliseconds**
‘explicit’	**Whether or not the track has explicit lyrics (true = yes it does; false = no it does not OR unknown)**
‘danceability’	**Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable**
‘energy’	**Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale**
‘key’	**The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, and so on. If no key was detected, the value is -1**
‘loudness’	**The overall loudness of a track in decibels (dB)**
‘mode’	**Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major**

	is represented by 1 and minor is 0
‘speechiness’	speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks
‘acousticness’	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic
‘instrumentalness’	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content
‘liveness’	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides

	strong likelihood that the track is live
‘valence’	**A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry)**
‘tempo’	**The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration**
‘time_signature’	**An estimated time signature. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures of 3/4, to 7/4.**
‘track_genre’	**The musical genre the track belongs to**

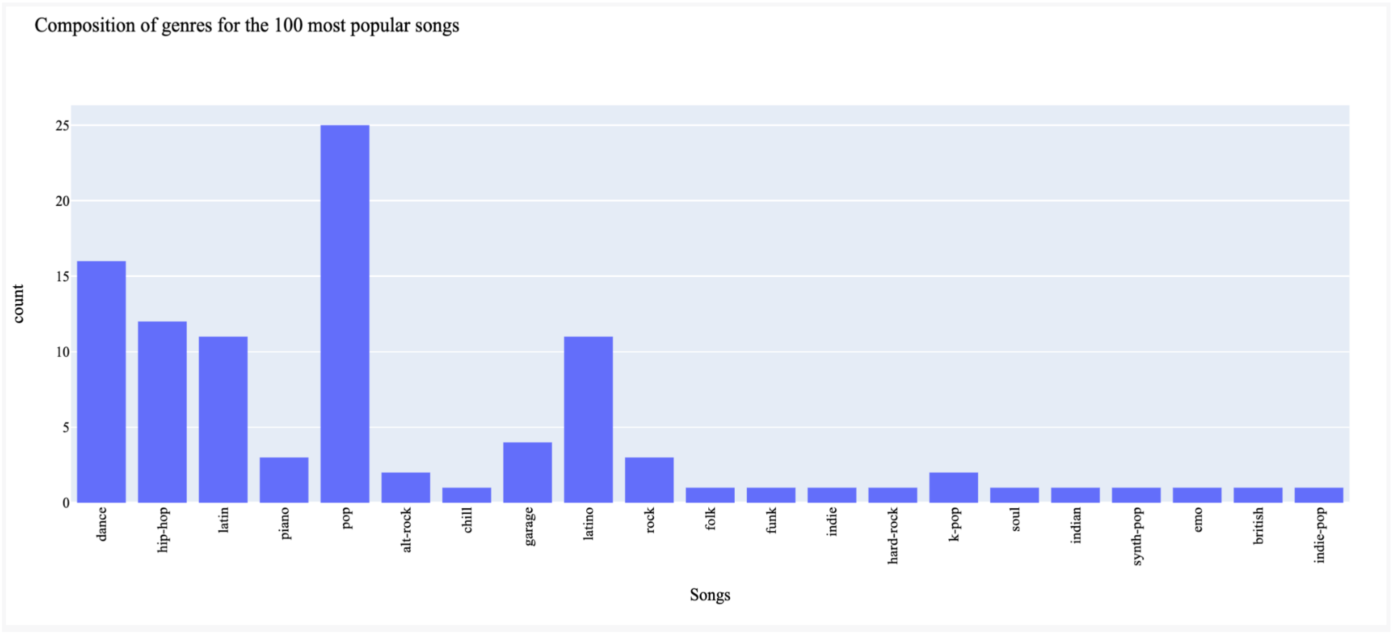
(Table 1: Dataset’s features Description)

4. Data Exploration and Visualization

The analysis of data collected for a specific purpose often requires a thorough exploration of its features, relationships, and patterns. To achieve this, data exploration and visualization techniques are often employed, such as summary statistics and graphical representations. In this context, the data was explored in order to better understand the relationship between its various features. By using summary statistics and various graphical techniques, the dataset's features and their interplay were examined, providing insights into the central tendency, variability, and distribution of the data, and highlighting any correlations or patterns that may exist. One of the key features of the dataset is the energy level of a music track, which is an essential factor in determining the mood and emotional impact of a song.

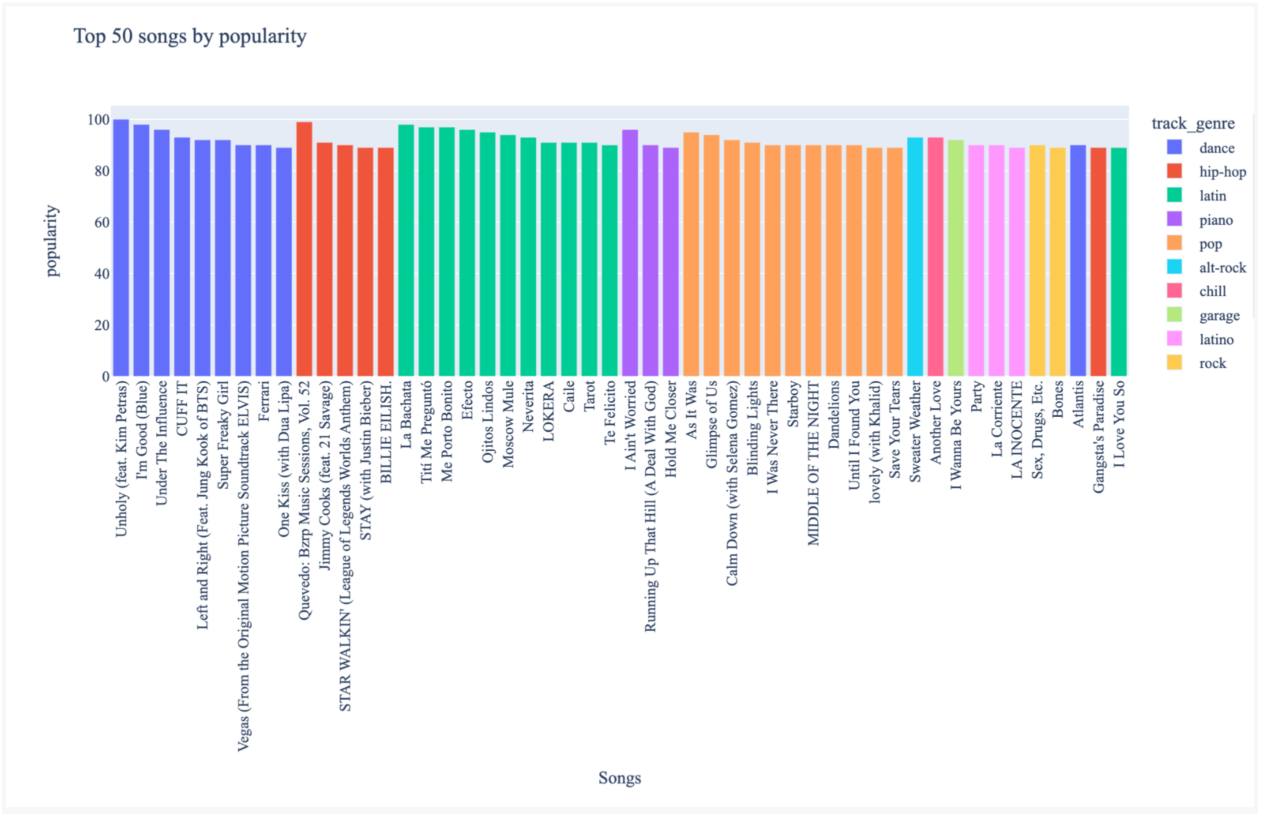
For such purpose scatterplots, histograms, and density plots were used to visualize the relationships between pairs of features and to identify any correlations or patterns that might exist, however, given the huge number of categories and subgroups of instances, in some cases a subselection of the dataset had to be done.

One of the first exploratory data visualizations conducted was a histogram that aimed at exploring the distribution of genres among the top 100 most popular songs. The results indicated that the pop genre was the most common among the most popular tracks, followed by the dance, hip-hop, and latin genres. This insight can provide valuable information for music industry professionals and researchers interested in understanding the current trends and preferences of music listeners.



(Figure 1: Histogram of composition of genres in the top 100 most popular songs)

Another visualization that was created to explore the relationship between music genres and popularity was a bar plot representing the 50 most popular songs, colored by their respective genres. This graph revealed that the most popular track in the dataset was "Unholy" by Sam Smith, which falls under the pop genre. The second and third most popular songs were "One Kiss" in the Dance genre and "La Bachata" in the Latin genre, respectively. These results provide additional information on which are the most popular music genres and were used to better understand the connection between the popularity of soundtracks, their music genres and the energy conveyed and perceived by listeners.



(Figure 2: barplot of the 50 most popular songs colored by genre)

To further understand these connections, additional exploratory data analysis was necessary. An association metric was required, and the selected method was correlation. Correlation is a statistical measure that describes the strength and direction of the linear relationship between two quantitative variables. The formula for correlation is:

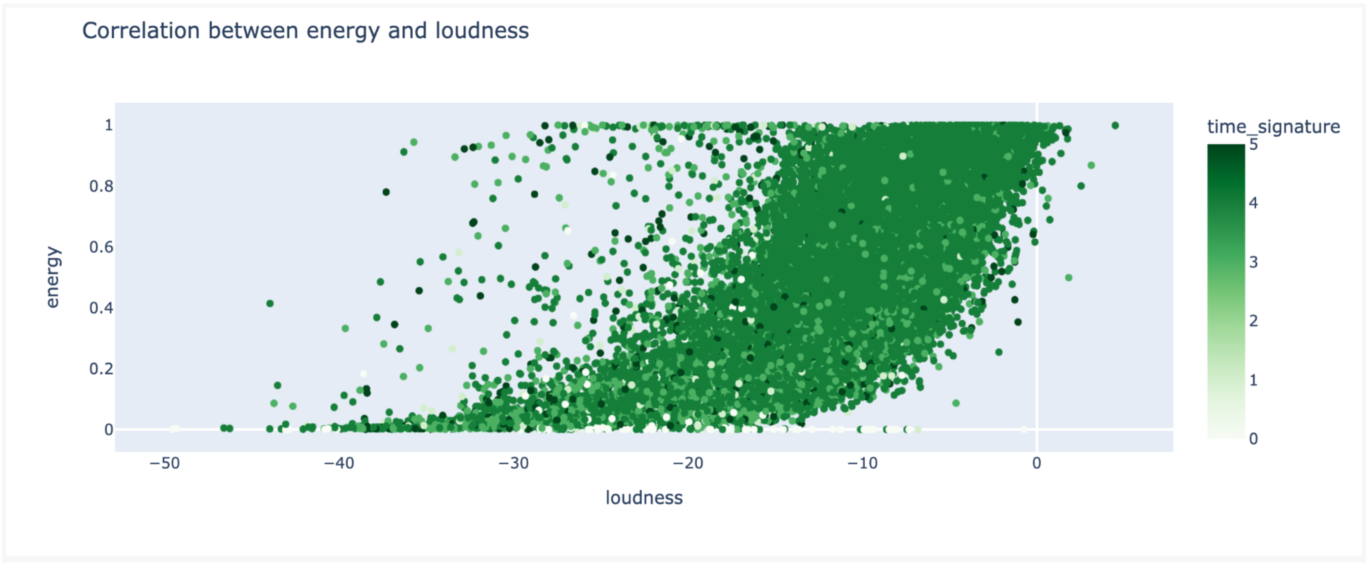
$$r = (n \sum_{i=0}^{n} x * y - (\sum_{i=0}^{n} x)(\sum_{i=0}^{n} y)) \div (\sqrt{(n \sum_{i=0}^{n} x^{2} - \sum_{i=0}^{n} x^{2}) * (n \sum_{I=0}^{n} y^{2} - \sum_{I00}^{n} y^{2})}$$

where:

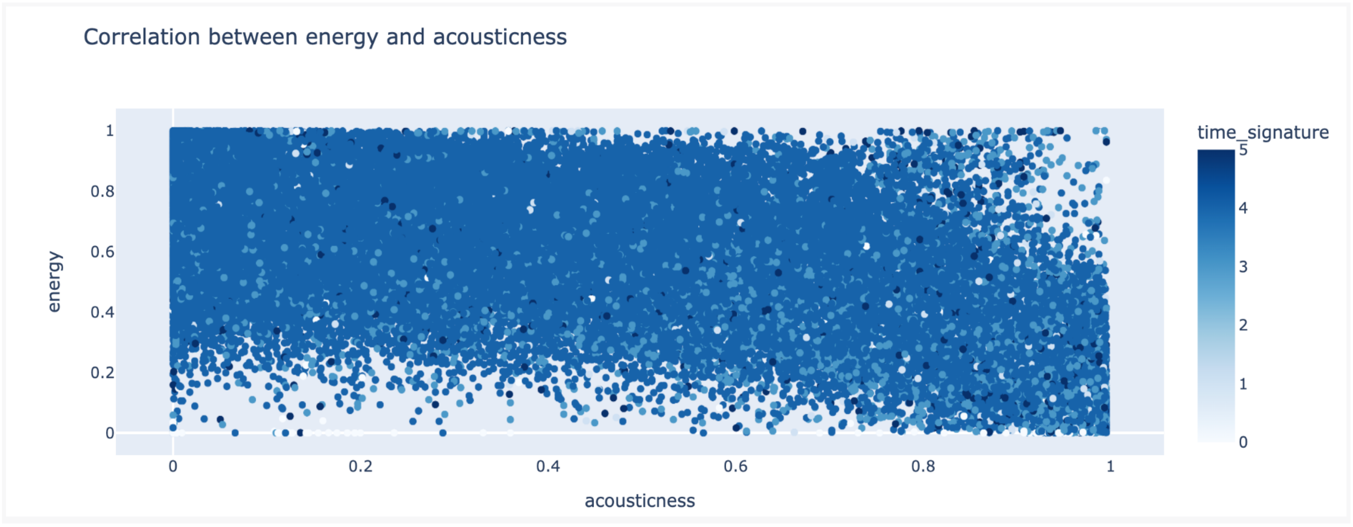
- x and y are two different features
- r is the correlation coefficient
- n is the number of data pairs
- ∑xy is the sum of the product of the corresponding x and y values
- ∑x and ∑y are the sums of the x and y values, respectively

The correlation coefficient, r, ranges between -1 and 1, with a value of 1 indicating a perfect positive correlation, a value of -1 indicating a perfect negative correlation, and a value of 0 indicating no correlation. A positive correlation means that as one variable increases, the other variable also tends to increase, while a negative correlation means that as one variable increases, the other variable tends to decrease.

To start, the correlation between variables that were believed to be the most closely related was computed. The first plot examined the relationship between the "energy" level and the "loudness" of a track, which were hypothesized to be closely related. The results showed that these variables were positively correlated with a coefficient of 0.76, indicating that as the energy level of a track increases, so does its loudness. This finding was already suggested by Chamorro-Premuzic, T., Fagan, P., & Furnham, A. (2009) who highlighted the importance of considering the role of loudness in music perception and its potential impact on listener experience. This high value of positive correlation indicates that there surely is a tendency for songs with louder tones to convey more energy.

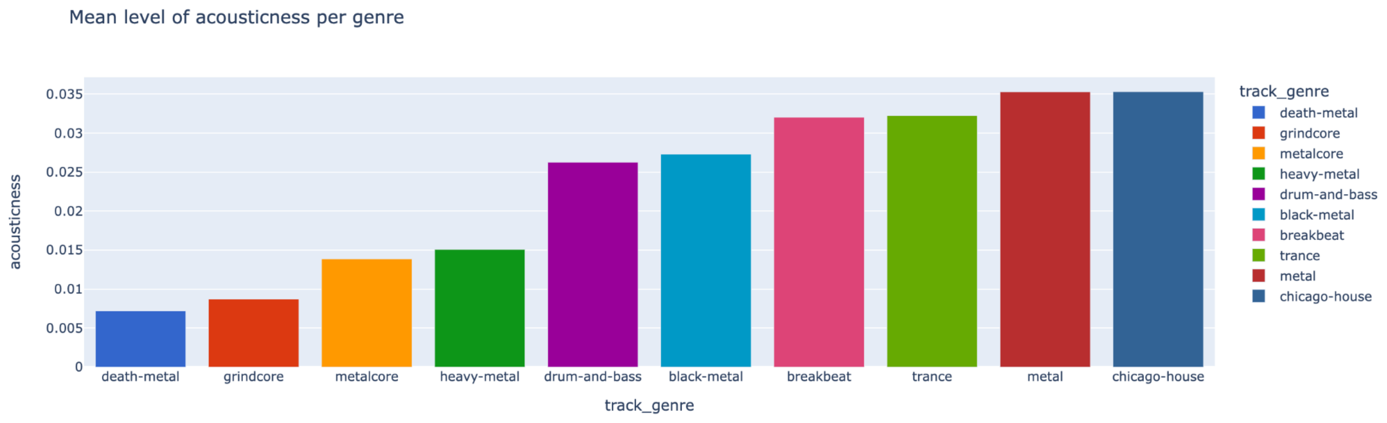


(Figure 3: correlation between energy and loudness)



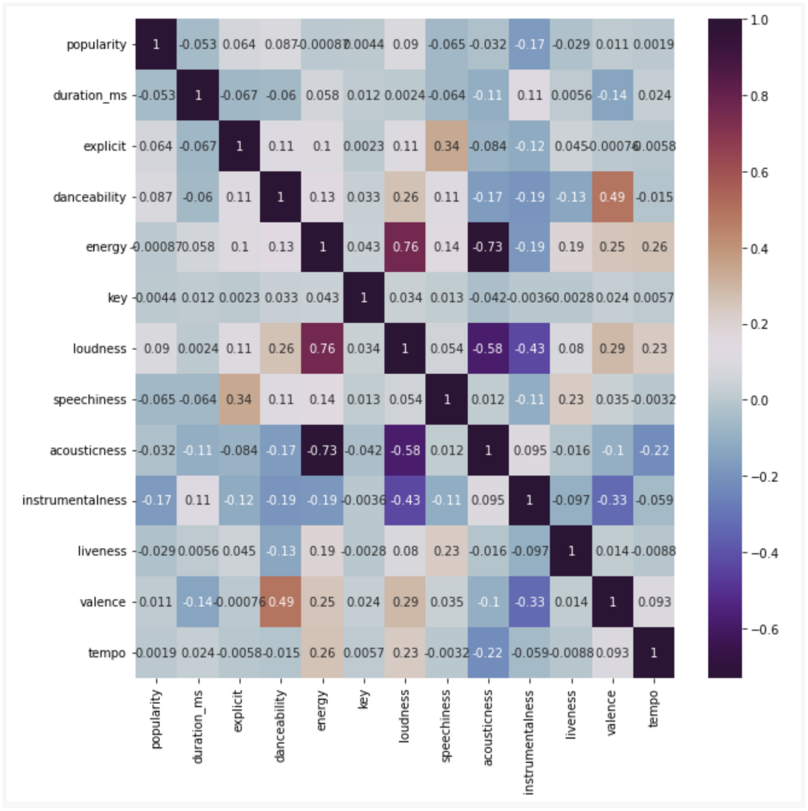
(Figure 4: correlation between energy and acousticness)

Subsequently, a second plot was generated to examine the relationship between the "energy" level and the "acousticness" score of a track. This plot revealed a negative correlation with a coefficient of -0.73, indicating that as the acousticness of a track increases, its energy level tends to decrease. This finding suggests that music tracks with higher acousticness scores may convey less energy overall. This contradicts what said by another study by Han et al. (2018) examined the relationship between acousticness and energy levels in music tracks who found that tracks with higher energy levels tend to have higher levels of acousticness, while tracks with lower energy levels tend to have lower levels of acousticness. Actually it seems like, people perceive as less energetic those songs whose level of acousticness is higher (e.g. classic music) and more energetic those with lower levels of acousticness (e.g. death metal).



(Figure 5: Mean level of acousticness of top 10 energetic genres)

Based on these intriguing results, a correlation matrix was constructed to investigate the correlations between all of the numeric variables:



(Figure 6: Correlation Matrix)

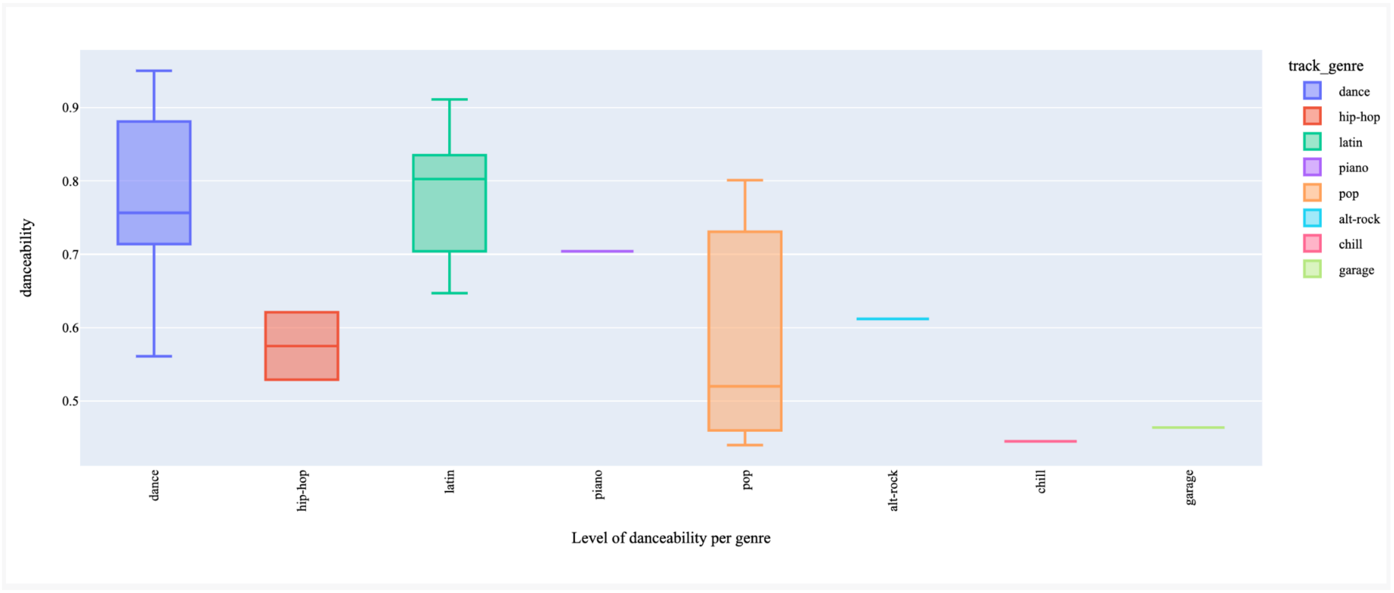
After generating a correlation matrix, the variables that were found to be most correlated with the target variable were identified. These variables were ordered according to their correlation coefficients, and the results are as follows:

1. Loudness → High positive correlation (0.76)
2. acousticness → High negative correlation (-0.73)
3. tempo → Low positive correlation (0.26)
4. valence → Low positive correlation (0.25)
5. liveness → Low positive correlation (0.19)

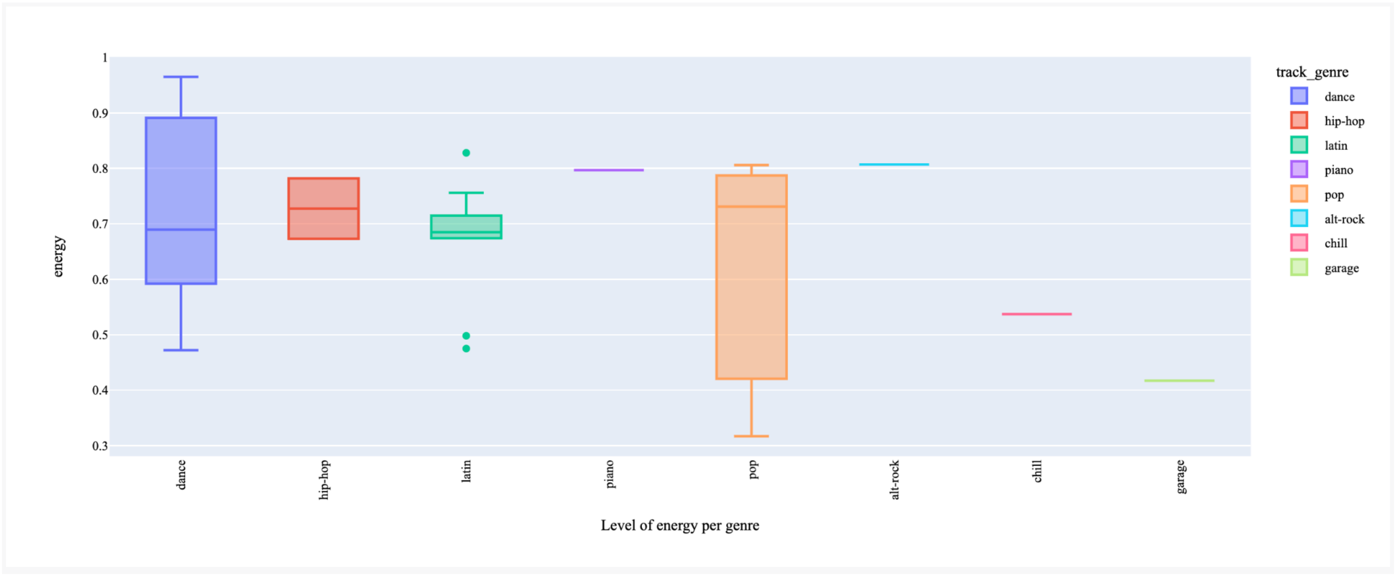
The correlation matrix revealed that, apart from "loudness" and "acousticness", the target variable "energy" had very weak correlation coefficients with the other variables in the dataset.

This suggests that a linear model may not be effective in accurately predicting the "energy" variable based solely on the other variables in the dataset. As will be discussed later, this was indeed found to be the case.

Lastly, two boxplots were generated to compare the danceability and energy levels of the top 100 most popular tracks across different genres. The boxplots revealed that the most danceable tracks were not necessarily the ones with the highest energy levels. In some cases, such as with hip-hop and all-rock, tracks that were not particularly danceable still conveyed a high level of energy.



(Figure 7: Boxplot of level of danceability per genre)



(Figure 8: Boxplot of level of danceability per genre)

Exploring and visualizing the data in this way provided valuable insights into the patterns and relationships within the dataset. These insights informed the subsequent steps of feature selection and model building. By identifying the variables that were most correlated with the target variable and understanding the relationships between different variables and genres, it was possible to select the most relevant features for building an accurate predictive model.

5. Modeling

5.a Tools and Libraries

The scripts used in this thesis project were written in Python, a high-level programming language that is known for its simple syntax and user-friendly interface. Python has become increasingly popular in data science due to the abundance of libraries and packages available, which make it easy to perform data analysis and machine learning tasks.

One of the most widely used libraries for data manipulation and preprocessing in Python is Pandas, which was extensively utilized in the script for data manipulation and preprocessing. Scikit-Learn is another crucial library that was employed in this study. It is a machine learning library that provides a wide range of algorithms for regression, classification, and clustering tasks. Scikit-Learn was used to build, train, and evaluate various regression models, such as Linear Regression, Ridge Regression, Lasso Regression, Random Forest Regression, and Gradient Boosting Regression

In addition to Scikit-Learn, two other libraries were imported for modeling and specifically for the development of Deep Learning models: TensorFlow and Keras. These libraries provide a powerful and flexible platform for building and training deep learning models for a range of applications, including computer vision, natural language processing, and speech recognition.

The script also employs other Python libraries for data visualization and model evaluation, including Plotly and Seaborn for creating interactive plots and graphs. Scikit-Learn's built-in functions were used for computing various performance metrics, such as Mean Absolute Error (MAE) and Rsquared score, which are the primary metrics used for evaluating the performance of the predictive model in this thesis whose choice is explained in the next sections.

Concerning the saving of the trained and optimized models and the development of the Web Application, the Pickle and Streamlit libraries were used. Streamlit was chosen for its simplicity and intuitiveness, since it allows users with little or no web development experience ro quickly create interactive apps and visualizations with just a few lines of Python code. Moreover it provides a fast feedback loop, allowing the real-time identification of changes as they’re made on the script. This makes it easy to experiment with different data visualizations and UI elements without having to spend a lot of time on development.

In addition to using various Python libraries, the Python file for this thesis project was developed using Visual Studio Code. Visual Studio Code (VS Code) is a free and open-source code editor developed by Microsoft which supports a wide variety of programming languages, including Python, JavaScript, and C#. VS Code has become increasingly popular among developers due to its ease of use, extensibility, and strong community support. It was used to write and edit the code for the machine learning and AI modeling, as well as for editing the streamlit_fhs.py file containing the code for creating the web app locally. Finally the web application was deployed on heroku.com which is a cloud platform providing free hosting servers for web applications. The choice fell on heroku.com thanks to: its easy deployment interface, which allows the user to create a web application without needing to worry about configuring servers or infrastructure, its scalability and its free tier, which offers users to host a small to medium web application without any cost.

5.b Data Preparation and Preprocessing

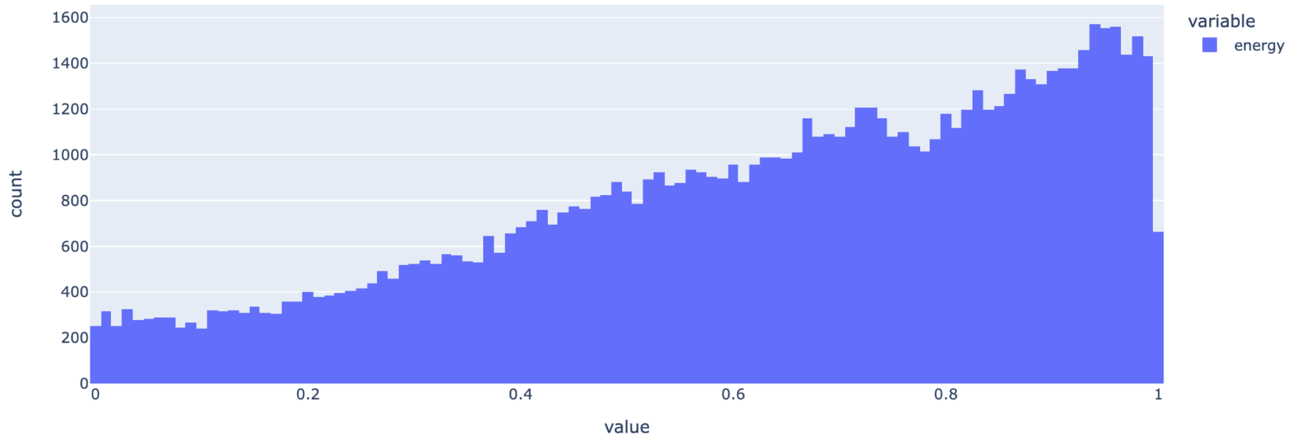
Data preprocessing is an important step in machine learning, as it involves transforming raw data into a format that can be easily used by algorithms. The first step in data preprocessing is loading the data into the environment. As specified earlier, in the given Python script, the Spotify dataset was loaded using the Pandas library. A priori by the type of task, data preprocessing and feature engineering are crucial steps in machine learning. These processes entail the cleaning, transforming, and identification of the most pertinent features from raw data, thus producing a dataset that is more amenable for use by algorithms.

Once the dataset was loaded, the focus shifted to identifying and handling missing values, duplicates, and outliers, as these issues can have a negative impact on model performance. The dataset was first examined for missing or duplicate values, with no missing values found and 32,656 instances removed as duplicates. Although some outliers were present, they were retained to allow the algorithms to understand the variability of the data, especially with regards to songs from different genres and their corresponding audio features.

Feature engineering was the next step, which involves creating new features from existing ones or selecting the most relevant ones for the given task. In this case, the objective was to predict the energy level of a song, so meaningful information was extracted from the Spotify features provided. After loading the dataset, the next step was to check for missing values, duplicates, and outliers. Missing values were not found, and duplicates were removed, while outliers were kept to provide algorithms the possibility to understand the variability of the data, especially regarding different track genres and related audio features. The audio features provided by Spotify were already normalized, scaled between 0 and 1. However, they needed to be appropriately transformed and standardized using the MinMaxScaler() module before training the models.

Feature engineering involves also data transformation and feature selection. Categorical features such as ‘track_genre’ and ‘explicit’ were converted into one-hot encoded variables to represent them as binary vectors. Feature selection techniques such as correlation analysis or feature importance rankings were used to identify the most relevant audio features for predicting energy level.

Standardizing the data is crucial to ensure that all features are treated equally by the machine learning algorithms, especially when the features have different units or scales. Standardization involves centering the data by subtracting the mean from each feature and scaling it by dividing by the standard deviation. This results in all features having zero mean and unit variance, which makes it easier for the algorithms to learn the relationships between them. The 'energy' feature as well, as can be denoted from the graph below, did not follow a Gaussian distribution and had to be scaled using an MinMaxScaler() to ensure its comparability with other features.

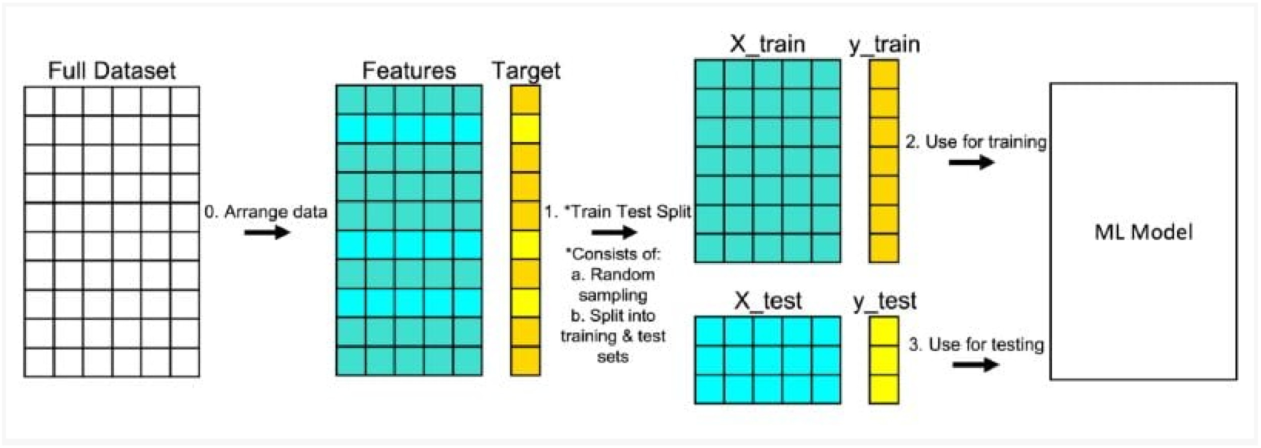


(Figure 9: Histogram of target variable “energy” ‘s distribution)

After scaling and transforming data and before training the models a subset of variables was chosen according to the logical relevance and the correlation found with the output variable. The subset of input variables chosen to train the statistical learning models was the following:

- **Popularity**
- **Duration_ms**
- **Danceability**
- **Key**
- **Loudness**
- **Speechiness**
- **Acousticness**
- **Instrumentalness**
- **Liveness**
- **Valence**
- **Tempo**
- **Explicit**
- **Track_genre (One-hot Encoded, 113 variables)**

The final step involved splitting the dataset into training and testing sets. The data was divided with a 80-20 ratio, where 80% of the data was used for training the models and 20% for testing their performance on unseen data. This helps in evaluating the effectiveness of the models on new data. Although attempts were made to use 5-fold cross-validation for model evaluation, it didn't result in any significant improvement in the results, and hence was discarded. The trained models were then evaluated on the test set using various metrics, which will be discussed in the upcoming section.



(Figure 10: Train-test split. Source builtin.com)

5.c Metrics and Hyperparameters Tuning

Evaluation metrics play a crucial role in machine learning as they enable the measurement of the performance of different models and facilitate the comparison of their accuracy. In this research , two evaluation metrics were utilized, namely the Mean Absolute Error (MAE) and the Rsquared (R2) coefficient, to assess the performance of the models.

The Mean Absolute Error (MAE) is a widely used metric that quantifies the accuracy of a regression model. It represents the average magnitude of the errors between the predicted and actual values of the response variable. The MAE is calculated by computing the absolute difference between the predicted and actual values of the response variable, and then taking the average of all these absolute differences. The formula for computing the MAE is:

$$MAE = 1/n * \sum_{i=1}^{n} |yi - \widehat{yi}|$$

Where:

- yi is the actual value of the target variable (energy level in this case)
- $\widehat{yi}$ is the value of the target variable predicted by the model
- n is the number of observations

The Mean Absolute Error (MAE) is a convenient evaluation metric due to its simplicity and ease of interpretation. It quantifies the average absolute difference between the predicted and actual values of the response variable, providing a straightforward intuitive understanding of the model's performance. Compared to the Root Mean Squared Error (RMSE), MAE is less affected by outliers, which makes it particularly useful when dealing with data that contains extreme values especially in a case like this where tracks belonging to different genres and with different audio metadata appear.

The Rsquared (R2) coefficient is a widely used metric for evaluating the performance of regression models. It quantifies the percentage of the variability in the response variable that is accounted for by the model. A value of 1 for R2 indicates a perfect fit of the model to the data, while a value of 0 indicates that the model cannot explain any of the variability in the response variable. R2 is easy to interpret and can be used to compare the performance of different models. However, it may not be the best metric for models that have a high variance and produce large errors, as it is less sensitive to these errors than MAE. The formula for computing the **Rsquared** (**R2**) coefficient is the following :

$$\textbf{Rsquared=}\ 1 - \left(\sum_{i=1}^{n} (yi - \widehat{yi})\hat{}2 \div \sum_{i=1}^{n} (yi - \overline{y})\hat{}2 \right)$$

Where:

- yi is the actual value of the target variable (energy level in this case)
- $\overline{y}$ is the mean value of the target variable
- n is the number of observations

The decision to use both MAE and R2 as evaluation metrics in this study was made to provide a more comprehensive evaluation of the models. MAE was chosen to assess the accuracy of the models, while R2 was used to provide an overall measure of how well the models fit the data. This approach was chosen to enable comparison with previous research by Zhang et al. (2020) and Kim et al. (2019), where MAE was commonly used to evaluate accuracy. By using both metrics, a more complete understanding of the performance of the models can be achieved. The models were first evaluated using MAE, and then the overall performance was compared using R2. This method enabled the identification of the models with the best balance between accuracy and fit.

A second essential aspect of the modeling part , besides the choice of the evaluation metrics, was Hyperparameters tuning. Hyperparameter tuning is a critical process in machine learning that involves selecting the optimal set of hyperparameters for a given model. In this study, hyperparameter tuning was achieved using the GridSearchCV method, which exhaustively searches over a predefined grid of hyperparameters to find the combination that results in the best performance. The hyperparameters were used to identify the most accurate model fit for this specific type of dataset by selecting among different combinations of grid values, such as alphas in Lasso and Ridge regression or the number of estimators, learning rate, and tree depth in tree-based methods.

5.d Statistical Learning Algorithms

The primary objective of this thesis is to develop a precise and reliable model capable of predicting the energy level of music tracks using various features such as tempo, loudness, and danceability. The models selected for this study were chosen based on their ability to handle complex and large datasets and their capacity to capture the relationships between the features and the target variable.

The algorithms used in this study were categorized into three classes: Linear models, Tree-based models, and Neural network-based models. Each of these models has its advantages and disadvantages. Linear models are relatively simple and interpretable but may not capture complex nonlinear relationships while Tree-based models can capture complex nonlinear relationships, but may overfit the data (Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, 2013). On the other hand, Neural network-based models are capable of modeling complex nonlinear relationships but can be pretty challenging to interpret and require a significant amount of data to train effectively.

The choice of the appropriate statistical learning algorithm relies on the particular problem at hand, the nature of the data, and the available resources. The following sections will provide a detailed explanation of each algorithm, its mechanisms, advantages, disadvantages, and performance on the dataset. Here is the list of statistical learning methods employed in this research classified by category:

- Linear models:
    - Simple Linear Regression
    - Multivariate Linear Regression
    - Linear Regression with interaction term

    - Ridge Regression
    - Lasso Regression
- Tree-based models and Non Linear Models:
    - Decision Tree Regressor
    - Random Forest Regressor
    - XGBoost Regressor
    - Gradient Boosting Regressor
    - AdaBoost Regressor
    - ExtraTrees Regressor
    - K-Nearest-Neighbors
- Neural Network-based models:
    - Multilayer Perceptron
    - Deep Neural Networks

5.d.i Linear Models

Linear Regression

Two typologies of linear regression were fitted in this research: a Simple Linear Regression and a Multivariate Linear Regression.

Simple linear regression is a type of linear regression where there is only one independent variable, which is used to predict a dependent variable. In the case of this dataset, the independent variable used is "loudness," and the dependent variable is "energy." The model estimates the relationship between these two variables by fitting a line that minimizes the sum of the squared errors between the predicted values and the actual values. The line is defined by the slope and intercept, which are estimated using the least squares method.

Here is the equation describing this Simple Linear Regression:

$$energy = \beta0 + \beta1 * loudness$$

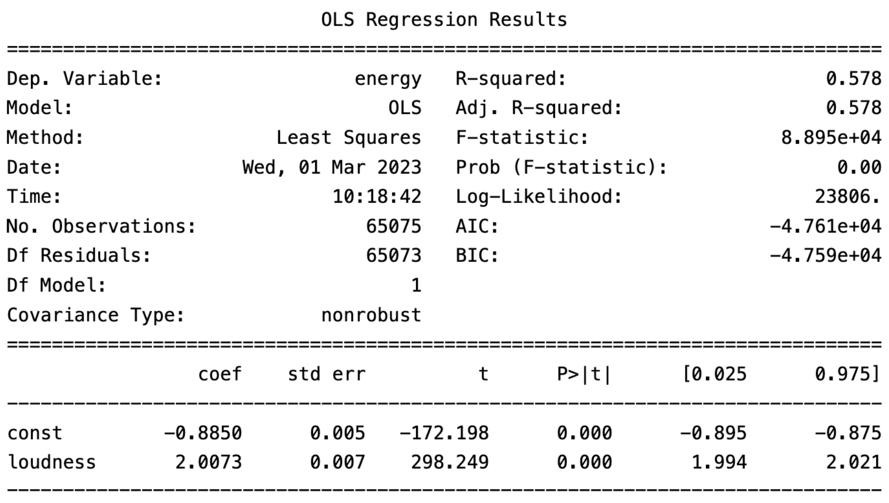
Where:

- β0 is the intercept coefficient
- β1 is the vector of coefficients for loudness

One of the main advantages of simple linear regression is its interpretability, as the slope represents the change in the dependent variable for every one-unit increase in the independent variable. However, a drawback of simple linear regression is that it assumes a linear relationship between the variables, which may not always hold true in real-world scenarios.

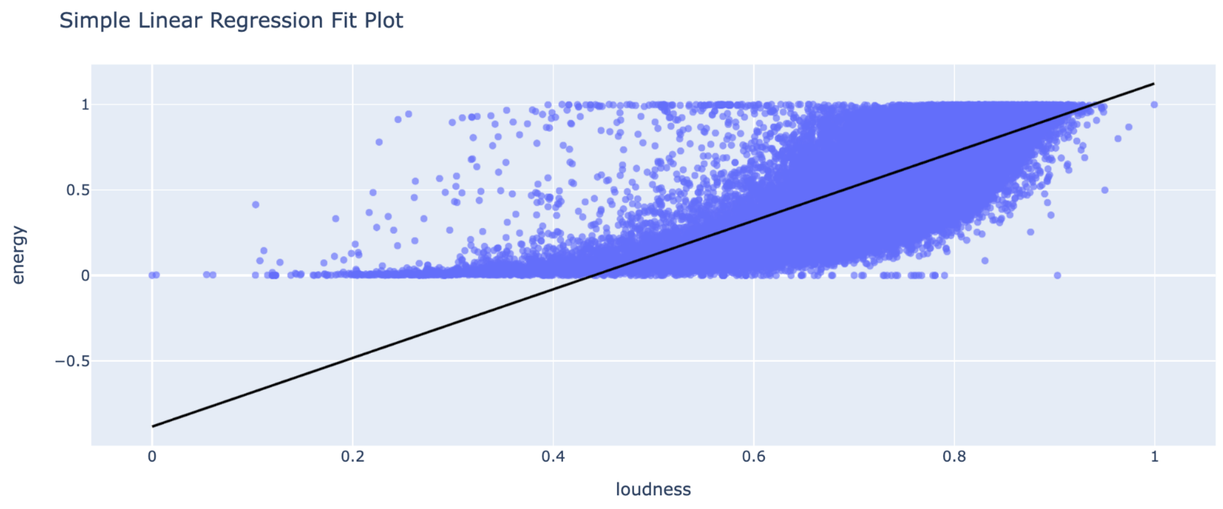
Once the model is fitted using the least squares approach, the summary statistics are displayed, which include the intercept, coefficients, Rsquared value, and p-value. The coefficient denotes the change in the dependent variable for each unit increase in the independent variable, and the intercept denotes the predicted value of the dependent variable when the independent variable is zero. The p-value indicates the possibility of observing a coefficient as extreme as the one obtained in the model if the null hypothesis (no relationship between the dependent and independent variables) were true. A p-value that is less than a predetermined threshold (usually 0.05) indicates the statistical significance of a predictor and its usefulness in the final estimation of a dependent variable.

In the case of Simple Linear Regression, if the p-value of the predictor 'loudness' is equal to 0, It indicates that the predictor is significantly related to the target variable and should be retained in the regression model. Moreover, by examining the coefficient value of the predictor, which is 2.007, the extent of the relationship between the two variables can be determined. Specifically, a one-unit increase in the independent variable will result in a twofold increase in the "energy" target variable. Another crucial finding that was obtained from the summary is the Rsquared value, which is one of the metrics used in this research. It appears that the "loudness" variable alone can account for more than 57% of the variation in the data utilized to train the model. The last important measure to compute was MAE that resulted to be 0.134.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 energy   R-squared:                       0.578
Model:                            OLS   Adj. R-squared:                  0.578
Method:                 Least Squares   F-statistic:                 8.895e+04
Date:                Wed, 01 Mar 2023   Prob (F-statistic):               0.00
Time:                        10:18:42   Log-Likelihood:                 23806.
No. Observations:               65075   AIC:                        -4.761e+04
Df Residuals:                   65073   BIC:                        -4.759e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.8850      0.005   -172.198      0.000      -0.895      -0.875
loudness       2.0073      0.007    298.249      0.000       1.994       2.021
------------------------------------------------------------------------------
```

(Figure 11: Simple Linear regression summary)



(Figure 12: Simple Linear Regression Fit)

Multivariate linear regression is employed when multiple independent variables are utilized to forecast a dependent variable, the “energy” variable. Like Simple Linear Regression, the algorithm calculates the coefficients of the independent variables through the least squares approach, but in this case, it involves more than one predictor. The resulting equation is then utilized to make predictions. Here is the equation of the Multivariate linear regression model:

$$energy = \beta 0 + \beta 1 * popularity + \beta 2 * duration + \ldots\ldots + \beta 126 * explicit$$

Multivariate linear regression enables the modeling of more intricate relationships and can be more precise than Simple Linear Regression. However, it also necessitates the fulfillment of more assumptions, such as the independence of the variables and normality of the residuals.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 energy   R-squared:                       0.804
Model:                            OLS   Adj. R-squared:                  0.804
Method:                 Least Squares   F-statistic:                     2131.
Date:                Wed, 01 Mar 2023   Prob (F-statistic):               0.00
Time:                        10:17:32   Log-Likelihood:                 48791.
No. Observations:               65075   AIC:                        -9.733e+04
Df Residuals:                   64949   BIC:                        -9.619e+04
Df Model:                         125
Covariance Type:            nonrobust
==================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
const                           -0.5312      0.008    -70.538      0.000      -0.546      -0.516
popularity                      -0.0220      0.003     -7.488      0.000      -0.028      -0.016
duration_ms                      0.0326      0.023      1.415      0.157      -0.013       0.078
danceability                    -0.1051      0.004    -28.027      0.000      -0.112      -0.098
key                              0.0022      0.001      1.615      0.106      -0.000       0.005
loudness                         1.4309      0.007    195.175      0.000       1.417       1.445
speechiness                      0.2008      0.005     37.370      0.000       0.190       0.211
acousticness                    -0.2449      0.002   -117.041      0.000      -0.249      -0.241
instrumentalness                 0.0761      0.002     38.829      0.000       0.072       0.080
liveness                         0.1131      0.003     44.674      0.000       0.108       0.118
valence                          0.1510      0.002     64.407      0.000       0.146       0.156
...
```

(Figure 13: Multivariate Linear Regression summary)

In the case of multivariate linear regression, it seems that the majority of the numeric variables in the dataset possess statistical significance in describing the variability of the output variable, with the exception of "duration_ms" and "key," which will still be retained and employed for training and fitting other models. The Rsquared and MAE values on the test set have both increased, reaching values of 0.806 and 0.087, respectively.

A final attempt was made by utilizing a Linear Regression model with an interaction term whose primary effects were "loudness" and "acousticness," the two most correlated features with the target variable. The interaction term was defined as follows:

$$Interaction\,Term = loudness\ x\ acousticness$$

Incorporating this interaction term resulted in a minor increase in the model's performance, with an Rsquared of 0.812 and a MAE of 0.085 achieved. However, its effectiveness was restricted to linear models, as utilizing it as a new feature in the input matrix did not provide any enhancements in the regularization and nonlinear models.

Given these performances, regularization techniques were implemented in order to improve the predictive performances.

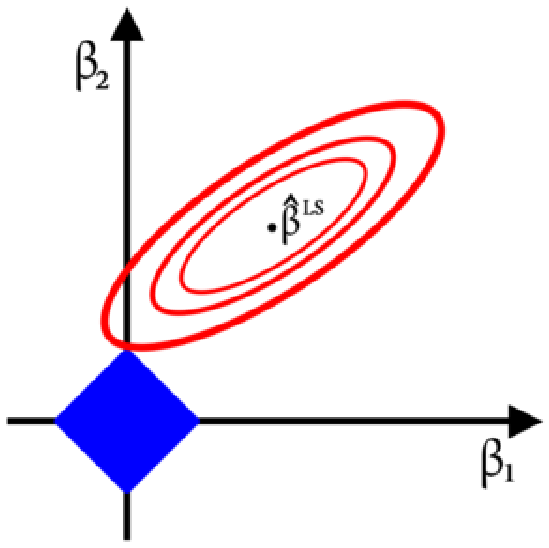
Lasso Regression

Lasso regression, also called L1 regularization, is a method that involves adding a penalty term to the least squares objective function. The penalty term λ is a combination of the sum of squared errors and the sum of the absolute values of the coefficients, multiplied by a tuning parameter alpha. This penalty term shrinks the coefficients of irrelevant features to zero, leading to a more parsimonious model and better interpretability. The formula for Lasso regression is as follows:

$$\text{Lasso} = \sum_{i=0}^{n} (yi - \beta 0 - \sum_{J=1}^{p} Xij\, \beta j)^2 + \lambda \sum_{J=1}^{p} |\beta j|$$

where

- yi is the target column value for instance i
- Xij is the value of an explanatory variable for instance i
- β is the vector of coefficients
- λ is the penalty term
- $\sum_{J=1}^{p} |\beta j|$ is the L1 norm of a coefficient vector β



(Figure 14: Lasso Regression coefficient shrinking plot. Source: www.natasshaselvaraj.com)

Lasso regression is advantageous because it can handle high-dimensional data and perform feature selection. However, one of its main disadvantages is that it may lead to biased coefficient estimates if the selected variables are correlated with each other.

In this case, the dataset consisted of 126 variables, making Lasso regression a potentially useful approach. However, the best hyperparameter for alpha was found to be e-0.5, and the resulting performances of the regularization algorithm were disappointing. The model achieved an Rsquared of 0.806 and a MAE of 0.087, which were almost equivalent to those obtained using the Multivariate Regression Model.

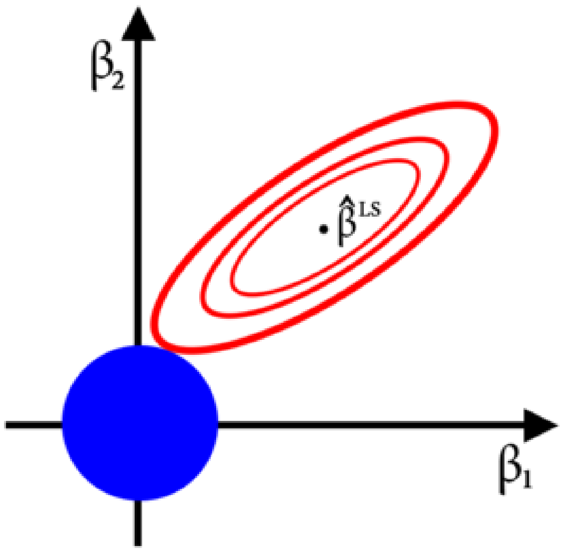
Ridge Regression

As said earlier, regularization techniques such as Ridge Regression and Lasso Regression can be used in linear regression to address overfitting. Both methods aim to reduce the magnitudes of the coefficients in the model, but they differ in how they apply the penalty term to the cost function. Lasso Regression adds a penalty term proportional to the absolute value of the coefficients, while Ridge Regression adds a penalty term proportional to the square of the coefficients. Ridge Regression tends to shrink the coefficients towards zero without necessarily setting them to zero, making it more suitable when all the features are potentially relevant for the outcome. The formula for Ridge Regression is the following:

$$\text{Ridge}= \sum_{i=0}^{n} (yi - \beta 0 - \sum_{J=1}^{p} Xij\, \beta j)^{2} + \lambda \sum_{J=1}^{p} \beta j^{2}$$

where

- yi is the target column value for instance i
- Xij is the value of an explanatory variable for instance i
- β is the vector of coefficients
- λ is the penalty term
- $\sum_{J=1}^{p} \beta j^{2}$ is the L2 norm of a coefficient vector β



(Figure 15: Ridge Regression coefficient shrinking plot. Source: www.natasshaselvaraj.com)

Both Lasso and Ridge have their strengths and weaknesses and are valuable tools in the machine learning toolbox.In this case, hyperparameter tuning was performed to find the optimal value of alpha, which was found to be 0.533 using gridsearch. However, the performance of Ridge Regression was similar to that of Lasso and Multivariate Linear Regression, with a MAE of 0.087 and an Rsquared of 0.806. This indicated that the only way to improve prediction accuracy would be to adopt non-linear models, which was taken as the next step.

5.d.ii Tree-Based and Non-Linear Models

Due to their versatility in handling categorical and numerical data as well as their interpretability, Tree-based models are a type of machine learning algorithms that have become more and more popular in recent years. Decision trees, hierarchical structures that divide the feature space into smaller areas based on straightforward decision rules, serve as the foundation for these models. The resulting tree is composed of splits, with each leaf node denoting a prediction and each node denoting a decision rule based on a feature. In comparison to other machine learning techniques, tree-based models have a number of benefits, such as the capacity to handle non-linear correlations between the features and the outcome, robustness to outliers, and simplicity of interpretation. Tree-based models, such as Random Forest and Gradient Boosting, have recently attained state-of-the-art performance on a variety of tasks, including image classification, speech recognition, natural language processing, and, as will become clear at the end, also on this particular application of machine learning.

However they present some drawbacks, such as their propensity to overfit when the tree is overly complicated and their sensitivity to the selection of hyperparameters. Because of this, hyperparameter tuning in this research was crucial and time-consuming, but it ultimately paid off and produced models that performed incredibly well, as will be explored in the following sections of this chapter.

Decision Tree Regressor

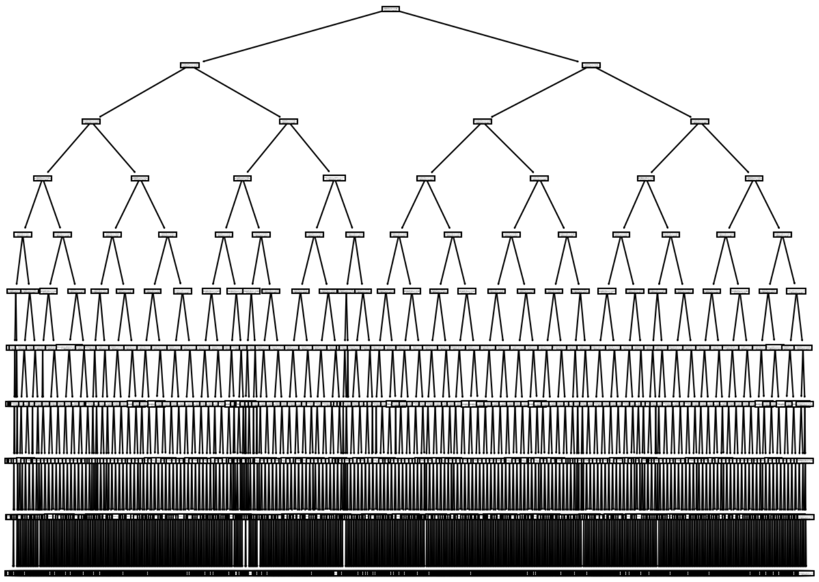
Decision trees are a widely used machine learning technique for both classification and regression tasks. This method involves creating a tree-like model that represents a series of decisions and their possible outcomes. The tree is made up of nodes, which correspond to a decision or test on a particular attribute, and branches, which represent the possible outcomes of that decision.The process of building a decision tree involves selecting the best attribute to split the data at each node, based on some criterion like information gain or Gini index. This process is repeated recursively until the tree is fully grown or some stopping criterion is met. To prevent overfitting, techniques like pruning and setting a minimum sample size at each node are often employed. Once the decision tree is constructed, it can be used to predict the class or value of new instances by traversing the tree from the root node to a leaf node that corresponds to a predicted outcome. Decision trees are also used in some ensemble methods that are explored in this thesis as random forests, where multiple decision trees are combined to improve the overall accuracy of the model.

One of the key advantages of decision trees is their interpretability. Each branch of the tree represents a distinct decision, making it easy to understand and explain how the model arrived at its predictions. Decision trees can also be a powerful tool for feature selection, as the top-level nodes of the tree represent the most important features for predicting the target variable.

After trying out many options and combination of hyperparameters, the best choice trough GridSearch for the model was the following :

- Max Depth =10
- Max Features = 120

The model's structure is depicted in the accompanying image, which shows the layout of the decision tree. While the image may not be very informative in terms of the logical flow of the model, it does reveal that the number of features in the model is 120 and that the number of root-node links is set to 10, as specified by the hyperparameters.



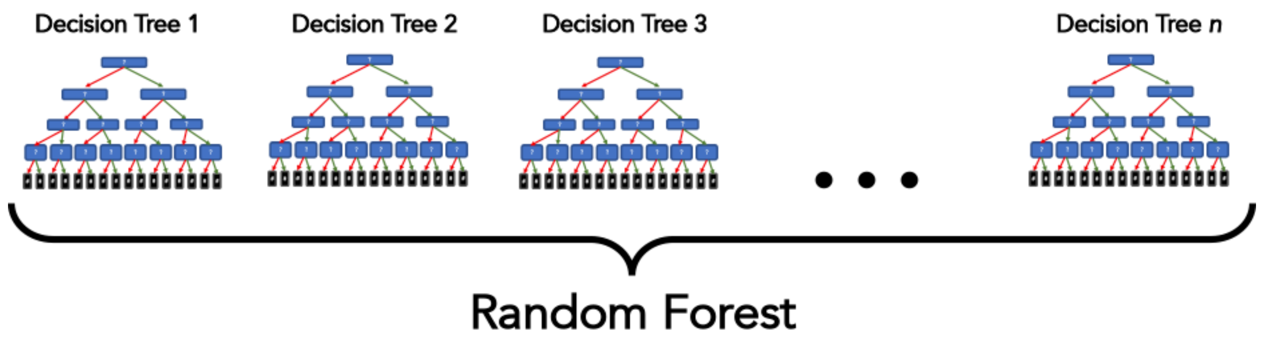
(Figure 16: Decision Tree Regressor developed on the dataset)

The performance of the Decision Tree Regressor model was not particularly impressive, as it produced scores that were close to those of the Ridge Regression regularization model with a 0.083 MAE and a 0.808 Rsquared. This was due to the fact that both models chose to exclude a subset of variables that were statistically insignificant, and this subset happened to be the same for both models.

Random Forest Regressor

Random forests are a type of ensemble learning technique that combines multiple decision trees to improve the accuracy and robustness of the model. This is achieved by creating a large number of decision trees, with each tree being trained on a randomly selected subset of the training data and a randomly selected subset of the features. This randomization helps to reduce overfitting and improve the generalization performance of the model.The predictions of each individual tree are then aggregated to obtain the final prediction of the random forest model.

In classification tasks, the most common prediction is taken through majority vote, while in regression tasks the mean of the predictions is taken. The use of multiple decision trees allows the model to capture complex interactions between features that may be missed by a single decision tree. Random forests are highly flexible and can handle large datasets with high-dimensional features. Additionally, they are scalable, making them suitable for handling large amounts of data.



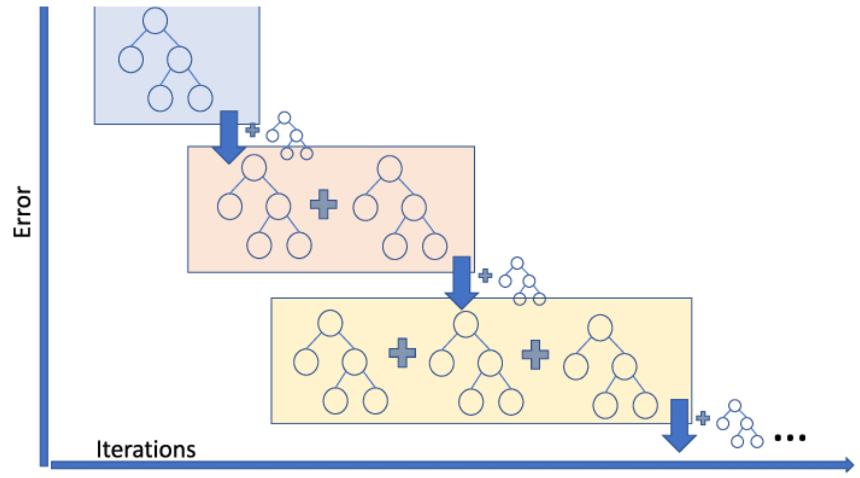
(Figure 17: Random Forest Regressor. Source thaddeus-segura.com)

Although Random Forests are highly flexible and can handle large datasets with high-dimensional features, one potential drawback is that they can be computationally expensive, especially for large datasets and a large number of trees. In this study, the process of fine-tuning the hyperparameters of the Random Forest model was particularly time-consuming, taking more than one hour to build all possible combinations of hyperparameters.

However, given the complexity of the data and the need to capture nonlinear relationships between the features, Random Forests were considered as a suitable choice, as demonstrated by the improved results. The fine-tuned Random Forest model achieved a lower MAE of 0.071 and a higher Rsquared value of 0.863 compared to the other models, indicating that non-linear models were more effective in achieving a highly accurate predictive algorithm.

Gradient Boosting Regressor

Boosting algorithms were used as the second set of statistical learning algorithms to create the best predictive model. These algorithms improve by focusing on the residuals and several boosting methods were tried, including XGBoost, Gradient Boosting, and ADABoost. Gradient Boosting was the first method tested, which uses decision trees to correct errors from previous trees and iteratively adds trees to the model. The difference between Gradient Boosting and other boosting methods, such as XGBoost, is in how the algorithm is optimized. XGBoost uses gradient-based sampling to select instances for each new tree, which reduces computation time and improves generalization. Additionally, XGBoost can handle missing data and has mechanisms to control model complexity. However, Gradient Boosting is less optimized, slower, and less scalable than other boosting methods. Gradient Boosting does not assign weights to instances and focuses on correcting errors from previous weak learners, making it less susceptible to overfitting. Despite its drawbacks, Gradient Boosting was chosen as the first trial to determine if it could improve prediction accuracy.



(Figure 18: Gradient Boosting algorithm. Source: Researchgate.com, Prediction of geometry deviations in additive manufactured parts: comparison of linear regression with machine learning algorithms)

The results of the Gradient Boosting algorithm were pretty good and encouraging since with a fine tuned learning rate of 0.5 it gave a 0.856 Rsquared and 0.073 MAE. Such an improvement in the accuracy paved the way to further increases in the accuracy by trying out more boosting algorithms which appeared to be the most effective methods for such data so far.

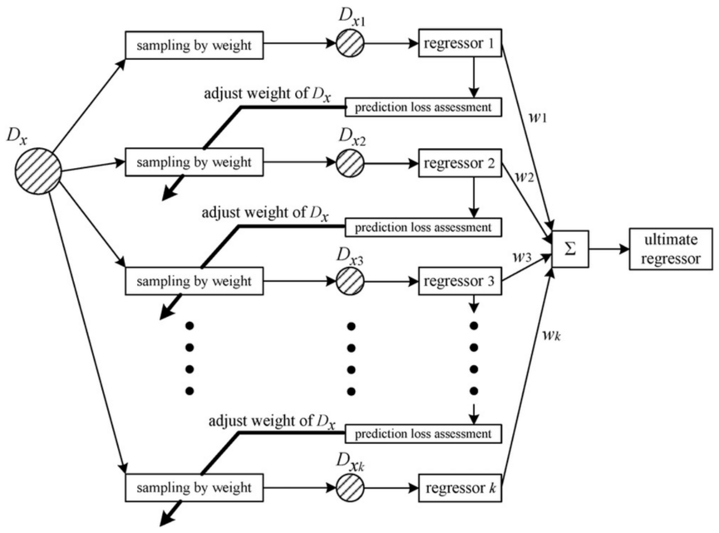
ADABoost Regressor

The second boosting algorithm used in the study was ADABoost, which is a popular ensemble learning technique that combines weak learners to form a strong learner. It works by adding weak learners iteratively to the model, where each new learner focuses on instances that were misclassified by previous learners. The algorithm assigns weights to each instance in the training set, with weights adjusted to focus on the most difficult-to-classify instances. The weak learners are then trained on the weighted data, and the final prediction is obtained by combining the predictions of all the weak learners.

ADABoost is easy to implement and generally performs well on classification tasks, especially when using decision trees as weak learners. It is also less prone to overfitting than bagging but more susceptible than Gradient Boosting.

Unlike Gradient Boosting and XGBoost, ADABoost assigns weights to training instances based on their classification difficulty before training the weak learners on the weighted data.

Another difference is that ADABoost is less computationally demanding than Gradient Boosting and XGBoost, making it a better choice for large datasets or limited computing resources.



(Figure 19: ADABoost algorithm. Source: Researchgate.com, Calibration of soft sensor by using Just-in-time modeling and AdaBoost learning method)

Unfortunately, the method architecture proved unsuitable for the task at hand, as the model performed worse than even the multivariate linear regression model, with a 0.108 MAE and a 0.729 Rsquared.

XGBoost Regressor

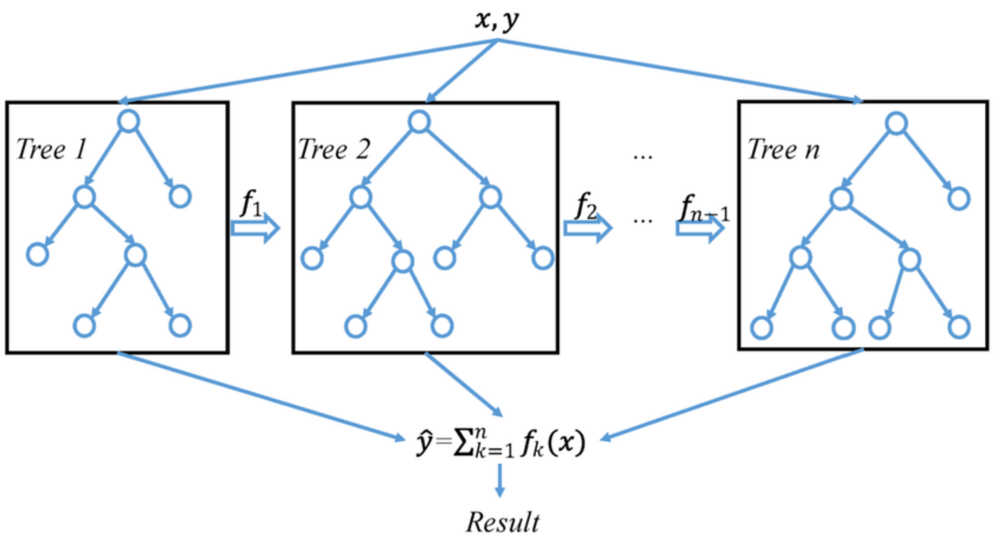
The last boosting algorithm tested in this study and which gave great results was XGBoost. XGBoost (Extreme Gradient Boosting) is a powerful ensemble learning technique that is commonly used for classification and regression tasks. It is an algorithm that builds on the concept of gradient boosting, which involves iteratively adding decision trees to the model to improve its accuracy.

The XGBoost algorithm works by optimizing a loss function using gradient descent. For example, in regression tasks, the loss function adopted might be the MSE or the MAE. In each iteration, a new decision tree is added to the model to reduce the residual error of the current prediction. The new tree is trained on the gradient of the loss function with respect to the current prediction, which helps to prioritize the instances that are most difficult to predict.

To prevent overfitting, XGBoost uses regularization techniques such as L1 and L2 regularization, which are the same techniques used by Lasso and Ridge, respectively. Additionally, XGBoost has built-in mechanisms to control the complexity of the model, such as the maximum depth of the decision trees and the minimum number of instances required to split a node.

One advantage of XGBoost over other gradient boosting methods is its use of "gradient-based sampling" to select the instances used to train each new tree. This technique helps to reduce computation time and improve the generalization performance of the model. XGBoost can also handle missing data and is highly scalable and efficient for large datasets. Another advantage of XGBoost is its interpretability. The algorithm provides several metrics for assessing the importance of each feature, and it also allows for visualization of the decision trees and their predictions. This can be useful for understanding how the algorithm is making its predictions and for identifying which features are most important for the task at hand.

In summary, XGBoost is surely the most powerful boosting algorithm that uses gradient boosting and optimization techniques to build accurate models for classification and regression tasks. It has several advantages, including the ability to handle missing data, scalability for large datasets, and interpretability through feature importance metrics and decision tree visualization.



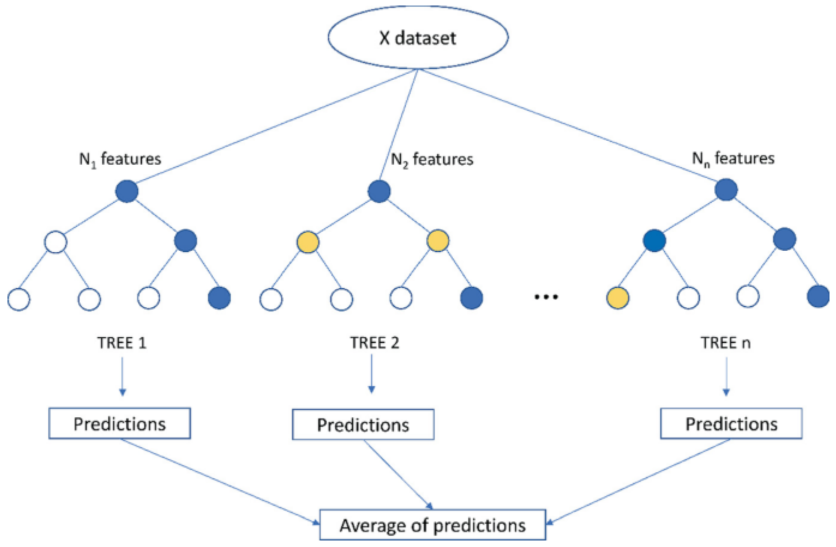
(Figure 20: XGBoost architecture. Source Researchgate.com, A hybrid ensemble method for pulsar candidate classification)

The initial experimentation with a limited number of tune parameters yielded unexpectedly favorable outcomes. Subsequent tuning utilizing additional parameters resulted in remarkable findings. Specifically, the XGBoost optimized algorithm, set to a learning rate of 0.05 produced a mean absolute error (MAE) lower than that of the Random Forest model equal to 0.067 and Rsquared of 0.878. This outcome served as evidence of the soundness and efficacy of this method to the dataset in question.

Extra Trees Regressor

The final tree-based method utilized in this study was the Extra Trees Regressor, a machine learning algorithm designed for regression problems that leverages an ensemble method by combining multiple decision trees to create a robust learner. This algorithm constructs numerous decision trees, each trained on a random subset of both the training data and features. Unlike other decision tree-based models, the algorithm selects the split point for each feature randomly, rather than based on the impurity of the data, which serves to mitigate overfitting.

One of the significant advantages of Extra Trees Regressor is its ability to reduce variance, which is a common problem with other ensemble methods, such as Random Forest. This lower variance results in improved generalization performance of the model. Furthermore, due to the random selection of features and split points, Extra Trees Regressor is less prone to overfitting than other decision tree-based algorithms. The difference between Extra Trees Regressor and other ensemble methods, such as Random Forest, lies in the feature and split point selection process. While Random Forest chooses the best feature and split point based on the impurity of the data, Extra Trees Regressor selects them randomly, which enhances the model's ability to generalize and reduces overfitting.Another advantage of Extra Trees Regressor is its speed, as it tends to outperform other decision tree-based algorithms, including Gradient Boosting and XGBoost, in terms of computation time taking more or less 15 minutes on this set of features. This makes it a preferred method for problems involving large datasets or limited computing resources as in this study.



(Figure 21: Extra Trees algorithm. Source: link.spinger.com, Extra Trees Ensemble: A Machine Learning Model for Predicting Blast-Induced Ground Vibration Based on the Bagging and Sibling of Random Forest Algorithm)

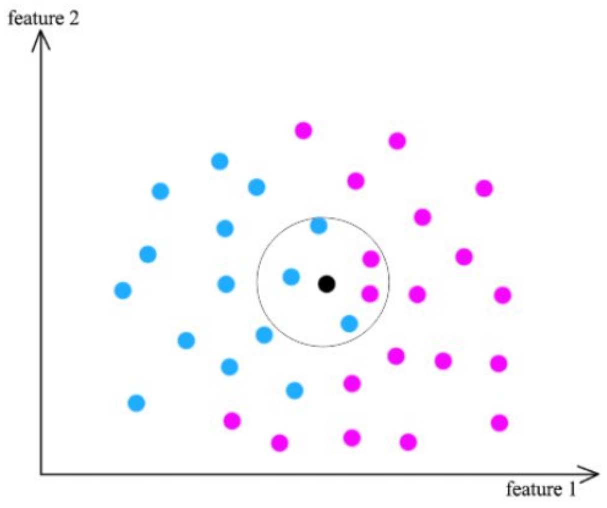
The results obtained by training and fitting this algorithm on the test set were really good but still inferior to those of the XGBoost model, since it had an Rsquared of 0.864 and a MAE of 0.070 with a fine tuned max depth of 36 root-node links.

K-Nearest-Neighbors

In this study was developed only one non-linear and non-tree-based model, the K-Nearest Neighbors (KNN), as it is not typically employed in precision-based regression problems. KNN is a machine learning non-parametric algorithm that doesn't make assumptions about the underlying data distribution. It functions by finding the K nearest neighbors to a specific data point in the training set and uses their labels to predict the label of the new data point.

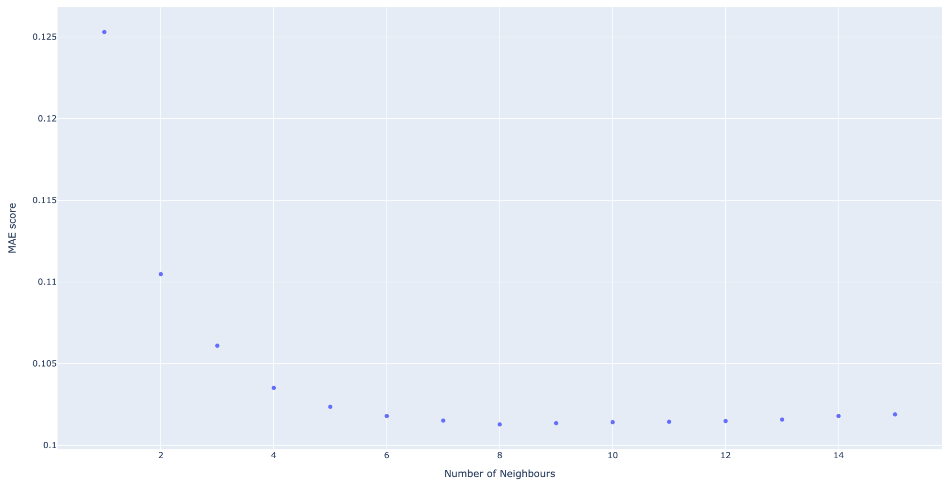
The main advantage of KNN is its simplicity and ease of implementation. It is suitable for both classification and regression problems and can handle non-linear decision boundaries. Furthermore, KNN is a lazy learning algorithm, meaning that it doesn't require training or model fitting, and can be used for online learning. However, one of the significant drawbacks of KNN is that it can be computationally expensive, particularly when dealing with large datasets.

Additionally, the algorithm is sensitive to the choice of distance metric used to measure the similarity between data points. Additionally, the KNN method can be sensitive to the selection of the parameter K, which represents the number of neighbors to consider when making a prediction.



(Figure 22: KNN Regressor (2 dimensions).Source: Datacademia.com)

Given the sensitivity of the algorithm to the number of neighbors chosen to fit the model, values of K (neighbors) from 1 to 15 were tried and evaluated, as is shown in the graph below.



(Figure 23: MAE score vs number of neighbours)

The results of the method after training and fitting it with the best number of neighbors, which was 15, gave a 0.125 MAE and a 0.583 Rsquared. By comparing it to the most underperforming non linear model, KNN proved to be one of the least suited algorithms for the object of the research and was soon discarded.As previously mentioned, the K-Nearest Neighbors algorithm is not commonly used in the context of continuous variable regression due to its coarseness.

5.d.iii Deep Learning Models

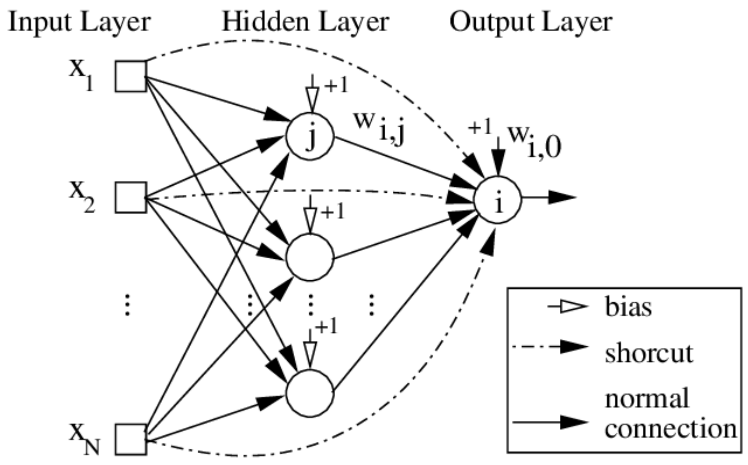
Neural networks represent a potent category of machine learning algorithms that emulate the structure and operation of the human brain. These networks employ interconnected nodes, also called neurons, to process and analyze complex data. A neuron receives input from one or more other neurons, performs computation on that input, and then transmits the output to one or more other neurons. One of the most significant benefits of neural networks is their ability to model and learn intricate, non-linear relationships in data. This makes them highly appropriate for a broad spectrum of applications such as speech and image recognition, natural language processing, and predictive modeling.

There exist various kinds of neural networks, each with its specific strengths and weaknesses. One popular type of neural network is the multilayer perceptron (MLP), which comprises a small amount layers of neurons that are linked by weighted edges. MLPs can be used for regression and classification problems and trained using backpropagation, a method that adjusts the weights of the edges to minimize the difference between the anticipated and actual outputs. Another type of neural network is the deep neural network (DNN), which comprises numerous layers of neurons. DNNs are frequently employed for intricate tasks such as image and speech recognition, and they require a substantial amount of data and computational resources to train. In this thesis, the difference between MLP and DNN was made evident. MLP is a simpler type of neural network that is more straightforward to interpret and train, while the DNN is a more intricate and powerful type of neural network that necessitates more data and resources to train.

Multilayer Perceptron

The Multilayer Perceptron (MLP) is a neural network comprising multiple layers of neurons that includes an input layer, one or more hidden layers, and an output layer. The neurons in each layer are entirely connected to the adjacent layers, and each neuron applies a non-linear activation function to the weighted sum of its inputs.

One of the most notable advantages of MLPs is their ability to learn and model intricate, non-linear relationships in data. This makes them highly suitable for a wide range of applications, including speech and image recognition, natural language processing, and predictive modeling.MLPs can be trained using a technique known as backpropagation, which involves iteratively modifying the weights of the edges between neurons in the network to minimize the discrepancy between the predicted and actual outputs. The backpropagation algorithm employs the chain rule of calculus to compute the gradient of the error function concerning the weights of the network and subsequently updates the weights using a gradient descent algorithm. In contrast to Deep Neural Networks (DNNs), MLPs typically have fewer layers and are more straightforward to train. However, they can still be used to model complex relationships in data and attain high levels of precision in various machine learning tasks. MLPs are frequently utilized as a baseline model for comparison with DNNs, which are more intricate and powerful.



(Figure 24: Multilayer Perceptron structure. Source: Researchgate.com, Simultaneous Evolution of Neural Network Topologies and Weights for Classification and Regression)

The decision to incorporate a Multilayer Perceptron (MLP) into the project was driven by the need to establish a baseline model that could be used to evaluate the performance of the Deep

Neural Network (DNN) model that was planned for later implementation. Initially, the MLP model was not expected to serve as the final predictive tool for the Web Application; however, the model showed decent performance. With a maximum number of optimization iterations set at 1500, the MAE was not as bad as expected reaching a value of 0.075, and the Rsquared increased to 0.851. These improvements compared to the linear models demonstrated the efficacy of the DNN models, which were expected to provide even better results in subsequent implementations.

Deep Neural Networks

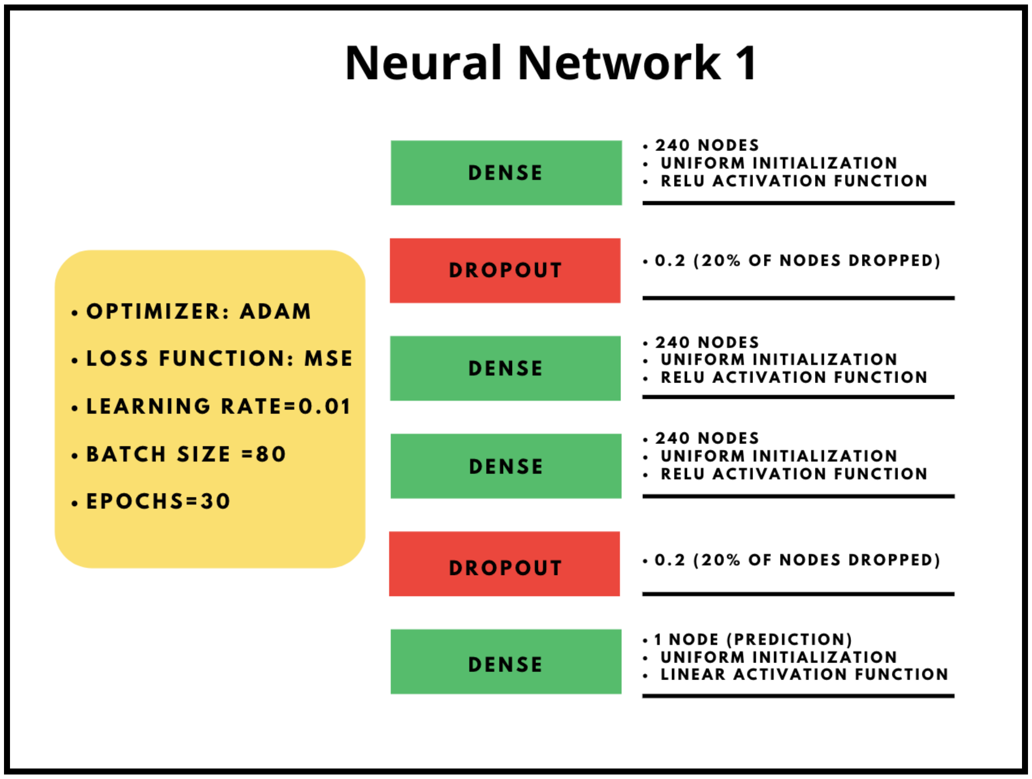
Deep Neural Networks (DNNs) are a type of neural network that consists of many layers of neurons, with each layer performing a specific computation on the input data. DNNs can have dozens or even hundreds of layers, making them much deeper and more complex than traditional neural networks like the Multilayer Perceptron (MLP). One of the main advantages of DNNs is their ability to automatically learn hierarchical representations of the input data, from low-level features like edges and corners to high-level concepts like objects and scenes. This makes them well-suited for a wide range of applications, including image and speech recognition, natural language processing, and predictive modeling.

As MLP, DNNs can be trained using a technique called backpropagation but require much more computational resources to train and deploy with respect to the former. They also require careful tuning of various hyperparameters, such as the number of layers, the number of neurons in each layer, and the choice of activation functions. One of the main advantages of DNNs is their ability to automatically learn and extract features from the input data, which eliminates the need for manual feature engineering. This can lead to more accurate and robust models, as well as faster development times.

In this application of Machine Learning two Deep Neural Networks models were implemented. The first Neural Network that was built is a sequential model consisting of three hidden layers and an output layer. The first hidden layer contains 240 nodes, with a ReLU activation function and uses uniform weight initialization. ReLU stands for Rectified Linear Unit and it transforms negative values in the input to zero while passing on positive values. This function helps to add non-linearity to the model and has been found to improve the performance of neural networks in various applications.Uniform weight initialization is a technique used to randomly initialize the weights in the neural network. The goal of this technique is to ensure that the weights are initialized with small random values that are uniformly distributed. This technique helps to prevent the weights from becoming too large or too small and can prevent issues such as vanishing gradients or exploding gradients. The second and third hidden layers also contain 240 nodes each with ReLU activation functions. The output layer has a single node with a linear activation function.

The model incorporates dropout regularization with a probability of 0.2 and was compiled using the Adam optimizer with a learning rate of 0.010, which has been chosen after trying out many possible combinations for it. The Adam optimizer was chosen for its reduced sensitivity to the initial learning rate compared to other optimization algorithms,which makes it easier to choose a good learning rate for the model. This is because the adaptive learning rate helps to compensate for the choice of a suboptimal learning rate. The batch size (80) and number of epochs (30) were set to these values after many trials that evidenced the fact that increasing the batch size would just make the model slower in its training without any affection on the performances and increasing the number of epochs was meaningless given the not improved performances of the model setting it higher.

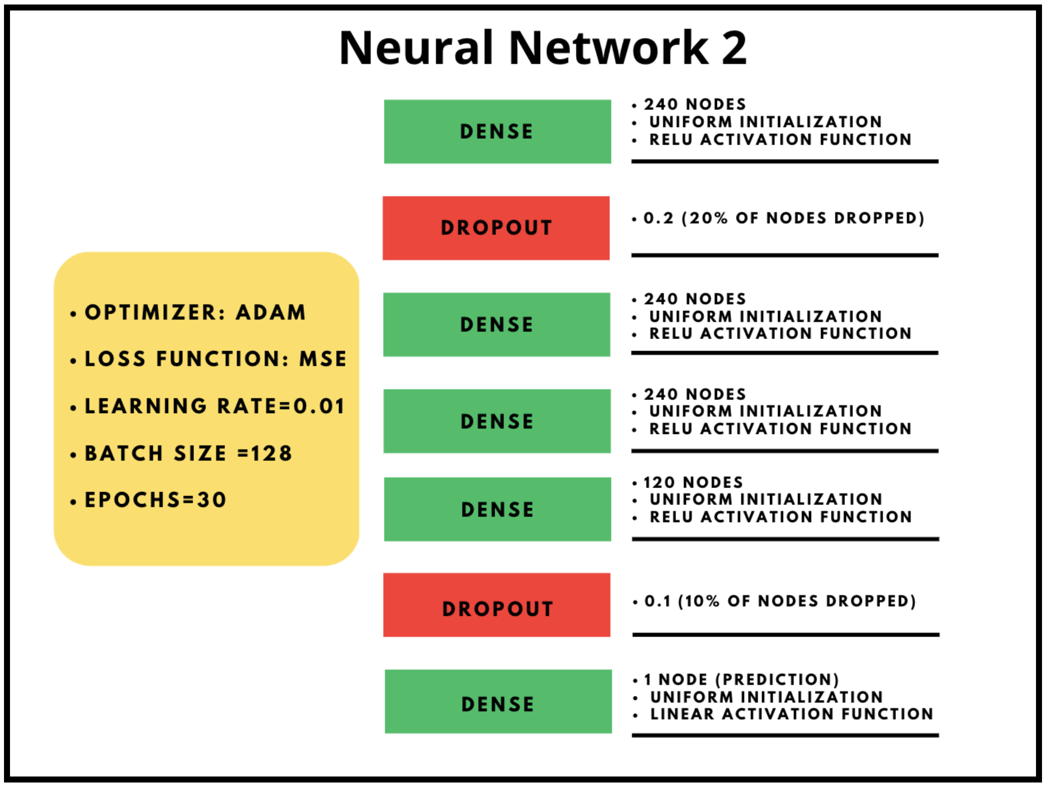
The performances of this first DNN consisted of a 0.071 MAE and 0.864 Rsquared which were quite close but didn't live up to those of the XGBoost model, therefore a second Neural Network was built.



(Figure 25: Structure of Neural Network 1)

The same principles were applied for the second Neural Network but little tweaks were made to the structure. The biggest change was the addition of a new Dense layer with the same uniform weight initialization and Relu activation function composed by 120 nodes. Additionally, after many attempts trying removing the second Dropout layer, the performances of the model dropped a bit so the percentage of nodes ignored by the model in the backpropagation phase was reduced to 0.1 (10%). Finally the batch size was set to 128, a multiple of two, in such a way to best exploit the GPU of the local machine used to execute the code.

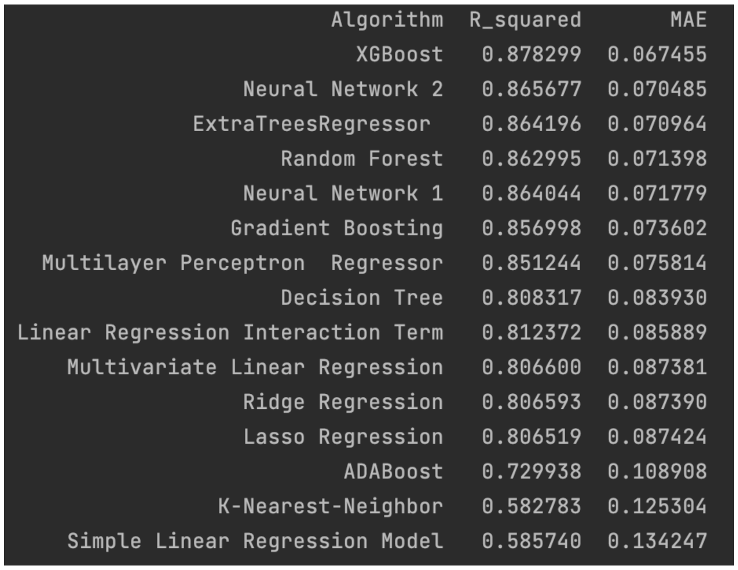
In this case the performances were quite good and almost close to those of the XGBoost model with a Rsquared of 0.865 and a MAE of 0.070. Even if this appeared to be the most suited ML algorithm to predict the energy level conveyed by music it wasn't the case since even after many attempts to optimize this appeared to be the best possible combination of parameters and still, it wasn't enough to overperform the XGBoost Tuned model.



(Figure 26: Structure of Neural Network 2)

6. Results Comparison

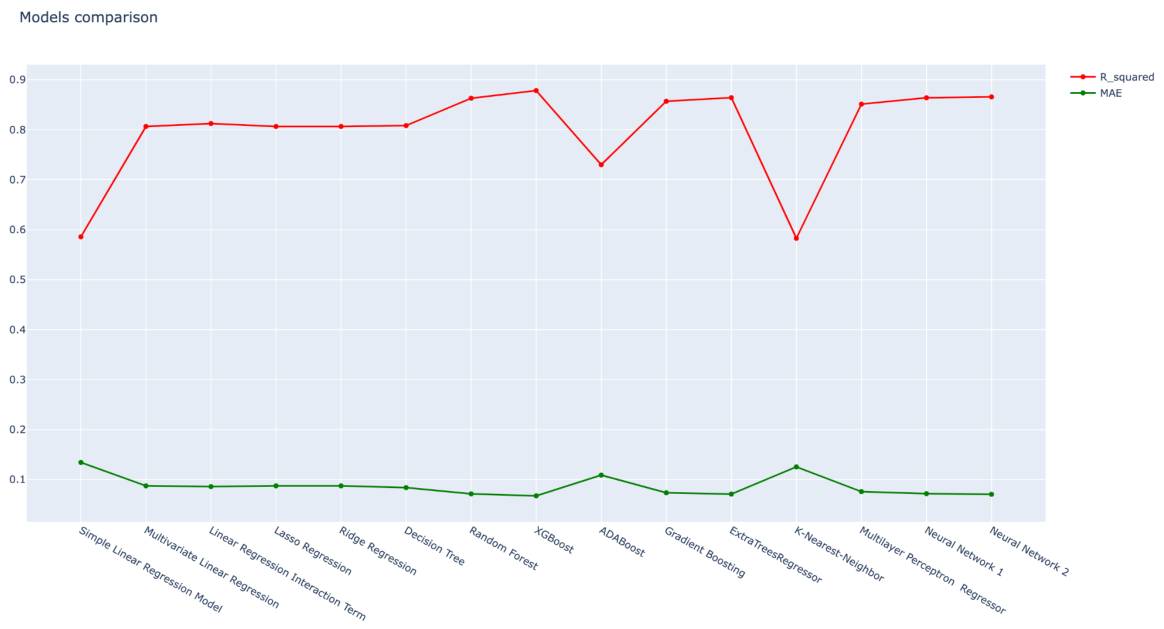
After trying out multiple machine learning models for regression, it was concluded that the XGBoost model was the most effective in predicting the energy level of music tracks in the given dataset. The XGBoost model outperformed all the other models, including the two neural networks created with TensorFlow and Keras, achieving an MAE of 0.067 and an Rsquared of 0.878 on the test set. The initial linear models were fundamental in underlining the magnitude of the relationships and associations between the target and independent variables suggesting that tempo, loudness, and spectral features were the most important features for predicting the energy level of a music track. Other models as ExtraTrees, Random Forest and Gradient Boosting proved to be efficient by scoring results not so far from those of the XGBoost model, while others such as KNN and ADABoost showed to be unsuited for such typology of data and task. The results of each model are shown in the table below:



```
                         Algorithm  R_squared       MAE
                           XGBoost   0.878299  0.067455
                  Neural Network 2   0.865677  0.070485
              ExtraTreesRegressor    0.864196  0.070964
                     Random Forest   0.862995  0.071398
                  Neural Network 1   0.864044  0.071779
                 Gradient Boosting   0.856998  0.073602
    Multilayer Perceptron  Regressor   0.851244  0.075814
                     Decision Tree   0.808317  0.083930
Linear Regression Interaction Term   0.812372  0.085889
    Multivariate Linear Regression   0.806600  0.087381
                  Ridge Regression   0.806593  0.087390
                  Lasso Regression   0.806519  0.087424
                          ADABoost   0.729938  0.108908
                K-Nearest-Neighbor   0.582783  0.125304
    Simple Linear Regression Model   0.585740  0.134247
```

(Figure 27: Table of models' performances ranked from the best to the worst)

The XGBoost model was selected as the primary predictor for the web application created as the final task of the thesis. Even if using a regression model for predicting the energy level of music tracks was found to be the most effective solution and with much greater predictive capabilities than those developed in earlier studies, the performances of the models could be improved with greater computational power. Due to the limited computing resources available on a simple local machine, hyperparameter tuning was restricted to a limited set of parameters given the lengthy execution times of code chunks.



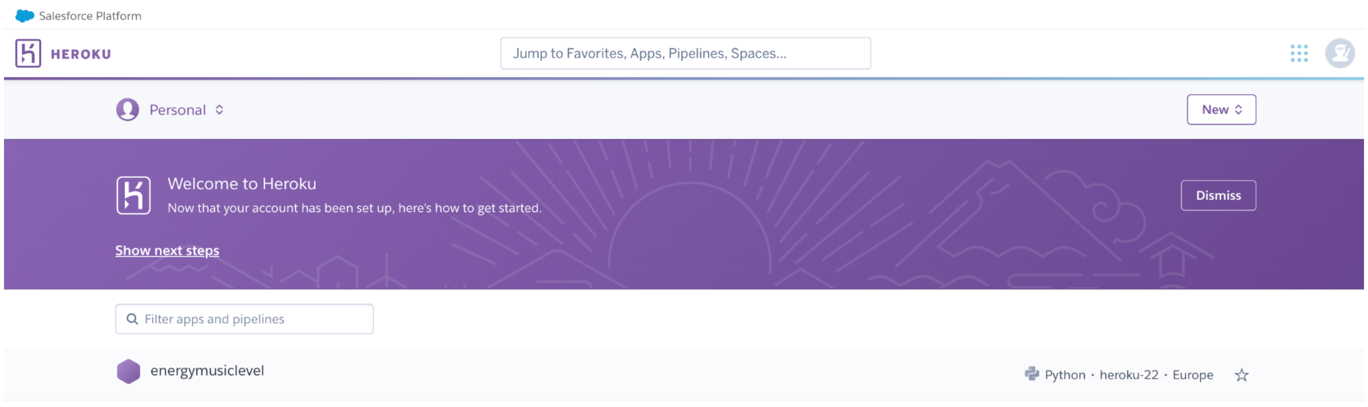
(Figure 28: Models performances. The green line indicates the trend of the MAE for each model while the red line the Rsquared. Generally these two present a negative correlation)

7. Web Application Implementation

Web application development has become an integral part of modern technology. It involves creating dynamic websites that enable users to interact with the application via a graphical user interface (GUI). Streamlit is a popular tool for building web applications, as it is a Python-based framework that makes it easy to create interactive applications with minimal integrative coding.

In this instance, the use of Streamlit was required to develop a simple web application that allows the user to input the name of a song. The application's goal is to to retrieve the track's features from the Spotify dataset and predicts its energy level using the previously trained XGBoost regression model. Once the prediction has been made by the model, the predicted energy level is displayed, along with the actual energy level of the song and some recommended tracks with similar energy levels.

The application was initially developed locally on the machine and accessed through the browser's localhost. It was subsequently deployed online using Heroku, a freemium cloud platform that enables developers to deploy, manage, and scale web applications quickly and easily.


Salesforce Platform
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
New
Dismiss
Show next steps
Filter apps and pipelines

Welcome to Heroku

Now that your account has been set up, here's how to get started.

energymusiclevel

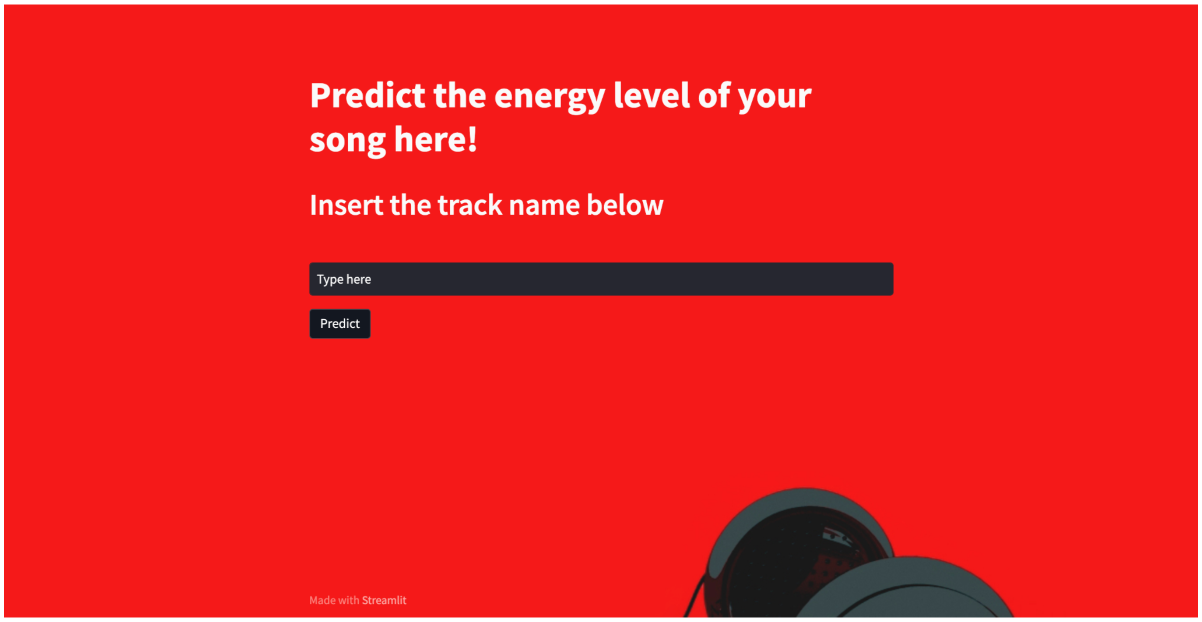
Python · heroku-22 · Europe

(Figure 29: Heroku Cloud Platform. Source: https://dashboard.heroku.com)

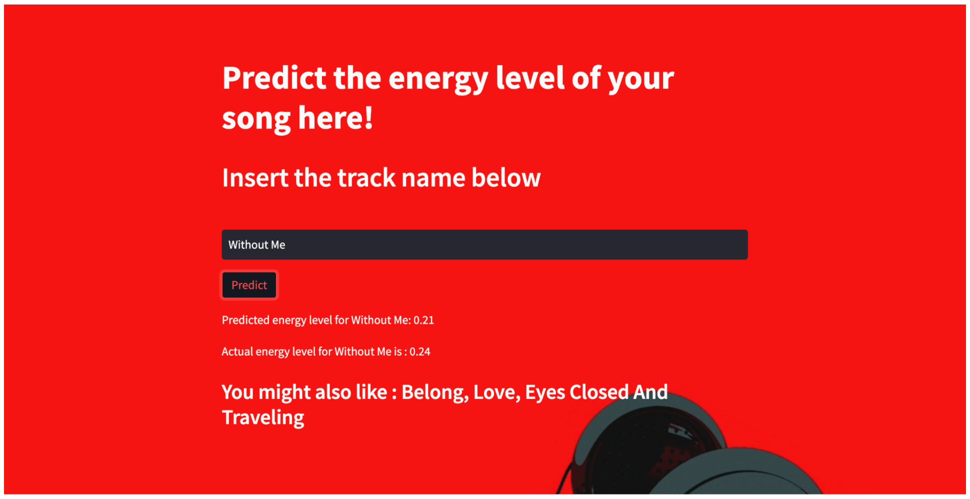
Deploying a web application is not so intuitive since many files have to be provided in order to set up the virtual environment. Firstly it is necessary to create a github repository and upload the necessary files in it. The files that need to be uploaded for creating a Machine Learning WebAPP on heroku are the following:

- Runtime.txt (A text file containing the python version of the virtual environment to set up on Heroku)
- Procfile (A shell script file which specifies which app to run and on which port to run the application)
- Setup.sh (A shell script file containing all the information related to the setup and build of the Heroku environment)
- Streamlit_fhs.py (A python file containing the script for the creation of the WebApp)
- XGBoostTuned.sav (A pickle file containing the pretrained XGBoost model)

After providing all the needed files the app, whose links for accessing it and its code are in the references, was deployed and launched online. These are the interfaces displayed by clicking the link of the app:


Predict the energy level of your song here!
Insert the track name below
Type here
Predict
Made with Streamlit

(Figure 30: Web Application Input Screen)


Predict the energy level of your song here!

Insert the track name below

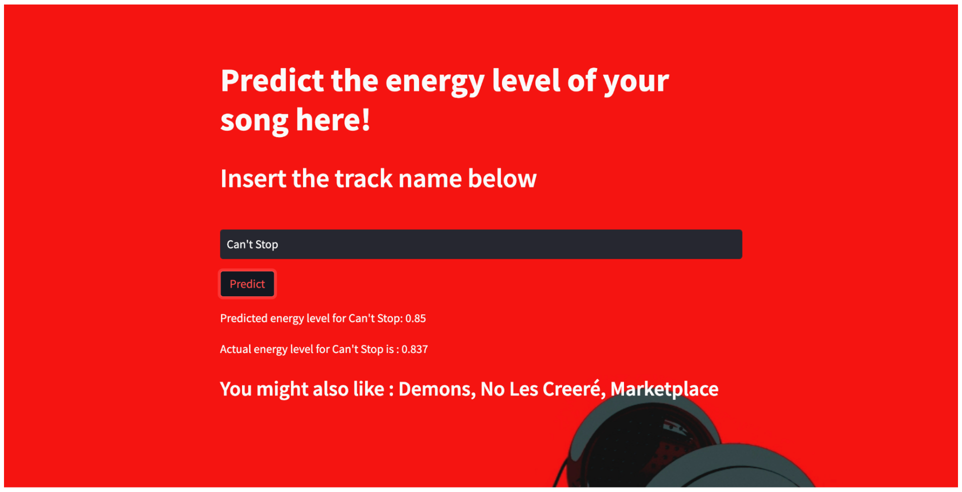
Without Me

Predict

Predicted energy level for Without Me: 0.21

Actual energy level for Without Me is : 0.24

You might also like : Belong, Love, Eyes Closed And Traveling


Predict the energy level of your song here!

Insert the track name below

Can't Stop

Predict

Predicted energy level for Can't Stop: 0.85

Actual energy level for Can't Stop is : 0.837

You might also like : Demons, No Les Creeré, Marketplace

(Figure 31: Web Application Displayed Results)

Once the user enters the track name into the Web Application and the song is found in the dataset, the XGBoost Regressor predicts the energy level of the track. The Web Application then displays the predicted value, the actual value, and the names of three songs that have a similar energy level to the submitted track, regardless of the genre.

This Recommendation system was conceived to suggest tracks with energy levels that are most similar to the ones anticipated by the machine learning regressor, ensuring a maximum energy value difference of 0.02 for tracks that have comparable energy levels. This feature is included to allow users to broaden their musical preferences and discover new songs that they may not have considered before.

8. Conclusions

The goal of this study was to predict the energy level of songs in the Spotify dataset using regression machine learning models and to develop a Web Application in order to display a potential application of this machine learning task in the music industry. The results of the models showed a marked improvement over previous studies, indicating that audio features alone are sufficient for accurately predicting energy levels. Through data visualization, exploration, and cleaning, the most correlated variables were identified, and several regression models were evaluated, with the XGBoost model performing the best. This Boosting model that, as specified earlier, was the best overall and was trained with a learning rate of 0.05 produced a 0.067 mean absolute error (MAE) way lower than the Random Forest model of Zhang et al. (2020) which was capable of attaining a 0.119 MAE by using a hybrid approach combining audio features and lyrics to predict the energy level of a music track. This finding effectively reinforces the central argument of the thesis that a superior regression algorithm can be developed for predicting the energy level of songs in the Spotify dataset by exclusively utilizing audio metrics and features.

The results of this research have practical applications in the music industry, especially in improving the classification and recommendation of songs based on their energy levels. Incorporating energy level as a key parameter in collaborative filtering can potentially enhance the accuracy and relevance of the recommendations, leading to improved user satisfaction and engagement. Another possible approach is to create a collaborative filtering recommendation system that identifies tracks with similar energy levels and uses them as a basis for generating recommendations. This could enable users to explore different genres they have not previously considered and expand their musical preferences. For instance, if a user typically listens to songs with high energy levels, the system could suggest other tracks with similar energy levels, but from different genres, such as rock or pop. This approach could introduce users to new songs and genres they may not have discovered otherwise, and help them discover new favorites. To wrap it up, incorporating energy levels into collaborative filtering can improve the music listening experience for users and provide a valuable tool for the music industry to better understand their audience's preferences and behaviors.

However, there are limitations to this study, including the specific dataset used, which may not generalize well to other datasets with different features, and computational power limitations, which hindered further fine-tuning of the models or training of neural networks.

In conclusion, this study provides a foundation for further investigation into the prediction of energy levels of songs and their impact on music analysis and recommendation by using energy levels as one of the parameters of collaborative filtering. The potential of AI in the music industry is significant, and there is a need to develop it further to achieve the best outcomes and progress into the future. In the words of Eleanor Roosevelt, "The future belongs to those who believe in the beauty of their dreams."

9. References

Baturynska, Ivanna & Martinsen, Kristian. (2021). Prediction of geometry deviations in additive manufactured parts: comparison of linear regression with machine learning algorithms. Journal of Intelligent Manufacturing. 32. 10.1007/s10845-020-01567-0.

Bonnin, G., Cornelis, O., & Vanhoof, K. (2019). Music popularity prediction using audio and user features in a random forest framework. In Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR 2019) (pp. 444–451).

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Choi, K., Fazekas, G., & Sandler, M. (2017). Automatic tagging using deep convolutional neural networks. Journal of the Acoustical Society of America, 141(5), EL409-EL415.

Gupta, P., & Dharmadhikari, S. (2021). Audio classification using deep learning techniques. In Proceedings of the 3rd International Conference on Intelligent Computing and Control Systems (ICICCS 2021) (pp. 146–151). IEEE.

Han, H., Park, H., & Lee, S. (2021). Convolutional neural network based popularity prediction model for music streaming services. Multimedia Tools and Applications, 80(1), 905–918. https://doi.org/10.1007/s11042-020-09988-4

Han, Y., & Seo, D. (2018). Predicting music popularity on social media by incorporating audio and text information. IEEE Transactions on Multimedia, 20(8), 2025-2035.

Hsieh, C. K., Yang, W. H., Lee, S. Y., & Lin, Y. H. (2019). Music genre classification using convolutional neural networks. In Proceedings of the 2019 International Conference on Audio, Language and Image Processing (ICALIP 2019) (pp. 1–5). IEEE.

Hu, X., & Downie, J. S. (2018). Exploring music popularity using enhanced metadata features. In Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018) (pp. 570–576).

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.

Kim, J. H., Choi, J., & Lee, J. H. (2019). A decision tree-based analysis of the effects of seasonal variation on music consumption. International Journal of Information Management, 44, 80-88.

Kwon, J., Lee, J. H., & Lee, J. H. (2019). Music genre classification using convolutional neural networks and transfer learning. Applied Sciences, 9(18), 3827.

McFee, B., Bertin-Mahieux, T., & Ellis, D. P. W. (2015). Learning content popularity from free-text metadata. In Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR 2015) (pp. 303–309).

Min, Huan & Luo, Xiong-Lin. (2016). Calibration of soft sensor by using Just-in-time modeling and AdaBoost learning method. Chinese Journal of Chemical Engineering. 24. 10.1016/j.cjche.2016.05.015.

Ngiam, J., Chen, C., & Teo, C. (2020). Predicting music popularity using user behavior features. In Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR 2020) (pp. 507–514).

Pons, J., Serra, X., & Müller, M. (2017). Towards a music popularity prediction model. In Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017) (pp. 140–146).

Chen, C. W., & Chen, M. S. (2016). Collaborative filtering for music recommendation: a review. In Proceedings of the 7th ACM international conference on multimedia systems (pp. 80-85).

Çakmak, E., & Yılmaz, O. T. (2019). A hybrid collaborative filtering and content-based music recommendation system. Journal of Ambient Intelligence and Humanized Computing, 10(1), 159-170.

Zhang, X., Liu, W., & Ma, L. (2018). An improved collaborative filtering recommendation algorithm for music service. Journal of Physics: Conference Series, 1069(1), 012007.

Tran, Quang-Hieu & Nguyen, Hoang & Bui, Xuan-Nam. (2022). Novel Soft Computing Model for Predicting Blast-Induced Ground Vibration in Open-Pit Mines Based on the Bagging and Sibling of Extra Trees Models. Computer Modeling in Engineering & Sciences. 134. 1-20. 10.32604/cmes.2022.021893.

Wang, Yuanchao & Pan, Z. & Zheng, J. & Qian, L. & Mingtao, Li. (2019). A hybrid ensemble method for pulsar candidate classification. Astrophysics and Space Science. 364. 10.1007/s10509-019-3602-4.

Wu, T., Zhang, W., Jiao, X., Guo, W., & Hamoud, Y. A. (2020). Comparison of five Boosting-based models for estimating daily reference evapotranspiration with limited meteorological variables. Plos One, 15(6), e0235324.

Zhang, S., Wang, J., Li, Y., Li, H., Li, P., & Guo, J. (2020). A Hybrid Model for Music Energy Estimation Based on Audio and Lyrics. IEEE Access, 8, 41116-41126.

Zhang, Y., & Li, S. (2019). Feature selection for music genre classification based on deep neural networks. IEEE Access, 7, 155182–155191.

Chamorro-Premuzic, T., Fagan, P., & Furnham, A. (2009). Personality and music: Can traits explain how people use music in everyday life? PLoS ONE, 4(11), e7793. doi:10.1371/journal.pone.0007793 https://energymusiclevel.herokuapp.com/

https://github.com/RobertoColangelo/Spotify-Music-Energy-Level-Web-APP

https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset

https://en.wikipedia.org/wiki/Correlation

https://pandas.pydata.org/

https://streamlit.io/

https://www.heroku.com/

https://scikit-learn.org/stable/

https://www.jetbrains.com/pycharm/

https://www.mygreatlearning.com/blog/gridsearchcv/#:~:text=Grid%20search%20is%20a%20method,learning%20rate%20of%20the%20optimiser).

https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d

https://medium.com/analytics-vidhya/mae-mse-rmse-coefficient-of-determination-adjusted-r-squared-which-metric-is-better-cd0326a5697e

https://en.wikipedia.org/wiki/Linear_regression

https://statisticsbyjim.com/regression/interaction-effects/

https://builtin.com/data-science/l2-regularization

https://c3.ai/glossary/data-science/tree-based-models/#:~:text=What%20are%20Tree%2DBased%20Models,or%20value%20of%20a%20home.

https://pro.arcgis.com/en/pro-app/latest/tool-reference/geoai/how-extra-tree-classification-and-regression-works.htm

https://towardsdatascience.com/decision-trees-and-random-forests-df0c3123f991

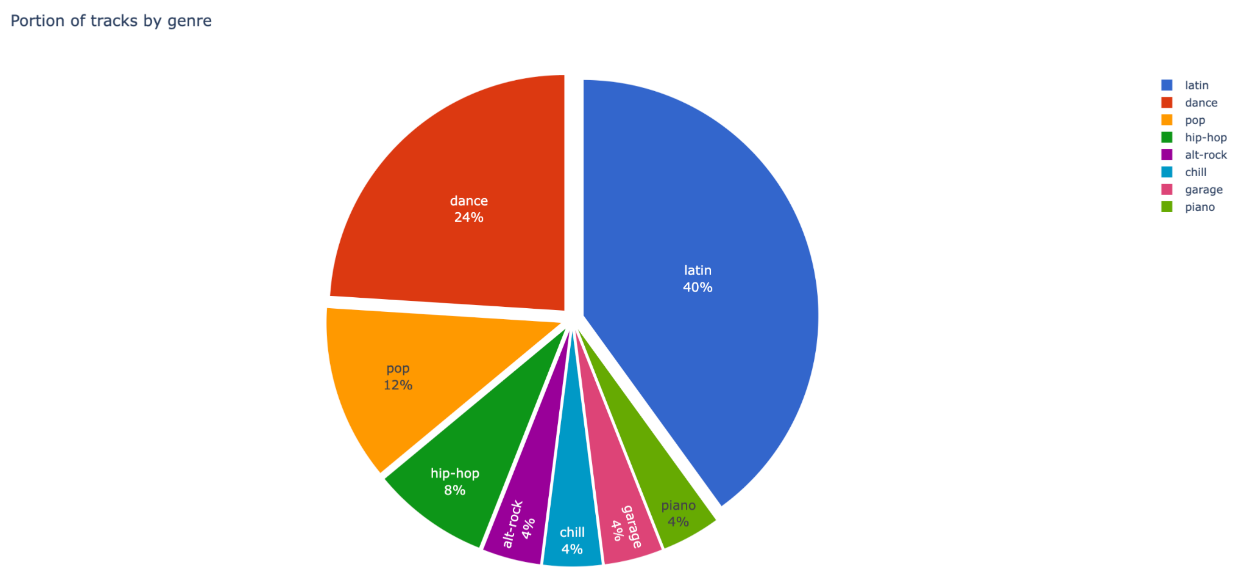
https://towardsdatascience.com/an-introduction-to-k-nearest-neighbours-algorithm-3ddc99883acd

https://www.baeldung.com/cs/mlp-vs-dnn

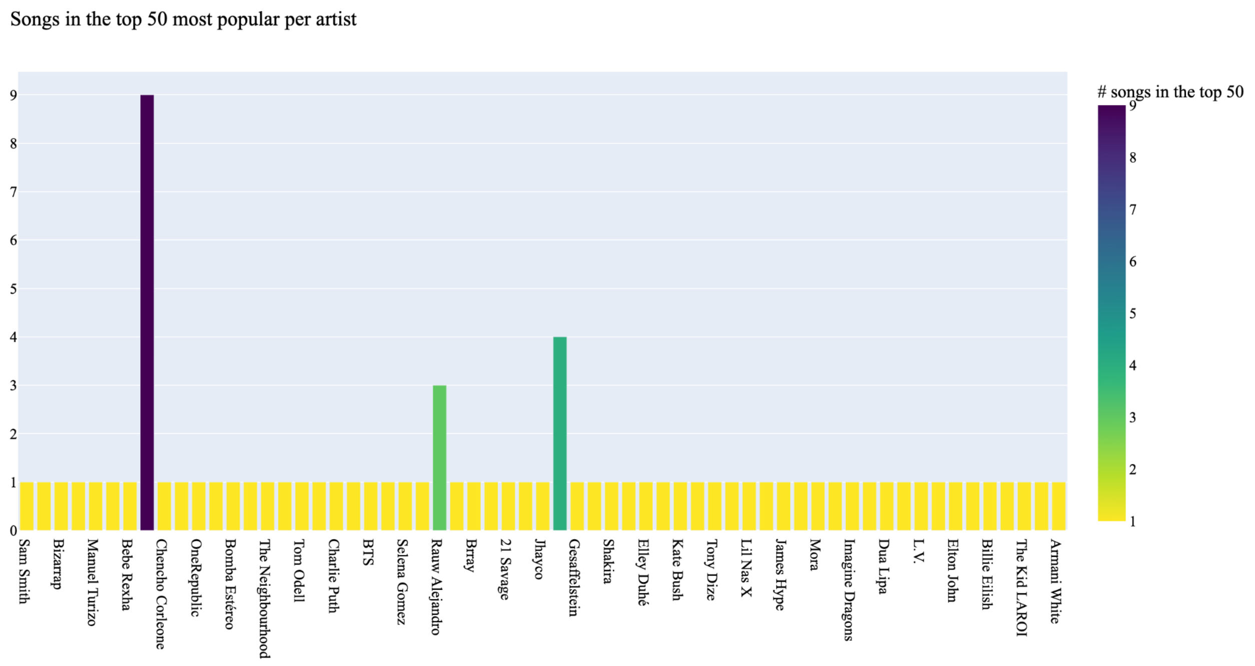
https://python.plainenglish.io/creating-an-awesome-web-app-with-python-and-streamlit-728fe100cf7

https://thaddeus-segura.com/random-forest

10. Appendix



(Figure A: Pie chart of portion of tracks by genre in the top 25 most popular tracks)



(Figure B: Histogram of songs per artist in the top 50 most popular)

Summary

Music is a language without words, a means of expression that can communicate directly with our feelings. Bypassing our rational mind, it can evoke a feeling of nostalgia, a sense of longing, a rush of excitement or a sense of peace and contentment. Since ancient times, music has been an essential part of human culture and has been used for various purposes, such as entertainment, relaxation, and communication. In recent years, the advent of digital music streaming services such as Spotify has led to the availability of large music datasets. These datasets contain a wealth of information about music, including audio features, information about artists, and user listening behavior. One of the audio features provided by Spotify is the energy level of a music track, which represents the perceived intensity and activity of the track. The energy level is a crucial factor in determining the mood and emotional impact of a song and can influence users' listening behavior.

Nowadays collaborative filtering is a popular recommendation system technique used by many music streaming services such as Spotify, to suggest new songs to their users based on their listening history and preferences. However, one limitation of collaborative filtering is that it may not always consider the energy level of songs as an important factor in generating recommendations and this is an unexploited element that might allow users to get to know more music and expand their music tastes and knowledge. Given such absence in the recommendation system sector, the aim of this thesis is that of developing an AI algorithm capable of predicting the energy level of track, improving the accuracy of already developed algorithms by using only the metadata provided by Spotify, and providing a new method for such systems consisting in incorporating the concept of energy levels into collaborative filtering. The objective is that of improving music recommendation systems by identifying songs with comparable energy levels and using them as the basis for generating recommendations. For instance, if a user often listens to high-energy songs, the system can suggest other tracks with similar energy levels. This can be accomplished by utilizing regression Machine Learning techniques, which involve analyzing the association between audio metrics such as tempo, loudness, and acousticness and a target variable that wants to be estimated, the energy level of a song. By integrating energy levels into collaborative filtering, the user experience can be enhanced by providing more personalized and

relevant recommendations. However, to achieve this, an accurate machine learning regression algorithm capable of predicting the energy level of a music track must be implemented.

Over the past years, several attempts have been made to predict the energy level of a music track. These studies, conducted by Han et al. (2018), Kim et al. (2019), and Zhang et al. (2020), involve regression or classification models that use different types of data such as album cover images, lyrics, and audio features/metrics. Despite previous attempts to predict the energy level of music tracks using different types of data such as album cover images, lyrics, and audio features/metrics, this research makes exclusively use of audio features and metrics from a dataset of 114,000 songs on Spotify. The main technical goal is be that of identifying the most effective algorithm for the purpose of developing a web application that can predict the energy level of a track when inputting its name and recommend tracks with similar energy levels.

To this end, two types of models were considered: regression models based on traditional machine learning algorithms and deep learning algorithms. In order to build the best regression model for the prediction several steps were followed including:

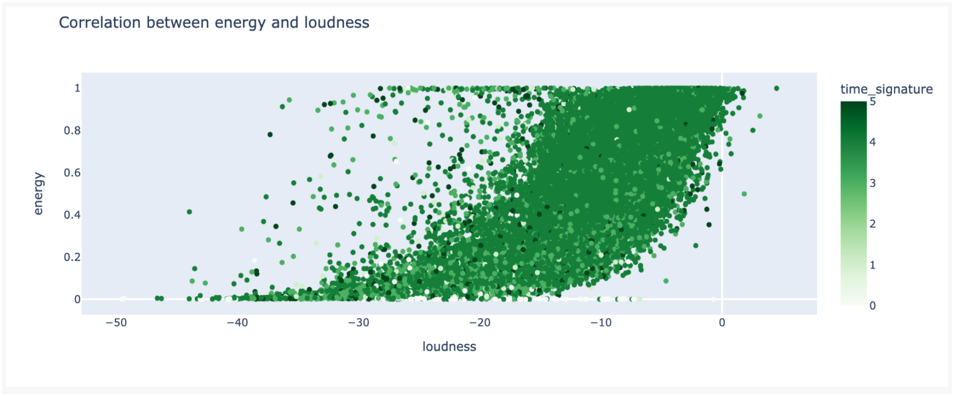
- Libraries import
- Data Cleaning
- Data Exploration /Visualization
- Data Preprocessing
- Feature Selection
- Modeling and hyperparameters Tuning
- Models comparison
- Model deployment

This research has significant implications for the development of music analysis and recommendation systems that can take into account the emotional impact of music on listeners. Additionally, differently from previous studies and researches which focused on the same task but by employing the Spotify audio metric in conjunction with other sources such as lyrics and images, and achieved as best results 80.3% accuracy with classification algorithms or 0.11 mean absolute error (MAE) with regression algorithms on the test set, the approach of this thesis aims to revolutionize the application of machine learning to music and its features by focusing solely on metadata to develop the regression models. Furthermore, the research provides a further secondary contribution to the literature since it involves the development of a web application system that displays predicted and actual values of energy for each track, as well as recommended tracks based on a subsequent affinity system, to provide an idea of the potential exploitation of such an algorithm in the music industry.

In the current study, secondary data was used, as the Spotify Tracks Dataset by Maharshipandya was downloaded in the form of a CSV file from a Kaggle repository. This dataset was originally obtained from the Spotify Web API in November 2022 and contains a comprehensive collection of music tracks and associated metadata. Specifically, the dataset comprises 114,000 songs, each of which includes 20 features, encompassing track, artist, and audio features. These features are categorized into 5 categorical and 15 numeric variables whose detailed descriptions can be found in the subsequent paragraph.

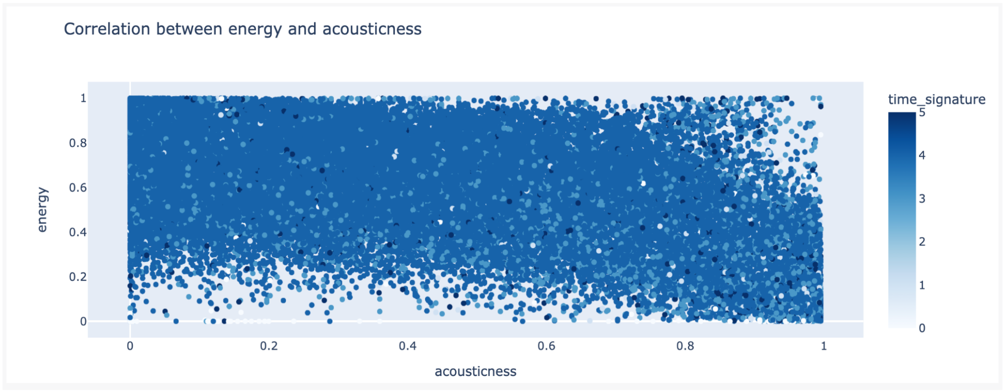
In order to better understand the relationship between its various features, data exploration and visualization techniques were employed, such as summary statistics and graphical representations. By using summary statistics and various graphical techniques, the dataset's features and their interplay were examined, providing insights into the central tendency, variability, and distribution of the data, and highlighting any correlations or patterns that may exist. One of the key features of the dataset is the energy level of a music track, which is an essential factor in determining the mood and emotional impact of a song.

Among all the information that was extrapolated from the plots made, the most important ones were that the relationship between the "energy" level and the "loudness" of a track, which were hypothesized to be two closely related variables, proved to be positively correlated with a coefficient of 0.76, indicating that as the energy level of a track increases, so does its loudness. This finding was already suggested by Chamorro-Premuzic, T., Fagan, P., & Furnham, A. (2009) who highlighted the importance of considering the role of loudness in music perception and its potential impact on listener experience. This high value of positive correlation indicates that there surely is a tendency for songs with louder tones to convey more energy.



(Figure : correlation between energy and loudness)

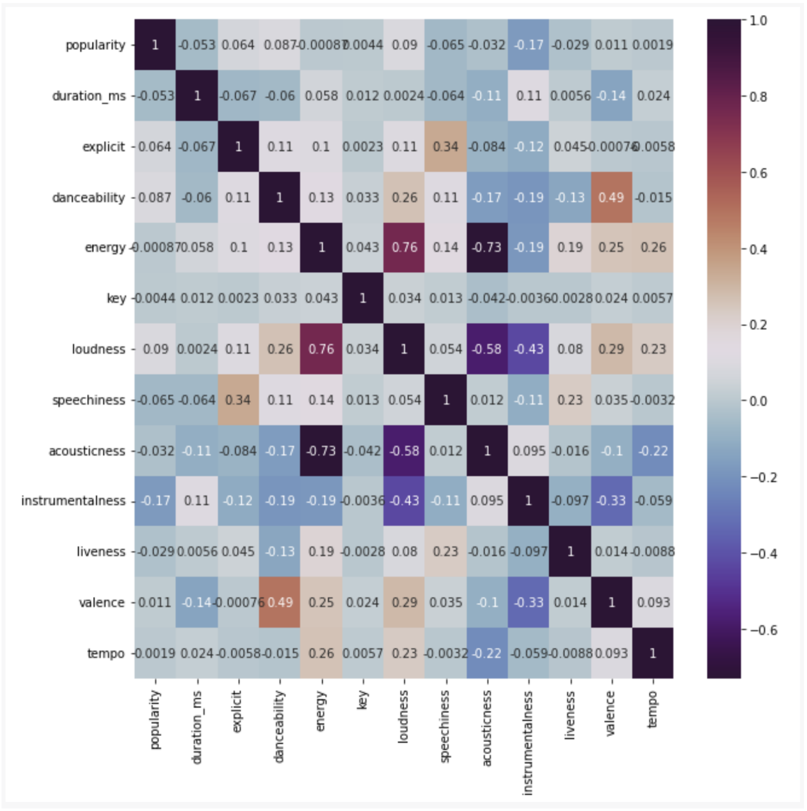
Subsequently, a second plot was generated to examine the relationship between the "energy" level and the "acousticness" score of a track. This plot revealed a negative correlation with a coefficient of -0.73, indicating that as the acousticness of a track increases, its energy level tends to decrease. This finding suggests that music tracks with higher acousticness scores may convey less energy overall. This contradicts what said by another study by Han et al. (2018) examined the relationship between acousticness and energy levels in music tracks who found that tracks with higher energy levels tend to have higher levels of acousticness, while tracks with lower energy levels tend to have lower levels of acousticness. Actually it seems like, people perceive as less energetic those songs whose level of acousticness is higher (e.g. classic music) and more energetic those with lower levels of acousticness (e.g. death metal).



(Figure : correlation between energy and acousticness)

After generating a correlation matrix, the variables that were found to be most correlated with the target variable were identified. These variables were ordered according to their correlation coefficients, and the results were as follows:

1. Loudness → High positive correlation (0.76)
2. acousticness → High negative correlation (-0.73)
3. tempo → Low positive correlation (0.26)
4. valence → Low positive correlation (0.25)
5. liveness → Low positive correlation (0.19)



(Figure : Correlation Matrix)

The second step in the research was the modeling and models performances comparison. In this research, two evaluation metrics were utilized to assess the performance of the models., namely the Mean Absolute Error (MAE) and the Rsquared (R2) coefficient. Evaluation metrics play a crucial role in machine learning as they enable the measurement of the performance of different models and facilitate the comparison of their accuracy.

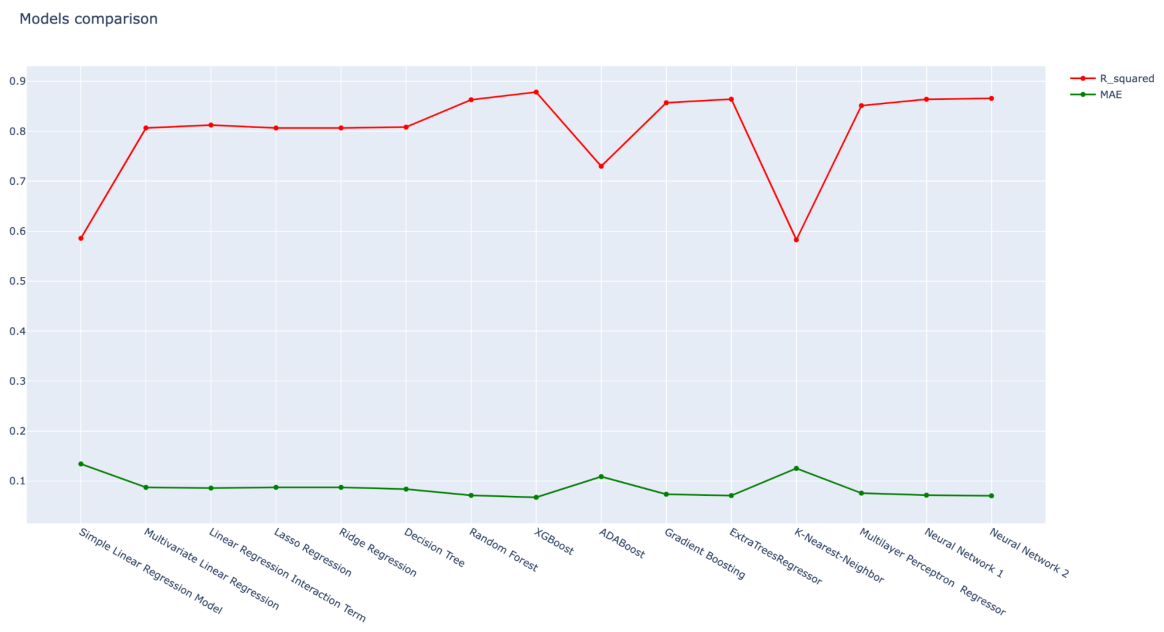
The decision to use both MAE and R2 as evaluation metrics in this study was made to provide a more comprehensive evaluation of the models. MAE was chosen to assess the accuracy of the models, while R2 was used to provide an overall measure of how well the models fit the data. This approach was chosen to enable comparison with previous research by Zhang et al. (2020) and Kim et al. (2019), where MAE was commonly used to evaluate accuracy. Generally by using more metrics, a more complete understanding of the performance of the models can be achieved. The models were first evaluated using MAE, and then the overall performance was compared using R2. This method enabled the identification of the models with the best balance between accuracy and fit.

A second essential aspect of the modeling part, besides the choice of the evaluation metrics, was Hyperparameters tuning. Hyperparameter tuning is a critical process in machine learning that involves selecting the optimal set of hyperparameters for a given model. In this study, hyperparameter tuning was achieved using the GridSearchCV method, which exhaustively searches over a predefined grid of hyperparameters to find the combination that results in the best performance. The hyperparameters were used to identify the most accurate model fit for this specific type of dataset by selecting among different combinations of grid values, such as alphas in Lasso and Ridge regression or the number of estimators, learning rate, and tree depth in tree-based methods.

The algorithms used in this study were categorized into three classes: Linear models, Tree-based models, and Neural network-based models. Each of these models has its advantages and disadvantages. Linear models are relatively simple and interpretable but may not capture complex nonlinear relationships while Tree-based models can capture complex nonlinear relationships, but may overfit the data (Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, 2013). On the other hand, Neural network-based models are capable of modeling complex nonlinear relationships but can be pretty challenging to interpret and require a significant amount of data to train effectively.

After trying out multiple machine learning models for regression, it was concluded that the fine-tuned XGBoost model was the most effective in predicting the energy level of music tracks in the given dataset. The XGBoost model outperformed all the other models, including the two neural networks created with TensorFlow and Keras, achieving an MAE of 0.067 and an Rsquared of 0.878 on the test set, a much better performance compared to the Random Forest model of Zhang et al. (2020) which was capable of attaining a 0.119 MAE by using a hybrid approach combining audio features and lyrics to predict the energy level of a music track. This finding effectively reinforces the central argument of the thesis that a superior regression algorithm can be developed for predicting the energy level of songs in the Spotify dataset by exclusively utilizing audio metrics and features.

The initial linear models were fundamental in underlining the magnitude of the relationships and associations between the target and independent variables suggesting that tempo, loudness, and spectral features were the most important features for predicting the energy level of a music track. Other models as ExtraTrees, Random Forest and Gradient Boosting proved to be efficient by scoring results not so far from those of the XGBoost model, while others such as KNN and ADABoost showed to be unsuited for such typology of data and task.

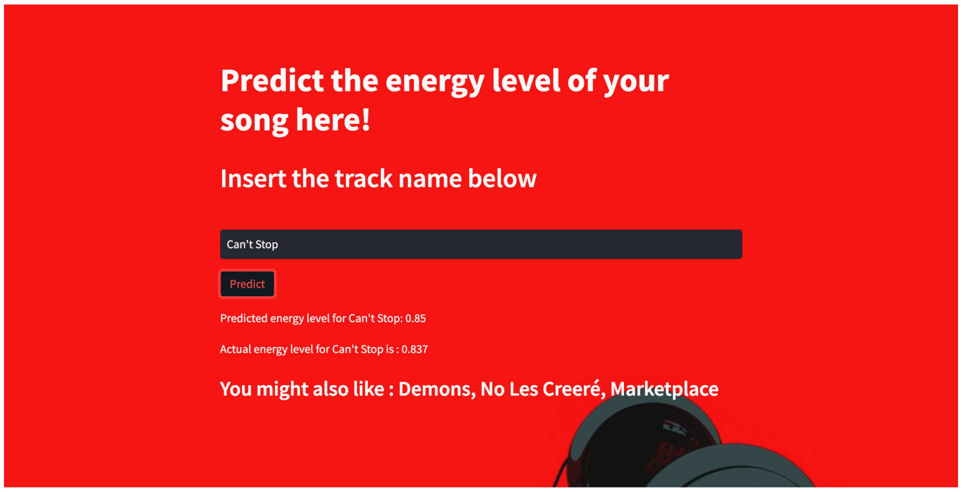


(Figure: Models performances. The green line indicates the trend of the MAE for each model while the red line the Rsquared. Generally these two present a negative correlation)

The XGBoost model was selected as the primary predictor for the web application created as the final task of the thesis. Even if using a regression model for predicting the energy level of music tracks was found to be the most effective solution and with much greater predictive capabilities than those developed in earlier studies, the performances of the models could be improved with greater computational power. Due to the limited computing resources available on a simple local machine, hyperparameter tuning was restricted to a limited set of parameters given the lengthy execution times of code chunks.

Once the machine learning algorithm was selected the Web application was developed on heroku.com. The Web application works in the following way: the user enters the track name into the Web Application and the song is found in the dataset, the XGBoost Regressor predicts the energy level of the track. The Web Application then displays the predicted value, the actual value, and the names of three songs that have a similar energy level to the submitted track, regardless of the genre.

This Recommendation system was conceived to suggest tracks with energy levels that are most similar to the ones anticipated by the machine learning regressor, ensuring a maximum energy value difference of 0.02 for tracks that have comparable energy levels. This feature is included to allow users to broaden their musical preferences and discover new songs that they may not have considered before.


Predict the energy level of your song here!

Insert the track name below

Can't Stop

Predicted energy level for Can't Stop: 0.85

Actual energy level for Can't Stop is : 0.837

You might also like : Demons, No Les Creeré, Marketplace

(Figure: Web Application Displayed Results)

In conclusion, the goal of this study was to predict the energy level of songs in the Spotify dataset using regression machine learning models and to develop a Web Application in order to display a potential application of this machine learning task in the music industry. The results of the models showed a marked improvement over previous studies, indicating that audio features alone are sufficient for accurately predicting energy levels. Through data visualization, exploration, and cleaning, the most correlated variables were identified, and several regression models were evaluated, with the XGBoost model performing the best. This Boosting model that, as specified earlier, was the best overall and was trained with a learning rate of 0.05 produced a 0.067 mean absolute error (MAE) way lower than the Random Forest model of Zhang et al. (2020) which was capable of attaining a 0.119 MAE by using a hybrid approach combining audio features and lyrics to predict the energy level of a music track. This finding effectively reinforces the central argument of the thesis that a superior regression algorithm can be developed for predicting the energy level of songs in the Spotify dataset by exclusively utilizing audio metrics and features.

The results of this research have practical applications in the music industry, especially in improving the classification and recommendation of songs based on their energy levels. Incorporating energy level as a key parameter in collaborative filtering can potentially enhance the accuracy and relevance of the recommendations, leading to improved user satisfaction and engagement. Another possible approach is to create a collaborative filtering recommendation system that identifies tracks with similar energy levels and uses them as a basis for generating recommendations. This could enable users to explore different genres they have not previously considered and expand their musical preferences. For instance, if a user typically listens to songs with high energy levels, the system could suggest other tracks with similar energy levels, but from different genres, such as rock or pop. This approach could introduce users to new songs and genres they may not have discovered otherwise, and help them discover new favorites. To wrap it up, incorporating energy levels into collaborative filtering can improve the music listening experience for users and provide a valuable tool for the music industry to better understand their audience's preferences and behaviors.

However, there were some limitations to this study, including the specific dataset used, which may not generalize well to other datasets with different features, and computational power limitations, which hindered further fine-tuning of the models or training of neural networks.

In conclusion, this study provides a foundation for further investigation into the prediction of energy levels of songs and their impact on music analysis and recommendation by using energy levels as one of the parameters of collaborative filtering. The potential of AI in the music industry is significant, and there is a need to develop it further to achieve the best outcomes and progress into the future. In the words of Eleanor Roosevelt, "The future belongs to those who believe in the beauty of their dreams."