Leopold-Franzens-University Innsbruck

Institute of Computer Science
Databases and Information Systems

Analyzing the Characteristics of Music Playlists using Song Lyrics and Content-based Features

Master Thesis

Stefan Wurzinger, BSc

supervised by
Dr. Eva Zangerle, Michael Tschuggnall, PhD
Univ.-Prof. Dr. Günther Specht

Innsbruck, October 2, 2017

To my loving family:
mom, dad and my brother Marc.

Abstract

In the recent years, music streaming services evolved which facilitate new research possibilities in the field of music information retrieval. Publicly available user-generated playlists offered by the music streaming platform Spotify allow to disclose properties of tracks shared within a playlist. Therefore, about 12,000 playlists consisting of more than 200,000 unique English tracks created by approximately 1,000 persons are explored through applying a multimodal supervised classification approach. Various state-of-the-art algorithms are surveyed while incorporating with a bunch of acoustic and lyrics (lexical, linguistic, semantic and syntactic) properties. A novel data set consisting of preprocessed lyrics gathered from ten different websites serves as a source for extracting lyrics features. Examinations revealed that acoustic features are superior than lyrics features in representing a music playlist with respect to the classification accuracy. Nonetheless, combinations of lyrics features are rather equally capable to capture the characteristics of playlists.

Contents

Chapter 1

Introduction

The consumption of music has changed substantially in the recent years as new cloud-based music services evolved who enable people to access, explore, share and preserve music as well as manage songs and personal playlists across different devices [39]. Parts of the emerging services, like the popular music streaming platform Spotify[1], offer valuable scientific data and consequently facilitate, among other research areas, new inspections of music playlists.

Previous explorations disclosed that human beings choose music for a purpose [8, 13, 14] and commonly consider mood, genre and artist of tracks during the creation of playlists [34]. Latter track properties are studied by means of classification tasks including acoustic and/or lyrics features [17, 28, 38, 45, 63]. The utilization of multimodal data sources, i.e., audio signals, song texts[2], meta-data about artists/albums, improved mood and genre classification tasks revealing an orthogonality of audio and lyrics features [38, 45]. Pre-assembled playlists are favored over shuffling while listening passively (e.g., during exercising) to music [34]. Most users of cloud-based music services listen to playlists and partly consume automatically created compilations [39]. Automated playlist generation algorithms usually rely on seed tracks and employ multimodal similarity measures to build playlists [8]. Hence, several studies already discovered information about the listening behaviors of users and the preparation of playlists. However, none of them analyzed the properties of individual tracks that are shared within a music playlist.

Therefore, this research assesses via supervised machine learning classification tasks the relevance of acoustic and lyrics features of tracks in representing a playlist. The least amount of tracks constituting a characteristic playlist is evaluated and the most discriminative features

[1] https://www.spotify.com/, accessed on 2017-06-25

[2] Note that song text is used as a synonym for lyrics throughout this document.

are investigated. Moreover, feature subset selection is performed to improve the classification task. Hence, the following research questions are elicited:

- To which extent do acoustic- and lyrics-based feature sets characterize a particular playlist?
- How many tracks are at least required to ensure that a playlist is well characterized?
- Which individual track features have the most predictive power in deciding whether a track fits into a playlist or not?

To answer the research questions, a collection of user-generated playlists extracted from Spotify by Pichl et al. [55] enriched with acoustic and lyrics features is explored. Former features are extracted from audio signals offered by Spotify while the latter ones are derived from a self-created lyrics collection. In total, about 12,000 playlists including more than 200,000 distinct English tracks generated by nearly 1,000 users are analyzed.
A detailed overview of the employed approach is illustrated in Figure 1.1, which outlines the acquisition of the data collection, the process of gathering features of tracks and the applied evaluation methodology to explore music playlists.

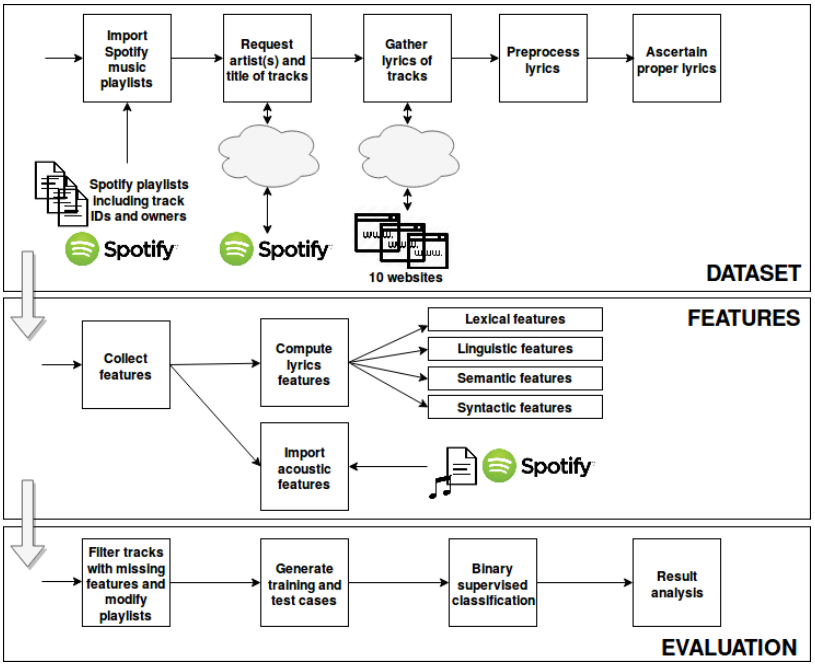

Figure 1.1: Approach overview.

Classification results are obtained through eight state-of-the-art machine learning algorithms on a per-playlist basis. They disclose that acoustic features are most discriminative in deciding whether a track fits into a playlist or not, as well as that the minimum amount of necessary tracks to characterize playlists is eight. Moreover, best classification results are achieved with feature subset selection gaining an accuracy of 71%.
Accordingly, this thesis gives an introduction into supervised classification in Chapter 2 by exemplifying the basic schemata, announcing commonly used features and algorithms, and presenting evaluation metrics. Chapter 3 covers present literature related to this research. Subsequently, in Chapter 4, the process of collecting data including playlists, tracks, and lyrics is described. The computation of various lyrics features based on the previously acquired data and the assembling of acoustic features is elucidated in Chapter 5. The research questions are answered in Chapter 6 through a supervised classification approach on a per-playlist basis. Finally, Chapter 7 concludes the thesis and presents future work.

Chapter 2

Supervised classification

Machine learning (ML), a subfield of artificial intelligence, is commonly applied in the extent literature to disclose hidden patterns in data collections (data mining) by observing data instances [37] and is employed in this research to reveal properties of playlists. Depending on the input sources a machine learning method makes use of, it either belongs to the supervised, unsupervised, semi-supervised or reinforcement learning category [2], each of it uncovers different types of patterns.

In supervised learning, data instances associated with labels, usually assigned by a domain expert, are observed while in unsupervised learning data instances without labels are analyzed. A combination of both types is named semi-supervised where data partially associated with labels is utilized. Reinforcement learning methods interact with their environment and learn from the impacts of their actions whilst dealing with a problem. The aim of (semi-)supervised methods is to distinguish relationships between inputs and desired outputs to infer a predictable mapping function. Unsupervised algorithms find similar classes of different inputs and reinforcement algorithms compute a sequence of actions with a maximum success outcome through trial-and-error runs regarding a given problem. [2, 37]

Accordingly, the research questions are answered by means of a supervised learning approach and properties of playlists are concluded through the performance analysis of learned models/mapping functions in classifying whether a track fits into a playlist or not. Hence, this chapter gives a brief introduction into supervised classification including the process of supervised machine learning, commonly applied features/algorithms, and model evaluation metrics.

2.1 Schema

The process of supervised machine learning, depicted in Figure 2.1, defines the necessary steps to build a classifier able to solve a certain problem. Depending on the problem domain, the necessary data set to learn a classifier needs to be acquired and afterwards preprocessed. The preprocessing step computes missing attributes/features valuable for the subsequent selected supervised machine learning algorithm. Attribute selection is performed to remove noisy data and to reduce data dimensionality as learning from large data sets is unfeasible. A parameterizable supervised algorithm is trained on the feature subset outputting a problem-oriented model usable for classification. If the resulting classifier is insufficient, previous conducted steps need to be adjusted until a desired state is achieved. [37]

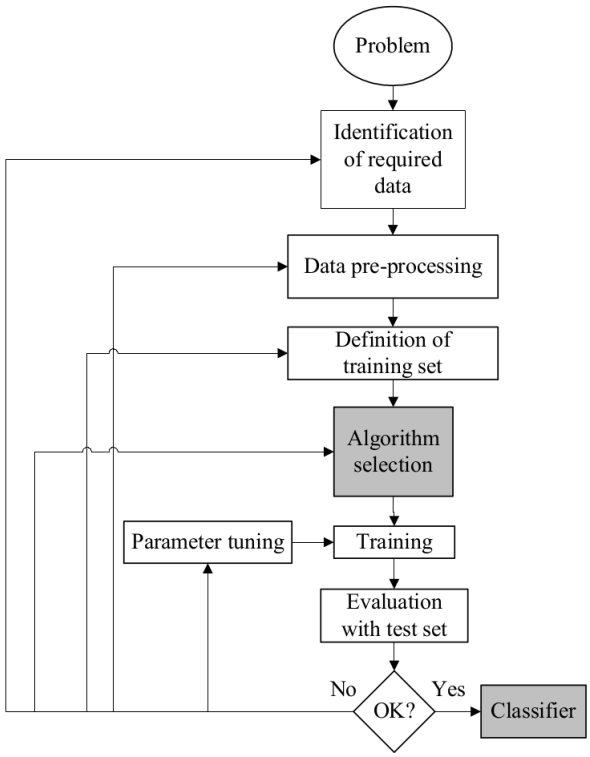

Figure 2.1: The process of supervised machine learning. [37]

2.2 Features

After the acquisition of an appropriate data set, feature extraction, also called attribute extraction, is performed to turn raw data into domain specific useful values to improve the accuracy of a model generated by an employed supervised learning algorithm [21]. Common techniques applied in this study are introduced below.

2.2.1 Bag-of-words model

The bag-of-words (BOW) model, also referred to as unigram language model, is a popular technique used in information retrieval (IR) to classify objects by simplifying the representation of the object contents. In the realm of text classification, a document is modeled as a collection of its words, including duplicates but ignoring contextual information like grammar and word ordering. Consequently, a document is represented as a feature vector of its word occurrences/frequencies. The *term frequency and inverse document frequency (tf-idf)* weighting schema, in conjunction with the bag-of-words model, is commonly applied to improve document classification. It overcomes the problem that words are usually not equally significant for a document by weighting a word according to its relevancy to a document compared to a collection. [40]

2.2.2 Part-of-speech tagging

Part-of-speech (POS) tagging/grammatical tagging describes the process of determining a proper morphosyntactic category (e.g., adjective, adverb, noun-singular) for each word in a text. Words are usually ambiguous and therefore belong to different parts of speech depending on their usage. For instance, consider the word *"flies"* which can be a noun (plural) or a verb. The process disambiguates a category for a word based on its definition and context. [61]
The grammatical tagged sentence of *"She flies to America."* using the *Penn Treebank POS tag set*[1] results in:

$$\text{"}[_{PRP}\ \textit{She}]\ [_{VBZ}\ \textit{flies}]\ [_{TO}\ \textit{to}]\ [_{NNP}\ \textit{America}]\ [_{.}\ .]\text{"}$$

Accordingly, *"She"* is a personal pronoun (PRP), *"flies"* is a third-person singular verb present (VBZ), and *"America"* is a proper noun (singular) (NNP). There is no distinction for the term *"to"*, whether it is an infinitival marker or a preposition. The *[.]*-tag marks the sentence-final punctuation (punctuations are marked as they appear in the text).

[1] Refer to Appendix A.2.1 for an overview of all Penn Treebank part-of-speech tags.

2.2.3 Text chunking

Text chunking is the task of splitting a text into non-overlapping groups of syntactically related words where each word belongs at most to one segment. Noun phrases (NP), verb phrases (VP), personal pronoun phrases (PP), and adjective phrases (ADJP) are samples of segment types. [68]
Depending on the employed chunking method, a possible chunking outcome of the sentence *"The look and feel of this smartphone is horrible"* using the *Penn Treebank phrase level tag set*[2] might be:

"[$_{NP}$ The look and feel] [$_{PP}$ of] [$_{NP}$ this smartphone] [$_{VP}$ is] [$_{ADJP}$ horrible] [$_{O}$.]"

The words within square brackets form a single segment/chunk. A tag at the beginning of each chunk indicates the type. The *"O"*-tag denotes a term outside of any segment.

2.3 Classification algorithms

Choosing a proper supervised learning algorithm is a crucial task and depends always on the application domain [37], thus different state-of-the-art algorithms are utilized in this work to determine the most appropriate classification algorithm for the specified research tasks. The functional principles of the classification algorithms *kNN*, *Bayes Net*, *Naïve Bayes*, *J48*, *PART* and *Support Vector Machine* are briefly introduced. For further information please refer to the referenced literature.

kNN

The *k-nearest neighbor (kNN)* classification discovers through a similarity/distance measure a cluster of k-closest training samples for an unlabeled instance and determines a class label with regards to the applied classes in the neighborhood. The performance of the kNN algorithm is influenced by the choice of k, the applied similarity/distance measure and the strategy of joining class labels of closest neighbors. If k is too small, then the classifier is sensitive to outliers, otherwise, if it is too large, the classification results get biased as class boundaries are less distinct. [72]

Bayes Net

Bayesian networks, often abbreviated as *Bayes Nets* but also known as *belief networks*, are probabilistic graphical models structured as directed

[2]Refer to Appendix A.2.2 for an overview of all Penn Treebank phrase level tags.

acyclic graphs where vertices constitute random variables and links indicate probabilistic dependencies between nodes. Moreover, a directed edge denotes an influence of a source node on a sink node. Inferences are possible through a subset of variables as subgraphs in a graphical model implicate conditional independencies facilitating local reasonings and further a simplification of a possible complex graph. [2, 5]

Naïve Bayes

Naïve Bayes builds a classifier by assuming independent feature values given a class. Through disregarding input correlations, a multivariate problem is turned into a set of univariate problems and thus a class probability for a feature vector $\mathbf{X}$ and a class C using Bayesian theorems corresponds to $P(\mathbf{X}|C) = \prod_{i=1}^{n} P(X_i|C)$, where $\mathbf{X} = \{X_1, \ldots, X_n\}$. By adding decision rules, for instance maximum a posteriori (MAP), a class for a feature vector is determined. [2, 59]

J48

J48 is the Java implementation of the C4.5 algorithm provided by Weka[3]. C4.5 is based on ID3 and belongs to the family of decision trees. An initial tree is generated from labeled data using a divide-and-conquer approach where nodes of a tree represent tests of single attributes and leafs state classes. [72]
The test attributes are ranked based on their corresponding information gain ratio. If C terms the amount of output classes, D denotes the set of training cases and $p(D, c)$ is the fraction of cases in D belonging to class $c \in C$, then the information gain of a test T with n outcomes yields to:

$$Info(D) = -\sum_{c \in C} p(D, c) \cdot log_2(p(D, c))$$

$$Gain(D, T) = Info(D) - \sum_{i=1}^{n} \frac{|D_i|}{|D|} \cdot Info(D_i)$$

$$Split(D, T) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \cdot log_2\left(\frac{|D_i|}{|D|}\right)$$

$$GainRatio(D, T) = \frac{Gain(D, T)}{Split(D, T)}$$

The highest gain ratio indicates the most discriminative test attribute and is accordingly selected as splitting attribute. After the tree has been constructed it is pruned to avoid overfitting. [57]

[3] http://www.cs.waikato.ac.nz/ml/weka/, accessed on 2017-04-18

PART

The *PART* algorithm infers classification rules from partial C4.5 decision trees. Based on the separate-and-conquer strategy employed by RIPPER [11], decision trees are iteratively created upon labeled training instances uncovered by previously generated rules until no instances remain. A rule is obtained from the most discriminating leaf of a pruned decision tree. After extracting a single rule the whole decision tree is discarded. The accuracy of PART is comparable to C4.5, however, it does not require a rather complex rule post-processing to improve classification. [19]

Support Vector Machine

A *support vector machine (SVM)*, also called support vector network, maps input vectors into a high dimensional feature space where a linear decision surface can be induced to separate classes. The optimal decision surface (hyperplane) has a maximum margin between vectors of different classes which assures the capability of high generalization. It is determined through support vectors which define the margin of largest separation between classes, as pictured in Figure 2.2. If training data cannot be separated without errors, soft margin hyperplanes can be defined to permit a minimal amount of misclassification. [12]

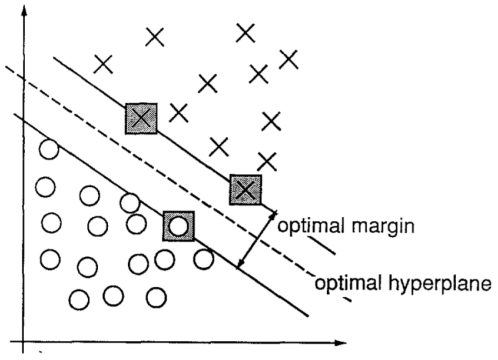

Figure 2.2: A separable classification problem in a two dimensional space. The margin of largest separation between the two classes is defined through support vectors (grey squares). [12]

2.4 Model evaluation

Depending on the domain and purpose of developed models particular metrics are applied in literature. Several metrics are derived from a

so-called *confusion matrix* which recaps the outputs of a model regarding to some test data. The confusion matrix represents the predicted classes of instances opposing them to their actual classes. Binary classifiers are used in this research, accordingly, a two-class confusion matrix discloses the performance for positive and negative classes as depicted in Figure 2.3. The resulting four values either indicate if instances are properly or improperly classified. A binary classifier can cause two types of errors: *false positives* (FP) and *false negatives* (FN). The false negative error denotes the number of misclassification of actual positive as negative instances. *True positives* (TP) and *true negatives* (TN) represent correct classifications. The total amount of instances per actual class or predicted class can be determined through the row-wise or colum-wise total, respectively. [61]

		Assigned Class	
		Positive	Negative
Actual Class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Figure 2.3: Binary classification outcomes divided into positive and negative classes. [61]

Before amplifying related metrics, a commonly employed test procedure is introduced to compute accurate confusion matrix values.

2.4.1 K-fold cross validation

K-fold cross validation is a test procedure to assess the predictive performance of models and is used to avoid overfitting. Training data is partitioned into k equal-sized and disjunctive subsets (folds), each used for the evaluation of a classifier while training on the remaining $k - 1$ subsets. The average error rate of all k evaluation runs correlates to the error rate of the classifier. [37, 61]

2.4.2 Metrics

Accuracy, *precision*, *recall*, and *F-Measure* are metrics derived from a confusion matrix and are frequently applied in (music) information retrieval. Consequently, these types are described, with respect to the above mentioned terminology used in a two-class confusion matrix.

Accuracy

How well a model predicts the correct classes of all instances is disclosed by the *accuracy* metric. It is defined as the proportion of proper classified instances to the total amount of instances:

$$Accuracy := \frac{TP + TN}{TP + FP + TN + FN}$$

A high *accuracy* measure indicates a proper model if and only if the actual classes are uniformly distributed. [50]

Precision

Precision, also known as *positive predictive value* [61], measures the fraction of truly positive instances to all positive assigned instances:

$$Precision := \frac{TP}{TP + FP}$$

In other words, *precision* quantifies the purity of positive predicted instances. [10]

Recall

Recall, often referred to as *sensitivity* or *true positive rate* [61], measures the proportion of positive instances which are correctly classified:

$$Recall := \frac{TP}{TP + FN}$$

Note that a perfect *recall* measure can always be achieved by simply classifying all instances as positive classes. *Recall* and *precision* are related to each other. The goal of a model is to achieve perfect measures for both metrics simultaneously. [10]

F-Measure

The harmonic mean of *precision* and *recall* is known as *F-Measure* or F_1-Score and is used to asses the accuracy of binary classification problems:

$$F_1 := 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

The score resides between the *precision* and *recall* measures but is closer to the minor one. A high *F-Measure* implies good *precision* and *recall* characteristics of a model. [61]

Chapter 3

Related work

The analysis of characteristics of music playlists is influenced by examinations in music management and music consumption behavior. Research disclosed that individuals choose music for a purpose and commonly consider mood, genre, and artists of tracks while creating playlists. Latter properties of tracks are already studied by means of a supervised classification approach in the realm of music information retrieval (MIR) and are thus related to this study. Findings in the fields of listening and music management behavior, mood classification, genre classification, and authorship attribution are incorporated in this work and are therefore presented in this chapter. The aim of mood classification is to categorize tracks based on the feelings they exhibit whereas the target of genre classification is to classify tracks according to human-defined genre labels. Recognizing the author (e.g., songwriter) of text documents by measuring textual features (stylometry) is the goal of authorship attribution.

3.1 Listening and music management behavior

Kamalzadeh et al. [34] researched in the area of music listening behavior and distinguished between active and passive listening. They examined that pre-assembled playlists and filters of album, artist, etc. are favored over shuffling when consuming music during performing other activities like exercising, commuting, or doing the housework. In addition, Kamalzadeh et al. confirmed previously conducted researches from Vignoli [71] as well as Bainbridge et al. [4] and parts from Stumpf and Muscroft [66] in the realm of music management behavior: artist, album and genre are the most significant attributes to manage music collections and mood, genre and artist were most relevant for constructing music playlists.

In the work of Demetriou et al. [14], the authors observed the listening behaviors of users too and pointed out that music is used as a technology to attain a desired internal state. Users choose music for a purpose and use it as a psychological tool to accomplish tasks more efficiently by achieving flow states through optimizing emotion, mood and arousal. The authors suggest that music information retrieval should consider the psychological impact of music.

An online survey of cloud music service usage performed by Lee et al. [39] revealed that 89.4% of participants use playlists. 53.1% consume automatically generated playlists in place of (or complementary to) creating their individual ones. Personal playlists are created on virtue of personal preference (72.9%), mood (59.9%), genre/style (55.4%), accompanying activity (50.8%), artists (35.6%) and recent acquisition (33.3%). Participants responded that online music services are dissatisfying because of suboptimal offered playlists or automated radio features.

3.2 Genre classification

Mayer et al. [46] computed rhyme, part-of-speech, bag-of-words, and text statistic features (e.g., words per line, characters per word, words per minute, counts of digits) from lyrics for genre classification and showed how values differ across several genres. Their obtained classification accuracies were inferior than assimilable achievements based on audio content. However, they demonstrated that lyrics features can be orthogonal to audio features and might be superior in determining different genres.

On grounds of the findings from [46], Mayer et al. [45] studied the combination of audio and lyric features and obtained higher genre classification accuracies than classifiers merely trained on audio features. The impact of individual features are investigated on a manually preprocessed and a non-preprocessed lyrics corpus. Best results for the non-preprocessed corpus could be achieved with a support vector machine (SVM) trained on audio content descriptors and text statistic features. Part-of-speech and rhyme features did not improve the SVM results. Content descriptors, text statistics and part-of-speech features worked best for preprocessed lyrics, again classified by a SVM. Lyrics preprocessing improved the classification accuracy by about 1% as against non-preprocessing. Mayer et al. [45] noted that preprocessing lyrics can enhance the performance of part-of-speech tagging and may thereupon increase classification accuracy.

A lyrics-based genre classification approach has been analyzed by Fell and Sporleder [17]. They trained SVMs with n-gram models combined with vocabulary, style, semantics, song structure, and orientation towards the world features to group songs into eight genres. Rap could be easily detected as this genre exhibits unique properties such as long lyrics, complex rhyme structures and quite distinctive vocabulary. Folk was frequently confounded with Blues or Country since they possess similar lexical characteristics. Musical properties improved the recognition of these genres. Experiments showed that length, slang use, type-token ratio, POS/chunk tags, imagery and pronouns features contribute most in genre classification.

3.3 Mood classification

Already one decade ago, Vignoli [71] mentioned the requirement to select music according to mood.

Laurier et al. [38] evaluated the influence of individual as well as the combination of audio and lyrics features in mood classification. Like in the realm of genre classification, they demonstrated the positive impact of multimodal data sources in mood classification. A song is not restricted to a single mood class and can belong to the groups *happy*, *sad*, *angry*, and *relaxed* which match the parts of Russell's mood model [60]. The audio-based classifier trained on timbral, rhytmic, tonal, and temporal features, achieved an accuracy of 98.1% for the mood category *angry*, 81.5% for *happy*, 87.7% for *sad* and 91.4% for *relaxed*. Inferior accuracies are attained with lyrics-based classifiers (based on similarity, latent semantic analysis and language model differences), but by mixing up the feature space the accuracy could be improved about 5% for the mood classes *happy* and *sad*.

In the work of Hu and Downie [28], 63 audio spectral features and various lyrics features, such as bag-of-words features, linguistic features and text stylistic features, including those proved beneficial in [45], are analyzed. Linguistic features are computed from sentiment lexicons and psycholinguistic resources like General Inquirer (GI) [64], Affective Norm of English Words (ANEW) [9] enriched with synonyms from WordNet [18], and WordNet-Affect [65]. The combination of content words, function words, GI psychological features, ANEW scores, affect-related words and text stylistic features performed best. Second best results could be gained by combining ANEW scores and text stylistic features, consisting only of 37 against ~115,000 features for the best lyric feature combination. Experiments discovered that content words are important in the

task of lyrics mood classification. Late fusion of audio and lyric classifiers outperformed a leading audio-only system by 9.6%.

An automatic mood classification approach based on lyrics using the information retrieval metric tf-idf has been proposed by Zaanen and Kanters [70]. Lyrics which manifest the same mood are merged together and represent a particular mood class. From these combined lyrics the relevancy of a word for a mood class is determined by applying the tf-idf weighting factor. Evaluations revealed that tf-idf can be used to detect words which characterize mood facets of lyrics and thus knowledge about mood can be exhibited from the lingual part of music.

3.4 Authorship attribution

Kırmacı and Oğul [35] dealt with the topic of author prediction solely based on song lyrics. They trained a linear kernel SVM with five feature sets, namely bag-of-words, character n-grams, suffix n-grams, global text statistics and line length statistics. The gained results pinpoint low precision (52.3%) and recall (53.4%) measures, indicating a non reliable classification accuracy. Nonetheless, an adequate ROC score of 73.9% was obtained too, illustrating the capability of the model to be applied as a supplementary method in music information retrieval and recommender systems. In addition, Kırmacı and Oğul investigated the performance of the model for genre classification and achieved higher precision (67.0%) and recall (67.7%) measures than for author prediction. Thus, song writers of the same music genre use similar linguistic and grammar forms, which simplifies genre classification but impedes author prediction.

Stamatatos [63] analyzed automated authorship attribution approaches and explored their characteristics for text representation and classification by focusing on the computational requirements. The survey presents various lexical, character, syntactic, semantic as well as application-specific measures and depicts how these so-called stylometric features contribute in authorship attribution. The bag-of-words model is the most (at least partially) applied lexical feature in authorship attribution approaches to exploit text stylistics. Function words are proven to be relevant as they are topic-independent and capable of determining stylistic choices of authors. Word n-grams capture contextual information and type-token ratios shed light on the vocabulary richness. Character n-grams of fixed or variable length capture nuances of style with lexical/contextual information, usage of punctuation/capitalization, etc. Similarly to words, the most popular n-grams are the most discrimina-

tive ones. Text chunks (i.e., phrases) and POS tags are used to derive syntactic style features like phrase counts, length of phrases or POS tag n-gram frequencies. Synonyms and hypernyms offer the possibility to reveal semantic information. Depending on the given text domain, particular features can be derived to improve the quantification of the writing style. For instance, in the domain of e-mail messages, structural measures such as the use of greetings or types of signatures can be computed. Stamatatos noted that an independent feature may not enhance a classification task but might be beneficial in combination with other feature types. Moreover, he mentioned that the accuracy of authorship attribution methods is influenced by the amount of candidate authors, the size of the training corpus and the length of the individual training and test texts.

Chapter 4

Dataset

Music information retrieval (MIR) research suffers from the scarcity of standardized benchmarks by reason of intellectual property and copyright issues [47, 48, 49]. There are MIR benchmarks publicly available (i.e., the *Million Song Dataset* [6]) which have already been used in literature, but to the best of the authors knowledge these do not possess a sufficient number of playlists and/or lyrics, hence they are not suited for this research purpose. Therefore, a novel test and training dataset is created consisting of user-generated playlists, meta data about tracks, and song texts. Accordingly, this chapter covers the process of gathering music playlists, tracks, and lyrics as well as the preparation of lyrics for further data evaluations.

4.1 Playlists

User-generated playlists form the basis of the self-created training and test corpus. They have been collected by Pichl et al. [55] who extracted them from the music platform *Spotify*. The dataset contains ~1,200,000 records where each record consists of a hashed user name, a *Spotify* track ID and a playlist name. This results in ~18,000 playlists of diverse size with ~670,000 tracks in total created by 1,016 users. The distribution of playlist sizes is pictured in Figure 4.1 and depicts that most playlists are compound of 9 to 14 tracks. Note that playlists consisting of only one track are not considered in later analysis.

4.2 Tracks

The dataset of Pichl et al. [55] doesn't offer any information about tracks except a *Spotify* ID which can be used for further analysis. Thus,

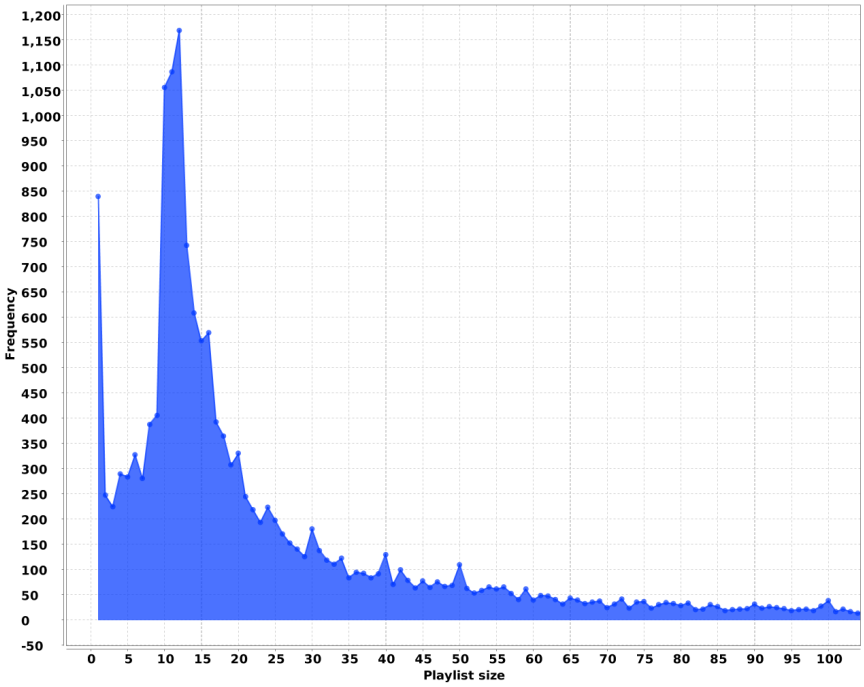

Figure 4.1: Distribution of playlist sizes.

the application programming interface[1] (API) provided by *Spotify* has been utilized to enrich the test corpus with meta data about tracks. The retrieved meta data exhibits valuable information like artist names and song titles which can be used to fetch lyrics from the World Wide Web automatically.

4.3 Lyrics

Well structured and correct song texts are crucial for this study, therefore the acquisition and preparation of lyrics is significant. In [20, 36, 58], the authors queried the *Google* search engine with the parameters artist name, track name, and the keyword "lyric" to automatically fetch lyrics from the Web. Knees et al. [36] used the retrieved lyrics to eliminate mistakes in lyrics like typos using a multiple sequence alignment technique. However, their outcome leads to a sequence of words without any word-wraps or punctuation and lacks therefore useful structural information. Geleijnse and Korst [20] investigated various versions of lyrics regarding a given song by assuming that lyrics within websites

[1] `https://api.spotify.com/v1/tracks/`, accessed on 2016-06-28

are not composed of HTML-tags except of end of line tags
, thus from the first 40 search engine results song texts are extracted using regular expressions. Ribeiro et al. [58] employed a lyrics detection and extraction procedure that uses all HTML-tags to locate lyrics within any website. An evaluation revealed that their developed *Ethnic Lyrics Fetcher (ELF)* tool outperforms the presented technique from Geleijnse and Korst [20]. A different approach has been applied by [17, 27, 45, 70], who utilized website specific crawlers to fetch accurate lyrics. As the *ELF* tool is currently not publicly available, the latter methodology has been pursued and user-contributed online lyrics databases are accessed and queried with specially implemented crawlers.

4.3.1 Collecting lyrics

As already mentioned, song titles and artist names are provided by *Spotify* and can therefore be used to fetch lyrics from the World Wide Web automatically. User-contributed lyrics databases are queried in the present literature to gather appropriate song texts for sundry analysis tasks. For instance, [29, 38] and [16] accessed the data sources *lyricwiki.org*[2] and *LYRICSMODE*[3], respectively. Moreover, [45] fetched lyrics from a collection of online databases by employing Amarok's[4] lyrics scripts. Accordingly, ten different user-contributed online lyrics platforms (most of them are queried by Amarok too) are used as data sources:

1. *ChartLyrics*[5]
2. *LYRICSnMUSIC*[6]
3. *LyricWikia*[7]
4. *eLyrics.net*[8]
5. *LYRICSMODE*
6. *METROLYRICS*[9]
7. *Mp3lyrics*[10]
8. *SING365*[11]
9. *SONGLYRICS*[12]
10. *Songtexte.com*[13]

The latter seven doesn't offer an API to request lyrics by artist and

[2] `http://www.lyricwiki.org` redirects to `http://lyrics.wikia.com/wiki/Lyrics\_Wiki`, accessed on 2016-06-28
[3] `http://www.lyricsmode.com/`, accessed on 2016-10-26
[4] `http://amarok.kde.org/`, accessed on 2016-10-26
[5] `http://www.chartlyrics.com/api.aspx`, accessed on 2016-06-28
[6] `http://www.lyricsnmusic.com/api`, accessed on 2016-06-28
[7] `http://lyrics.wikia.com/api.php`, accessed on 2016-06-28
[8] `http://www.elyrics.net/`, accessed on 2016-10-26
[9] `http://www.metrolyrics.com`, accessed on 2016-06-28
[10] `http://mp3lyrics.com/`, accessed on 2016-10-26
[11] `http://www.sing365.com/`, accessed on 2016-10-26
[12] `http://www.songlyrics.com/`, accessed on 2016-10-26
[13] `http://www.songtexte.com`, accessed on 2016-06-28

song title, thus classical web-crawling techniques have been applied to grab lyrics from those web systems. The language of each song text is identified with the content analysis toolkit *Apache Tika*[14] to filter English lyrics as some of the employed text features can not be computed for all languages. The result of the lyrics acquisition is illustrated in the subsequent Table 4.1, which is itemized by data source and lyrics language.

Language code	ChartLyrics	eLyrics.net	LYRICSMODE	LYRICSnMUSIC	LyricsWikia	METROLYRICS	Mp3Lyrics	SING365	SONGLYRICS	Songtexte.com
be	5	0	0	0	7	2	0	0	2	2
ca	821	1,148	180	162	624	568	322	65	934	1,235
da	40	238	32	44	101	173	49	26	238	173
de	1,309	3,337	355	422	759	2,490	557	232	2,512	37,017
el	0	0	0	0	0	6	0	0	0	3
en	**266,036**	**584,624**	**195,919**	**231,675**	**267,837**	**502,308**	**251,468**	**201,478**	**574,040**	**264,436**
eo	1,015	1,300	557	707	1,082	2,282	754	496	1,579	1,086
es	9,120	21,813	4,607	6,017	12,127	21,032	8,798	1,240	28,993	14,486
et	1,287	3,650	751	1,478	1,153	3,178	1,324	757	3,413	1,444
fa	0	0	0	1	9	4	1	0	3	39
fi	718	1,636	504	354	1,400	1,563	573	221	2,190	1,906
fr	4,478	6,483	1,728	1,307	3,164	4,644	2,382	466	9,039	4,013
gl	3,153	11,231	1,628	1,935	3,380	5,887	2,884	689	8,239	3,956
hu	1,261	1,512	546	647	817	1,291	891	473	1,758	778
is	405	604	222	207	386	520	425	189	653	395
it	9,431	6,227	1,175	2,038	13,524	10,118	1,902	1,922	13,226	2,393
lt	555	954	200	33,659	2,506	87,442	416	140	658	7,919
nl	1,843	1,792	384	155	804	805	271	48	1,726	991
no	6,010	8,932	3,753	4,620	5,772	10,151	6,092	3,489	9,375	6,085
pl	354	663	147	230	339	459	271	134	516	375
pt	746	1,297	230	441	846	1,021	703	74	2,133	1,088
ro	586	909	227	304	380	936	414	227	1,153	484
ru	0	0	1	2	6	3	3	1	15	35
sk	1,432	2,689	763	1,016	1,235	3,248	1,127	684	2,273	1,342
sl	247	2,188	312	331	387	1,468	397	213	1,125	575
sv	785	1,522	429	163	1,455	1,649	325	108	3,469	2,331
th	0	0	0	0	0	0	0	0	1	7
uk	9	2	1	5	9	10	6	5	7	9
??	0	0	0	1	1	0	1	1	0	0
∑	**311,646**	**664,751**	**214,651**	**287,921**	**320,110**	**663,258**	**282,356**	**213,378**	**669,270**	**354,603**

Table 4.1: Amount of retrieved lyrics from ten different data sources grouped by language (ISO 639 code).

4.3.2 Data preparation

Due to the use of user-generated data sources, challenges like data noise, quality issues and the utilization of different lyrics notation styles have to be mastered, otherwise the evaluation results get tampered. To mitigate these problems all lyrics need to be sanitized and carefully selected. In the field of genre categorization, Mayer et al. [45] already indicated improved classification accuracies through lyrics preprocessing.

[14] `https://tika.apache.org/1.13/detection.html`, accessed on 2016-10-26

Typical characteristics of lyrics have been pointed out by [16, 26, 29, 36, 70] and are listed below:

- Song structure annotations:
  Lyrics are often structured into segments like intro, interlude, verse, bridge, hook, pre-chorus, chorus and outro. Several lyrics exist with explicit type annotations on their segments.
- References and abbreviations of repetitions:
  Song texts are seldom written completely, instead instructions for repetitions, sometimes with a reference to a previous segment, are used (e.g., "~Chorus (x1)~", "(x3)", "[repeat thrice]", etc.).
- Annotation of background voices/sounds:
  Occasionally there are background voices (yeah yeah yeah, etc.) or sounds (e.g., *scratching*, ~fade out~, etc.) denoted in lyrics.
- Song remarks:
  Information about the author (e.g., written by . . .), performing artists, publisher, song title, total song duration (e.g., Time: 3:01), chords or even the used instruments are sometimes remarked in song texts.

All these characteristics need to be considered when preprocessing lyrics. The usage of different notation styles impede this task. Figure 4.2 depicts a couple of these properties by comparing three syntactical different, but semantical equivalent versions of the song "Tainted Love" performed by "Soft Cell". Hu [26] manually created a list with commonly used repetition and annotation patterns, which takes the before mentioned traits into account. The list has been adopted and slightly modified such that it can be used as a guideline for sanitizing lyrics. The adapted list of lyrics repetition and annotation patterns can be found in Appendix A.1. Accordingly, the following outlined preprocessing steps are conducted on lyrics, which are exemplified in Figure 4.3:

1. Remove/replace superfluous whitespaces
   (a) remove leading and trailing newlines
   (b) remove leading and trailing whitespaces (except newlines) from each line
   (c) replace consecutive whitespaces (except newlines) with a single whitespace
   (d) replace three or more consecutive newlines with two newlines

Version 1	Version 2	Version 3
... (CHORUS) Once I ran to you (I ran) Now I'll run from you This tainted love you've given I give you all a boy could give you Take my tears, that's not nearly all Tainted love (oh) Tainted love ... ---------- (CHORUS) ---------- ... Tainted love (oh) (4x) Touch me, baby, tainted love (2x) Tainted love (oh) (2x) Tainted love Tainted love fade out Written by: Ed Cobb Copyright: Ed Cobb	Time: 3:01 ... Chorus: Once I ran to you (I ran) Now I'll run from you This tainted love you've given I give you all a boy could give you Take my tears, that's not nearly all Tainted love (oh) Tainted love ... [repeat chorus]: ... (interlude) Tainted love (oh) (x4) Touch me, baby, tainted love (2x) Tainted love (oh) (repeat twice) Tainted love Tainted love	intro ... {chorus} once I ran to you (I ran) now I will run from you this tainted love you've given I give you all a boy could give you take my tears, that's not nearly all tainted love (oh) tainted love ... ~chorus (x1)~ ... verse 6: tainted love (oh) (x4) touch me, baby, tainted love (x2) tainted love (oh) (x2) tainted love (x2) outro

Figure 4.2: Three syntactically different, but semantically equivalent lyrics excerpts of the song “Tainted Love” by “Soft Cell” pointing out some typical lyrics characteristics.

2. Remove/replace special characters
   (a) replace characters due to mismatched encodings
   (b) remove lines which contain only special characters (e.g., used as segment separators “———————”)
3. Remove music chords (e.g., “A /A”, “E7”, etc.) [16]
4. Remove song remarks [26]
   (a) remove artist name(s) and song title information
   (b) remove pronunciation hints (e.g., “whispered”, “laughing”, etc.)
   (c) remove publisher, producer, song writer, copyright, song duration, etc. from the beginning and end of segments

(d) ...

5. Remove hyperlinks [26]
6. Reduplicate designated segments and lines [26, 36]
7. Remove song structure annotations [26]

4.3.3 Ascertaining proper lyrics

User-contributed online data sources provide materials which are not always reliable and accurate due to wrong or incomplete (on purpose or unintentionally) published data from several users. Consequently, the correctness of the fetched content needs to be revised to minimize the likelihood of considering wrong song texts in the experiments. Based on the assumption that content errors occur platform independently, valuable content can be detected through comparing results of multiple user-contributed data sources. Accordingly, user-generated content is distinguished as worthwhile, if per song minimum three of ten accessed online platforms offer lyrics which possess a similar lexical content. A platform offers a lyric version, iff the fetched song text is comprised of at least ten lines, each line consists of maximum 200 characters (similar to [16]) and the corresponding download URL is not multiple times used to fetch lyrics except for tracks with the same artist names and song title. The similarity of two song texts is investigated via the Jaccard index [32], also referred to as Jaccard similarity coefficient. The Jaccard index measures the similarity of finite sets, thus the user-generated song texts are transformed into sets of lowercased word bigrams. Let A and B be two finite sets then the Jaccard index is defined as:

$$jaccard(A, B) := \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

The function ranges from 0.0 to 1.0, where the closer to 1.0, the more similar are the sets. Two song texts are considered as lexical similar if the Jaccard similarity measure exceeds a manually investigated threshold of 0.6. To ensure the aforementioned criteria, all obtained contents are pair-wisely compared of which at least three lyrics need to exhibit a similarity measure above the threshold. If so, the most proper song text out of the retrieved lyrics is selected which is considered for further playlist analysis otherwise the gathered content is inappropriate. The choice of the most proper song text is precisely described in the sample below.

Raw song text	Intermediate result	PPS	Sanitized song text
Tainted Love Lyrics	~~Tainted Love Lyrics~~	4(a)	
	-	1(a)	…
Time: 3:01	~~Time: 3:01~~	4(b)	
			Once I ran to you (I ran)
(instrumental)	~~(instrumental)~~	4(d)	Now I'll run from you
			This tainted love you've given
…	…		I give you all a boy could give you
			Take my tears and that's not nearly all
----------	----------	2(b)	Tainted love (oh)
CHORUS:	~~CHORUS:~~	6	Tainted love
F#m	~~F#m~~	3	
Once I ran to you (I ran)	____Once I ran to you (I ran)	1(b)	...
A	~~A~~	3	
Now I'll run from you	Now I'll run __from you	1(c)	Once I ran to you (I ran)
D	~~D~~	3	Now I'll run from you
This tainted love you've given	This tainted love you've given		This tainted love you've given
Bm	~~Bm~~	3	I give you all a boy could give you
I give you all a boy could give you	I give you all a boy could give you		Take my tears and that's not nearly all
Take my tears and that's not nearly all	Take my tears and that's not nearly all	2(a)	Tainted love (oh)
Tainted love (oh)	Tainted love (oh)		Tainted love
Tainted love	Tainted love		
----------	----------	2(b)	...
…	…		Tainted love (oh)
			Tainted love (oh)
[repeat CHORUS]	~~[repeat CHORUS]~~	6	Tainted love (oh)
	Once I ran to you (I ran)		Tainted love (oh)
…	Now I'll run from you		Touch me, baby, tainted love
	This tainted love you've given		Touch me, baby, tainted love
Tainted love (oh) (x4)	I give you all a boy could give you		Tainted love (oh)
[whispered]	Take my tears and that's not nearly all		Tainted love (oh)
Touch me, baby, tainted love (2x)	Tainted love (oh)		
Tainted love (oh) (repeat twice)	Tainted love		Tainted love
			Tainted love
	…		
Tainted love (x2)	Tainted love (oh) ~~(x4)~~	6	
	Tainted love (oh)		
outro	Tainted love (oh)		
	Tainted love (oh)		
http://lyrics.com/sc-tainlove.html	~~[whispered]~~	4(c)	
Copyright: Ed Cobb	Touch me, baby, tainted love ~~(2x)~~	6	
	Touch me, baby, tainted love		
	Tainted love (oh) ~~(repeat twice)~~	6	
	Tainted love (oh)		
	-	1(d)	
	-	1(d)	
	Tainted love ~~(x2)~~	6	
	Tainted love		
	~~outro~~	7	
	~~http://lyrics.com/sc-tainlove.html~~	5	
	~~Copyright: Ed Cobb~~	4(b)	

Figure 4.3: Example of sanitizing a user-generated song text including preprocessing steps (PPS). The sample represents the song "Tainted Love" performed by "Soft Cell".

Example:

Assume, the online platforms $P := (p_i \mid 1 \leq i \leq 5)$ are accessed who offer a song text t for a song s. Moreover, let $bigrams(t)$ denote the set of lowercased word bigrams for t. To simplify the example, s portrays the particular song "Wonderwall" performed by Oasis and the song texts $T := (t_p \mid p \in P)$ are comprised only of a single line. The process of choosing proper lyrics is elucidated for the following song text excerpts:

$$
\begin{aligned}
t_{p_1} &:= \text{“and all the roads we have to walk are winding”} \\
t_{p_2} &:= \text{“You never have to walk alone”} \\
t_{p_3} &:= \text{“And all the roads we have to walk are blinding”} \\
t_{p_4} &:= \text{“And all the roads we have to walk are winding”} \\
t_{p_5} &:= \textit{no song text provided}
\end{aligned}
$$

Thus, the excerpts of platform p_1 and p_4 are correct and the song text from p_2 is almost right. Platform p_3 provides a wrong lyric version and p_5 doesn't offer one. A valuable song text exists, iff at least three of all data sources provide similar song text versions. To ensure this criteria, the Jaccard similarity of all song text pairs is computed. The Jaccard index requires finite sets as input, thus all song texts are transformed into lowercased word bigram sets, denoted as $B := \{bigrams(t) \mid t \in T\}$. For instance, the bigram sets $b_{p_1}, b_{p_2} \in B$ arise from the song texts $t_{p_1}, t_{p_2} \in T$, respectively:

$$
\begin{aligned}
b_{p_1} = \{&\text{“and all”}, \text{“all the”}, \text{“the roads”}, \text{“roads we”}, \\
&\text{“we have”}, \text{“have to”}, \text{“to walk”}, \text{“walk are”}, \\
&\text{“are winding”}\} \\
b_{p_2} = \{&\text{“you never”}, \text{“never have”}, \text{“have to”}, \text{“to walk”}, \\
&\text{“walk alone”}\}
\end{aligned}
$$

The application of the Jaccard index for b_{p_1} and b_{p_2} results in:

$$
\begin{aligned}
jaccard(b_{p_1}, b_{p_2}) &= \frac{|b_{p_1} \cap b_{p_2}|}{|b_{p_1} \cup b_{p_2}|} = \frac{|b_{p_1} \cap b_{p_2}|}{|b_{p_1}| + |b_{p_2}| - |b_{p_1} \cap b_{p_2}|} \\
&= \frac{|\{\text{“have to”}, \text{“to walk”}\}|}{|b_{p_1}| + |b_{p_2}| - |\{\text{“have to”}, \text{“to walk”}\}|} \\
&= \frac{2}{9 + 5 - 2} \sim 0.17
\end{aligned}
$$

The song texts are quite different as the outcome is close to zero. The following similarity matrix S is obtained by comparing all pairs of song texts bigrams:

$$S := \{(jaccard(b_{p_i}, b_{p_j}))_{ij} \mid b_{p_i}, b_{p_j} \in B \wedge 0 < i < j < |B| \wedge i \neq j\}$$

S	b_{p_1}	b_{p_2}	b_{p_3}	b_{p_4}	b_{p_5}
b_{p_1}	-	0.17	0.8	1.0	0.0
b_{p_2}	0.17	-	0.17	0.17	0.0
b_{p_3}	0.8	0.17	-	0.8	0.0
b_{p_4}	1.0	0.17	0.8	-	0.0
b_{p_5}	0.0	0.0	0.0	0.0	-

These measures reveal that the song texts from the platforms p_1, p_3 and p_4 are similar. Hence, the criteria is fulfilled and valuable data is existent. Finally, a song text needs to be chosen for further playlists analysis. Therefore, all similarity values above the similarity threshold (≥ 0.6) are row-wise summed up to indicate the most agreeable lyrics version. This leads to the subsequent result:

S	b_{p_1}	b_{p_2}	b_{p_3}	b_{p_4}	b_{p_5}	$\sum$
b_{p_1}	-	0.17	0.8	1.0	0.0	1.8
b_{p_2}	0.17	-	0.17	0.17	0.0	0
b_{p_3}	0.8	0.17	-	0.8	0.0	1.6
b_{p_4}	1.0	0.17	0.8	-	0.0	1.8
b_{p_5}	0.0	0.0	0.0	0.0	-	0.0

The lyrics from data source p_1 and p_4 are the most proper lyrics as they have the highest row-wise summed up similarity value. A random lyric out of the most proper song texts is chosen if no exclusive song text can be distinguished.

Through this method, 226,747 proper English lyrics could be distinguished for 671,650 tracks. This corresponds to a percentage of 33.76%.

Chapter 5

Features

Based on previously discussed findings, features particularly used in the fields of mood classification, genre classfication, and authorship attribution are considered to reveal characteristics of playlists. Several researchers [26, 28, 38, 45, 47] indicated that audio features and lyrics features are orthogonal to each other and accordingly illustrated the improvements of classification systems by employing multimodal features. Consequently, this chapter introduces acoustic and lyrics features and describes in detail how they are extracted. An overview of all computed features is given in Table 5.1.

Feature sets	Features
Acoustic (10)	danceability, energy, speechiness, liveness, acousticness, valence, tempo, duration, loudness, instrumentalness
Lexical (35)	bag-of-words (5), token count, unique token ratios (3), average token length, repeated token ratio, hapax/-dis-/tris-/legomenon, unique tokens/line, average tokens/line, line counts (5), words/lines/characters per minute, punctuation and digit ratios (9), stop words ratio, stop words per line
Linguistic (39)	uncommon words ratios (2), slang words ratio, lemma ratio, Rhyme Analyzer features (24), echoisms (3), repetitive structures (8)
Semantic (52)	Regressive imagery (RI) conceptual thought features (7), RI emotion features (7), RI primordial thought features (29), SentiStrength sentiment ratios (3), AFINN valence score, Opinion Lexicon opinion, VADER sentiment ratios/scores (4)
Syntactic (85)	pronouns frequencies (7), POS frequencies (54), text chunks (23), past tense ratio

Table 5.1: Overview of extracted features per track. The numbers in parenthesis pinpoint the number of features per individual feature set.

5.1 Acoustic features

Similar to the *Million Song Dataset* [6], ten acoustic features for 587,400 tracks are introduced for later analysis tasks, collected from *Spotify* and the music intelligence and data platform *Echo Nest*[1] by Pichl et al. [55]. According to the documentation from *Echo Nest* [33], meaningful information is extracted from audio signals with proprietary machine listening techniques which simulate the musical perception of persons. Moreover, musical content is obtained by modeling the physical and cognitive process of human listening through employing principles of psychoacoustics, music perception, and adaptive learning. The consulted acoustic attributes are defined by *Echo Nest* [15] and Spotify [62] as follows:

1. **Danceability** expresses how applicable an audio track is for dancing. Tempo, rhythm stability, beat strength and overall regularity of musical elements contribute to this measurement.

2. **Energy** is a perceptual measure of intensity and activity. Energetic tracks usually feel fast, loud and noisy (e.g., death metal has high whilst a Bach prelude has low energy). Energy is computed from various perceptual features like dynamic range, perceived loudness, timbre, onset rate and general entropy.

3. **Speechiness** indicates the likelihood of an audio file to be speech by determining the existence of spoken words.

4. **Liveness** describes how likely an audio file has been recorded live or in a studio by recognizing the attendance of an audience in the composition.

5. **Acousticness** predicts if an audio track is composed of only voice and acoustic instruments. Songs with electric guitars, distortion, synthesizers, auto-tuned vocals and drum machines are resulting in low acousticness. Music tracks with high acousticness contain orchestral instruments, acoustic guitars, unaltered voice and natural drum kits.

6. **Valence** predicts the musical positiveness of a track. The higher the valence value is the more positive a track sounds. The combination of valence and energy is an indicator of acoustic mood.

7. **Tempo** is the estimated speed or pace of a track in beats per minute (BPM) derived from the average beat duration.

8. **Duration** is the total time of a track.

[1] `http://the.echonest.com/`, accessed on 2016-09-07

9. **Loudness** describes the sound intensity of a track in decibels (dB). The average of all volume levels across the whole track yields the loudness measure.

10. **Instrumentalness** estimates if a track includes vocals or not. "Ooh" and "aah" sounds are considered as non vocals. Typical "vocals" are rap or spoken word songs.

5.2 Lyric features

A range of lyric features are introduced in this section based on the aforementioned research in Chapter 3. Those can be grouped into the following categories: *lexical features*, *linguistic features*, *syntactic features*, and *semantic features*.

Basic natural language analysis is the preliminary step towards deriving features from lyrics, therefore each song text is primarily analyzed with the well-known *Stanford Core NLP Natural Language Processing Toolkit* [41]. The toolkit provides a set of natural language processing components, from tokenization to sentiment analysis. The applied techniques on song texts are tokenization, part-of-speech (POS) tagging and lemmatization. The *Stanford Tokenizer*[2], more precisely the *Penn Treebank Tokenizer*[3] *(PTBTokenizer)* which is applicable for English text, is used to divide lyrics into lines, each comprised of a sequence of tokens. For every token the part of speech is determined based on its definition and textual context. Some possible POS categories are nouns, verbs, adjectives, adverbs, and prepositions. The *Stanford Log-linear Part-Of-Speech Tagger* [69] is utilized to lexically categorize a song text, whereat each line is treated as a particular unit. The tagger labels each token with one of the available tags in the *Penn Treebank tag set*[4]. It is trained on news articles from the Wall Street Journal due to missing training corpora consisting of POS tagged lyrics, but according to Hu [26], the tagger performs well for lyrics although news articles and lyrics differ in their text genres. Finally, a morphological analysis is conducted with the *Stanford MorphaAnnotator*[5] which computes the lemma (base form) of English words.

[2] `http://nlp.stanford.edu/software/tokenizer.shtml`, accessed on 2016-09-08

[3] `http://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/process/PTBTokenizer.html`, accessed on 2016-09-08

[4] see Appendix A.2 for all Penn Treebank tags

[5] `http://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/pipeline/MorphaAnnotator.html`, accessed on 2016-09-08

For the following feature definitions let s be a song text and $lines(s)$ be a sequence of all lines comprised in s in their natural order. A line $l \in lines(s)$ consists of a list of tokens in its natural order typified by $tokens(l)$ whereas $tokens(s)$ constitutes a natural ordered sequence of all tokens comprised in a song text s. The expression $chars(t)$ represents the characters for a token t and $bigrams(x)$ as well as $trigrams(x)$ the lists of word bigrams and trigrams for any text x.

5.2.1 Lexical features

Lexical features can be determined independent of language and text corpus and just require a tokenizer [63]. Through different text representations it is possible to discover various text stylometric features which contribute in authorship attribution and text genre categorization [3]. According to the survey of Stamatatos [63], most authorship attribution researches employ (at least partly) lexical features to describe style. In the field of music genre classification, Mayer et al. [46] pinpointed the beneficial use of text style features.

Bag-of-words

Hu et al. [29] investigated the performance of BOW features for lyrics mood classification and noted that choosing a set of words to assemble the bag-of-words set is a crucial task. Owing to mixed effects of stemming[6] in text classification, Hu et al. analyzed the influence of non-stemmed and stemmed words by excluding stop words[7]. Moreover, they modeled stop words and POS tags as BOW features, since stop words are stated to be effective in text style analysis and part-of-speech feature types are commonly applied in text sentiment as well as text style analysis. The stop word list of Argamon et al. [3], who combined the function words from Mitton [52] and a list of stop words special to the newsgroup domain gathered from a website listing, has been utilized by Hu et al. to identify stop words.

The features described by Hu et al. [29] are considered as *tf-idf* measures in this work, too, but instead of involving word stems in the feature set the word lemmata are used. Lemmatization is compared to stemming more accurate as it does a morphological analysis on words rather than a rough heuristic analysis to crop word ends. Stop words are recognized with the function word list of Mitton [52] and the modern *long stop*

[6] Stemming is the process of reducing a word to its word stem.

[7] Stop words, also called function words, are usually the most used words in a language and carry little or no information. They may be filtered out to reduce the feature space and improve the classification accuracy.

word list[8] of *ranks.nl*. The consolidation of both lists results in 732 stop words. Beside the four BOW features of Hu et al., an additional BOW feature is introduced, which is composed of all non-lemmatized words including stop words. Therefore, the incorporated feature models are:

1. **Entire words model:** includes all non-lemmatized words of a song text s.
2. **Stop words model:** includes only words of a song text s that are present in the stop word list.
3. **Content words model:** includes all non-lemmatized words of a song text s except stop words.
4. **Lemmatized content words model:** includes all lemmatized words of a song text s except function words.
5. **Part-of-speech tags model:** includes all POS tags assigned by the *Stanford Log-linear Part-Of-Speech Tagger* for the words of a song text s.

Text stylistics

Elementary text statistical/stylistic measures are extracted from lyrics based on word or character frequencies and are confirmed to be viable in mood [28] and genre classification [45, 46]. Mayer et al. [46] analyzed style properties for musical genre classification and discovered that plenty of exclamation marks are employed in Hip-Hop, Punk Rock and Reggae lyrics. Further, they noticed that Hip-Hop, Metal and Punk Rock apply more digits in lyrics than other genres and that Hip-Hop uses by far most words per minute. Text stylistics as individual features performed poorest in mood classification but together with ANEW features (second worst individual features) it gained similar results as the best feature type combination with only 37 instead of 107,000 dimensions [26]. The influence of text stylometrics on playlists are analyzed by means of features already applied in mood classification [26, 28], genre classification [17, 45, 46] and authorship attribution [63].
To be able to define parts of the following characteristics let $freq(t)$ denote the amount of occurrences of a token t within a song text s, whereat $t \in tokens(s)$. Moreover, let $isDigit(c)$ be the evaluation if a character c is a digit or not and let $isStopWord(t)$ indicate if a token t is present in the stop word list previously defined for the bag-of-words features. The duration of a song in minutes is expressed by $duration_{min}(s)$. Note that each feature is extracted from lowercased song texts.

[8] `http://www.ranks.nl/stopwords`, accessed on 2016-10-05

1. **Token count:** amount of total song text tokens. [26]

$$tokenCount(s) := |tokens(s)|$$

2. **Unique tokens ratio:** amount of unique tokens normalized with the total amount of song text tokens (indicates the vocabulary richness). [17, 26, 63]

$$uniqueTokensRatio(s) := \frac{|\{t \mid t \in tokens(s)\}|}{|tokens(s)|}$$

3. **Unique token bigrams ratio:** amount of unique token bigrams normalized with the total amount of song text token bigrams. [17, 63]

$$uniqueBigramsRatio(s) := \frac{|\{t \mid t \in bigrams(s)\}|}{|bigrams(s)|}$$

4. **Unique token trigrams ratio:** amount of unique token trigrams normalized with the total amount of song text token trigrams. [17, 63]

$$uniqueTrigramsRatio(s) := \frac{|\{t \mid t \in trigrams(s)\}|}{|trigrams(s)|}$$

5. **Average token length:** average amount of characters per token. [26, 46, 63]

$$averageTokenLength(s) := \frac{1}{|tokens(s)|} \cdot \sum\nolimits_{t \in tokens(s)} |t|$$

6. **Repeated token ratio:** proportion of repeated tokens. [26]

$$repeatedTokenRatio(s) := \frac{|tokens(s)| - |\{t \mid t \in tokens(s)\}|}{|tokens(s)|}$$

7. **Hapax legomenon ratio:** tokens that exactly occur once within a song text. [63]

$$hapaxLegomenonRatio(s) := \frac{|\{t \mid t \in tokens(s) \ \wedge \ freq(t) = 1\}|}{|tokens(s)|}$$

8. **Dis legomenon ratio:** tokens that exactly occur twice within a song text.

$$disLegomenonRatio(s) := \frac{|\{t \mid t \in tokens(s) \ \wedge \ freq(t) = 2\}|}{|tokens(s)|}$$

9. **Tris legomenon ratio:** tokens that exactly occur thrice within a song text.

$$trisLegomenonRatio(s) := \frac{|\{t \mid t \in tokens(s) \ \wedge \ freq(t) = 3\}|}{|tokens(s)|}$$

10. **Unique tokens per line:** amount of unique tokens normalized with the total amount of song text lines. [26, 46]

$$uniqueTokensPerLine(s) := \frac{|\{t \mid t \in tokens(s)\}|}{|lines(s)|}$$

11. **Average tokens per line:** average amount of tokens per line. [26]

$$averageTokensPerLine(s) := \frac{|tokens(s)|}{|lines(s)|}$$

12. **Line count:** total amount of song text lines. [17, 26]

$$lineCount(s) := |lines(s)|$$

13. **Unique line count:** amount of unique song text lines. [26]

$$uniqueLineCount(s) := |\{l \mid l \in lines(s)\}|$$

14. **Blank line count:** amount of blank song text lines. [26]

$$blankLineCount(s) := |(l \mid l \in lines(s) \wedge |l| = 0)|$$

15. **Blank line ratio:** amount of blank song text lines normalized with the total amount of song text lines. [26]

$$blankLineRatio(s) := \frac{|(l \mid l \in lines(s) \wedge |l| = 0)|}{|lines(s)|}$$

16. **Repeated line ratio:** amount of repeated song text lines normalized with the total amount of song text lines. [26]

$$repeatedLineRatio(s) := \frac{|lines(s)| - |\{l \mid l \in lines(s)\}|}{|lines(s)|}$$

17. **Words per minute:** amount of words spoken per minute. [26, 46]

$$wordsPerMin(s) := \frac{|tokens(s)|}{duration_{min}(s)}$$

18. **Lines per minute:** amount of lines spoken per minute. [26]

$$linesPerMin(s) := \frac{|lines(s)|}{duration_{min}(s)}$$

19. **Characters per minute:** amount of characters spoken per minute.

$$charactersPerMin(s) := \frac{1}{duration_{min}(s)} \cdot \sum_{t \in tokens(s)} |t|$$

20. **Exclamation marks ratio:** amount of occurrences of exclamation marks within a song text normalized with the total amount of song text characters. [26, 46]

$$exclMarksRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`!'})|}{\sum_{t \in tokens(s)} |t|}$$

21. **Question marks ratio:** amount of occurrences of question marks within a song text normalized with the total amount of song text characters. [46]

$$qstMarksRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`?'})|}{\sum_{t \in tokens(s)} |t|}$$

22. **Digits ratio:** amount of digits (0-9) occurrences within a song text normalized with the total amount of song text characters. [46]

$$digitsRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ isDigit(c)|}{\sum_{t \in tokens(s)} |t|}$$

23. **Colons ratio:** amount of occurrences of colons within a song text normalized with the total amount of song text characters. [46]

$$colonsRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`:'})|}{\sum_{t \in tokens(s)} |t|}$$

24. **Semicolons ratio:** amount of occurrences of semicolons within a song text normalized with the total amount of song text characters. [46]

$$semicolonsRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`;'})|}{\sum_{t \in tokens(s)} |t|}$$

25. **Hyphens ratio:** amount of occurrences of hyphens within a song text normalized with the total amount of song text characters. [26, 46]

$$hyphensRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`-'})|}{\sum_{t \in tokens(s)} |t|}$$

26. **Dots ratio:** amount of occurrences of dots within a song text normalized with the total amount of song text characters. [46]

$$dotsRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`.'})|}{\sum_{t \in tokens(s)} |t|}$$

27. **Commas ratio:** amount of occurrences of commas within a song text normalized with the total amount of song text characters. [46]

$$commasRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ = \ \text{`,'})|}{\sum_{t \in tokens(s)} |t|}$$

28. **Single quotes ratio:** amount of occurrences of single quotes (' and ‘) within a song text normalized with the total amount of song text characters. [46]

$$singleQuotesRatio(s) := \frac{\sum_{t \in tokens(s)} |(c \mid c \in chars(t) \ \wedge \ c \ \in \{`, '\}|}{\sum_{t \in tokens(s)} |t|}$$

29. **Stop words ratio:** amount of used stop words normalized with the total amount of song text tokens.

$$stopWordsRatio(s) := \frac{|(t \mid t \in tokens(s) \ \wedge \ isStopWord(t))|}{|tokens(s)|}$$

30. **Stop words per line:** amount of used stop words normalized with the total amount of song text lines.

$$stopWordsPerLine(s) := \frac{|(t \mid t \in tokens(s) \ \wedge \ isStopWord(t))|}{|lines(s)|}$$

5.2.2 Linguistic features

Particular linguistic features for lyrics have been examined by [16, 23, 45]. Fell [16] analyzed slang words, echoisms, and repetitive structures in lyrics and detected genre specific deviations. Differences in rhyming frequency and applied rhyming types per genre have been detected by Mayer et al. [45]. Subsequent linguistic features are adopted from [16] and described in detail.

Nonstandard words

Slang words contribute in identifying different types of genres as demonstrated by [16, 45]. For example, Mayer et al. [45] distinguished through tf-idf weighting that the words 'nuh', 'fi', and 'jah' are especially used in the genre of Reggae. Similar observations have been made by Fell [16]. Beside the ranking of words, Fell identified slang words and uncommon words through the usage of the resources *Urban Dictionary*[9] and *Wiktionary*[10].
The features of [16] and the ratio of unique uncommon words are considered in the experiments. Uncommon words are specified as terms that are not contained in the *Wiktionary*: $uncommonWords(s) := (t \mid t \in tokens(s) \ \wedge \ t \notin Wiktionary)$. Slang words are words not available in the *Wiktionary* but existent in the *Urban Dictionary*: $slangWords(s) := \{t \mid t \in tokens(s) \ \wedge \ t \notin Wiktionary \ \wedge \ t \in UrbanDictionary\}$. Based on these definitions three lyric characteristics are computed:

[9] `http://www.urbandictionary.com/`, accessed on 2016-09-08
[10] `http://en.wiktionary.org/`, accessed on 2016-09-08

1. **Uncommon words ratio:** amount of uncommon words normalized with the total amount of song text tokens

$$uncommonWordsRatio(s) := \frac{|uncommonWords(s)|}{|tokens(s)|}$$

2. **Unique uncommon words ratio:** fraction of unique uncommon words to all song text tokens

$$uniqUncommonWordsRatio(s) := \frac{|\{t \mid t \in uncommonWords(s)\}|}{|tokens(s)|}$$

3. **Slang words ratio:** proportion of slang words to all words

$$slangWordsRatio(s) := \frac{|slangWords(s)|}{|tokens(s)|}$$

Fell [16] pointed out that in the genre of Reggae several words are used that are not contained in the Urban Dictionary due to their flexible typing, e.g. "onno" is also spelled as "unnu". To be able to measure those words, the degree of lemmata of words which are equal to the words themselves is examined as unknown words are not lemmatized with *Stanford MorphaAnnotator* and consequently will stay the same. Fell already confirmed that the highest lemma ratio appears in Reggae but compared to other genres the difference isn't really significant. Nonetheless, the feature is taken into account.
For the following definition, let $lemma(t)$ be the function that determines the lemma for a token t.

4. **Lemma ratio:** percentage of words which are identical to their lemma

$$lemmaRatio(s) := \frac{|\{t \mid t \in tokens(s) \wedge t = lemma(t)\}|}{|\{t \mid t \in tokens(s)\}|}$$

Rhymes

Mayer et al. [46] extracted several rhyme descriptors from lyrics and discovered that the genres Folk and Reggae use most whilst R&B, Slow Rock and Grunge use least unique rhyme words. Moreover, Reggae, Grunge, R&B and Slow Rock exhibit a significant amount of blocks with subsequent pairs of rhyming lines (AABB rhymes). Highest usage of rhyming patterns arise in the genre of Reggae. The authors transcribed lyrics to a phonetic representation to be able to recognize rhyming words since from their point of view similar-sounding words are rather composed of identical or akin phonemes than lexical word endings. Related to [46], Hirjee and Brown [22] designed a system to automatically identify music rhymes in rap lyrics by employing a probabilistic model. The *Carnegie Mellon University (CMU) Pronouncing Dictionary* expanded

with slang terms along with the *Naval Research Laboratory's* text-to-phoneme rules are used to convert lyrics into sequences of phonemes and stress markings. Similarity scores for all syllable pairs are calculated by measuring the coexistence of phonemes in rhyming phrases. Phonemes which co-occur more often than excepted by chance receive positive scores, else negative scores. A rhyme is detected when the total score of a region of syllables matched to each other exceeds a particular threshold. Their experiments showed that their probabilistic model is superior in recognizing perfect and imperfect rhymes than other simpler rules-based approaches. Furthermore, estimated high-level rhyme scheme features (e.g., rhyme density) arose to be useful in examining characteristics about artists and genres. The 24 high-level features, depicted in Figure 5.1, can be computed with the *Rhyme Analyzer* tool from Hirjee and Brown [23] and are therefore included in the experiments, like Fell [16] did it for genre classification.

Feature	Description
Syllables per Line	Average number of syllables per line
Syllables per Word	Average word length in syllables
Syllable Variation	Standard deviation of line lengths in syllables
Novel Word Proportion	Average percentage of words in the second line in a pair not appearing in the first
Rhymes per Line	Average number of detected rhymes per line
Rhymes per Syllable	Average number of detected rhymes per syllable
Rhyme Density	Total number of rhymed syllables divided by total number syllables
End Pairs per Line	Percentage of lines ending with a line-final rhyme
End Pairs Grown	Percentage of rhyming couplets in which the second line is more than 15% longer (in syllables) than the first
End Pairs Shrunk	Percentage of rhyming couplets in which the first line is more than 15% shorter (in syllables) than the second
End Pairs Even	Percentage of rhyming couplets neither grown or shrunk
Average End Score	Average similarity score of line final rhymes
Average End Syl Score	Average similarity score per syllable in line final rhymes
Singles per Rhyme	Percentage of rhymes being one syllable long
Doubles per Rhyme	Percentage of rhymes being two syllables long
Triples per Rhyme	Percentage of rhymes being three syllables long
Quads per Rhyme	Percentage of rhymes being four syllables long
Longs per Rhyme	Percentage of rhymes being longer than four syllables
Perfect Rhymes	Percentage of rhymes with identical vowels and codas
Line Internals per Line	Number of rhymes with both parts falling in the same line divided by total number of lines
Links per Line	Average number of link rhymes per line
Bridges per Line	Average number of bridge rhymes per line
Compounds per Line	Average number of compound rhymes per line
Chaining per Line	Total number of words or phrases involved in chain rhymes divided by total number of lines

Figure 5.1: Description of 24 higher-level rhyme features computed by the *Rhyme Analyzer* tool. [24]

Echoisms

Fell [16] defines echoisms as expressions in which characters or terms are repeated in a particular manner and deals with three types of echoisms which are applied in lyrics: musical words, reduplications, and rhyme-alikes. Musical words like "uhhhhhhh", "aaahhh", or "shiiiiine" and reduplications such as "honey honey" or "go go go" are used to accentuate importance or emotion, or to bypass the problem that less syllables than notes are available to be sung. Rhyme-alikes including "burning turning" or "where were we" are not proper echoes, but are applied to produce uniformly sounding sequences and rhymes. Reduplications and rhyme-alikes are made up of at least two words unlike musical words which can be recognized from a single word, too. Thus, the feature set contains single and multi word echoisms which are computed by Fell [16] as subsequently described.
A word is classified as a musical word/single word echoism if the ratio of unique characters per word (letter innovation) is below an experimentally investigated hard threshold (0.4) or is lower than a soft threshold (0.5) and the word itself is not present in the *Wiktionary*. Hence, a token t is a musical word iff:

$$\begin{aligned} musicalWord(t) := & \frac{|\{chars(t)\}|}{|(chars(t))|} < 0.4 \vee \\ & \vee \frac{|\{chars(t)\}|}{|(chars(t))|} < 0.5 \ \wedge \ t \notin Wiktionary \end{aligned}$$

Consecutive pairs of a token sequence $(t_i)_{i=a}^{b} \subseteq l \wedge l \in lines(s)$ covered from the a-th to the b-th token of l form a multi word echoism if the edit distance between words is below 0.5. The edit distance $edit(A, B)$ employed by [16] is based on the Damerau-Levensthein edit distance $lev(A, B)$ and measures the proportion of operations needed to commute token A into token B and vice versa:

$$edit(A,B) := \sqrt{\frac{lev(A,B)}{|A|} \cdot \frac{lev(B,A)}{|B|}} = \sqrt{\frac{lev(A,B)^2}{|A| \cdot |B|}} = \frac{1}{\sqrt{|A| \cdot |B|}} \cdot lev(A,B)$$

Depending on the lemmata of constituent words a multi word echoism is further assigned to one of the aforementioned echoism types.

1. If all words in the multi word echosim exhibit the same lemma
   (a) and the lemma is listed in the *Wiktionary*, it is classified as a reduplication.
   (b) otherwise, it is classified as a musical word.

2. If not all words in the multi word echoism exhibit the same lemma

   (a) and all lemmata are listed in the *Wiktionary*, it is classified as a rhyme-alike.

   (b) and no lemma is present in the *Wiktionary*, it is classified as a musical word.

   (c) otherwise, it is undefinded.

The multi-word echoisms are counted per type, discriminating between a length of $= 1$, $= 2$ and > 2. Finally, the ratio of these values to all song texts tokens is computed and included in the experiments. The same applies to musical words (single word echoism).

Repetitive structures

Lyrics consist of more or less large proportions of replicated words or phrases which are not always exact duplicates but share at least a similar structure or wording. Fell [16] proposed a procedure to quantify the repetitive content in song texts by identifying identical line pairs and aligning similar successive and previous line pairs to form repetitive blocks. In the collaborative work of Fell and Sporleder [17] they adjusted this approach to enable more fuzzy matches as they do not search for exact copies of lines to build blocks. Based on lemma and POS bigrams, [16] defined a weighted similarity measure assembled of a word similarity and a structure similarity to identify related lines. Consider two lines x, y and let $bigrams_{lem}(l)$ represent the finite set of lemma bigrams of any song text line l, then the word similarity among x, y is specified as:

$$sim_{word}(x, y) = \frac{|bigrams_{lem}(x) \cap bigrams_{lem}(y)|}{max(|bigrams_{lem}(x)|, |bigrams_{lem}(y)|)}$$

The structural sameness of a line pair is investigated via part-of-speech tags. Thereby, for each line x, y a set of POS tag bigrams is generated which fulfill the requirement that their associated lemma bigrams belong to the symmetric difference of lemma bigram sets x, y (lemma bigram overlaps are discarded). Formally, a lemma bigram set x' for line x and line pair x, y consists only of bigrams which satisfy: $x' = disj(x, y) := \{b \mid b \in (bigrams_{lem}(x) \oplus bigrams_{lem}(y)) \wedge b \in bigrams_{lem}(x)\}$. Let $bigrams_{pos}(s)$ denote the set of corresponding POS tag bigram set for a lemma bigram set s then the structural similarity sim_{struct} of line pair x, y can be computed as described below. The structural similarity is squared to (heuristically) balance it with the word similarity since much less POS tags than words exist.

$$sim_{struct}(x,y) = \left(\frac{|bigrams_{pos}(disj(x,y)) \cap bigrams_{pos}(disj(y,x))|}{max(|bigrams_{pos}(disj(x,y))|, |bigrams_{pos}(disj(y,x))|)} \right)^2$$

Finally, the total similarity score $sim(x,y)$ for a line pair x, y arises from above mentioned similarity measures. The measures are weighted to enforce a higher significance on structural similarity if x and y use dissimilar tokens, otherwise the word similarity is ought to be more relevant.

$$sim(x,y) = sim^2_{word}(x,y) + (1 - sim_{word}) \cdot sim_{struct}(x,y)$$

After introducing the similarity measure of Fell [16] the process of distinguishing repetitive phrases can be amplified. The approach described in [16, 17] has been adopted to find repetitive phrases, but instead of comparing all line pairs of a song text, only lines from different segments are tried to align with the similarity measure. Hence, repetitive structures coexist at least once in two segments. Two lines x, y are aligned if $sim(x,y) \geq 0.25$. So, repetitive blocks are recognized by computing the similarity of all lines from different segments and finding consecutive and disjunctive ranges of aligned lines with maximum size afterwards. Samples of how lyrics are scanned for repetitive structures are illustrated in Figure 5.2 and Figure 5.3.

Ref.	Text	Similar lines		Blocks
1.1	Last christmas, I gave you my heart	(1.1, 2.1)		(1.1, 2.1)
1.2	But the very next day, you gave it away	(1.2, 2.2)	(1.2, 3.1)	(1.2, 2.2)
1.3	This year, to save me from tears	(1.3, 2.3)	(1.3, 3.2)	(1.3, 2.3)
1.4	I will give it to someone special	(1.4, 2.4)	(1.4, 3.3)	(1.4, 2.4)
2.1	Last christmas, I gave you my heart	(2.1, 1.1)		(2.1, 1.1)
2.2	But the very next day, you gave it away	(2.2, 1.2)	(2.2, 3.1)	(2.2, 1.2)
2.3	This year, to save me from tears	(2.3, 1.3)	(2.3, 3.2)	(2.3, 1.3)
2.4	I will give it to someone special	(2.4, 1.4)	(2.4, 3.3)	(2.4, 1.4)
3.1	But the very next day, you gave it away	(3.1, 1.2)	(3.1, 2.2)	(3.1, 1.2)
3.2	This year, to save me from tears	(3.2, 1.3)	(3.2, 2.3)	(3.2, 1.3)
3.3	I will give it to someone special	(3.3, 1.4)	(3.3, 2.4)	(3.3, 1.4)

Figure 5.2: Recognizing repetitive structures in lyrics. Detect similar lines and find maximum sized blocks of similar lines. Lines within a segment can belong at most to one block.

Based on the located blocks, Fell [16] educed eight measures to represent phrase repetitions. Let $blocks(s)$ be the collection of repetitive blocks

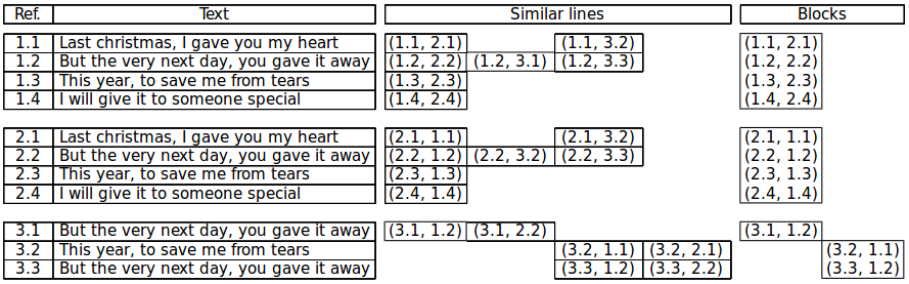

Figure 5.3: Variegated example of Figure 5.2 to get a finer grasp on detecting repetitive structures.

comprised in a song text s. Then the features, which are included in this study, are defined as:

1. **Block count:** amount of repetitive blocks

$$blockCount(s) := |blocks(s)|$$

2. **Average block size:** average amount of lines comprised in a block

$$averageBlockSize(s) := \frac{1}{|blocks(s)|} \cdot \sum_{b \in blocks(s)} |b|$$

3. **Blocks per line:**

$$blocksPerLine(s) := \frac{|blocks(s)|}{|lines(s)|}$$

4. **Repetitivity:** amount of lines which belong to a repetitive block

$$repetitivity(s) := \frac{|(l \mid l \in lines(s) \wedge \exists b \in blocks(s) \wedge l \in b)|}{|lines(s)|}$$

5. **Block reduplication:** ratio of unique blocks to all blocks

$$blockReduplication(s) := \frac{|\{b \mid b \in blocks(s)\}|}{|blocks(s)|}$$

6. **Type token ratio of lines:**

$$typeTokenRatio_{lines}(s) := \frac{|\{l \mid l \in lines(s)\}|}{|lines(s)|}$$

7. **Type token ratio inside lines:**[11]

$$typeTokenRatio_{inlines}(s) := \frac{1}{|lines(s)|} \cdot \sum_{l \in lines(s)} \frac{|\{lemma(t) \mid t \in l\}|}{|l|}$$

[11] Note that [16] divided by $|\{t \mid t \in l\}|$ instead of $|l|$.

8. **Average alignment score:** average line alignment score of all repetitive lines.

The genre classification evaluations of Fell [16] revealed that lines are seldom duplicated in Rap songs and block duplicates are less used in Metal. Highest inline type token ratios can be found in Country and fewest in Metal.

5.2.3 Semantic features

A semantic analysis of lyrics is potentially profitable as a bunch of semantics can only be inferred from lyrics (i.e., the topic of a song) and not from audio signals [45] or can be partly derived from lyrics like in the case of mood [38]. Especially the mood analysis may be beneficial for playlists classification as many playlists are created on basis of emotional states [34]. Similar to [16, 26], semantic lexicons are employed to distinguish the imageries and emotions of song texts.

Regressive imagery

The psychologist Colin Martindale [43, 44] established a dictionary based on theoretical and empirical researches on regressive thought to quantify the primordial and conceptual cognition in texts. The primordial cognition is present in dreams, fantasies, and reveries and can be specified as concrete, associative, and almost unrealistic whereas conceptual cognition is abstract, logical, and reality oriented and tends to solve problems [43, 44, 56]. The so-called *Regressive Imagery Dictionary* (RID), available from [56], consists of ~3,000 English words belonging to the main categories primordial cognition, conceptual cognition, and emotions, each in turn divided into various subcategories. There are 29 primordial thought, 7 conceptual thought and 7 emotions subcategories which are illustrated in Figure 5.4 - 5.7.

$$\text{Regressive Imagery} := \begin{cases} \text{Conceptual Thought} \\ \text{Emotions} \\ \text{Primordial Thought} \end{cases}$$

Figure 5.4: Main categories of the regressive imagery.

Fell [16] excluded sparsely represented classes in his experiments and observed that Rap noticeable exert more words of the categories *Primordial Thought→Need→Sex* and *Primordial Thought→ Need → Anal-*

Conceptual Thought →	Abstractions	→ {figure, reality, think, ...}
	Instrumental Behavior	→ {apply, gain, win, ...}
	Moral Imperative	→ {honest, law, should, ...}
	Order	→ {ordinal, balance, uniform, ...}
	Restraint	→ {bar, jail, police, ...}
	Social Behavior	→ {ask, explain, talk, ...}
	Temporal References	→ {ago, date, full-time, ...}

Figure 5.5: Sub categories of conceptual thought.

Emotions →	Affection	→ {attractive, cherish, kindness, ...}
	Aggression	→ {axes, battle, kill, ...}
	Anxiety	→ {cares, fearful, scare, ...}
	Expressive Behavior	→ {cry, roar, yell, ...}
	Glory	→ {fame, hero, rich, ...}
	Positive Affect	→ {flirt, grateful, love, ...}
	Sadness	→ {pain, sad, unhappy, ...}

Figure 5.6: Sub categories of emotions.

ity than other genres. Moreover, most aggressive expressions (*Emotions→Aggression*) can be found in Metal. Highest amount of *Emotions→Glory* and *Primordial Thought→Icarian Imagery→Water* contents are manifested in religious songs and Folk, respectively. In contrast to Fell, the lyrics in the own specifically generated corpus posses a valuable amount of words for each *Regressive Imagery Dictionary* class. Hence, word frequencies normalized with the total song text tokens of all categories are involved in the experiments.

SentiStrength

The SentiStrength[12] application is optimized for general social web text and consists of a lexicon, an emoticon list, an idiom list, and additional sentiment rules to detect the sentiment strength of a text. Based on the Linguistic Inquiry and Word Count (LIWC) [54] dictionary and the General Inquirer list of sentiment terms [64], the lexicon contains 2,310 sentiment words and word stems and provides an average positive or negative sentiment score for each included word. The sentiment score of

[12] `http://sentistrength.wlv.ac.uk/`, accessed on 2017-03-01

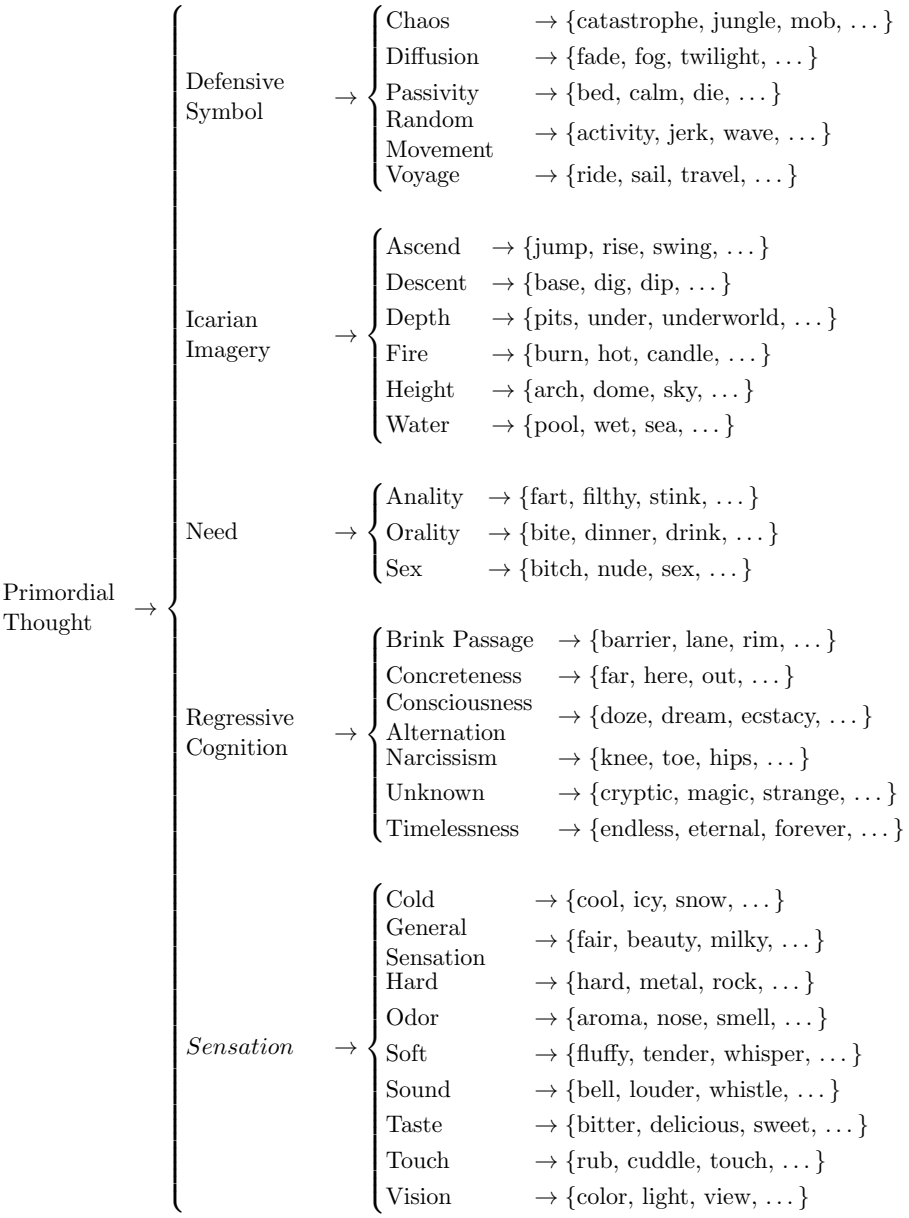

Figure 5.7: Sub categories of primordial thought.

a text conforms to the maximum positive and negative magnitude of any word element (dual positive/negative scoring), except some of the additional rules come to bear due to the presence of negations, booster words, emoticons, exclamtion marks, repeated punctuations, etc. Despite the fact that SentiStrength has its weaknesses in recognizing sarcasm, it still

provides near-human accuracies on general short social web texts. [67]
As mood is strongly considered during playlist creation, it is worthwhile to determine it in lyrics.
Let $sentiStrength_{pos}(t), sentiStrength_{neg}(t)$, and $sentiStrength_{neu}(t)$ be the SentiStrength positive, negative, and neutral score for a text t, respectively, then the emotion scores for a song text s are calculated as follows:

1. **Positive sentiment ratio:**

$$positiveSentimentRatio(s) := \frac{\sum_{l \in lines(s)} sentiStrength_{pos}(l)}{|lines(s)|}$$

2. **Negative sentiment ratio:**

$$negativeSentimentRatio(s) := \frac{\sum_{l \in lines(s)} sentiStrength_{neg}(l)}{|lines(s)|}$$

3. **Neutral sentiment ratio:**

$$neutralSentimentRatio(s) := \frac{\sum_{l \in lines(s)} sentiStrength_{neu}(l)}{|lines(s)|}$$

AFINN

AFINN [53] is a sentiment lexicon that consists only of English words which are rated for valence with an integer score between -5 (most negative) and +5 (most positive). The current version of *AFINN* (AFINN-111) contains 2,477 manually labeled words and phrases and is especially designed for microblogs. Conducted experiments from Nielsen, the originator of the *AFINN lexicon*, showed that the *AFINN lexicon* slightly outperforms the *ANEW lexicon (Affective Norms for English Words)* with respect to the mood detection of Twitter microblogs, but provides inferior results compared to *SentiStrength*. To get an additional grasp of the emotion exposed by lyrics the *AFINN* valence score for lyrics is further dissected.
Let $score(t)$ return the valence score of a token t out of the *AFINN* lexicon. Then the overall valence score $valence(s)$ for a song text s is defined as:

$$valence(s) := \frac{\sum_{t \in tokens(s) \ \wedge \ t \in AFINN} score(t)}{|(t \mid t \ \in \ tokens(s) \ \wedge \ t \in AFINN)|}$$

Opinion Lexicon

The *Opinion Lexicon* has been modeled by Hu and Liu [25] and includes ~4,780 negative and ~2,000 positive adjectives. Therefore, they crawled online customer product reviews, determined sentences describing product features, and defined opinion words as the adjacent adjectives of product feature nouns. The semantic orientation of opinion words has been investigated by devoting the adjective synonym and antonym sets in WordNet [51]. In contrast to *SentiStrength* and *AFINN* the *Opinion Lexicon* does not use a scoring range. Let $score(t)$ be a function that returns 1 if token t is a positive adjective or -1 if it is a negative adjective. Then, an opinion value for a song text s results from:

$$opinion(s) := \frac{\sum_{t \in tokens(s) \wedge t \in OpinionLexicon} score(t)}{|(t \mid t \in tokens(s) \wedge t \in OpinionLexicon)|}$$

VADER

VADER (Valence Aware Dictionary for sEntiment Reasoning) [31] is a simplistic rule-based system for general sentiment analysis. It uses a valence-based lexicon, comprised of 7,517 lexical features, and five generalizable rules, based on grammatical and syntactical cues, that consider word-order sensitive relations amongst terms to compute both sentiment polarity and sentiment intensity for a text. Experiments revealed that in the domains of tweets, movie reviews, product reviews, and opinion news articles the system performs equally well as or even better than other sentiment analysis tools (i.e., GI, ANEW, LIWC, ...), although it was especially adjusted for sentiments conveyed on social media platforms.
The fully open-sourced VADER tool [30] computes four different scores for a particular text: compound, positive, negative and neutral opinion score. Regarding to [30], the compound score results from the overall sum of all valence scores of each word in the lexicon, which are adequately changed by the defined rules, and afterwards normalized to retrieve a value between -1 (most extreme negative) and +1 (most extreme positive). The positive, neutral, and negative scores reveal the proportions of text that reside in each class. Therefore, the sum of these scores is equal to 1.
Let $vader_{com}(t)$, $vader_{pos}(t)$, $vader_{neg}(t)$, and $vader_{neu}(t)$ be the VADER compound, positive, negative, and neutral score for a text t, respectively. Hence, four VADER sentiment features are derived for a song text s:

1. **Positive sentiment ratio:**

$$positiveSentimentRatio(s) := \frac{\sum_{l \in lines(s)} vader_{pos}(l)}{|lines(s)|}$$

2. **Negative sentiment ratio:**

$$negativeSentimentRatio(s) := \frac{\sum_{l \in lines(s)} vader_{neg}(l)}{|lines(s)|}$$

3. **Neutral sentiment ratio:**

$$neutralSentimentRatio(s) := \frac{\sum_{l \in lines(s)} vader_{neu}(l)}{|lines(s)|}$$

4. **Compound sentiment:** normalized, weighted composite score

$$compoundSentimentScore(s) := \frac{\sum_{l \in lines(s)} vader_{com}(l)}{|lines(s)|}$$

5.2.4 Syntactic features

Syntactic informations are commonly extracted out of texts to reveal stylistic characteristics of authors [63] and may be helpful in assigning tracks to playlists, too, as artist attributes maybe relevant properties for users to create personal playlists [34]. Part-of-speech features have already been used by Mayer et al. [45] to categorize the genre of lyrics. Their evaluation showed that adverbs are higher employed in Pop and R&B and less frequently used in Hip-Hop. Fell [16] also discovered discriminative syntactic features for genre classification. Primarily relying on the study of Fell [16], pronouns, part-of-speech, text chunks and past tense features are derived from lyrics.

Pronouns

Based on the count of pronouns, Fell [16] introduced seven features to improve the genre classification of songs. Lyric characteristics have been deduced by grouping similar pronouns together and counting their occurrences in a song text. The total amount of pronouns occurrences is distinguished for the five abstract groups *I*, *You*, *It*, *We*, and *They*, which are compound as defined below:

$$\begin{aligned}
I &\rightarrow \{\text{I, me, my, mine, myself}\} \\
You &\rightarrow \{\text{you, your, yours, yourself, yourselves}\} \\
It &\rightarrow \{\text{he, him, his, himself, she, her, hers, herself, it, its, itself}\} \\
We &\rightarrow \{\text{we, us, our, ours, ourselves}\} \\
They &\rightarrow \{\text{they, them, their, theirs, themselves}\}
\end{aligned}$$

Experiments from Fell depicted that pronouns of the abstract group *They* are noticeable often used in the genre of Rap and concludes this with the fact that in many Rap songs someone is speaking about other people. In the genre of Blues the *We*-pronouns are little used as it might be the case that "you and me" is utilized instead of "we". Fell relies therefore on the relatively increased usage of pronouns from the groups *I* and *You*.

From the abstract pronoun group counts the additional lyric properties *excentricity* and *IvU* are imposed by Fell. The first property aims to determine how far song writers are concerned with themselves in contrast to other persons or groups they do not belong to, whereas the second property identifies if the narrators put the emphasis on the "I" or the "you" in their texts.

Let $f(x)$ denote the frequency in which an abstract pronoun group $x \in \{I, You, It, We, They\}$ is contained in a song text s, then the latter features are formally defined as follows:

$$excentricity(s) := \begin{cases} f_s(I) + f_s(We) \geq & \\ \qquad 2 \cdot (f_s(You) + f_s(It) + f_s(They)) & \rightarrow 1 \\ f_s(You) + f_s(It) + f_s(They) \geq & \\ \qquad 2 \cdot (f_s(I) + f_s(We)) & \rightarrow 0 \\ else & \rightarrow \frac{1}{2} \end{cases}$$

$$IvU(s) := \begin{cases} f_s(I) \geq 2 \cdot f_s(You) & \rightarrow 1 \\ f_s(You) \geq 2 \cdot f_s(I) & \rightarrow 0 \\ else & \rightarrow \frac{1}{2} \end{cases}$$

For the purpose of this work, the absolute frequencies of all five abstract pronoun groups normalized by all song text tokens are considered as features as well as the *excentricity* and *IvU* values.

Part-of-speech

Mayer et al. [46] examined that the combination of rhyme, part-of-speech, and simple text statistic features can improve genre classification results. Also, Fell [16] demonstrated that POS tags contribute in genre categorization tasks. Syntactic features are derived via the occurrence frequency of each part-of-speech tag type within a song text as well as via the amount of six POS tag groups as introduced by [16]. The groups are made up of similar POS tags and are called verbs (V), nouns ($NOUN$), adjectives (ADJ), adverbs (ADV), wh-questions (WH), and special

characters (?) and are defined as follows:[13]

$$
\begin{aligned}
\text{V} &\rightarrow \{\text{VB, VBG, VBP, VBZ, VBN, VBD}\} \\
\text{NOUN} &\rightarrow \{\text{NN, NNS, NNP, NNPS}\} \\
\text{ADJ} &\rightarrow \{\text{JJ, JJR, JJS}\} \\
\text{ADV} &\rightarrow \{\text{RB, RBR, RBS}\} \\
\text{WH} &\rightarrow \{\text{WDT, WP, WP\$, WRB}\} \\
? &\rightarrow \{\#, \$, ., ,, :, \text{-RRB-, -LRB-, “, ”}\}
\end{aligned}
$$

All features are normalized with the total amount of tokens of a particular song text. The study of Fell [16] revealed that foreign words (FW) are most common in Reggae since this genre commonly uses words with a nonstandard spelling which are unknown to the POS tagger and hence “foreign”.

Text chunks

Text chunking, also known as shallow parsing or partial parsing, is a technique to recover syntactic information from text efficiently [1] and is for instance used in the realm of authorship attribution [63]. A popular partial parsing tool is the *Apache OpenNLP Chunker*[14] which recognizes syntactically correlated parts of words, like noun, verb or other phrases. The chunker requires as input a POS tagged text and outputs word phrases labeled with a tag out of the *Penn Treebank* phrase level tag set (see Appendix A.2.2). Text chunks are computed for each song text and afterwards processed to model syntactical features. The amount of each phrase tag as well as the average length of each phrase tag is investigated to characterize lyrics. Moreover, two additional features are derived through the approach from Fell [16] who formed new phrase groups by combining the subsequent tags:

$$
\begin{aligned}
\text{AP} &\rightarrow \{\text{ADJP, ADVP}\} \\
\text{XP} &\rightarrow \{\text{CONJP, INTP, LST, PRT, UCP}\}
\end{aligned}
$$

Thus, the amount of each phrase tag group is considered as a feature. Again, to facilitate a comparison of values all tag counts are normalized with the amount of tokens of each analyzed song text.

[13] ( ), [ ], { } are mapped to -LRB-, -RRB-, for additional information, refer to `http://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/process/PTBTokenizer.html`, accessed on 2017-03-01

[14] `https://opennlp.apache.org/documentation/1.5.2-incubating/manual/opennlp.html\#tools.chunker`, accessed on 2016-08-13

Past Tense

To indicate if a song deals with past or present/future happenings or occurrences, Fell [16] computed the proportion of past tense verb forms to all verb forms for each song text. He discovered that past tense forms are rarely found in Reggae and supposes that this is partly caused by nonstandard word spellings for which the past tense of "to be" is not correctly recognized as such. The tense of a verb is detected by the *Stanford POS tagger* that labels each word with one tag out of the *Penn Treebank* part-of-speech tag set. The past tense ratio feature of [16] is assessed as described below, whereat the function $tag(t)$ represents the *Penn Treebank* part-of-speech tag for a token t.

$$
\begin{aligned}
Verbs &:= \{\text{VB, VBG, VBP, VBZ, VBD, VBN}\} \\
VerbsPast &:= \{\text{VBD, VBN}\} \\
pastTenseRatio(s) &:= \frac{|(t \mid t \in tokens(s) \wedge tag(t) \in VerbsPast)|}{|(t \mid t \in tokens(s) \wedge tag(t) \in Verbs)|}
\end{aligned}
$$

Chapter 6

Evaluation

Exploring characteristics of playlists leads to an understanding of how users create music compilations which in turn may be applied to improve playlist recommender systems or other music classification tasks. To disclose characteristics of playlists, this chapter investigates the properties shared among most tracks within a playlist by employing a data classification approach on a per-playlist basis. Thus, in a preliminary step, a multimodal playlist test/training collection is assembled based on the novel dataset consisting only of tracks with acoustic and lyric features to ensure reasonable results. With regards to the data classification results across all playlists, a minimum size of characteristic playlists is figured out and most important individual attributes are distinguished through information gain computation. Moreover, an attribute selection experiment is conducted to improve classification results.

6.1 Test/training data collection

For 17,889 playlists consisting of 671,650 total tracks, 587,400 audio features and 226,747 lyrics could be acquired. Consequently, mandatory lyrics and audio data is missing for most of the tracks. Due to that reason, the playlist test collection needs to be adapted. Tracks without available audio features and/or lyrics are removed from the test corpus and related playlists are accordingly modified. The overall distribution of playlist sizes, pictured in Figure 6.1, significantly changed compared to the original distribution illustrated in Figure 4.1. About 4,400 playlists do not possess tracks anymore and ~1,300 single sized playlists exist now.
Playlists which are comprised of at most one track are excluded as they do not satisfy the definition of playlists anymore. Hence, 12,159 playlists with 203,217 unique tracks created by 953 users remain. An overview of the adapted playlist data collection is given in Table 6.1.

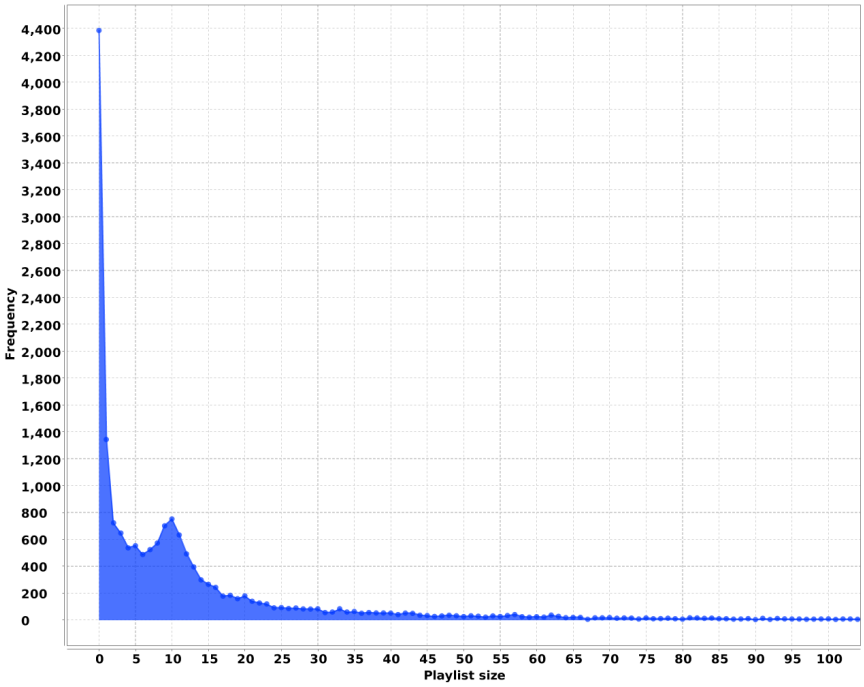

Figure 6.1: Distribution of playlist sizes after removing tracks without mandatory song texts and/or audio features.

Test cases are generated on per-playlist basis. Each is constructed by doubling the playlist length by adding s random tracks which are not present in the original playlist. Hence, binary classification experiments can be conducted on the resulting 12,159 test cases with equally weighted correct and incorrect tracks.

	Characteristic	**Value**
	Users	953
	Playlists	12,159
	Tracks	203,188
Amount of playlists containing	2–5 tracks	2,464
	6–10 tracks	3,043
	11–25 tracks	3,615
	26–50 tracks	1,404
	51–100 tracks	858
	> 100 tracks	775

Table 6.1: Overview of the test/training data collection.

6.2 Classification algorithms

The state-of-the-art classification algorithms *BayesNet*[1], *Naïve Bayes*[2], *kNN*[3], *LibLinear*[4], *LibSVM (C)/LibSVM (nu)*[5], *PART*[6] and *J48*[7] decision trees which have already been applied in the realm of music classification [26, 47] are employed to reveal properties of playlists. All of these algorithms are provided by Weka and are utilized with their default configuration settings, merely the SVM versions (LibLinear, LibSVM (C), LibSVM (nu)) are further parameterized to normalize the input data[8].

6.3 Coherent features sets

Distinguishing coherent features between tracks within playlists is only worthwhile iff classifiers who incorporate with the extracted features are able to outperform the baseline in assigning tracks to playlists. As described above, all test cases consist of equally distributed correct/incorrect tracks, so the baseline is naturally 50%. A 5-fold cross validation has been accomplished for all test cases to evaluate the performance of each classifier and feature type combination. Table 6.2 depicts the obtained averaged accuracies and reveals that the classifiers provide in most cases superior classification results than the baseline.
The best results of each classifier indicate that acoustic features (AU) represent the main characteristics of playlists except for Bayes Net which represents playlists best with acoustic, linguistic, and syntactic features (AU+LI+SY). Highest accuracy is gained by LibLinear (~67%) and worst by LibSVM (C) (~55%) who outperforms the baseline only twice (AU and AU+LI). Best accuracy result (~61%) by solely relying on lyrics features is attained with LibLinear and the combination of all lexical, linguistic, semantic and syntactic lyrics features (LX+LI+SE+SY).

[1] `http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/BayesNet.html`, accessed on 2017-07-30

[2] `http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayes.html`, accessed on 2017-07-30

[3] `http://weka.sourceforge.net/doc.dev/weka/classifiers/lazy/IBk.html`, accessed on 2017-07-30

[4] `http://weka.sourceforge.net/doc.stable/weka/classifiers/functions/LibLINEAR.html`, accessed on 2017-07-30

[5] `http://weka.sourceforge.net/doc.stable/weka/classifiers/functions/LibSVM.html`, accessed on 2017-07-30

[6] `http://weka.sourceforge.net/doc.dev/weka/classifiers/rules/PART.html`, accessed on 2017-07-30

[7] `http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html`, accessed on 2017-07-30

[8] Classification pre-tests indicated a performance improvement of all SVM versions after turning on the input normalization.

Feature set	Bayes Net	J48	kNN	LibLinear	LibSVM (C)	LibSVM (nu)	Naïve Bayes	PART	Max.
AU	*0.618*	***0.608***	***0.638***	***0.669***	***0.548***	***0.653***	***0.636***	***0.607***	***0.669***
AU+LI	0.627	0.587	0.600	0.641	0.508	0.624	0.592	0.587	**0.641**
AU+LI+SY	**0.630**	0.576	0.585	0.626	0.498	0.612	0.571	0.577	**0.630**
AU+LI+SE+SY	0.629	0.570	0.585	0.624	0.497	0.610	0.560	0.570	**0.629**
ALL	*0.629*	*0.565*	*0.550*	*0.619*	*0.487*	*0.603*	*0.569*	*0.565*	***0.629***
AU+LI+SE	0.627	0.573	0.588	0.622	0.499	0.607	0.561	0.573	**0.627**
AU+LX+LI+SE	0.627	0.565	0.543	0.616	0.484	0.601	0.567	0.566	**0.627**
AU+LX+SE+SY	0.627	0.565	0.546	0.616	0.485	0.601	0.568	0.565	**0.627**
AU+LX+LI+SY	0.626	0.566	0.546	0.616	0.485	0.603	0.566	0.567	**0.626**
AU+LX+SY	0.624	0.566	0.541	0.613	0.483	0.599	0.565	0.567	**0.624**
AU+LX+LI	0.624	0.566	0.538	0.613	0.482	0.600	0.565	0.567	**0.624**
AU+LX+SE	0.624	0.565	0.538	0.612	0.482	0.598	0.565	0.565	**0.624**
AU+SY	0.622	0.578	0.580	0.618	0.491	0.604	0.572	0.578	**0.622**
AU+SE+SY	0.621	0.569	0.580	0.614	0.490	0.600	0.558	0.570	**0.621**
AU+LX	0.621	0.566	0.532	0.609	0.480	0.598	0.564	0.566	**0.621**
AU+SE	0.616	0.573	0.581	0.608	0.489	0.593	0.559	0.573	**0.616**
LX+LI+SE+SY	*0.603*	*0.545*	*0.544*	*0.608*	*0.484*	*0.595*	*0.560*	*0.546*	***0.608***
LX+LI+SY	0.598	0.545	0.540	0.604	0.483	0.594	0.558	0.546	**0.604**
LX+SE+SY	0.599	0.544	0.539	0.603	0.482	0.592	0.559	0.544	**0.603**
LX+LI+SE	0.599	0.545	0.536	0.603	0.481	0.592	0.558	0.545	**0.603**
LX+SY	0.594	0.544	0.535	0.600	0.480	0.590	0.556	0.544	**0.600**
LX+LI	0.593	0.545	0.531	0.600	0.478	0.591	0.556	0.545	**0.600**
LX+SE	0.593	0.543	0.531	0.599	0.478	0.589	0.557	0.543	**0.599**
LX	0.586	0.542	0.526	0.595	0.476	0.588	0.556	0.543	**0.595**
LI+SE+SY	0.585	0.536	0.566	0.595	0.482	0.582	0.540	0.536	**0.595**
LI+SY	0.577	0.537	0.561	0.589	0.479	0.577	0.541	0.538	**0.589**
LI+SE	0.566	0.532	0.556	0.582	0.476	0.567	0.532	0.532	**0.582**
LI	0.547	0.534	0.552	0.579	0.470	0.564	0.538	0.534	**0.579**
SE+SY	0.566	0.527	0.555	0.577	0.472	0.565	0.533	0.528	**0.577**
SY	0.555	0.526	0.548	0.568	0.466	0.557	0.533	0.527	**0.568**
SE	0.525	0.517	0.535	0.547	0.454	0.535	0.517	0.516	**0.547**
Baseline	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	
Max.	***0.630***	***0.608***	***0.638***	***0.669***	***0.548***	***0.653***	***0.636***	***0.607***	

Table 6.2: Average accuracies of all test cases across all feature set combinations and classifiers. Results are ordered by the maximum achieved accuracy of each feature set combination.

Overall worst accuracies are obtained with semantic features (SE) followed by syntactic (SY) and linguistic features (LI). They seem not to serve as primary link among tracks and playlists.
Supplementing audio features with lyric features decreases the accuracy of all classifiers except for the Bayes Net version. Hence, an orthogonality of content-based and lyrics features can only be assessed for the latter classifier. All feature set combinations containing audio features are superior than those without.

6.4 Minimum/maximum playlist size

The previous results are further analyzed to determine if a minimum amount of tracks is required to characterize playlists well. Due to data

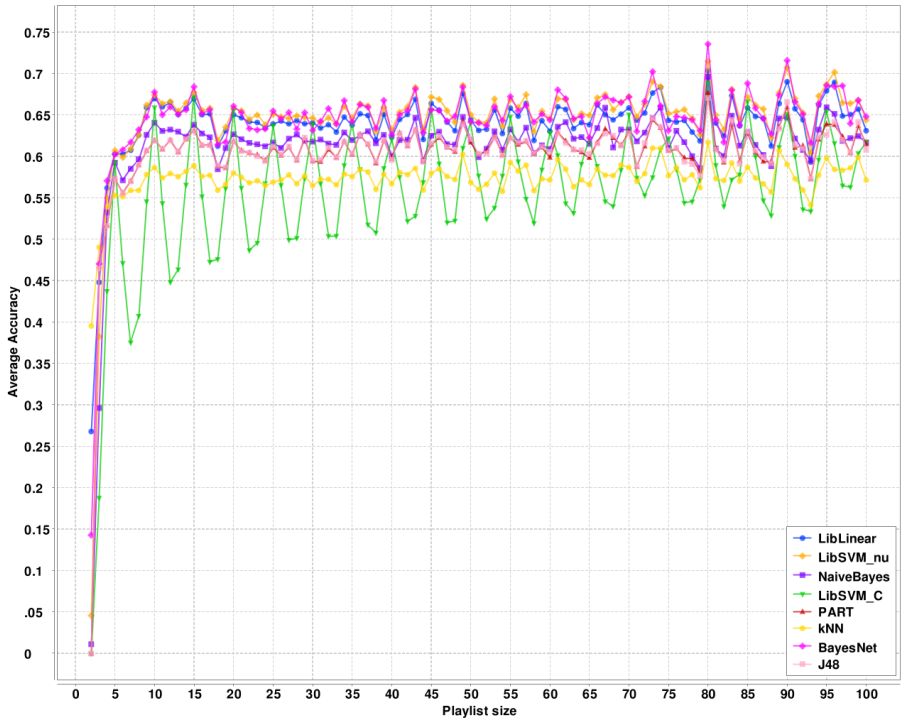

Figure 6.2: Average accuracies of all features sets and utilized classifiers across different playlist sizes.

set limitations it is not possible to investigate if there is an upper bound for tracks which are still characteristic since only few compilations of playlists consisting of at least 100 tracks are available. Thus, the maximum amount of tracks is set to at most 100 tracks for representable results.

The averaged accuracies of all feature sets grouped by classifiers and the amount of tracks which reside in each playlist is illustrated in Figure 6.2. It discloses that a minimum amount of tracks is necessary to distinguish playlists properly.

At least four tracks, precisely depicted in Table 6.3, are required to surpass the baseline by each classifier except for LibSVM (C) which exceeds the baseline for the first time with five tracks but provides partly inferior results up to a quantity of 28 tracks. According to Figure 6.2, a minimum amount of eight tracks is viable for all classifiers as an amount of seven tracks generates a significant better accuracy than the baseline (except LibSVM (C)) but leads to overall worse results than playlists with a size of 8 to 100 tracks.

Excluding playlists with less than 8 and more than 100 tracks reduces the test collection from 12,159 to 7,905 playlists. Those are created

Tracks	Bayes Net	J48	kNN	LibLinear	LibSVM (C)	LibSVM (nu)	Naïve Bayes	PART
2	0.143	0.000	0.396	0.268	0.000	0.046	0.011	0.000
3	0.471	0.465	0.491	0.448	0.188	0.383	0.297	0.468
4	**0.571**	**0.517**	**0.540**	**0.562**	0.438	**0.550**	**0.532**	**0.517**
5	0.603	0.573	0.553	0.603	**0.593**	0.607	0.593	0.573
6	0.608	0.556	0.552	0.603	0.471	0.599	0.571	0.556
7	0.618	0.570	0.559	0.608	0.375	0.610	0.585	0.570
8	0.633	0.590	0.559	0.626	0.408	0.626	0.597	0.590
9	0.648	0.607	0.578	0.659	0.546	0.662	0.626	0.607
10	0.678	0.619	0.587	0.670	0.659	0.674	0.641	0.620
11	0.651	0.609	0.574	0.660	0.544	0.664	0.631	0.609
12	0.660	0.620	0.580	0.665	0.448	0.666	0.633	0.620
Baseline	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***	***0.500***

Table 6.3: Average accuracies of all features sets and classifiers combinations across playlists consisting of maximum twelve tracks.

by 876 users and contain 162,376 unique tracks. Table 6.4 depicts the impact on the averaged results after constraining playlists and discloses the relative improvement (Δ_{max}) of each classifier compared to their maximum average accuracy among all playlists listed in Table 6.2.
Higher accuracies are gained by all classifiers after constraining playlists. Best accuracy of 70% is scored by LibLinear and LibSVM (nu) incorporating only acoustic features. Similar results are achieved by the Bayes Net (69.4%) and Naïve Bayes (69.6%) algorithms. The highest relative improvement (Δ_{max}) is attained by the LibSVM (C) classifier, which enhances the maximum average accuracy by 7.3% from 54.8% to 62.1%. Acoustic features have still the most cohesive power. The highest accuracy of a non-acoustic feature combination is again achieved by all lyric features (LX+LI+SE+SY) and is merely ~3% inferior than the best results of LibLinear and LibSVM (nu).

6.5 Most discriminative individual features

In this final experiment the relevancy of individual features is analyzed and feature selection is applied to further improve the accuracies of all classifiers. Similar to [45], the information gain selector[9] of Weka, again with standard configuration, is utilized to detect the importance of each extracted attribute. The information gains are determined across all test cases and feature set combinations. Computing the information gain mean of all features and ranking them by the highest measures

[9] `http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html`, accessed on 2017-07-30

Feature set	Bayes Net	J48	kNN	LibLinear	LibSVM (C)	LibSVM (nu)	Naïve Bayes	PART	Max.
AU	*0.645*	***0.661***	***0.660***	***0.700***	***0.621***	***0.700***	***0.696***	***0.660***	***0.700***
ALL	***0.694***	*0.623*	*0.575*	*0.675*	*0.552*	*0.677*	*0.636*	*0.624*	***0.694***
AU+LX+LI+SE	0.692	0.624	0.566	0.671	0.548	0.674	0.632	0.624	**0.692**
AU+LX+LI+SY	0.691	0.625	0.570	0.672	0.550	0.675	0.631	0.625	**0.691**
AU+LX+SE+SY	0.691	0.624	0.570	0.670	0.550	0.673	0.633	0.624	**0.691**
AU+LX+LI	0.689	0.627	0.558	0.667	0.546	0.671	0.626	0.627	**0.689**
AU+LX+SE	0.689	0.624	0.559	0.665	0.546	0.669	0.627	0.624	**0.689**
AU+LX+SY	0.688	0.625	0.563	0.667	0.547	0.670	0.628	0.625	**0.688**
AU+LX	0.686	0.626	0.552	0.660	0.543	0.666	0.622	0.626	**0.686**
AU+LI	0.663	0.642	0.621	0.674	0.571	0.681	0.664	0.642	**0.681**
AU+LI+SE+SY	0.677	0.627	0.602	0.665	0.558	0.675	0.630	0.627	**0.677**
AU+LI+SY	0.673	0.633	0.605	0.664	0.558	0.673	0.643	0.632	**0.673**
AU+LI+SE	0.669	0.628	0.607	0.659	0.561	0.670	0.629	0.628	**0.670**
LX+LI+SE+SY	*0.664*	*0.600*	*0.568*	*0.665*	*0.549*	*0.668*	*0.627*	*0.600*	***0.668***
AU+SE+SY	0.665	0.625	0.597	0.652	0.550	0.661	0.625	0.625	**0.665**
LX+LI+SE	0.659	0.600	0.558	0.659	0.545	0.664	0.623	0.600	**0.664**
LX+LI+SY	0.659	0.599	0.562	0.660	0.546	0.664	0.623	0.599	**0.664**
LX+SE+SY	0.660	0.599	0.562	0.660	0.546	0.663	0.624	0.599	**0.663**
AU+SY	0.662	0.633	0.598	0.652	0.550	0.660	0.643	0.632	**0.662**
LX+SY	0.654	0.598	0.555	0.654	0.544	0.660	0.619	0.598	**0.660**
LX+LI	0.652	0.600	0.550	0.655	0.542	0.660	0.617	0.600	**0.660**
LX+SE	0.653	0.598	0.551	0.653	0.542	0.658	0.618	0.598	**0.658**
LX	0.645	0.598	0.544	0.646	0.539	0.654	0.613	0.598	**0.654**
AU+SE	0.653	0.626	0.601	0.640	0.550	0.650	0.621	0.625	**0.653**
LI+SE+SY	0.625	0.587	0.582	0.633	0.542	0.642	0.606	0.588	**0.642**
LI+SY	0.613	0.587	0.579	0.623	0.538	0.630	0.609	0.587	**0.630**
LI+SE	0.601	0.581	0.575	0.615	0.535	0.622	0.596	0.580	**0.622**
SE+SY	0.601	0.576	0.571	0.611	0.530	0.620	0.596	0.576	**0.620**
LI	0.573	0.581	0.573	0.607	0.529	0.610	0.601	0.580	**0.610**
SY	0.585	0.572	0.566	0.596	0.523	0.602	0.596	0.573	**0.602**
SE	0.550	0.561	0.552	0.572	0.511	0.580	0.572	0.560	**0.580**
Max.	***0.694***	***0.661***	***0.660***	***0.700***	***0.621***	***0.700***	***0.696***	***0.660***	
Δ_{max}	***+0.064***	***+0.053***	***+0.022***	***+0.031***	***+0.073***	***+0.047***	***+0.060***	***+0.053***	

Table 6.4: Average accuracies of constrained test cases including 8 to 100 original tracks ordered by the maximum achieved accuracy of each feature set combination.

leads to the following top 25 attributes for the "ALL" feature set which performs best for the Bayes Net classifier:

1. Audio: loudness
2. Text style: words per minute
3. Text style: chars per minute
4. Text style: token count
5. Audio: acousticness
6. Audio: energy
7. Audio: instrumentalness
8. Text style: lines per minute
9. BOW-POS: personal pronoun
10. Pronouns: excentricity
11. Pronouns: IvU
12. Text style: line count
13. Audio: danceability
14. Audio: speechiness
15. BOW-POS: noun, singular or mass
16. Text style: unique line count
17. Text style: repeat word ratio
18. Text style: unique token ratio
19. BOW-POS: verb, non-3rd ps. sing. present
20. Text style: hapax legomenon ratio
21. BOW-POS: verb, base form
22. BOW-POS: preposition/subordinating conjunction
23. Text style: unique bigram ratio
24. Text style: unique trigram ratio
25. BOW-POS: determiner

The ranking suggests that acoustic and text style features provide the most relevant individual attributes while linguistic and semantic attributes seem to be less important since none of these are present. Referring to Stamatatos [63], the latter ones may be beneficial in combination with other feature types. All of the above listed features belong to only four out of 15 feature set groups: acoustic features (6x), text stylistic features (11x), pronouns features (2x), and part-of-speech features modeled as bag of words (6x). This correlates to the work of [45] who achieved best accuracy results with audio, text statistic, and part-of-speech features in genre classification. Also [17] revealed that length, type-token ratios, part-of-speech tags and pronouns contribute in genre classification using lyrics only.
Feature selection has been performed based on the mean and standard deviation of all information gains. Only features with an information gain higher than the difference of mean and standard deviation are chosen to train/test all classifiers, however the outcome did not lead to any improvement. Same applied for feature selection with a threshold equal to the information gain mean. Nevertheless, through a manual subset selection slight enhancements could be achieved: Features are ranked by the highest mean value but only the top 100 features are chosen. If less than 100 features are available for a feature set combination then 80% of all features are picked. The resulting achievements are listed in Table 6.5 as well as the relative improvements per classifier (Δ'_{max}) and feature set combination (Δ''_{max}) compared to their maximum average accuracy attained on constrained playlists pinpointed in Table 6.4.
The impact of feature selection is low but the results could be enhanced in general. However, only fractions of extracted features are necessary to gain equal outcomes, so the learning phase of classifiers can be reduced without worsening their performance. Highest accuracy is obtained by LibSVM (nu) incorporating only eight audio features:

1. Audio: loudness
2. Audio: acousticness
3. Audio: energy
4. Audio: instrumentalness
5. Audio: danceability
6. Audio: speechiness
7. Audio: duration
8. Audio: valence

(Audio: tempo)$_{\text{excluded}}$

(Audio: liveness)$_{\text{excluded}}$

The ranking of audio features conforms to the relative ranking of the top "ALL" features. By excluding the *tempo* and *liveness* attributes the LibSVM (nu) model could be improved by 1%. Latter classifier outperforms all other classifiers on every feature set combination after feature selection. Superior results could be achieved for all feature combinations except for SE, SE+SY, LX+SY, LX+SE+SY and LX+LI+SE+SY. The

Feature set	Bayes Net	J48	kNN	LibLinear	LibSVM (C)	LibSVM (nu)	Naïve Bayes	PART	Max.	Δ''_{max}
AU	*0.643*	***0.668***	***0.671***	***0.708***	***0.632***	***0.710***	***0.699***	***0.668***	***0.710***	***+0.010***
AU+LX+LI+SE	0.689	0.650	0.656	0.698	0.601	**0.710**	0.688	0.650	**0.710**	***+0.018***
AU+LX+SE	0.685	0.650	0.653	0.693	0.595	0.706	0.680	0.649	**0.706**	***+0.017***
AU+LX+LI	0.685	0.650	0.651	0.692	0.596	0.704	0.681	0.650	**0.704**	***+0.015***
ALL	***0.690***	*0.651*	*0.647*	*0.692*	*0.600*	*0.703*	*0.691*	*0.651*	***0.703***	***+0.009***
AU+LX	0.683	0.649	0.651	0.690	0.589	0.701	0.674	0.649	**0.701**	***+0.015***
AU+LX+LI+SY	0.688	0.650	0.645	0.689	0.598	0.700	0.689	0.651	**0.700**	***+0.009***
AU+LX+SE+SY	0.689	0.649	0.644	0.688	0.597	0.699	0.688	0.650	**0.699**	***+0.008***
AU+LX+SY	0.685	0.648	0.640	0.682	0.593	0.693	0.682	0.649	**0.693**	***+0.005***
AU+LI+SE+SY	0.675	0.638	0.618	0.681	0.572	0.690	0.674	0.638	**0.690**	***+0.013***
AU+LI	0.663	0.647	0.628	0.681	0.582	0.687	0.675	0.647	**0.687**	***+0.006***
AU+LI+SY	0.672	0.638	0.609	0.669	0.565	0.679	0.662	0.636	**0.679**	***+0.006***
AU+LI+SE	0.669	0.633	0.612	0.666	0.567	0.677	0.642	0.633	**0.677**	***+0.007***
LX+LI+SE	*0.646*	*0.616*	*0.629*	*0.659*	*0.582*	*0.669*	*0.656*	*0.616*	***0.669***	***+0.005***
AU+SE+SY	0.663	0.632	0.602	0.660	0.557	0.668	0.650	0.631	**0.668**	***+0.003***
LX+LI+SE+SY	*0.653*	*0.617*	*0.626*	*0.657*	*0.584*	*0.668*	*0.664*	*0.617*	***0.668***	***0.000***
LX+SE	0.641	0.612	0.624	0.654	0.576	0.665	0.648	0.612	**0.665**	***+0.007***
LX+LI	0.640	0.614	0.624	0.652	0.578	0.664	0.649	0.613	**0.664**	***+0.004***
AU+SY	0.662	0.635	0.599	0.654	0.556	0.664	0.649	0.635	**0.664**	***+0.002***
LX+LI+SY	0.648	0.614	0.622	0.650	0.581	0.662	0.658	0.614	**0.662**	***+0.002***
LX+SE+SY	0.648	0.613	0.621	0.650	0.579	0.661	0.655	0.614	**0.661**	***-0.002***
LX	0.634	0.611	0.621	0.649	0.571	0.660	0.641	0.611	**0.660**	***+0.006***
AU+SE	0.653	0.632	0.607	0.649	0.557	0.658	0.633	0.631	**0.658**	***+0.005***
LX+SY	0.642	0.611	0.616	0.643	0.575	0.654	0.649	0.611	**0.654**	***-0.006***
LI+SE+SY	0.619	0.590	0.591	0.638	0.548	0.646	0.632	0.590	**0.646**	***+0.004***
LI+SY	0.612	0.589	0.581	0.625	0.541	0.632	0.615	0.589	**0.632**	***+0.002***
LI+SE	0.600	0.584	0.576	0.618	0.539	0.626	0.601	0.583	**0.626**	***+0.004***
SE+SY	0.599	0.578	0.573	0.612	0.534	0.620	0.605	0.578	**0.620**	***0.000***
LI	0.572	0.582	0.575	0.610	0.535	0.612	0.605	0.583	**0.612**	***+0.002***
SY	0.585	0.574	0.567	0.597	0.526	0.604	0.597	0.574	**0.604**	***+0.002***
SE	0.549	0.561	0.553	0.574	0.514	0.580	0.573	0.561	**0.580**	***0.000***
Max.	***0.690***	***0.668***	***0.671***	***0.708***	***0.632***	***0.710***	***0.699***	***0.668***		
Δ'_{max}	***-0.004***	***+0.007***	***+0.011***	***+0.008***	***+0.011***	***+0.010***	***+0.003***	***+0.008***		

Table 6.5: Average accuracies after feature selection including test cases with 8 to 100 original tracks ordered by the maximum achieved accuracy of each feature set combination.

performance of all classifiers increased but not for Bayes Net (-0.4%). Consequently, the experiments indicate with respect to the accuracy results that acoustic features act as the main link between tracks within the same playlist and are possibly intuitively/on purpose considered by human beings during playlist generation.

To be complete, the corresponding precision, recall, and F_1 measures after performing feature selection are explored. Results are depicted in Table 6.6 for the classifiers LibLinear, LibSVM (nu) and Naïve Bayes as they gained highest accuracies. The measures reveal that models trained solely on acoustic features provide highest precision values of about 74% for all analyzed algorithms. Best recall measure of 70.3% is achieved by Naïve Bayes incorporating with the AU+LI+SE+SY feature combination. Superior precision (74%) and F_1 measures (70.2%) are gained by LibSVM (nu).

The aim of each model is to achieve perfect measures for precision and recall, however those are typically inversely related. Thus, depending on the application domain, specific different feature sets can be applied to improve either precision, recall or the harmonic mean F_1.

	LibLinear			**LibSVM (nu)**			**Naïve Bayes**		
Feature set	P	R	F_1	P	R	F_1	P	R	F_1
AU	**0.737**	0.657	**0.690**	**0.740**	0.661	0.693	**0.735**	0.636	0.672
LX	0.659	0.631	0.640	0.670	0.649	0.654	0.650	0.632	0.633
AU+LX	0.705	0.665	0.680	0.715	0.683	0.694	0.679	0.679	0.672
LI	0.619	0.592	0.601	0.621	0.596	0.604	0.610	0.604	0.599
AU+LI	0.698	0.652	0.670	0.704	0.660	0.678	0.678	0.681	0.673
LX+LI	0.664	0.631	0.643	0.676	0.647	0.657	0.656	0.645	0.643
AU+LX+LI	0.709	0.665	0.682	0.719	0.681	0.695	0.684	0.691	0.681
SE	0.581	0.560	0.565	0.586	0.572	0.575	0.578	0.575	0.569
AU+SE	0.662	0.625	0.638	0.669	0.643	0.651	0.634	0.651	0.636
LX+SE	0.665	0.634	0.645	0.676	0.651	0.659	0.654	0.649	0.644
AU+LX+SE	0.710	0.666	0.683	0.721	0.684	0.698	0.683	0.691	0.680
LI+SE	0.629	0.599	0.608	0.634	0.616	0.621	0.602	0.618	0.603
AU+LI+SE	0.682	0.641	0.655	0.689	0.662	0.671	0.639	0.671	0.648
LX+LI+SE	0.671	0.636	0.649	0.681	0.651	0.662	0.661	0.657	0.652
AU+LX+LI+SE	0.715	**0.670**	0.687	0.726	**0.686**	**0.702**	0.690	0.700	0.689
SY	0.607	0.579	0.588	0.613	0.591	0.598	0.600	0.602	0.596
AU+SY	0.668	0.632	0.644	0.675	0.647	0.657	0.647	0.669	0.653
LX+SY	0.653	0.623	0.634	0.665	0.638	0.647	0.656	0.645	0.643
AU+LX+SY	0.697	0.657	0.672	0.707	0.671	0.685	0.685	0.691	0.682
LI+SY	0.637	0.605	0.615	0.642	0.617	0.625	0.616	0.629	0.617
AU+LI+SY	0.685	0.644	0.659	0.692	0.658	0.671	0.657	0.689	0.667
LX+LI+SY	0.662	0.627	0.640	0.673	0.642	0.654	0.663	0.657	0.653
AU+LX+LI+SY	0.705	0.662	0.679	0.715	0.676	0.691	0.691	0.698	0.688
SE+SY	0.624	0.595	0.603	0.629	0.609	0.614	0.604	0.624	0.609
AU+SE+SY	0.674	0.637	0.650	0.681	0.650	0.661	0.644	0.683	0.658
LX+SE+SY	0.662	0.629	0.641	0.673	0.642	0.653	0.660	0.655	0.651
AU+LX+SE+SY	0.704	0.661	0.678	0.714	0.676	0.691	0.690	0.699	0.688
LI+SE+SY	0.651	0.617	0.629	0.658	0.629	0.639	0.630	0.652	0.635
AU+LI+SE+SY	0.697	0.653	0.670	0.705	0.667	0.681	0.669	**0.703**	0.681
LX+LI+SE+SY	0.669	0.633	0.647	0.680	0.647	0.659	0.668	0.664	0.660
ALL	0.708	0.664	0.681	0.718	0.678	0.694	0.693	0.702	**0.691**
Max.	**0.737**	**0.670**	**0.690**	**0.740**	**0.686**	**0.702**	**0.735**	**0.703**	**0.691**

Table 6.6: Average precision (P), recall (R), and F_1 measures after feature selection including test cases with 8 to 100 original tracks.

Chapter 7

Conclusion

This research analyzed the cohesive characteristics of tracks shared within a playlist on the basis of a supervised classification approach. According to the discussed process of supervised classification, first the acquisition of a data collection consisting of playlists, tracks, and lyrics has been presented including the preprocessing of lyrics and the ascertaining of proper lyrics versions. Second, the accumulation of acoustic features and the computation of lexical, linguistic, semantic and syntactic lyrics features has been elucidated with respect to the findings of the mentioned related studies. Finally, characteristics of playlists are revealed by employing eight different state-of-the-art classification algorithms trained on previously gathered track features. The analysis of about 12,000 playlists consisting of more than 200,000 unique English tracks created by almost 1,000 persons disclosed that acoustic features act as a major link between tracks within a playlist with respect to classification accuracies. The best accuracy result of 71% could be achieved by applying feature selection and training the LibSVM (nu) classifier on eight audio features. Slightly inferior accuracy results are gained by the LibSVM (nu) machine learning algorithm while incorporating with lyrics features only (~67%). Moreover, it has been figured out that a minimum amount of eight tracks is necessary to constitute a characteristic playlist.

To revise the results of this research, future work might employ the findings in a real-world application like a playlist recommender system and validate if they enhance user satisfaction. Several experiments regarding the improvement of classification should be performed including various parameterizations of the classification algorithms and the application of different feature selection techniques. Considering a late fusion of audio and lyrics classifiers might increase the accuracy of classification models and may reveal an orthogonality of audio and lyrics features regarding the characterization of playlists like in the realm of mood and

genre classification. Furthermore, the employed dataset should be extended to disclose information about the upper bound of characteristic playlists.

Appendix

A.1 Lyrics annotation and repetition patterns

To get a slight notion of how crawled lyrics are sanitized, this section lists some commonly used repetition and annotation patterns and depicts how they are processed.
Annotations are remarks in lyrics that provide some meta information about the song: involved artists, song title, song duration, song structure (verse, chorus, bridge, ...), used instruments, used chords, lyrics source, producer, copyright, etc. This data may indicate some useful information for an automated playlist prediction, however in this work it is (partly) removed as only the content/raw structure of lyrics is analyzed.
Repetition patterns are frequently used in lyrics to avoid writing of recurring parts again and again. More precisely, repetition patterns are instructions which are used to duplicate lines or whole (labeled) segments. This instructions need to be recognized to fully expand the lyrics to its actual form.
Regarding to the list of notation patterns elaborated by Hu [26], following annotations and repetition patterns are considered and processed or marked as future improvements. By reason of the huge amount of pattern possibilities only some examples of handled patterns are facilitated. To be able to recognize as many patterns as possible multiple consecutive whitespaces between words/symbols are treated as a single whitespace character and the letter casing is ignored.

A.1.1 Annotations

Annotations are removed from lyrics except those which are necessary to dissolve the repetition patterns (e.g, repeat, chorus, refrain, ...).
For clarity, let <*song structure*> be the placeholder for the words "*repeat*", "*solo*", "*verse*", "*intro*", "*chorus*", "*refrain*", "*outro*", "*bridge*", "*interlude*", "*pre-chorus*", "*end chorus*", and "*end refrain*". Moreover, consider <*decoration*> as one term out of: "*()*", "*[]*", "<>", "∼∼", "***", and "{}".

1. <*song structure*> decorated with <*decoration*> sometimes with letters and numbers:
   e.g., [outro], ~bridge~, (Chorus A)
2. <*song structure*> at the beginning of a line with a ":":
   e.g., Solo: It's my ...
3. <*song structure*> at the beginning of a line with a preceding "end of":
   e.g., End of solo Here are ...
4. Similar to above but with a ":":
   e.g., End of bridge: Let the ...
5. Instrumental annotations like "instrument", "instrumental", "instrumental break", "piano", and "violin" boxed in a <*decoration*>:
   e.g., (instrumental break)
6. Similar to above but with a ":":
   e.g., (piano):
7. Artist name at the top of a segment or at the beginning of lines along with a ":" or "-":
   e.g., Justin Timberlake - Ohhhh ...,
   Mel B:
   If you wanna be ...
8. Artist name which is decorated with <*decoration*> and followed by an optional ":" or "-":
   e.g., <OutKast>, [Rihanna:], (Bob Marley):
9. Artist name at the beginning of a line followed by ":" or "-":
   e.g., DMX - Give it ...
10. Lines which contains the words "transcribed from", "written by", "sung by", "spoken words by", "words by", "lyrics by", "music by", "mixed by", "copyright", "producer", and "vocals by" at the beginning of segments or at the end of song text:
    e.g., Transcribed from H. Simpson recordings
11. Lines which contains the word "Time" followed by a duration declaration in the format "xx:xx" or "x:xx" at the beginning of segments or at the end of song text:
    e.g., Time 3:10, Time: 10:45 ...
12. Lines which contains the words "Inc." and "Music" or "Publishing" at the beginning of segments or at the end of song text:
    e.g., H. Simpson Music, Inc.
13. Phrase "fade to end" at the end of a song text
14. Phrase "music fade" at end of a segment

15. Lines consisting only of music chords:
    e.g., F#m, A /A
16. Lines with both “Lyrics” and the title of a song
17. Lines with http://, https://, or www.
18. Any words in between “*” and “*”:
    e.g., *Radio announcer*

A.1.2 Repetitions

Different types of repetition patterns are used within song texts. Particularly line, segment or chorus repetition notations are applied whereas the latter one is a specialized form of the segment repetition.
Again, consider <*decoration*> as one term out of: “*()*”, “*[]*”, “<>”, “∼∼”, “***”, and “{}”. Subsequent repetition patterns are covered in the song text preprocessing phase:

Chorus repetitions

Reoccurring chorus segments are every now and then not entirely written in song texts. Rather references to chorus segments are used to denote that a chorus should be repeated. To enable referencing the first occurrence of a chorus segment is usually labeled with some appropriate name. Therefore, the first appearance of the below-mentioned terms are considered as labels and do not cause any repetitions.

1. Lines consisting only of “chorus” or “refrain” optional decorated with <*decoration*> are figured out as repetitions:
   e.g., (chorus), <chorus>, ∼refrain∼, {chorus}, chorus
2. Similar to 1., but with an additional colon:
   e.g., [chorus]:, (refrain:)
3. Similar to 1., but followed by a hint of how often the chorus should be repeated:
   e.g., [Chorus (2x)], refrain 3x, Chorus (4x), *chorus* <2x>
4. Similar to 3., but without spaces between chorus tag and the amount of repetitions:
   e.g., {chorus2x}
5. Similar to 3., but without “x” in front of the amount of repetitions:
   e.g., *chorus x2*
6. Similar to 3./5., but with a space between the amount of repetitions:
   e.g., {chorus 2 x}, Refrain x 4,

7. Similar to 5., but instead of “x” a “*” is used:
   e.g., (Refrain *2)
8. Similar to above patterns, but between the chorus tag and the amount of repetitions a “:” or “-” is used:
   e.g., chorus: 3x , *chorus - x2*
9. Similar to the above patterns, but the number of repetitions is in front of the chorus tag:
   e.g., 2x Chorus, {*2 refrain}
10. Similar to 9., but with a “:” at the end of the chorus tag:
    e.g., <2x Chorus:>, {x3 chorus:}
11. Similar to 3, but “x” gets replaced with “times”:
    e.g., {3 times chorus:}, Chorus 3 times, [Chorus (2 times)]
12. Similar to the above patterns, but numbers get replaced with words:
    e.g., [Two times chorus:], [Chorus - (three times)], Chorus twice
13. Similar to above patterns, but with the word “repeat” beside the chorus tag:
    e.g., (repeat refrain), *repeat chorus thrice*, <repeat chorus [x 2]>, (chorus, repeat 2x), chorus (repeat), chorus: (repeat 2x)
14. Chorus tag with phrases “till end”, “till fade“, “and fade“:
    e.g., [repeat chorus and fade], {repeat chorus till end}:

Unnamed segment repetitions

Similar to chorus repetitions unnamed segment repetitions are assimilated of the form:

1. “repeat” with/without a <*decoration*> at the end of a segment:
   e.g., I don’t quite know
   How to say
   How I feel
   Those three words
   repeat
2. “repeat” with/without a <*decoration*> at the end of a segment with amount of repetition times, similar to chorus repetition the amount of repeatings could be written in numbers or words:
   e.g., I’m in love
   I’m in love
   I’m in love shape of dance
   (repeat three times)

3. “repeat to fade” boxed in a <*decoration*> at the end of a segment

4. “repeated till end” boxed in a <*decoration*> at the end of a segment

Line repetitions

Song texts have to property of containing single lines that are successively repeated. Consequently, there is a syntax to abbreviate recurring lines too:

1. Repetition instruction at the end of a line boxed in a <*decoration*>, again with different possibilities to specify the amount of repeatings:
   e.g., Tainted love (oh) (x4),
   Oh oh oh <repeat 7 more times>,
   And I would walk 500 miles {four times},
   I was made for loving you baby *repeat four more times*

2. Similar to above pattern, but with an additonal “:”, “-” or “,” between repeat and the repetition amount:
   e.g., Wake me up before you go go ~repeat, six times~,
   Do you really love me? {repeat - twice}

3. “repeat to fade”, “repeat to fade...” boxed in a <*decoration*> at the end of a line:
   e.g., Rock, rock on (*repeat to fade*)

4. “repeated till end” decorated with a <*decoration*> at the end of a line

A.1.3 Future improvements

The lyrics sanitizing is not yet perfect but building such one is out of scope of this thesis. Nevertheless, for future improvements someone should be aware of following annotation patterns:

1. Identifying and reduplicating multiple chorus:
   e.g., Chorus A/Chorus B

2. Multiple chorus repetitions defined per line:
   e.g., (chorus) (chorus) (chorus), (chorus 1 + 2), *chorus A* & *chorus B*

3. Repeat chorus with (multiple) artist name(s) in between:
   e.g., Chorus: DMX *repeat x3*

4. Repeat annotation at the beginning of a segment

5. Reduplicate named segments other than those labeled as chorus:
   e.g., {verse 1 twice}

6. Repeat annotation at the beginning of a line:
   e.g., (repeat 2x) oh yeaaaahhh

7. Remove chords from the beginning/end of a line

8. Remove pitch informations:
   e.g., (1/2 step higher)

9. Advanced repetition instructions:
   e.g., repeat last two lines 3 times, <repeat last verse>, (repeat and then chorus twice)

10. Segmentation annotation along with specific instruction:
    e.g., *repeat chorus, insert “Hello” before ”is it me you looking for“*

A.2 Penn Treebank tag sets

For the sake of completeness the Penn Treebank part-of-speech tags and phrase level tags outputted by the employed natural language processing tools are listed.

A.2.1 Part-of-speech tag set

The part-of-speech tag set defined in the Penn Treebank project used by the *PTBTokenizer* is depicted in Table A.2.1. For detailed information, please refer to Marcus et al. [42].

Tag	Description	Tag	Description
CC	Coordinating conjunction	TO	*to*
CD	Cardinal number	UH	Interjection
DT	Determiner	VB	Verb, base form
EX	Existential *there*	VBD	Verb, past tense
FW	Foreign word	VBG	Verb, gerund/present participle
IN	Preposition/subordinating conjunction	VBN	Verb, past participle
JJ	Adjective	VBP	Verb, non-3rd ps. sing. present
JJR	Adjective, comparative	VBZ	Verb, 3rd ps. sing. present
JJS	Adjective, superlative	WDT	*wh*-determiner
LS	List item marker	WP	*wh*-pronoun
MD	Modal	WP$	Possessive *wh*-pronoun
NN	Noun, singular or mass	WRB	*wh*-adverb
NNS	Noun, plural	#	Pound sign
NNP	Proper noun, singular	$	Dollar sign
NNPS	Proper noun, plural	.	Sentence-final punctuation
PDT	Predeterminer	,	Comma
POS	Possessive ending	:	Colon, semi-colon
PRP	Personal pronoun	(	Left bracket character
PP$	Possessive pronoun	)	Right bracket character
RB	Adverb	″	Straight double quote
RBR	Adverb, comparative	‘	Left open single quote
RBS	Adverb, superlative	“	Left open double quote
RP	Particle	’	Right close single quote
SYM	Symbol (mathematical or scientific)	”	Right close double quote

Table A.2.1: The Penn Treebank part-of-speech tag set. [42]

A.2.2 Phrase level tag set

The *Apache OpenNLP Chunker* applies chunk tags out of Table A.2.2. For detailed information, please refer to Bies et al. [7].

Tag	Description
ADJP	Adjective Phrase. Phrasal category headed by an adjective (including comparative and superlative adjectives). Example: *outrageously expensive*.
ADVP	Adverb Phrase. Phrasal category headed by an adverb (including comparative and superlative adverbs). Examples: *rather timidly*, *very well indeed*, *rapidly*.
CONJP	Conjunction Phrase. Used to mark certain "multi-word" conjunctions, such as *as well as*, *instead of*.
FRAG	Fragment.
INTJ	Interjection. Corresponds approximately to the part-of-speech tag UH.
LST	List marker. Includes surrounding punctuation.
NAC	Not A Constituent; used to show the scope of certain prenominal modifiers within a noun phrase.
NP	Noun Phrase. Phrasal category that includes all constituents that depend on a head noun.
NX	Used within certain complex noun phrases to mark the head of the noun phrase. Corresponds very roughly to N-bar level but used quite differently.
PP	Prepositional Phrase. Phrasal category headed by a preposition.
PRN	Parenthetical.
PRT	Particle.
QP	Quantifier Phrase (i.e., complex measure/amount phrase); used within NP.
RRC	Reduced Relative Clause.
UCP	Unlike Coordinated Phrase.
VP	Verb Phrase. Phrasal category headed a verb.
WHADJP	*Wh*-adjective Phrase. Adjectival phrase containing a *wh*-adverb, as in *how* hot.
WHADVP	*Wh*-adverb Phrase. Introduces a clause with an ADVP gap. May be null (containing the 0 complementizer) or lexical, containing a *wh*-adverb such as *how* or *why*.
WHNP	*Wh*-noun Phrase. Introduces a clause with an NP gap. May be null (containing the 0 complementizer) or lexical, containing some *wh*-word, e.g. *who*, *which book*, *whose daughter*, *none of which*, or *how many leopards*.
WHPP	*Wh*-prepositional Phrase. Prepositional phrase containing a *wh*-noun phrase (such as *of which* or *by whose authority*) that either introduces a PP gap or is contained by a WHNP.
X	Unknown, uncertain, or unbracketable. X is often used for bracketing typos and in bracketing *the...the*-constructions.

Table A.2.2: The Penn Treebank phrase level tag set. [7]

Bibliography

[1] S. Abney. Part-of-Speech Tagging and Partial Parsing. In *Corpus-Based Methods in Language and Speech*, pages 118–136. Kluwer Academic Publishers, 1996.

[2] E. Alpaydin. *Introduction to Machine Learning.* The MIT Press, 2nd edition, 2010.

[3] S. Argamon, M. Šarić, and S. S. Stein. Style Mining of Electronic Messages for Multiple Authorship Discrimination: First Results. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 475–480, New York, NY, USA, 2003. ACM.

[4] D. Bainbridge, S. J. Cunningham, and S. J. Downie. How People Describe Their Music Information Needs: A Grounded Theory Analysis of Music Queries. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR 2003)*, Baltimore, Maryland , USA, October 2003.

[5] I. Ben-Gal. Bayesian Networks. *Encyclopedia of statistics in quality and reliability*, 2007.

[6] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere. The Million Song Dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, pages 591–596, 2011.

[7] A. Bies, M. Ferguson, K. Katz, R. MacIntyre, V. Tredinnick, G. Kim, M. A. Marcinkiewicz, and B. Schasberger. Bracketing Guidelines for Treebank II Style Penn Treebank Project. Technical report, University of Pennsylvania, 1995.

[8] G. Bonnin and D. Jannach. Automated Generation of Music Playlists: Survey and Experiments. *ACM Computing Surveys*, 47(2):26:1–26:35, 2014.

[9] M. M. Bradley and P. J. Lang. Affective Norms for English Words (ANEW): Instruction Manual and Affective Ratings. Technical report, Technical report C-1, the center for research in psychophysiology, University of Florida, 1999.

[10] M. K. Buckland and F. C. Gey. The Relationship between Recall and Precision. *JASIS*, 45(1):12–19, 1994.

[11] W. W. Cohen. Fast Effective Rule Induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[12] C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[13] S. J. Cunningham, D. Bainbridge, and A. Falconer. 'More of an Art than a Science': Supporting the Creation of Playlists and Mixes. In *ISMIR 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 8-12 October 2006, Proceedings*, pages 240–245, 2006.

[14] A. M. Demetriou, M. A. Larson, and C. C. Liem. Go With the Flow: When Listeners Use Music as Technology. In *17th International Society for Music Information Retrieval Conference (ISMIR)*, pages 292–298, New York, USA, August 2016.

[15] EchoNest. Acoustic Attributes Overview. URL `http://developer.echonest.com/acoustic-attributes.html`. Accessed on 2016-09-07.

[16] M. Fell. Lyrics Classification. Master's thesis, Saarland University, 2014.

[17] M. Fell and C. Sporleder. Lyrics-based Analysis and Classification of Music. In J. Hajic and J. Tsujii, editors, *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 620–631. ACL, 2014.

[18] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[19] E. Frank and I. H. Witten. Generating Accurate Rule Sets Without Global Optimization. In J. Shavlik, editor, *Fifteenth International Conference on Machine Learning*, pages 144–151. Morgan Kaufmann, 1998.

[20] G. Geleijnse and J. H. M. Korst. Efficient Lyrics Extraction from the Web. In *ISMIR 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 8-12 October 2006, Proceedings*, pages 371–372, 2006.

[21] I. Guyon and A. Elisseeff. An Introduction to Feature Extraction. In I. Guyon, M. Nikravesh, S. Gunn, and L. A. Zadeh, editors, *Feature Extraction: Foundations and Applications*, pages 1–25, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[22] H. Hirjee and D. G. Brown. Automatic Detection of Internal and Imperfect Rhymes in Rap Lyrics. In K. Hirata, G. Tzanetakis, and K. Yoshii, editors, *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe International Conference Center, Kobe, Japan, October 26-30, 2009*, pages 711–716. International Society for Music Information Retrieval, 2009.

[23] H. Hirjee and D. G. Brown. Rhyme Analyzer: An Analysis Tool for Rap Lyrics. In J. S. Downie and R. C. Veltkamp, editors, *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010*. International Society for Music Information Retrieval, 2010.

[24] H. Hirjee and D. G. Brown. Using Automated Rhyme Detection to Characterize Rhyming Style in Rap Music. *Empirical Musicology Review*, 5(4):121–145, 2010.

[25] M. Hu and B. Liu. Mining and Summarizing Customer Reviews. In W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177. ACM, 2004.

[26] X. Hu. *Improving Music Mood Classification Using Lyrics, Audio and Social Tags*. PhD thesis, University of Illinois, Champaign, IL, USA, 2010.

[27] X. Hu, K. Choi, and J. S. Downie. A Framework for Evaluating Multimodal Music Mood Classification. *Journal of the Association for Information Science and Technology*, 2016.

[28] X. Hu and J. S. Downie. Improving Mood Classification in Music Digital Libraries by Combining Lyrics and Audio. In J. Hunter, C. Lagoze, C. L. Giles, and Y. Li, editors, *Proceedings of the 2010 Joint International Conference on Digital Libraries, JCDL 2010,*

Gold Coast, Queensland, Australia, June 21-25, 2010, pages 159–168. ACM, 2010.

[29] X. Hu, J. S. Downie, and A. F. Ehmann. Lyric Text Mining in Music Mood Classification. In K. Hirata, G. Tzanetakis, and K. Yoshii, editors, *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe International Conference Center, Kobe, Japan, October 26-30, 2009*, pages 411–416. International Society for Music Information Retrieval, 2009.

[30] C. J. Hutto. VADER-Sentiment-Analysis. URL `https://github.com/cjhutto/vaderSentiment/`. Accessed on 2017-28-02.

[31] C. J. Hutto and E. Gilbert. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. In E. Adar, P. Resnick, M. D. Choudhury, B. Hogan, and A. H. Oh, editors, *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014*. The AAAI Press, 2014.

[32] P. Jaccard. The Distribution of the Flora in the Alpine Zone. *New Phytologist*, 11(2):37–50, Feb. 1912.

[33] T. Jehan and D. DesRoches. Analyzer Documentation. Technical report, The Echo Nest Corporation, 48 Grove St. Suite 206, Somerville, MA 02144, January 2014. Analyzer Version 3.2. URL `http://developer.echonest.com/docs/v4/\_static/AnalyzeDocumentation.pdf`, accessed on 2016-09-07.

[34] M. Kamalzadeh, D. Baur, and T. Möller. A Survey on Music Listening and Management Behaviours. In *International Society for Music Information Retrieval Conference (ISMIR)*, pages 373–378, October 2012.

[35] B. Kırmacı and H. Oğul. Evaluating Text Features for Lyrics-based Songwriter Prediction. In *Intelligent Engineering Systems (INES), 2015 IEEE 19th International Conference on*, pages 405–409, Bratislava, Slovakia, September 2015. IEEE.

[36] P. Knees, M. Schedl, and G. Widmer. Multiple Lyrics Alignment: Automatic Retrieval of Song Lyrics. In *ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 11-15 September 2005, Proceedings*, pages 564–569, 2005.

[37] S. B. Kotsiantis. Supervised Machine Learning: A Review of Classification Techniques. *Informatica (Slovenia)*, 31(3):249–268, 2007.

[38] C. Laurier, J. Grivolla, and P. Herrera. Multimodal Music Mood Classification Using Audio and Lyrics. In M. A. Wani, X. Chen, D. Casasent, L. A. Kurgan, T. Hu, and K. Hafeez, editors, *Seventh International Conference on Machine Learning and Applications, ICMLA 2008, San Diego, California, USA, 11-13 December 2008*, pages 688–693. IEEE Computer Society, 2008.

[39] J. H. Lee, Y. Kim, and C. Hubbles. A Look at the Cloud from Both Sides Now: An Analysis of Cloud Music Service Usage. In *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016*, pages 299–305, 2016.

[40] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[41] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014.

[42] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[43] C. Martindale. *Romantic Progression: The Psychology of Literary History*. Washington, DC: Hemisphere, 1975.

[44] C. Martindale. *The Clockwork Muse: The Predictability of Artistic Change*. New York: Basic Books, 1990.

[45] R. Mayer, R. Neumayer, and A. Rauber. Combination of Audio and Lyrics Features for Genre Classification in Digital Audio Collections. In *Proceedings of the 16th International Conference on Multimedia 2008, Vancouver, British Columbia, Canada, October 26-31, 2008*, pages 159–168, 2008.

[46] R. Mayer, R. Neumayer, and A. Rauber. Rhyme and Style Features for Musical Genre Classification by Song Lyrics. In *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008*, pages 337–342, 2008.

[47] R. Mayer and A. Rauber. Music Genre Classification by Ensembles of Audio and Lyrics Features. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR*

2011, Miami, Florida, USA, October 24-28, 2011, pages 675–680, 2011.

[48] B. McFee, T. Bertin-Mahieux, D. P. W. Ellis, and G. R. G. Lanckriet. The Million Song Dataset Challenge. In A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich, and S. Staab, editors, *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*, pages 909–916. ACM, 2012.

[49] B. McFee, E. J. Humphrey, and J. Urbano. A Plan for Sustainable MIR Evaluation. In M. I. Mandel, J. Devaney, D. Turnbull, and G. Tzanetakis, editors, *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016*, pages 285–291, 2016.

[50] C. E. Metz. Basic Principles of ROC Analysis. In *Seminars in Nuclear Medicine*, volume 8, pages 283–298, 1978.

[51] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to WordNet: An On-line Lexical Database*. *International Journal of Lexicography*, 3(4):235–312, 1990.

[52] R. Mitton. Spelling Checkers, Spelling Correctors and the Misspellings of Poor Spellers*. *Information Processing and Management*, 23(5):495–505, 1987.

[53] F. Å. Nielsen. A New ANEW: Evaluation of a Word List for Sentiment Analysis in Microblogs. In M. Rowe, M. Stankovic, A. Dadzie, and M. Hardey, editors, *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages, Heraklion, Crete, Greece, May 30, 2011*, volume 718 of *CEUR Workshop Proceedings*, pages 93–98. CEUR-WS.org, 2011.

[54] J. W. Pennebaker, M. R. Mehl, and K. G. Niederhoffer. Psychological Aspects of Natural Language Use: Our Words, Our Selves. *Annual review of psychology*, 54(1):547–577, 2003.

[55] M. Pichl, E. Zangerle, and G. Specht. Understanding Playlist Creation on Music Streaming Platforms. In *Proceedings of the IEEE Symposium on Multimedia (ISM)*. IEEE, 2016.

[56] Provalis Research. Regressive Imagery Dictionary. URL `https://provalisresearch.com/products/content-analysis-software/wordstat-dictionary/regressive-imagery-dictionary/`. Accessed on 2017-03-26.

[57] J. R. Quinlan. Improved Use of Continuous Attributes in C4.5. *J. Artif. Intell. Res. (JAIR)*, 4:77–90, 1996.

[58] R. P. Ribeiro, M. Almeida, and C. N. Silla Jr. The Ethnic Lyrics Fetcher Tool. *EURASIP J. Audio, Speech and Music Processing*, 2014:27, 2014.

[59] I. Rish. An Empirical Study of the Naive Bayes Classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, pages 41–46. IBM New York, 2001.

[60] J. A. Russell. A Circumplex Model of Affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, 1980.

[61] C. Sammut and G. I. Webb, editors. *Encyclopedia of Machine Learning*. Springer, New York, 2011.

[62] Spotify. Audio Features Description. URL `https://developer.spotify.com/web-api/get-audio-features/`. Accessed on 2016-09-07.

[63] E. Stamatatos. A Survey of Modern Authorship Attribution Methods. *Journal of the American Society for Information Science and Technology (JASIST)*, 60(3):538–556, 2009.

[64] P. J. Stone, D. C. Dunphy, and M. S. Smith. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, 1966.

[65] C. Strapparava and A. Valitutti. WordNet Affect: An Affective Extension of WordNet. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*. European Language Resources Association, 2004.

[66] S. Stumpf and S. Muscroft. When Users Generate Music Playlists: When Words Leave Off, Music Begins? In *Proceedings of the 2011 IEEE International Conference on Multimedia and Expo, ICME 2011, 11-15 July, 2011, Barcelona, Catalonia, Spain*, pages 1–6. IEEE Computer Society, 2011.

[67] M. Thelwall. The Heart and Soul of the Web? Sentiment Strength Detection in the Social Web with SentiStrength. In J. A. Holyst, editor, *Cyberemotions: Collective Emotions in Cyberspace*, pages 119–134, Cham, 2017. Springer International Publishing.

[68] E. F. Tjong Kim Sang and S. Buchholz. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proceedings of the 2nd Workshop*

on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning, volume 7 of *ConLL '00*, pages 127–132, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[69] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[70] M. van Zaanen and P. Kanters. Automatic Mood Classification Using TF*IDF Based on Lyrics. In J. S. Downie and R. C. Veltkamp, editors, *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010*, pages 75–80. International Society for Music Information Retrieval, 2010.

[71] F. Vignoli. Digital Music Interaction Concepts: A User Study. In *ISMIR 2004, 5th International Conference on Music Information Retrieval, Barcelona, Spain, October 10-14, 2004, Proceedings*, 2004.

[72] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. F. M. Ng, B. Liu, P. S. Yu, Z. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg. Top 10 Algorithms in Data Mining. *Knowl. Inf. Syst.*, 14(1):1–37, 2008.