Automatic Music Moods Classification (AM^{2}C)

A PROJECT REPORT

Submitted to

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal

In partial fulfillment of the degree

of

Bachelor of Engineering

in

INFORMATION TECHNOLOGY

Project Guide:

Dr. Satyendra Singh Chouhan

Submitted By:

Apurva Maheshwari [0801IT141014]

Ashish Kumar Patel [0801IT141018]

Ekta Raghuwanshi [0801IT141030]

Department of Information Technology

Shri Govindram Seksaria Institute of Technology and Science
Indore-452003

2017-2018

Automatic Music Moods Classification (AM^{2}C)

A PROJECT REPORT

Submitted to

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal

In partial fulfillment of the degree

of

Bachelor of Engineering

in

INFORMATION TECHNOLOGY

Project Guide:

Dr. Satyendra Singh Chouhan

Submitted By:

Apurva Maheshwari [0801IT141014]

Ashish Kumar Patel [0801IT141018]

Ekta Raghuwanshi [0801IT141030]

Department of Information Technology

Shri Govindram Seksaria Institute of Technology and Science
Indore-452003

2017-2018

Shri Govindram Seksaria Institute of Technology and Science
Indore-452003

RECOMMENDATION

This is to certify that the project report entitled Automatic Music Moods Classification (AM^{2}C) submitted by Apurva Maheshwari, Ashish Kumar Patel and Ekta Raghuwanshi towards the partial fulfillment of degree of Bachelor of Engineering (Information Technology) from Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, is a satisfactory account of their project work and is approved for the award of degree.

Project guide	Head of Department
Dr. Satyendra Singh Chouhan	Dr. Sunita Verma

Shri Govindram Seksaria Institute of Technology and Science

CERTIFICATE

This is to certify that the project report entitled Automatic Music Moods Classification (AM^2C) submitted Apurva Maheshwari, Ashish Kumar Patel and Ekta Raghuwanshi towards the partial fulfillment of degree of Bachelor of Engineering (Information Technology) from Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, is a satisfactory account of their project work and is approved for the award of degree.

Internal Examiner **External Examiner**

Acknowledgement

The completion of any task depends upon the cooperation, coordination and consolidated efforts of several resources of knowledge, energy, time and above on all the proper guidance.

We owe the moment of satisfaction with deep sense of gratitude to our project guide
Dr. Satyendra Singh Chouhan for his technical guidance, persistent encouragement, perpetual motivation and everlasting patience. Without him this project would not have been successfully completed. Working under his guidance has been very fruitful and has proved to be a valuable experience.

We are also grateful to Dr. Sunita Verma (Head of department Information Technology) for providing us with numerous facilities and opportunities in the form of laboratories and equipment.

We are sincerely thankful to all the staff members of the department Information Technology, SGSITS Indore and all those who have helped us directly or indirectly throughout project work.

Apurva Maheshwari

Ashish Kumar Patel

Ekta Raghuwanshi

ABSTRACT

Music has been part of human lives since ancient times. We have hundreds of millions of songs representing different cultures, mood and genres. Though they are readily accessible using internet and streaming services, discovery of the right music piece to listen to is hard and an automated assistance to find the right song among the millions is required.

Attempts have been done to classify music on the basis of their genres but their efforts have not been much fruitful because of lack of good and large datasets. Identifying the right features to represent the music in summarized way is also a challenging task in itself. To create a large training set as well as to eliminate the subjectivity of one's perception of mood on a song, we use crowdsourcing platforms to get labels for songs. We take two distinct and extreme mood categories, Happy and Sad. We evaluate the system using three classification algorithms.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Emotion is fundamental to human experience, influencing cognition, perception, and everyday tasks such as learning, communication, and even rational decision-making. Affective Computing is computing that relates to, arises from, or deliberately influences emotion or other affective phenomena [1].Affective Computing combines engineering and computer science with psychology, cognitive science, neuroscience, sociology, education, psychophysiology, value-centered design, ethics, and more.

In recent years, the demand for techniques and products that provide affective computing and perceptual interaction capabilities has rendered affective computing more important than ever before. As a branch of affective computing, since music can convey emotion-related information, the study of music emotion has become the focus of researchers and considerable effort has been expended on it during the past decade. Emotion is the essence of music, and the emotion information of music can be widely used in music retrieval and recommendation. Almost all music pieces are created to convey feelings; composers create music to resonate with their listeners; and performers use the language of music to elicit the emotional responses of audiences [2]. Meanwhile, massive cross-cultural studies of the power of music have indicated that common music psychological and emotional cues exit in music that can transcend the limits of language and achieve cultural infiltration simultaneously [3] [4] [5]. For these reasons, a technique for organizing and retrieving music using an emotion-based approach is feasible, and the core of such technique is the automatic recognition of music emotion information. At present, methods for using the physical, audial, and semantic features of music signals to achieve the automatic recognition of music emotion information have become an important part of the research on digital music applications. The study of music emotion recognition has thus become an extremely urgent issue.

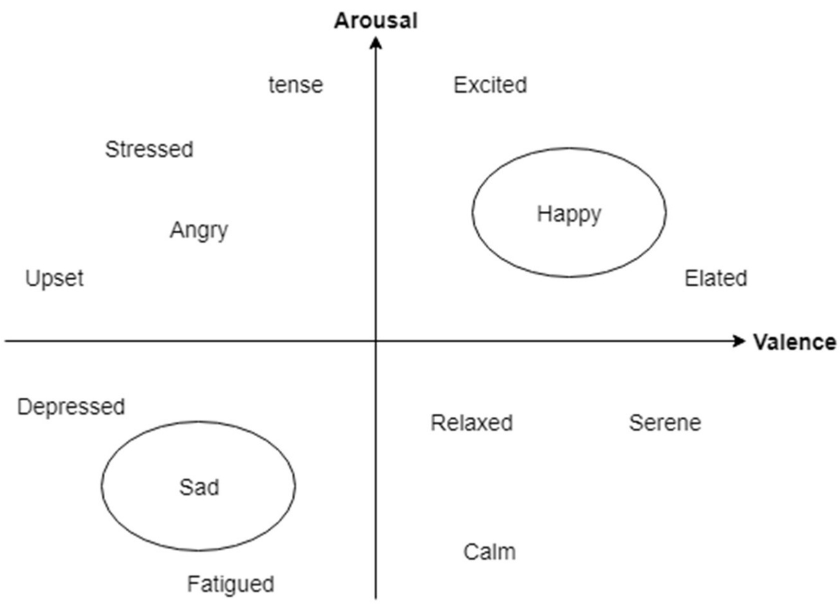

Figure 1: Mood categories on VA plane

The research for detecting the emotion of songs is called music emotion recognition (MER) [6]. According to psychological theory, music emotion recognition system can be roughly divided into parametric models and categorical models. Parametric approaches represent emotion in music using the valence and arousal (VA) values. Figure 1 displays the VA plane and the four main classes of human emotion- angry, happy, sad and relaxed. The emotion of a music clip can be described as a point in VA plane. However, the emotion tags in the VA plane may be inconvenient for users to quickly search songs. Another way to represent emotions in music is categorical approach that tags songs with emotion labels or adjectives, such as angry, bored, calm, happy, and peaceful.

1.1 Objective

In this work of MER, we aim to achieve an Automated Music Mood Classification system. We take a categorical approach and consider two categories Happy and Sad.

We aim to conduct a comparative study of different classification algorithms to achieve our objective. Moreover, we aim to assess the effectiveness of emotion metadata available in online music communities for the stated objective.

1.2 Scope

The automatic classification of the songs has many applications. For example, automatic cataloging of musical pieces and automatic generation of music playlists based on a similarity criterion of mood. The term “automatic” is used explicitly to underline the fact that these playlists and cataloging are produced based entirely on the extraction and analysis of songs features, without the use of any kind of descriptive text tags (typical ones are artist, album, genre, etc.).

In the more specific context of mood analysis, the applications can be extended to the generation of music playlists based in certain moods, without bothering the user with the task of browsing his personal musical collection (which can easily reach (dozens of) thousands of songs) to manually choose the songs. This will simultaneously be both more comfortable and time sparing to the user.

For instance, if the user wants music to play while he is jogging or doing sports (imagining the application is being used in a portable music player) he may want to hear some joyful, “happy” songs, beat-driven and/or with rhythm. On the other hand, if someone is driving he may want to be presented with some quiet, relaxing music. The same would apply to a stressful situation, where the user just wants to sit and calm down, or a psychological treatment. Even a shop owner or a DJ could benefit from this type of application, by automatically selecting cheerful music to serve as his store’s background music or selecting music based on the mood of the venue he is playing in (excitement for a club where rock tunes are played traditionally, for instance), respectively. Finally, another possible daily situation where this type of software could be applied in a useful way is the selection of music for a party, where the user could choose the input mood based on the theme (or even dress code) of it.

Apart from the use cases stated above, this work aims to find the usefulness of online emotional tags. Accurate emotion metadata will enable creation of large labelled datasets for MER and other related work.

1.3 Problems in Music Moods Classification

Music Mood Classification (MMC) has several issues. The first one is the subjectivity of ones perception while labeling a song. Due to the subjective nature of human perception, classification of the emotion of music is a challenging problem. Listening mood, environment, personality, age, cultural background etc., can influence the emotion perception. Because of these factors, classification methods that simply assign one emotion class to each song in a deterministic manner do not perform well in Practice.

The Second one is ambiguous mood taxonomy. One typical and common problem in this area is the existence of dozens (even hundreds or thousands) of different words or terms to describe moods, most them describing somewhat similar and redundant ones, for example depressed and sad. Usually these words are adjectives, but there is the need of normalizing the terms used, since there is not a standard mood taxonomy. Thus, there is a need that mood categories must be very distinct and non-overlapping like Happy and Sad.

Third one is, computing content based audio features and presence on noise and distortion in audio content. In addition, lack of large and unbiased data set in public domain is also a major concern.

1.4 Organization of the Report

The remainder of this paper is organized as follows. Chapter 2 presents the related work. In Chapter 3, we discuss the hardware and software platform requirements. In Chapter 4 we present the proposed music mood classification system. Performance evaluation is given in Chapter 5. Chapter 6 presents the conclusion and future directions.

CHAPTER 2: Literature Survey

This section, we first discuss the difference music mood recognition (MMR) method used in the literature. Next, we discuss the works that are closely related to the work presented in this paper.

The state-of-the-art MMR methods use different types of ground truth data about the music emotions. The ground truth data can be numeric or label type thus both regression and classification methods have been used for MMR [7]

Table 1: State of the art MMR methods

Paper Ref	Models	Accuracy	Dataset Size	is CrowdSourced	Feature Types
[7]	Hierarchical model, Gaussian mixture model, Expectation Maximization (EM) algorithm	86.3%	800	No, Three experts	Low-Level, Acoustic
[8]	Agglomerative clustering and SOM neural network	80%	104+70	No	MIDI, Pitch, duration and velocity
[6]	Deep GP	71.3%	1080	No	Rhythm, dynamics, timbre, pitch and tonality
[9]	W-D-KNN, K-NN and SVM	96.7 %	60	No	PsySoundb and Marsyasb

MMR classification methods can be categorized into single-label classification [9] [6] [8] [10] [11] multi-label classification [12] [13] [14] and a special case of multi-label classification called fuzzy classification [13] [15]. Single-label classification expresses the music emotion as a certain single emotion label that is mostly an adjective. Multi-label classification classifies the emotion of music segment into a number of emotion categories. Fuzzy classification expresses the emotion of music segment as the discrete possibility distribution of a number of emotion categories.

Since, we present a single label classification method, therefore, now we discuss the works that closely related to our work.

In [8], an approach for Mood Detection and Tracking of Music Audio Signals is proposed. It divides the music track into frames and uses the frame-based intensity features, timbre features, and rhythm features as feature set. It uses a hierarchical framework consisting of Gaussian mixture model (GMM) with Expectation Maximization (EM) algorithm. It uses four target classes in hierarchical fashion Group 1 (Contentment and Depression), Group 2 (Exuberance and Anxious/Frantic). It achieves an accuracy of 86.3% in a dataset of 800 (75%: 25%) using 10 fold cross validation.

In [9], it uses MIDI. It applies unsupervised agglomerative clustering algorithm. It divides a song into segments and calculates pitch, duration and velocity as features for each segment. The final feature vector consists of music scale, accuracy, sound intensity, basic sound, interval, direction, velocity and duration of notes. It applies clustering for 5, 12, 20 number of clusters. Finally a SOM neural network has been shown for visualization of song segments. The system achieved an accuracy of 80% on a dataset of 104/70.

In [6], this paper proposes a system for detecting emotion in music that is based on a deep Gaussian process (GP).A song is divided into frames. For each frame features such as rhythm, dynamics, timbre, pitch and tonality are calculated. Next, statistical values, such as mean and standard deviation, of frame-based features are calculated to generate a 38-dimensional feature vector. For emotion classification a deep GP is utilized. It treats classification problem from the perspective of regression. Finally, 9 classes of emotion are categorized by 9 one-versus-all classifiers. The system achieves an accuracy of 71.3 % on a dataset of 1080.

In [10], it uses dataset of 60 songs. It extracts a total of 45 feature using two free toolkits PsySound and Marsyas. It compares W-D-KNN with KNN and SVM. The highest recognition rate is 96.7% with W-D-KNN classifier weighted by Fibonacci series scheme. The accuracy of other classifiers ranges from 78.3% ~ 90.0%

In table, we present summarized study of state of the art Music Mood Classifiers in comparison with AM2C.

Table 2: Comparing state of the art Music Emotion Recognition Frameworks

Paper ID		[8]	[9]	[6]	[10]	**AM^{2}C**
Crowd Sourced Data		No	No	No	No	**Yes**
Features Type	High level	No	No	No	No	**Yes**
	Low Level	No	No	No	No	**Yes**
Size of Dataset		800	174	1080	174	**16527**
Classification Technique		Gaussian Mixture Model	Agglomerative Clustering	Deep Gaussian Process	W-D-KNN, KNN, SVM	**ANN, SVM, DT**

CHAPTER 3: System Requirement Analysis

3.1 Platform Specification

Machine learning algorithms are computationally heavy and will require a decent hardware infrastructure for efficient execution. While software requirements can easily be met using open source frameworks. In following sections 3.1.1 and 3.1.2 we discuss the system platform and the software platforms required for this project.

3.1.1 System Platform

We use a Linux based machine. It uses Ubuntu 16.04 LTS as operating system and has Java, Python and GUI capabilities. Below, we list the hardware specifications for the system we use for the experiment:

- 8GB of Memory
- 2 GB ATI Radeon GPU
- Intel Core i5 processor
- Storage Space of 5 GB.

3.1.2 Software Platform

We use a variety of languages and tools for the various phases of this project. For building the dataset from APIs, Node JS is used. For experiments, we use a Java Based GUI tool, Weka. For visualization using PCA we use Python (Anaconda). In the following sections we give a brief of each of these platforms.

3.1.2.1 WEKA

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to many of the machine learning functionalities. Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. All of Weka's techniques are predicated on the assumption that the data is available as one flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric or nominal attributes, but some other attribute types are also supported).

We list some of the features of Weka below:

- Platform independent
- Open source and free
- Different machine learning algorithms for Data Mining
- Easy to use
- Data Preprocessing tools
- Flexibility for scripting experiments
- Graphical User Interface

One can download Weka from the official website http://www.cs.waikato.ac.nz/ml/weka/.

Weka application interfaces

There are totally five application interfaces available for Weka. When we open Weka, it will start the Weka GUI Chooser screen from where we can open the Weka application interface. The Weka GUI screen and the available application interfaces are seen in Figure 2.

Weka data formats
Weka uses the Attribute Relation File Format for data analysis, by default. But listed below are some formats that Weka supports, from where data can be imported:

- CSV
- ARFF
- Database using ODBC

Attribute Relation File Format (ARFF):
This has two parts:
1) The header section defines the relation (data set) name, attribute name and the type.
2) The data section lists the data instances.
An ARFF file requires the declaration of the relation, attribute and data. Figure 3 is an example of an ARFF file.

- *@relation:* This is the first line in any ARFF file, written in the header section, followed by the relation/data set name. The relation name must be a string and if it contains spaces, then it should be enclosed between quotes.
- *@attribute:* These are declared with their names and the type or range in the header section. Weka supports the following data types for attributes:
    - Numeric
    - <nominal-specification>
    - String
    - date
    - @data – Defined in the Data section followed by the list of all data segments

Weka Explorer
The Weka Explorer is illustrated in Figure 2. It contains a total of six tabs.
The tabs are as follows.
1) *Preprocess:* This allows us to choose the data file.
2) *Classify:* This allows us to apply and experiment with different algorithms on preprocessed data files.
3) *Cluster:* This allows us to apply different clustering tools, which identify clusters within the data file.
4) *Association:* This allows us to apply association rules, which identify the association within the data.
5) *Select attributes:* These allow us to see the changes on the inclusion and exclusion of attributes from the experiment.
6) *Visualize:* This allows us to see the possible visualization produced on the data set in a 2D format, in scatter plot and bar graph output.
The user cannot move between the different tabs until the initial preprocessing of the data set has been completed.

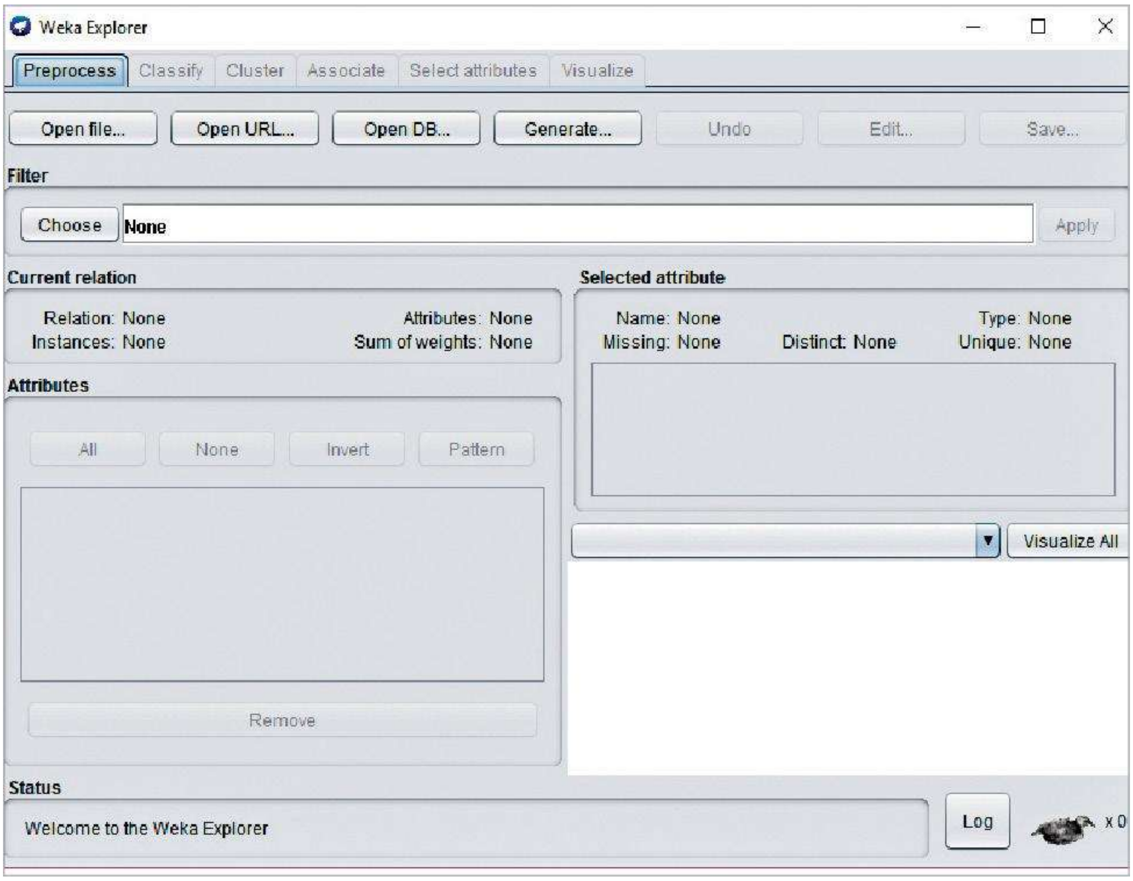

Figure 2: Weka explorer GUI

Preprocessing

Data preprocessing is a must. There are three ways to inject the data for preprocessing:

- Open File – enables the user to select the file from the local machine
- Open URL – enables the user to select the data file from different locations
- pen Database – enables users to retrieve a data file from a database source

After loading the data in Explorer, we can refine the data by selecting different options. We can also select or remove the attributes as per our need and even apply filters on data to refine the result.

Classification

To predict nominal or numeric quantities, we have classifiers in Weka. Available learning schemes are decision-trees and lists, support vector machines, instance-based classifiers, logistic regression and Bayes' nets. Once the data has been loaded, all the tabs are enabled. Based on the requirements and by trial and error, we can find out the most suitable algorithm to produce an easily understandable representation of data.

Before running any classification algorithm, we need to set test options. Available test options are listed below.

Use training set: Evaluation is based on how well it can predict the class of the instances it was trained on.

Supplied training set: Evaluation is based on how well it can predict the class of a set of instances loaded from a file.

Cross-validation: Evaluation is based on cross-validation by using the number of folds entered in the 'Folds' text field.

Split percentage: Evaluation is based on how well it can predict a certain percentage of the data, held out for testing by using the values entered in the '%' field.
To classify the data set based on the characteristics of attributes, Weka uses classifiers.

Visualization: The user can see the final piece of the puzzle, derived throughout the process. It allows users to visualize a 2D representation of data, and is used to determine the difficulty of the learning problem. We can visualize single attributes (1D) and pairs of attributes (2D), and rotate 3D visualizations in Weka. It has the Jitter option to deal with nominal attributes and to detect 'hidden' data points.

3.1.2.2 Anaconda

Anaconda is a free and open source distribution of the Python and R programming languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment. To know more about Anaconda please visit: https://anaconda.org/anaconda/python

We used from Scipy, Pandas and Matplotlib packages from Anaconda. SciPy is an open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hard copy formats and interactive environments across platforms.

3.2 Algorithms Used

We employed three machine learning algorithms in our project and they include ANN (Artificial Neural Network), SVM (Support Vector Machine) and Decision Tree. Following sections discusses them one by one in detail.

3.2.1 Artificial neural networks

Artificial neural networks (ANNs) are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" (i.e. progressively improve performance on) tasks by considering examples, generally without task-specific programming. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any a priori knowledge about cats, e.g., that they have fur, tails, whiskers and cat-like faces. Instead, they evolve their own set of relevant characteristics from the learning material that they process.

An ANN is based on a collection of connected units or nodes called artificial neurons (a simplified version of biological neurons in an animal brain). Each connection (a simplified version of a synapse) between artificial neurons can transmit a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it.

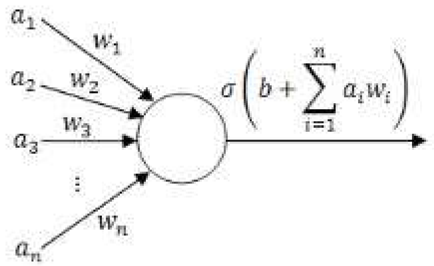

Figure 3: A single perceptron and its incoming connections

In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is calculated by a nonlinear function of the sum of its inputs. Artificial neurons and connections typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that only if the aggregate signal crosses that threshold is the signal sent. Typically, artificial neurons are organized in layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first (input), to the last (output) layer, possibly after traversing the layers multiple times.

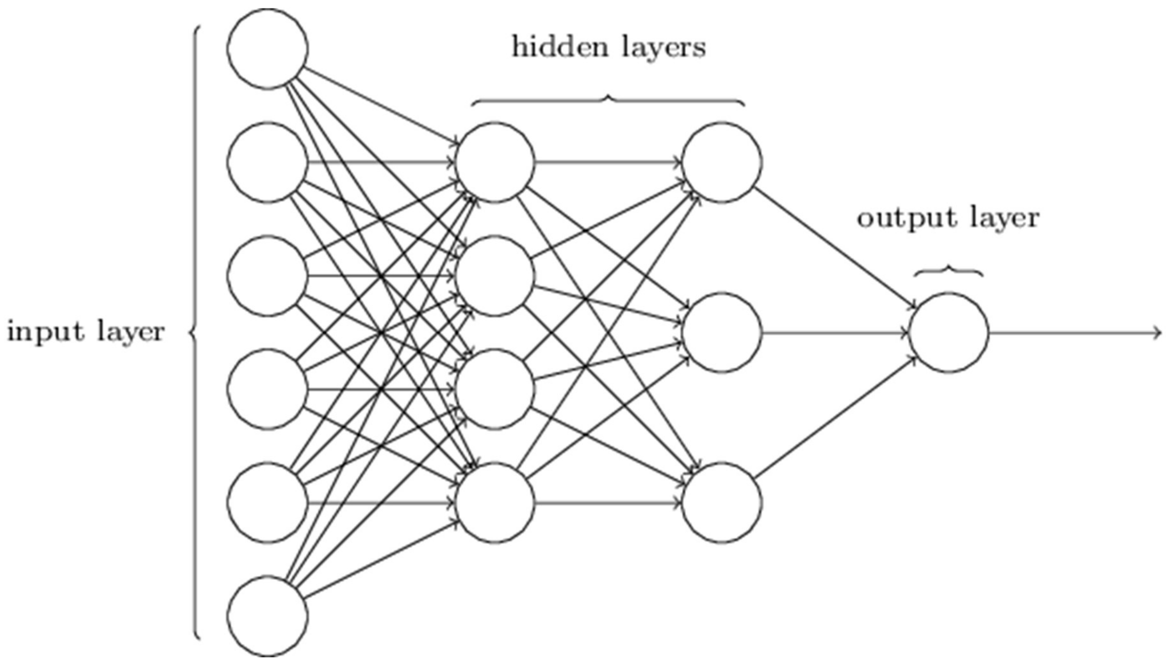

Figure 4: A network of perceptron's i.e. Neural Network

3.2.2 SVM

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

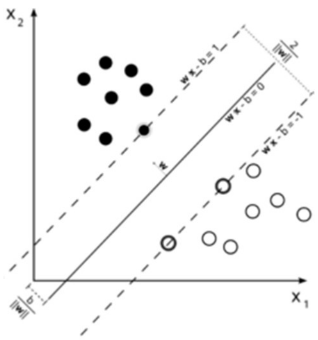

Figure 5: Support vectors and hyper plane

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

3.2.3 Decision Tree

A Decision Tree(DT) is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.
A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.
In decision analysis, a decision tree and the closely related influence diagram are used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated.
A decision tree consists of three types of nodes:-

1. Decision nodes – typically represented by squares
2. Chance nodes – typically represented by circles
3. End nodes – typically represented by triangles

Let’s look at the basic terminology used with Decision trees:

1. **Root Node:** It represents entire population or sample and this further gets divided into two or more homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.

3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
4. **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.
5. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
6. **Branch / Subtree:** A subsection of entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes whereas sub-nodes are the child of parent node.

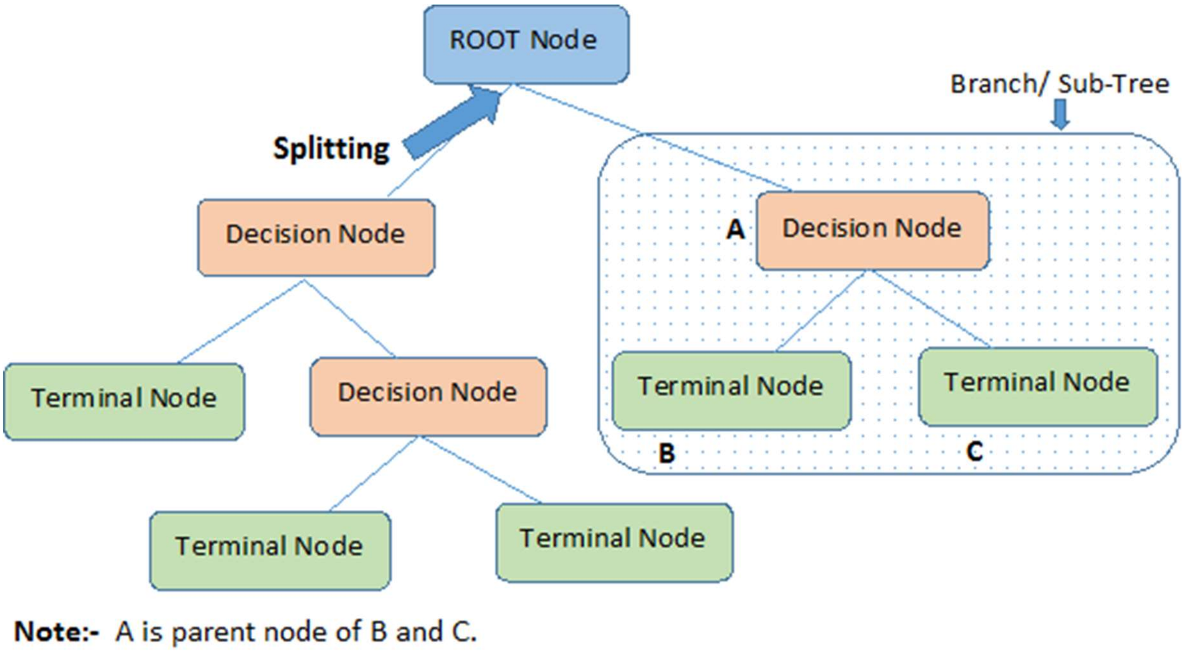

Figure 6: Decision tree with internal decision nodes

Types of decision tree is based on the type of target variable we have. It can be of two types:

1. Categorical Variable Decision Tree: Decision Tree which has categorical target variable then it called as categorical variable decision tree. Example:- In above scenario of student problem, where the target variable was “Student will play cricket or not” i.e. YES or NO.
2. Continuous Variable Decision Tree: Decision Tree has continuous target variable then it is called as Continuous Variable Decision Tree.

Example: - Let’s say we have a problem to predict whether a customer will pay his renewal premium with an insurance company (yes/ no). Here we know that income of customer is a significant variable but insurance company does not have income details for all customers. Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product and various other variables. In this case, we are predicting values for continuous variable.

How does a tree decide where to split?

The decision of making strategic splits heavily affects a tree’s accuracy. The decision criteria is different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable. Decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

3.2.4 SVM RFE

SVM Recursive Feature Elimination algorithm is also used for the selection of significant features. This algorithm fits the model according to all the features post which each feature is ranked using its importance to the model. At each iteration of feature selection, sequence of the top ranked features are retained, the model is refit and performance is assessed. So, in this algorithm, the training data is used for three purposes namely- Predictor or feature selection, model fitting and performance evaluation. It falls under the embedded method of feature selection which learns the features that best contribute to the accuracy of the model while the model is being created. Its computational complexity lies in between the filter and wrapper methods (greater than filter method and less than wrapper method).

The main purpose of SVM-RFE is to compute the ranking weights for all features and sort the features according to weight vectors as the classification basis. SVM-RFE is an iteration process of the backward removal of features. Its steps for feature set selection are shown as follows.

1. Use the current dataset to train the classifier.
2. Compute the ranking weights for all features.
3. Delete the feature with the smallest weight.

Implement the iteration process until there is only one feature remaining in the dataset; the implementation result provides a list of features in the order of weight. The algorithm will remove the feature with smallest ranking weight, while retaining the feature variables of significant impact. Finally, the feature variables will be listed in the descending order of explanatory difference degree. SVM-RFE's selection of feature sets can be mainly divided into three steps, namely, (1) the input of the datasets to be classified, (2) calculation of weight of each feature, and (3) the deletion of the feature of minimum weight to obtain the ranking of features. The computational step is shown as follows:

(*1) Input*

- Training sample: $X_0 = [x_1, x_2, \ldots, x_m]^T$.
- Category: $y = [y_1, y_2, \ldots, y_m]^T$.
- The current feature set: $s = [1, 2 \ldots n]$.
- Feature sorted list: $r = []$.

(*2) Feature Sorting*

- Repeat the following process until $s = []$.
- To obtain the new training sample matrix according to the remaining features: $X = X_0(:, s)$.
- Training classifier: α = SVM-train(X, y).
- Calculation of weight: $w = \sum_k \alpha_k y_k x_k$.
- Calculation of sorting standards: $c_i = (w_i)^2$.
- Finding the features of the minimum weight: $f = \arg\min \quad (c)$.
- Updating feature sorted list: $r = [s(f), r]$.
- Removing the features with minimum weight: $s = s(1{:} -1, f + 1{:}\ \mathrm{length}(s))$.

(*3) Output: Feature Sorted List r*. In each loop, the feature with minimum $(w_i)^2$ will be removed. The SVM then retrains the remaining features to obtain the new feature sorting.

SVM-RFE repeatedly implements the process until obtaining a feature sorted list. Through training SVM using the feature subsets of the sorted list and evaluating the subsets using the SVM prediction accuracy, we can obtain the optimum feature subsets.

CHAPTER 4: Proposed Work: Automatic Music Moods Classification (AM^{2}C)

The overall architecture of the proposed system can be summarized by Figure 7. The architecture consists of three phases. The first phase combines data aggregation and data cleaning. The output of the first phase were evaluated using feature selection algorithms and this concludes our second phase. The System uses three machine learning models ANN, DT, and SVM for performance evaluation.

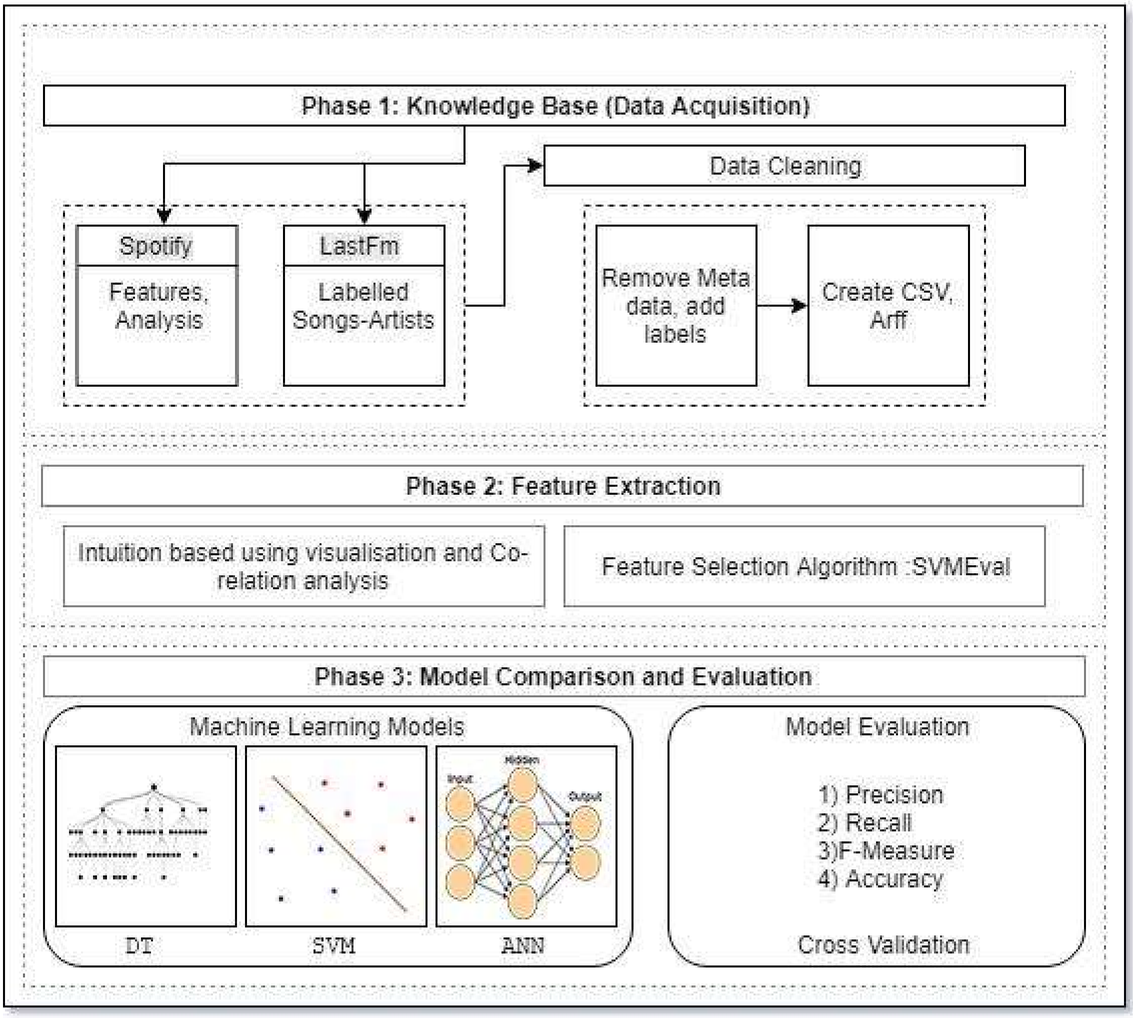

Figure 7: Architecture of the system

4.1 Data Acquisition and Data Cleaning

To achieve the system, a dataset of each category of song is needed. First, a list of songs is needed and then each song needs to be classified into a category. Traditionally, a group of experts or non-experts (subjects) may be asked to classify these songs manually. This approach is not scalable as building large datasets is tedious. Moreover, the categories may be biased with respect to the perceived emotion. Existing online music communities in the internet contain valuable metadata about music tracks like emotion tags. We fetch a list of songs using emotion tags from Last.fm that is a music website, which has a large corpora of music metadata and user listening habits. The procedure is shown in Figure 8.

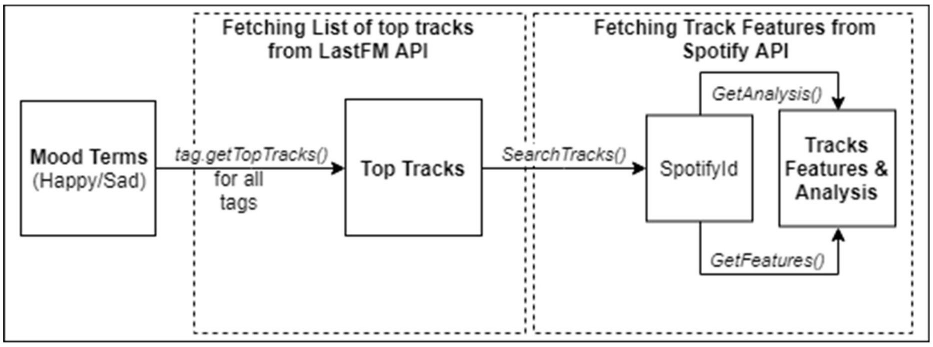

Figure 8: Data aggregation from LastFm and Spotify APIs

Once the list of songs has been compiled, we need a set of features to classify them. Music is a time-series signal, sampled at a frequency of 44 KHz (for a CD quality mp3). Using music in its raw form requires a large storage space and heavy computation. Thus, features are needed that can summarize music without losing much information. Coming up with the right feature set is difficult and is a challenging task. We get features for a track from Spotify, which is a digital music streaming service and provides music features through its API. The significance of the features from Spotify is that, they contain high level semantic information like danceability and accoustiness. The high-level semantic features are closer to human perception of music. The procedure is shown in Figure 8.

4.2 Data Exploration

After obtaining a set of feature, to get a sense of distribution of data, some statistics for each feature is calculated. For numeric type features we calculate minimum value, maximum value, standard deviation and mean. We then plot the frequency distribution charts for each feature, using different color codes for the two classes. For example, Figure 9 is a plot for feature energy.

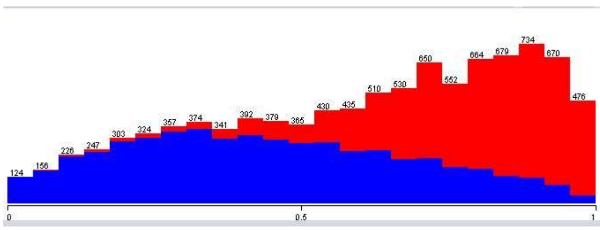

Figure 9: Frequency plot for Energy

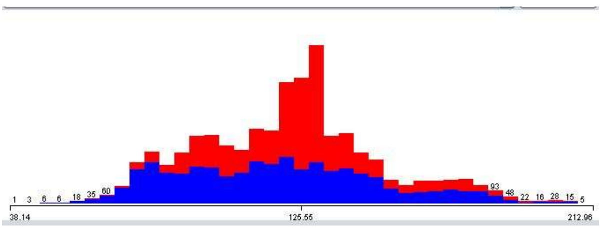

Figure 10: Frequency plot for Tempo

The dataset contains a total of 18 features; track_mode_confidence, track_key_confidence, Track_time_signature_confidence, track_time_signature, track_tempo_confidence, track_end_of_fade_in, duration_ms, tempo, valence, liveness, instrumentalness, accousticness, speechiness, mode, loudness, key, energy, danceability, and mood.

In, next step, we determine the usefulness of all the 18 features. We utilize the frequency distribution plots for this purpose. For feature *energy* Figure 9, most of the happy (red) songs have values greater than 0.5, while most Sad (blue) songs have values less than 0.5. We use this approach to intuitively evaluate features usefulness. Features like tempo Figure 10Figure 10 seems less useful as the distribution of both the categories of songs looks similar for it.

After primarily analysis, to come up with the right set of features, we apply SVM recursive feature elimination (SVM-RFE) algorithm. It ranks the features according to their merit of distinguishing the target class. It proceeds in recursive manner, we stop the procedure after no feature remains to be eliminated below a certain threshold merit value. To validate the process, we use 10 fold cross validation. The results of SVM-RFE elimination step is reported in section 4.

4.3 AI Classifiers

In this phase, we discuss the three algorithms ANN, SVM and Decision Tree that we use for the classification. In the following paragraphs, we start with the discussion of how we minimize the sampling biases using k-fold cross validation. Then we briefly discuss the classification algorithm. Finally, we discuss metrics used to evaluate and compare the algorithms.

In experiments that use k-fold cross validation, the data set is randomly partitioned into k subsets of approximately equal size. Then, the model is evaluated k-times i.e. k-folds. In each fold, one of the k subsets is hold back from training process for validating the model. Finally averaged result of the k-folds is reported. We use a variation of k-fold cross validation called stratified k-fold cross validation, in which each subset contain roughly the same proportions of class labels. To keep a balance between performance and computational time [ref from stock], we choose 10 number of folds.

Artificial Neural Networks (ANN) are biologically inspired computational network. They have been widely employed to solve pattern recognition, classification, and regression problems. They have been used previously for Music moods classification [2] [14]. In this paper we use sigmoid activation function for all the layers. We have also employed Multi-Layer Perceptron (MLP) learning model that uses backpropagation algorithm.

Decision trees are also widely used classification algorithms. They are popular for their simplicity and efficiency. The dataset is partitioned at root node, by applying a splitting criteria on a single feature. The resultant smaller datasets are further partitioned recursively, until maximal pure subsets are obtained. Some of the popular Decision Tree algorithms include ID3, C4.5, and CART. We use J48, which is an implementation of C4.5 in Java.

The third classification algorithm we use is SVM. A number of Music Classification works have shown their effectiveness in Music Moods Classification [10]. To classify, the data vectors are mapped to a high-dimensional space, followed by finding a hyperplane that separates the classes with maximum margin. We use a method of SVM implementation called SMO that avoid quadratic programming problem with SVM.

To compare the different models and their respective performance we use 1) Accuracy 2) F-Score 3) Recall. The selected metrics are suitable for binary classification problems and are widely used in other Machine learning literatures as well. For details of their calculation and their implications, readers are advised to go throw relevant reading sources.

CHAPTER 5: Results and Discussion

In this section, we discuss the classification performance observed for the three classifiers. Before that, we first discuss the feasibility of classification using visualization. Then, we discuss the features obtained using SVM-RFE. On the basis of which we take three scenarios of the used features. Finally we discuss the results of the three scenarios with respect to the three models.

We constructed a dataset of 16527 songs using the method discussed in section 4.1. To check the feasibility of classification and to get a sense of it we applied PCA for visualization. Following Figure 11 illustrates the plot obtained.

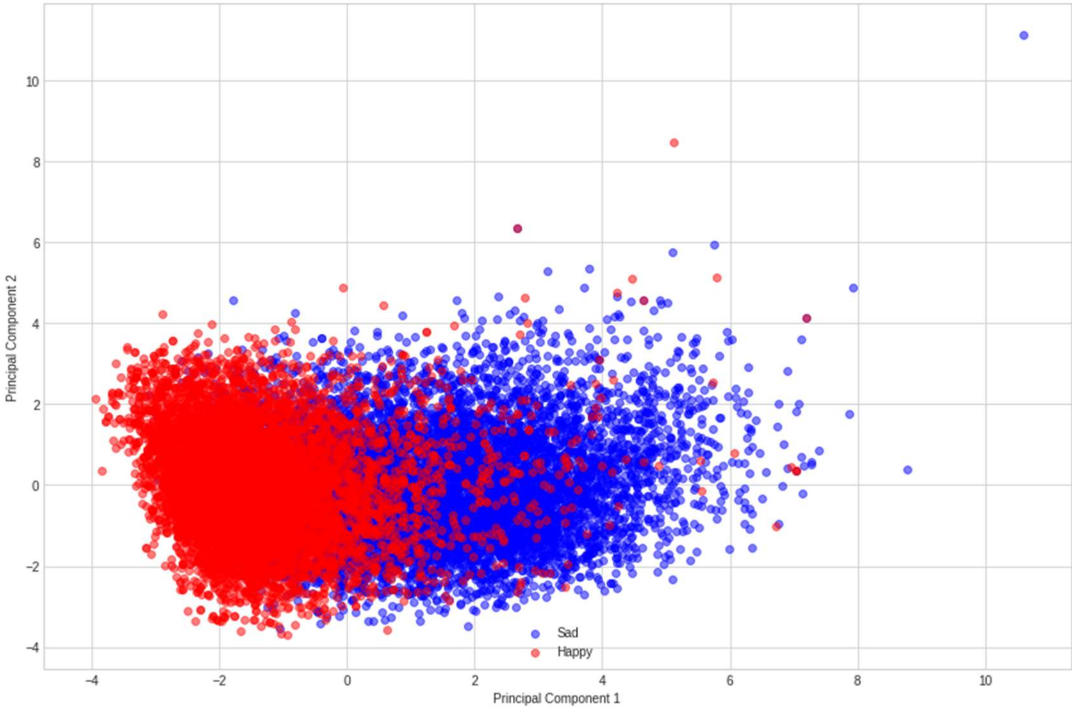

Figure 11: Data visualization through PCA

Following conclusions were made from above plot:
a) We can see that the positions of the red dots and those of the blue dots are such that it leads to the formation of two clusters. This vivid formation of the two clusters was because of the crowdsourced labels that we utilized to our advantage from LastFm. This gave the go ahead to perform linear classification.
b) We can see that the red dots are very much near to each other as compared to the position of blue dots. This suggests that there is relatively less variation in happy songs as compared to that obtained in the sad songs.

From data exploration we found the dataset to contain 8399 sad and 8128 happy songs. Features that appear good for classification on the basis of frequency plots are Track_tempo_confidence, valence, energy, danceability and loudness appear to be good choice for classification.

Table 3: Attributes ranked using SVM-RFE

Attribute	Average Rank	Average merit
Speechiness	1 +- 0	18 +- 0
Energy	2 +- 0	17 +- 0
Danceability	3 +- 0	16 +- 0
loudness	4 +- 0	15 +- 0
valence	5 +- 0	14 +- 0
duration_ms	6 +- 0	13 +- 0
accousticness	7.2 +- 0.6	11.8 +- 0.6
track_mode_confidence	8.6 +- 0.92	10.4 +- 0.917
instrumentalism	9.1 +- 1.14	9.9 +- 1.136
track_key_confidence	9.6 +- 0.92	9.4 +- 0.917
track_time_signature	10.9 +- 1.22	8.1 +- 1.221
liveness	11.7 +- 0.46	7.3 +- 0.458
track_tempo_confidence	13.1 +- 0.54	5.9 +- 0.539
track_end_of_fade_in	14.1 +- 0.7	4.9 +- 0.7
tempo	14.7 +- 0.46	4.3 +- 0.458
track_time_signature_confidence	16.5 +- 0.67	2.5 +- 0.671
key	16.8 +- 0.75	2.2 +- 0.748
mode	17.7 +- 0.46	1.3 +- 0.458

Table 3, displays the results obtained by SVM-RFE. The results are averaged for 10 folds of the cross validation. From this result, we observer that Speechiness, Energy, Danceability, Loudness, Valence, and duration_ms are top six valuable features.

5.1 Model Evaluation

Before Evaluating with actual features we applied SVM on the PCA, to obtain a minimum threshold of classification accuracy. The following Figure 12 shows the results.

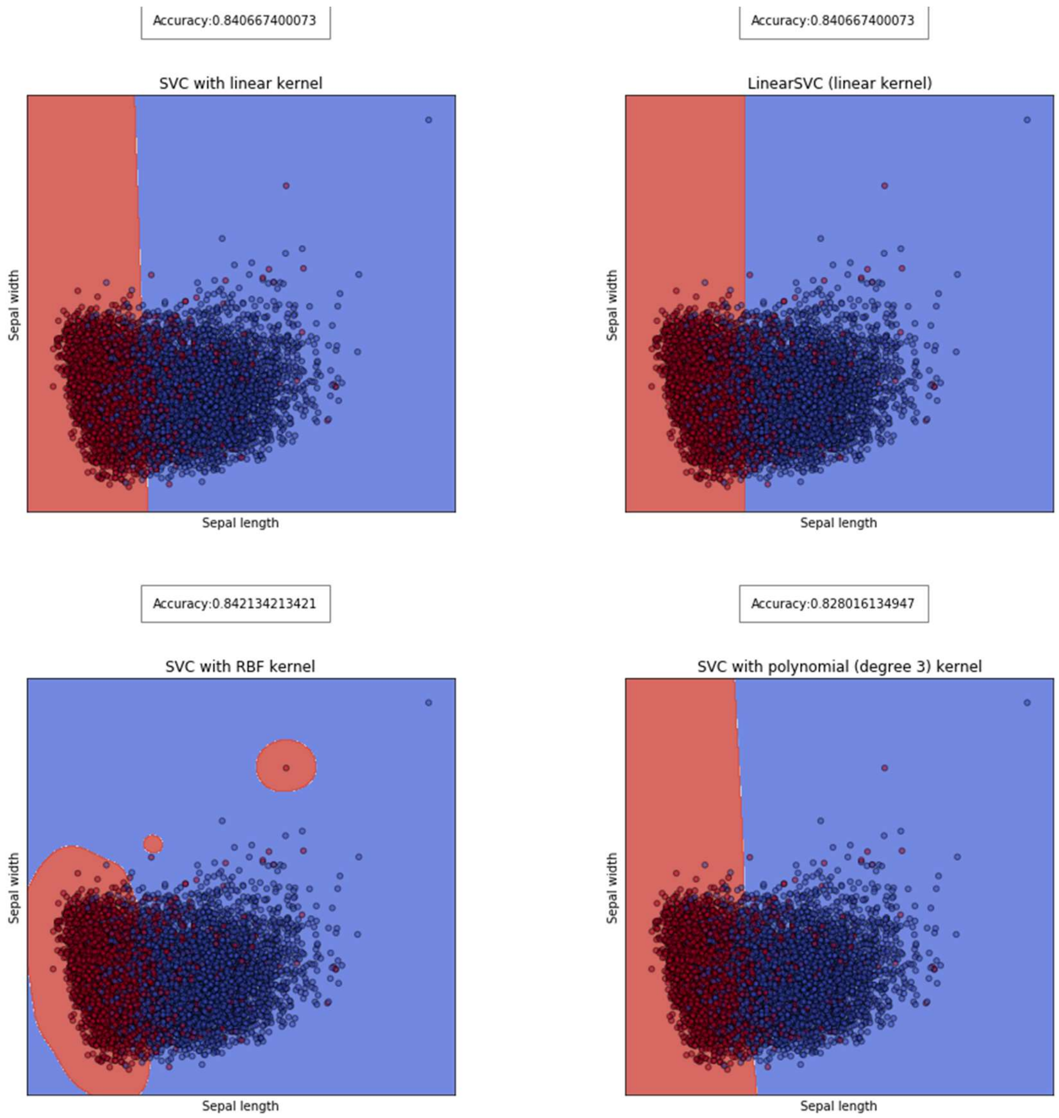

Figure 12: Different SVM Kernels applied on PCA Dataset

From above experiment we can expect to get at least 84.21 % accuracy.

Next, we take three experimental scenarios, the first one include all the features. The second scenario is on the basis of the merit obtained through SVM-RFE, it includes features with merit greater than 10.0. In third scenario we remove the last feature from the second scenario. Table x depicts the three different scenarios.

Table 4: Features in different scenarios

Scenario	Features		
One	Speechiness	accousticness	track_tempo_confidence
	Energy	track_mode_confidence	track_end_of_fade_in
	Danceability	instrumentalism	tempo

	loudness	track_key_confidence	track_time_signature_confidence
	valence	track_time_signature	key
	duration_ms	liveness	mode
Two	Speechiness	loudness	acousticness
	Energy	valence	track_mode_confidence
	Danceability	duration_ms	
Three	Speechiness	loudness	acousticness
	Energy	valence	
	Danceability	duration_ms	

In Table 5, we report the Precision, Recall and F-Measure, for the three scenarios. We observe that both ANN and SVM perform almost equal and better than Decision tree.

Table 5: Metrics for all the scenarios

Scenario One			
Algorithm	Precision	Recall	F Measure
ANN	**0.868**	**0.868**	**0.868**
SVM	0.863	0.862	0.862
Decision Tree	0.842	0.842	0.842
Scenario Two			
Algorithm	Precision	Recall	F Measure
ANN	0.867	0.867	0.867
SVM	0.86	0.859	0.859
Decision Tree	0.854	0.854	0.854
Scenario Three			
Algorithm	Precision	Recall	F Measure
ANN	0.866	0.866	0.866
SVM	0.859	0.858	0.858
Decision Tree	0.855	0.854	0.854

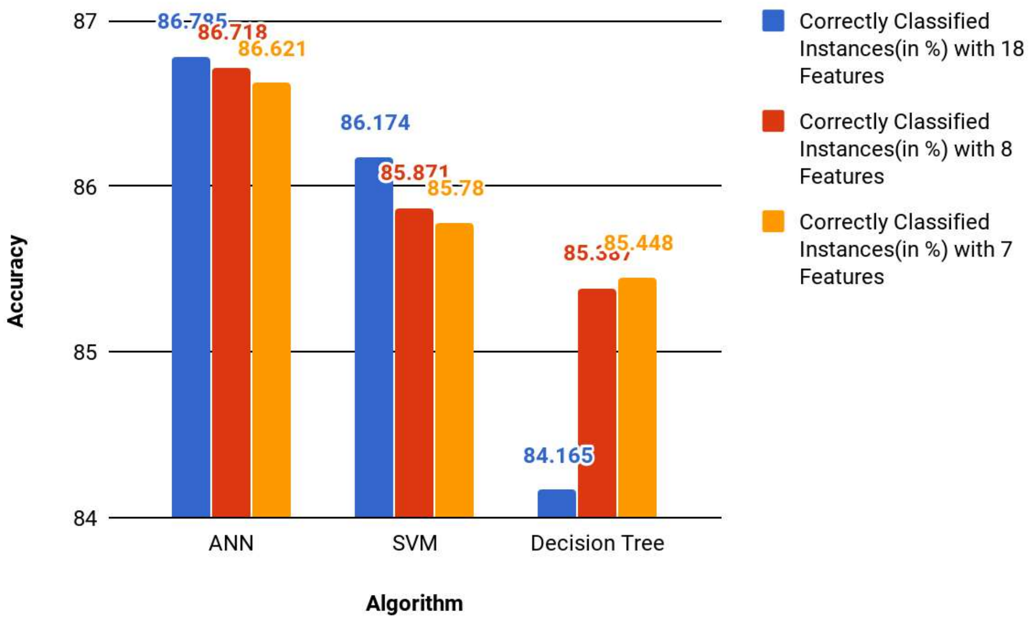

Figure 13: Accuracy of the three models in all the scenarios

Figure 13 compares the accuracy of the three classifiers. Artificial Neural Network has outperformed he other two algorithms by achieving an accuracy of 86.785%. This corresponds to the case where we have used all the 18 features. Also, it is evident that there is a very little difference between the accuracy results corresponding to the scenario with 8 features and the one with 7 features. This suggests that the contribution of the remaining features is relatively less in comparison to the ones in latter scenarios.

5.2 Discussion

The work we presented effectively classifies music in two classes. The effectiveness can be attributed to the large dataset we built using the LastFm platform. More dataset can be made using tags from other music communities like AllMusic (https://www.allmusic.com/). The sematic features from Spotify are very effective in classification, but that limits the ability of the system to classify songs available on Spotify.

CHAPTER 6: Conclusion

In this work, we addressed music mood classification problem and presented an Automated Music Mood Classification system (AM^2C). First we presented the literature survey of state-of-the-arts in music mood recognition and presented theoretical analysis of AM^2C with existing work.

The significant feature of AM^2C is to use crowd-source platforms to label the songs which eliminates the subjectivity of one's perception of mood on songs.

We have evaluated our system on a large and unbiased dataset of songs and the experimental results show that ANN performing best with 86.785% accuracy.

6.1 Future Work

The system is effective and shows the usability of online music metadata. It can be extended in various ways. We discuss some of them in the following list:

- More mood classes like Angry, Depressed, and Relaxed etc. can be added.
- Metadata from other music communities like Allmusic can be used to build bigger dataset.
- Sound Analysis tools like LibRosa can be used to get features for songs not on Spotify.
- Songs from variety of cultural and lingual background can be used.
- Probable application: playlist generator using proposed algorithms.

LIST OF REFERENCES

[1] X. Hu and J. Downie, When Lyrics Outperform Audio for Music Mood Classification: A Feature Analysis., vol. 53, 2010, pp. 619-624.

[2] B. G. Patra, D. Das and S. Bandyopadhyay, "Unsupervised Approach to Hindi Music Mood Classification," in *Mining Intelligence and Knowledge Exploration*, Springer International Publishing, 2013, pp. 62-69.

[3] T. Fritz, S. Jentschke, N. Gosselin, D. Sammler, I. Peretz, R. Turner, A. D. Friederici and S. Koelsch, "Universal Recognition of Three Basic Emotions in Music," *Current Biology,* vol. 19, pp. 573-576, 4 2009.

[4] X. Hu and Y.-H. Yang, "The mood of Chinese Pop music: Representation and recognition," *Journal of the Association for Information Science and Technology,* vol. 68, pp. 1899-1910, 6 2017.

[5] X. Yang, Y. Dong and J. Li, "Review of data features-based music emotion recognition methods," *Multimedia Systems,* 8 2017.

[6] Y. Feng, Y. Zhuang and Y. Pan, "Popular music retrieval by detecting mood," in *Proceedings of the 26th annual international {ACM} {SIGIR} conference on Research and development in informaion retrieval - {SIGIR} {\textquotesingle}03*, 2003.

[7] S.-H. Chen, Y.-S. Lee, W.-C. Hsieh and J.-C. Wang, "Music emotion recognition using deep Gaussian process," in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference ({APSIPA})*, 2015.

[8] B. Wu, E. Zhong, A. Horner and Q. Yang, "Music Emotion Recognition by Multi-label Multi-layer Multi-instance Multi-view Learning," in *Proceedings of the {ACM} International Conference on Multimedia - {MM} {\textquotesingle}14*, 2014.

[9] Y.-H. Yang, C.-C. Liu and H. H. Chen, "Music emotion classification," in *Proceedings of the 14th annual {ACM} international conference on Multimedia - {MULTIMEDIA} {\textquotesingle}06*, 2006.

[10] H. Ahsan, V. Kumar and C. V. Jawahar, "Multi-label annotation of music," in *2015 Eighth International Conference on Advances in Pattern Recognition ({ICAPR})*, 2015.

[11] J. A. Russell and M. Bullock, "Multidimensional scaling of emotional facial expressions: Similarity from preschoolers to adults.," *Journal of Personality and Social Psychology,* vol. 48, pp. 1290-1298, 1985.

[12] B. Patra, P. Maitra, D. Das and S. Bandyopadhyay, MediaEval 2015: Music Emotion Recognition based on Feed-Forward Neural Network, vol. 48, Springer Nature, 2015, pp. 633-651.

[13] H. He, J. Jin, Y. Xiong, B. Chen, W. Sun and L. Zhao, "Language Feature Mining for Music Emotion Classification via Supervised Learning from Lyrics," in *Advances in Computation and Intelligence*, Springer Berlin Heidelberg, 2008, pp. 426-435.

[14] M. Bartoszewski, H. Kwasnicka, U. Markowska-Kaczmar and P. B. Myszkowski, "Extraction of Emotional Content from Music Data," in *2008 7th Computer Information Systems and Industrial Management Applications*, 2008.

[15] P. N. Juslin and P. Laukka, "Expression, Perception, and Induction of Musical Emotions: A Review and a Questionnaire Study of Everyday Listening," *Journal of New Music Research,* vol. 33, pp. 217-238, 9 2004.

[16] M. Zentner, D. Grandjean and K. R. Scherer, "Emotions evoked by the sound of music: Characterization, classification, and measurement.," *Emotion,* vol. 8, pp. 494-521, 2008.

[17] P. N. Juslin, S. Liljeström, P. Laukka, D. Västfjäll and L.-O. Lundqvist, "Emotional reactions to music in a nationally representative sample of Swedish adults," *Musicae Scientiae,* vol. 15, pp. 174-207, 7 2011.

[18] K. R. Scherer, "Emotion as a multicomponent process: A model and some cross-cultural data.," *Review of Personality \& Social Psychology,* vol. 5, pp. 37-63, 1984.

[19] K. R. Scherer, "Emotion as a multicomponent process: A model and some cross-cultural data.," *Review of Personality \& Social Psychology,* vol. 5, pp. 37-63, 1984.

[20] X. Hu and Y.-H. Yang, "Cross-Dataset and Cross-Cultural Music Mood Prediction: A Case on Western and Chinese Pop Songs," *{IEEE} Transactions on Affective Computing,* vol. 8, pp. 228-240, 4 2017.

[21] Y.-H. Yang and X. Hu, "Cross-cultural Music Mood Classification: A Comparison on English and Chinese Songs," in *ISMIR*, 2012.

[22] X. Hu and Y.-H. Yang, "Cross-cultural mood regression for music digital libraries," in *{IEEE}/{ACM} Joint Conference on Digital Libraries*, 2014.

[23] T.-L. Pao, Y.-M. Cheng, J.-H. Yeh, Y.-T. Chen, C.-Y. Pai and Y.-W. Tsai, "Comparison between Weighted D-KNN and Other Classifiers for Music Emotion Recognition," in *2008 3rd International Conference on Innovative Computing Information and Control*, 2008.

[24] R. Mayer, R. Neumayer and A. Rauber, "Combination of audio and lyrics features for genre classification in digital audio collections," in *Proceeding of the 16th {ACM} international conference on Multimedia - {MM} {\textquotesingle}08*, 2008.

[25] M. Huang, W. Rong, T. Arjannikov, N. Jiang and Z. Xiong, "Bi-Modal Deep Boltzmann Machine Based Musical Emotion Classification," in *Artificial Neural Networks and Machine Learning – ICANN 2016*, Springer International Publishing, 2016, pp. 199-207.

[26] L. Lu, D. Liu and H.-J. Zhang, "Automatic mood detection and tracking of music audio signals," *{IEEE} Transactions on Audio, Speech and Language Processing,* vol. 14, pp. 5-18, 1 2006.

[27] S. Goyal and E. Kim, "Application of fuzzy relational interval computing for emotional classification of music," in *2014 {IEEE} Conference on Norbert Wiener in the 21st Century (21CW)*, 2014.

[28] E. Nest, *Analyzer Documentation,* vol. 52, E. Nest, Ed., 1999, p. vii.

[29] P. Ekman, "An argument for basic emotions," *Cognition and Emotion,* vol. 6, pp. 169-200, 5 1992.

[30] J. L. Armony, "Affective Computing," *Trends in Cognitive Sciences,* vol. 2, p. 270, 7 1998.

[31] P. Ekkekakis and S. J. Petruzzello, "Acute Aerobic Exercise and Affect," *Sports Medicine,* vol. 28, pp. 337-374, 1999.

[32] L.-L. Balkwill and W. F. Thompson, "A Cross-Cultural Investigation of the Perception of Emotion in Music: Psychophysical and Cultural Cues," *Music Perception: An Interdisciplinary Journal,* vol. 17, pp. 43-64, 10 1999.