Songs Analysis

Thomas Astad Sve
University of California,
San Diego
& Norwegian University of
Science and Technology
thomaasv@stud.ntnu.no

Executive Summary

This project does analytics work on the million song dataset. Through doing exploratory data analysis we try to find patterns in the data that can help understand the music better and how the music is developing. The analytical work focuses on the following questions, can a songs popularity be foreseen? Can you predict the release year or song genre with only looking at the track metadata? By the track metadata, this is values such as a tracks loudness, duration and tempo. We did predictive analytics using different classification models (SVM, Random Forest and Decision Tree) trying to answer the questions mentioned above. This paper failed to find clear correlations between the track metadata and its popularity, genre or year release. The predicitve models struggeled to get good results on the validation sets, even after massive tuning of parameters. With this we can only conclude that the track metadata is not sufficient enough to predict a songs popularity, release year or genre.

Contents

1 Introduction

Music has had an important role in our culture troughout human history. As a result of this importance, follows a billion dollar industry. In 2014 alone, this industry generated 15 billion dollars [1]. Most of the income is made from mainstream popular songs and goes straight to the hands of record labels companies. Having a good understanding and of songs is a valuable skillset to have in this industry. Not only is it important as for finding hits, but it is also important for people working with radio stations or with social happenings to have the right music for the right crowd.

Our project will try to see if the next hit can be predicted, a song genre can be predicted and if it can label different songs to different years or decades. If these predictions are successful, it can mean that the record labels and radio stations will get a easier job in finding the next hit or the perfect song for the right radio station. For instance, a radio station with music targeted for the fans of the rock genre wants to find songs that can fit this audience. Music taste is different from people to people, and being able to find the next hit for the right audience can make a big economical difference. Also by trying to predict a songs year release, we can try and find a pattern in how the music is changing over the years and maybe we can predict how the future music will look like.

2 Related Work

The problem of predicting song popularity has been heavily researched. Three students at Stanford University, Pham, Kyauk and Park, did a song popularity prediction with a computer science view [7]. In their work they took the one percentage subset of the million song dataset and then labeled the top 25% hottest songs as popular. They had a more complex classification and feature extraction than we will do in this project. For instance they included a feature named artist familiarity, which as not surprisingly turned out to be the most important feature when predicting the song popularity. We will not include features such as these, and will focus on the track metadata to see if we can get similar results.

Salganik, Dodds, and Watts conducted an study on popularity and concluded that a songs quality only partially influences whether or not a song becomes popular [8]. If we get the similar results, we might not be able to predict a songs popularity just from its track metadata. We would have to include artists popularity or familiarity as the stanford students did to add the social influence of popularity.

3 Dataset

3.1 Data

We use song track data from The Million Song Dataset [3]. The Million Song Dataset is a exhaustive collection of audio features and metadata on a million different songs. In total the collection has a size of 280GB, which is of a size we will not be able to download due to limited space on the computer. Even if we don't want to use the audio features, we would still have to download the dataset in its full size before extracting the features we want to work on. Therefore we will work on two different subsets of the dataset. One is a subset with all the information included, but reduced

to 10 000 songs instead of a million. This dataset has the perk of having all the information and am therefore able to work with the dataset as I please. On the webpage of The Million Song Dataset is a dataset where they have extracted and by combining The Million Song Dataset with GZTAN genre dataset to create a subset with genre included [6]. This is the subset I will be using to do genre predictions, and has a total of 59 600 samples. I am not going to use this subset for year and popularity prediction due to the lack of the year and song hottness features.

3.2 Features

The dataset can be split into two different parts. The first part contains the track metadata, and the second the sequenced data. From the track metadata we have features such as duration, tempo, loudness, confidence, song hotttnesss, tempo and year. From the sequence data we have timing information (start, duration of sequence) as well as loudness, pitch and timbre features in each sequence. For this project I will not include the sequenced data in my work, because using the sequenced data I will be trying to figure out a deep psychological patterns in what kind of music appeals to people. Using only the track metadata, it will therefore be easier to read and understand the results we are getting. For more information about the sequenced data, I suggest reading the echonest Analyser documentation [10].

For the two different subsets we will be working on there is a slight difference in features and table 1 gives a small description on the features included in the two datasets. For a full description on all the features in The Millon Song Dataset there excist a more detailed description of all the features on their website [5].

Feature	One-Percentage MSD	Genre MSD	Explanation
Track ID	Yes	Yes	The Echo Nest ID of this track
Genre	No	Yes	Genre of the track
Artist Name	Yes	Yes	The name of the artist
Title	Yes	Yes	Title of the song
Loudness	Yes	Yes	General loudness of the track
Tempo	Yes	Yes	Tempo in BPM
Time Signature	Yes	Yes	Time signature of the song
Key	Yes	Yes	Estimation of the key the song is in
Mode	Yes	Yes	Estimation of the mode the song is in
Duration	Yes	Yes	Duration of the track in seconds
Song Hotttnesss	Yes	No	Song hottnesss on a scale from 0 and 1
Year	Yes	No	Year when this song was released

Table 1: A description of the features included in the datasets, MSD = Million Song Dataset

3.2.1 Song Popularity Problem

For predicting a songs popularity we will take the song hottness feature and label the 25% with the highest value as popular and the other as not popular. Doing this we are adding a new feature named popular with 0 and 1 with popular as being 1. Not all the 10 000 songs has the song hotttness feature, these songs we will therefore leave out, and after extracting only the songs with the song hottness feature, we are left with a total of 4513 songs, where 1129 is popular and 3384 labeled as

not popular. This is a very small dataset for the work we want to do, but we might be able to get good results even with the small dataset. Comparing the size to the work of Pham, Kyaud and Park they ended up with a total of 2717 tracks from the same dataset when they also labeled the top 25% as popular so we have a significantly bigger dataset than theirs, but still very small one.

3.2.2 Song Year Problem

We want to predict a songs year release date using the same data as the previous works.

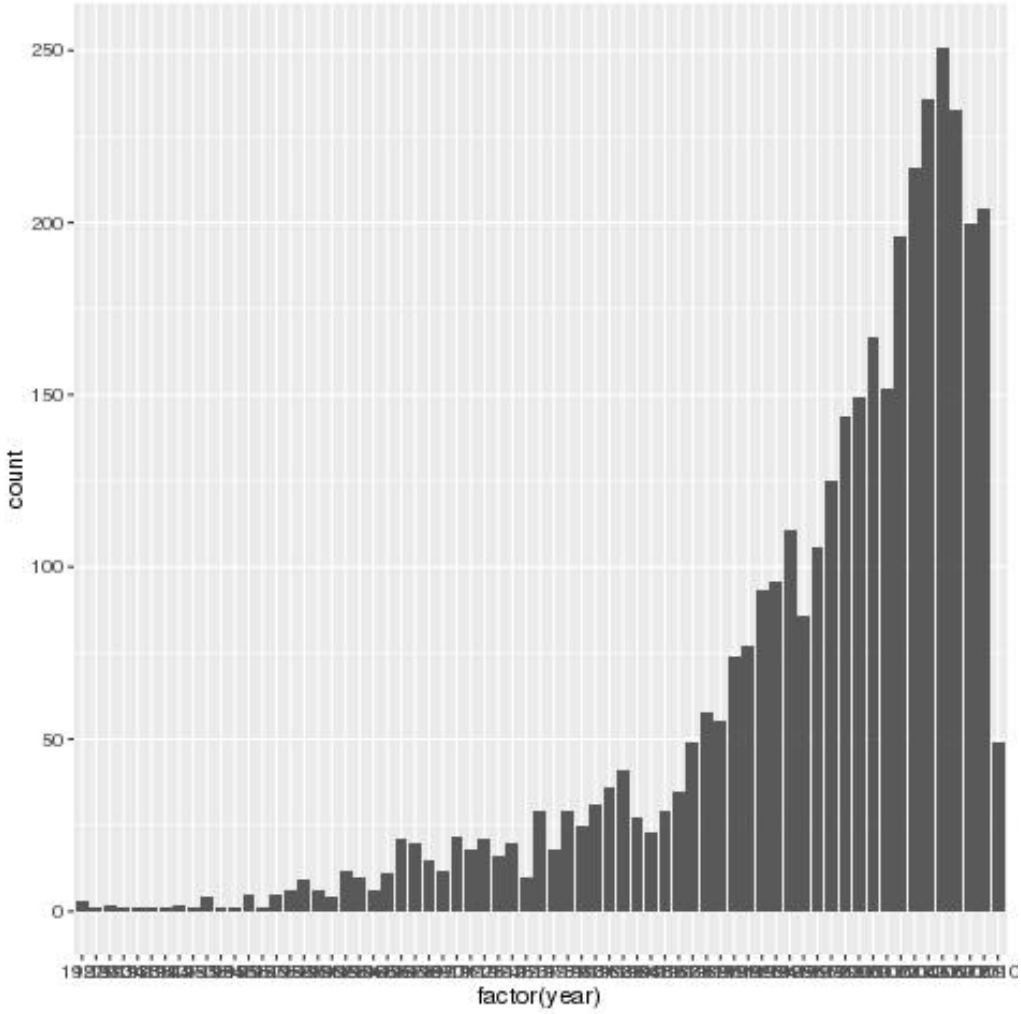


Figure 1: The year distribution

Looking at figure 1 we can see that there are too many different year classes, in fact in total there are 67 different years on all of the songs. You can also see that some years have significantly

more songs than others, and trying to preprocess this data can be a challenge. Therefore we decided to cut this number of classes down to five.

A new feature in the dataset was now created, decade. We labeled each song into the 10 years period it beloged to. For instance, songs released between 1985 and 1994 got 1985 as label. All songs released earlier than 1974 got labeled as 1965. We chose to split on half decade (1985-1994 and not 1980-1989) because the class 1990-1999 was much larger than the other classes. Splitting the classes on 1995, 1985 etc helped getting a more even distribution.

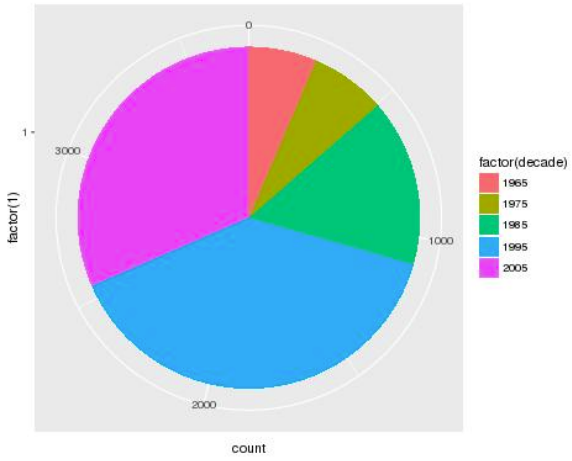


Figure 2: Distribution of the years class data set after preprocessing

Figure 2 shows the different year decades, or periods. The distribution is not very even with the two classes having the 90s in them is significantly larger than the others. One solution is to split the classes in different ways where we dont follow a ten years period on all the classes. For instance having classes like 1990-1994, 1995-1999 and 2000 - 2016 could improve the distribution. Because we want to find a change over the years, we dont want to go here as it could mean that its even harder to do a good analysis with the data over the years.

3.2.3 Song Genre Problem

When trying to predict the song genre, there already is a complete genre dataset from the Million Song Dataset we will use, this dataset has a few of the audio features included, but we will not use these and only focusing on the track metadata as mentioned before.

From figure 3, we can see the distrubution between the different genres in the dataset. There is clearly a majority of the

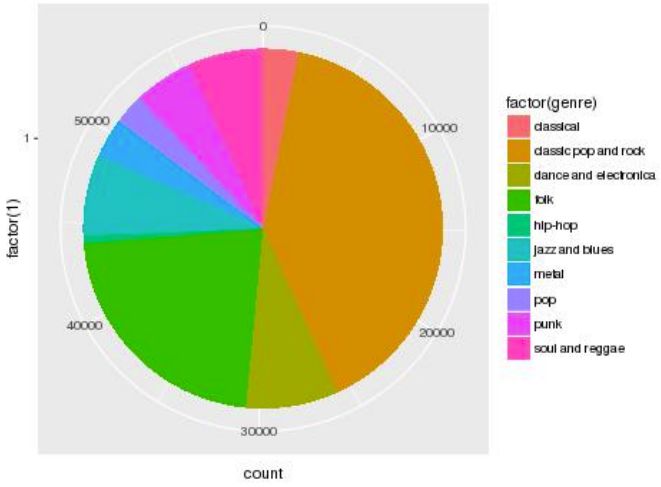


Figure 3: Distribution of the genres in the genre data set

classic pop and rock genre, while the simple pop genre have few samples. This is a typical problem in data sets that one class have more samples than the other classes, and can lead to a bad results. This can be a challenge thats hard to fix without increasing the number of samples to even out the distribution.

4 Methods and Quantitative Results

4.1 Exploratory Data Analysis

With exploratory data analysis we are analysing the data sets to summarize their characteristics, using visual methods. In this section we will visualize the datasets to find correlation between different features.

From figure 4 we are trying to see if genre and loudness shows correlation. From knowing a little about music you would guess some genres have higher loudness than others. Even if we can see that the metal genre is the loudess genre, we can also see that many genres have loudness values all over the scale, making it hard to define the genre by just looking at the loudness. Later we will do a predictive analysis on the genre, and then we may see more on if the models can find a way to classify the different genres

Figure 5 shows the correlation between duration and year. We can see similar results as with figure 4, that all the songs from the different decades or year periods have values all over the scale and its hard to predict the year by just looking at the duration.

4.2 Predictive Analytics Model

For predictive analysis, this project have used three different classification methods. This is used becuase often different classification models can be better for different kinds of problems. Also there is a significant time difference in how long it take to train a model, depending on the problem and model. By using several different models, we have a better chance of getting a better total results. The three different methods used is **Decision Tree**, **Support Vector Machine (SVM)** and **Random Forest**. When doing predictive analytics we have splitted the datasets into training, validation and test set. The results showed in this section, is the results on the testset after training on trainingset and tuning the parameters with the validationset.

4.2.1 Predicting if a song is popular

When predicting songs popularity we decided to try and see if the top 25 percentage of the hottest songs had something in common. First we tried with a smaller percentage, but because the popular class now had a very low samples in the class we decided to increase to 25 percentage.

Figure 7 shows the predicted vs observed plots on all three classifiers. This shows how all the different classifiers struggeles to classify this problem.

The results correlates with Salganik, Dodds and Watts conclusion. As mentioned in the related work section, they concluded that a songs quality only partially influences whether or not a song becomes popular. Also when looking closer to Pham, Kyauk and Parks work their most important

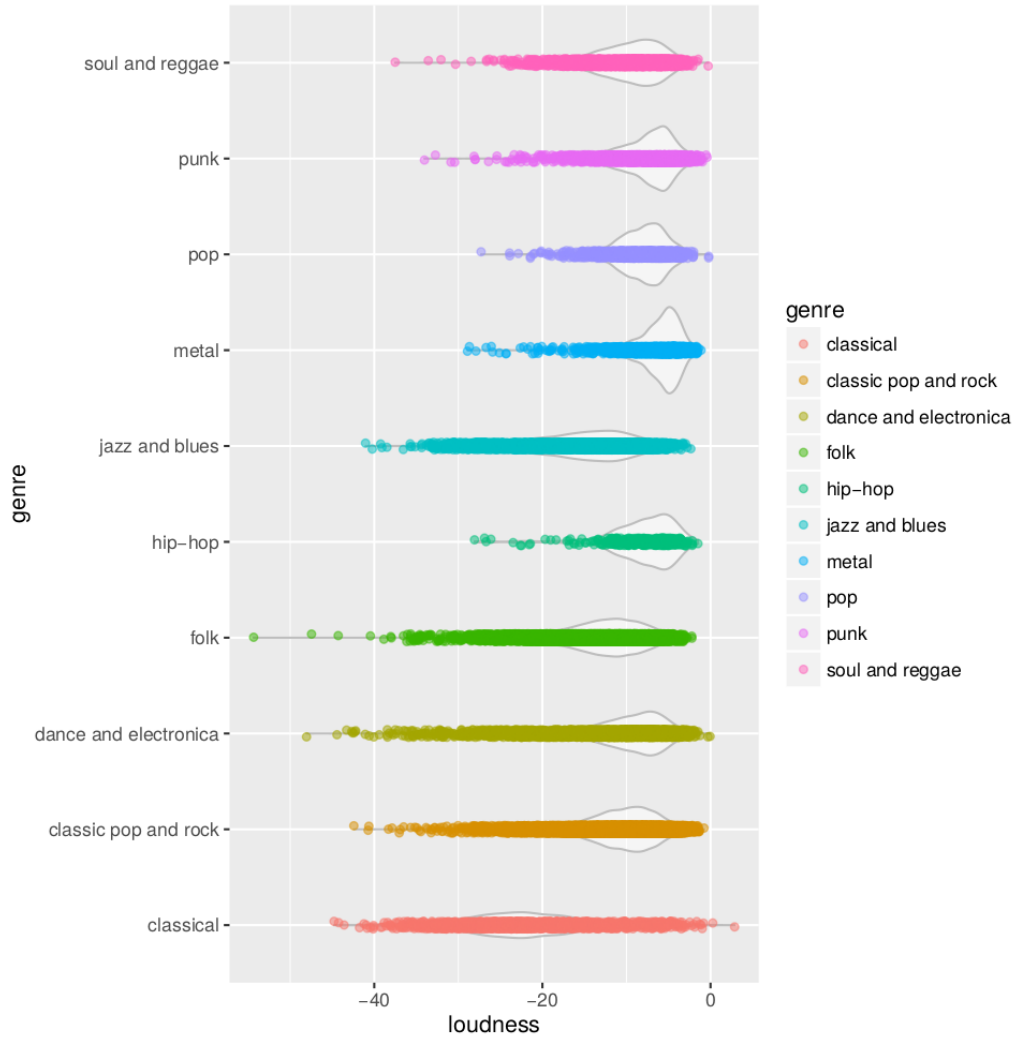


Figure 4: The correlation between Genre class and loudness

	Predicted					
	Decision Tree		Random Forest		Support Vector Machine	
Actual	0	1	0	1	0	1
0	487	29	471	45	511	5
1	133	27	127	33	151	9
Error	24%		25%		23%	

Table 2: Error Matrix for the three different classifiers for song popular prediction

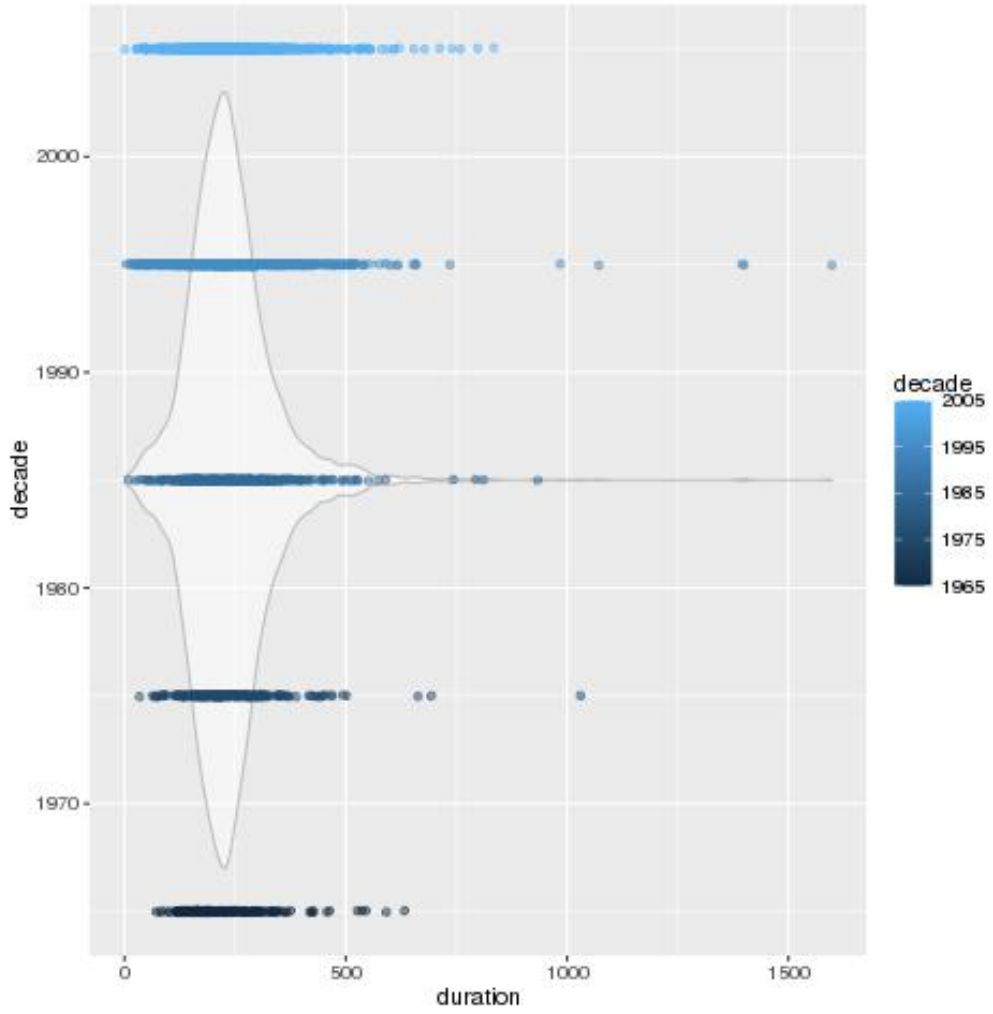


Figure 5: The correlation between duration and year

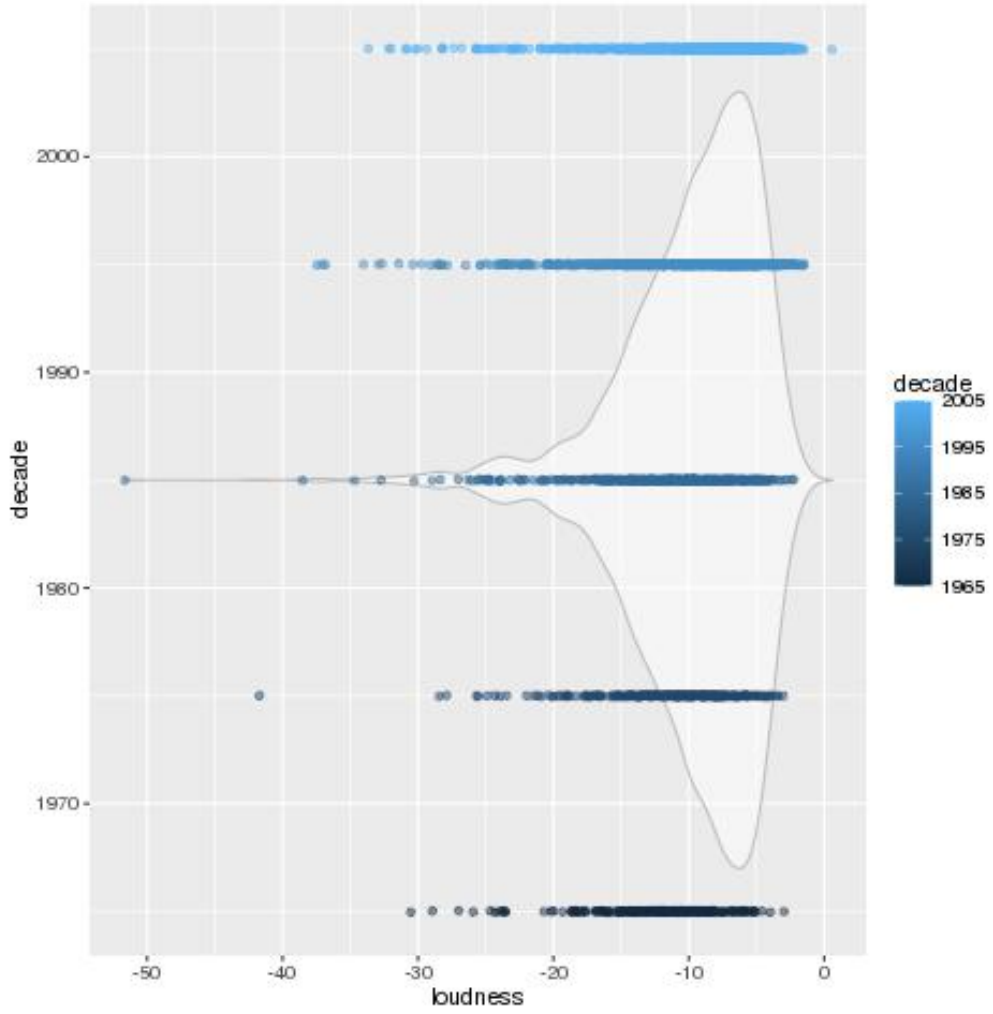
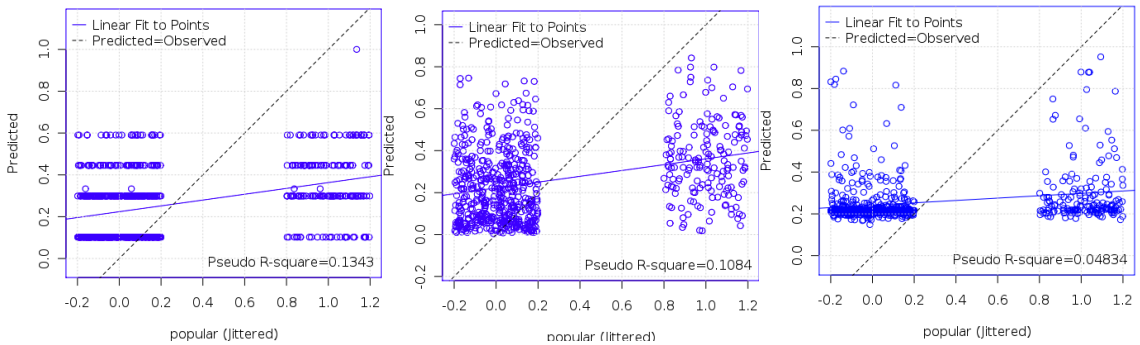


Figure 6: The correlation between loudness and year



(a) Decision Tree (b) Random Forest (c) Support Vector Machine

Figure 7: Predicted vs observed plots on testset

feature for song popularity prediction is the artist familiarity feature, which is a feature we do not have.

4.2.2 Predicting a song release year

	Predicted														
	Decision Tree					Random Forest					Support Vector Machine				
Actual	65	75	85	95	05	65	75	85	95	05	65	75	85	95	05
1965	0	0	0	42	1	0	1	8	6	1	0	0	2	16	2
1975	0	0	0	34	2	0	1	6	8	6	0	0	0	21	3
1985	0	0	0	88	6	0	0	7	27	15	0	0	0	46	14
1995	0	0	0	179	38	0	1	8	88	54	0	0	0	101	47
2005	0	0	0	122	45	0	0	9	67	47	0	0	0	63	60
Error	60%					60%					57%				

Table 3: Error Matrix for the three different classifiers used for the song year prediction on the test set

From table 3 we see that all the three classifiers struggeles to classify the problem. Trying to tune the classifyers, such as prune the decision tree, had no success. There is small differences in the classifyers with the results. The decision tree only seperate between the 1995 and 2005 classes, SVM classifier almost have the same

4.2.3 Predicting a song genre

From the error matrix in table 4 we can see that the Decision Tree classifyer only classify the object into three different classes. This may be a result that it struggeles to find good splits between the different classes.

The error matrix in table 5 we have similar results as from the decision tree matrix. While the decision tree only classifies three classes, the random forest classifier tries to classify on all classes

	Predicted										
Actual	1	2	3	4	5	6	7	8	9	10	Class error
"classical"	0	93	0	225	0	241	0	0	0	0	1.00
"classic pop and rock"	0	6238	0	714	0	168	0	0	0	0	0.12
"dance and electronica"	0	1263	0	192	0	99	0	0	0	0	1.00
"folk"	0	2887	0	826	0	194	0	0	0	0	0.77
"hip-hop"	0	120	0	7	0	0	0	0	0	0	1.00
"jazz and blues"	0	737	0	228	0	352	0	0	0	0	0.78
"metal"	0	621	0	20	0	2	0	0	0	0	1.00
"pop"	0	445	0	9	0	4	0	0	0	0	1.00
"punk"	0	954	0	34	0	2	0	0	0	0	1.00
"soul and reggae"	0	1109	0	76	0	20	0	0	0	0	1.00
Error	**59%**										

Table 4: Error Matrix using Decision Tree as classification model

	Predicted										
Actual	1	2	3	4	5	6	7	8	9	10	Class error
"classical"	173	98	23	154	0	113	3	2	1	1	1.00
"classic pop and rock"	32	5950	147	749	1	127	62	5	109	18	0.07
"dance and electronica"	54	912	243	150	0	81	27	0	24	11	1.00
"folk"	59	2552	35	1078	0	141	17	1	37	3	0.81
"hip-hop"	0	99	5	1	0	2	10	0	12	5	1.00
"jazz and blues"	86	597	37	242	0	313	8	0	6	3	0.82
"metal"	1	366	23	22	2	6	133	2	29	6	1.00
"pop"	0	398	19	24	0	2	6	9	7	6	1.00
"punk"	2	653	13	53	0	6	31	1	199	6	1.00
"soul and reggae"	3	992	31	78	0	21	38	2	13	24	1.00
Error	**54%**										

Table 5: Error Matrix using Random Forest as classification model

and is successfull to reduce the error percentage down from 59% to 54%. Still also this is not a great results, and the classifier struggeles to predict genre using the track metadata.

None of the different classifiers is successful in doing predictive analysis on the genre using the data provided.

5 Conclusion

In this project we have tried to do predictive and exploratory analysis work on three different problems using two subsets of The Million Song Dataset. This project could not find a good way to do predictive analytics and the exploratory data analysis did not give any good results either. Therefore we can conclude that having a dataset with the track metadata can not predict a songs popularity, genre or release year.

6 Acknowledgement

Many thanks to Professor Roger Bohn for providing me with his guidance and advise throughout the project.

References

[1] Tim Ingham, "Global record industry income drops below 15BN dollars for first time in decades", paril 14 2015, abailable at: http://www.musicbusinessworldwide.com/global-record-industry-income-drops-below-15bn-for-first-time-in-history/

[2] Mike Masnick, Yes, "Major Record Labels Are Keeping Nearly All The Money They Get From Spotify, Rather Than Giving It To Artists", Feb 5th 2015, available at: https://www.techdirt.com/articles/20150204/07310329906/yes-major-record-labels-are-keeping-nearly-all-money-they-get-spotify-rather-than-giving-it-to-artists.shtml

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011), 2011.

[4] Million Song Dataset, official website by Thierry Bertin-Mahieux, available at: http://labrosa.ee.columbia.edu/millionsong/

[5] An Example Track Description, The Million Song Dataset, available at: http://labrosa.ee.columbia.edu/millionsong/pages/example-track-description

[6] Deriving a genre dataset, The Million Song Dataset, available at: http://labrosa.ee.columbia.edu/millionsong/blog/11-2-28-deriving-genre-dataset

[7] James Pham, Edric Kyauk and Edwin Park. "Predicting Song Popularity" (2015) http://cs229.stanford.edu/proj2015/140$_r$*eport.pdf*

[8] Salganik, Matthew J., Peter Sheridan Dodds, and Duncan J. Watts. "Experimental study of inequality and unpredictability in an artificial cultural market." (2006)

[9] Derek Thompson. "The Shazam Effect". The Atlantic (2014). http://www.theatlantic.com/magazine/archive/2014/12/the-shazam-effect/382237/

[10] Tristan Jehan and David DesRoches. "Analyzer Documentation". The Echonest http://developer.echonest.com/docs/v4/$_s$*tatic/AnalyzeDocumentation.pdf*

Appendices

A Project code

A.1 Preprocessing data

A.1.1 Preprocessing dataset for year prediction

```
##
## Thomas Astad Sve
## Script to clear the year dataset, so there is no samples with NA or 0 value in year
## This script also creates a new column to reduce the number of classes
##

## -- Load dataset --
print("Loading dataset...")
MSongs <- read.csv("../datasets/msd_simple.csv", header = TRUE)

## Removing duplicates
print("Removing duplicates...")
MSongs <- MSongs[!duplicated(MSongs[,c('artist_name', 'title', 'track_id')]),]

## Remove columns where year is zero, year has column 11
print("Removing columns where year is 0...")
cleanYMS <- MSongs[apply(MSongs[11], 1, function(z) any(z!=0)),]

## Reducing classes to five, using decades
print("Reducing to five classes")
cleanYMS$decade <- ifelse(cleanYMS$year>=2005, 2005,
                          ifelse(cleanYMS$year>= 1995, 1995,
                                 ifelse(cleanYMS$year>=1985, 1985,
                                        ifelse(cleanYMS$year>= 1975, 1975, 1965))))

print("Saving new clean dataset...")
write.csv(cleanYMS, "../datasets/msd_year.csv")
print("Saved!")

library(ggplot2, quietly=TRUE)
cat(paste("Saving a plot for year distribution \n"))
jpeg('year_distribution_full.jpg')
c <- ggplot(cleanYMS, aes(factor(year)))
plot(c + geom_bar())
dev.off()
```

A.1.2 Preprocessing dataset for popularity prediction

```
##
## Thomas Astad Sve
## Script to create a popular column in the msd dataset, and cleans duplicates
##

## -- Load dataset --
print("Loading dataset...")
file <- "../datasets/msd_simple.csv"
MSongs <- read.csv(file, header = TRUE)

## Removing duplicates
MSongs <- MSongs[!duplicated(MSongs[,c('artist_name', 'title')]),]

## Sort data by hotness
print("Sorting dataset by hottnessss")
MSongs <- na.omit(MSongs[order(-MSongs$song_hotttnesss),])

## Insert new value with the most popular songs as 1, rest 0
print("Adding new variable, popular")

## Take top 25% of the hottest songs as popular
n = 25
MSongs$popular <- ifelse(MSongs$song_hotttnesss < quantile(MSongs$song_hotttnesss,prob=1-n/100),0,1)

print(paste("Num samples: ", nrow(MSongs)))

numUnPop <- length(MSongs$popular[MSongs$popular == 0])
numPop <- length(MSongs$popular[MSongs$popular == 1])

print(paste("Num Popular: ", numPop, " Num unPopular: ", numUnPop))

## Save dataset
print("Saving the new popular dataset")
write.csv(MSongs, "../datasets/msd_pop.csv")
print("Saved!")
```

A.2 Predictive analytics and plot generating

A.2.1 Song popularity prediction and plots

```
##
## Thomas Astad Sve
##
##

## Create new environment, classification
cls <- new.env()

##
## Load dataset
##
## Predict popularity
cls$dt <- read.csv("../datasets/msd_pop.csv")

## Build the training/validate/test datasets.
## Split into 70/15/15 train/test/val
val <- 0.15
test <- 0.15
train <- 0.7

cls$nrow <- nrow(cls$dt)
cat(paste("Number of rows: ", cls$nrow, "\n"))

cls$sample <- cls$train <- sample(cls$nrow, train*cls$nrow)
cls$validate <- sample(setdiff(seq_len(cls$nrow), cls$train), val*cls$nrow)
cls$test <- setdiff(setdiff(seq_len(cls$nrow), cls$train), test*cls$nrow)

## The following variable selections have been noted.

cls$input <- c("loudness", "tempo", "time_signature", "key", "mode", "duration")
cls$numeric <- c("loudness", "tempo", "time_signature", "key", "mode", "duration")

cls$target  <- "popular"
cls$ident   <- "track_id"
cls$ignore  <- c("artist_name", "title", "energy", "danceability", "song_hotttnesss")

##
## Decision Tree
##

library(rpart, quietly=TRUE)
print("Classifying using decision tree...")
cls$dtfit <- rpart(popular ~ .,
                   data=cls$dt[cls$train, c(cls$input, cls$target)],
                   method="class")

print("Finished classifying using Decision Tree")

##
## SVM
##

library(e1071, quietly=TRUE)
print("Classifying using SVM...")
cls$svmfit <- svm(popular ~ .,
                  data=cls$dt[cls$train, c(cls$input, cls$target)])

print("Finished classifying using SVM")

##
## Random Forest
##

library(randomForest, quietly=TRUE)
print("Classifying using Random Forest...")
cls$rffit <- randomForest::randomForest(popular ~ .,
                                        data=cls$dt[cls$train, c(cls$input, cls$target)],
                                        ntree=500,
                                        mtry=2,
                                        importance=TRUE,
                                        na.action=randomForest::na.roughfix,
                                        replace=FALSE)

print("Finished classifying using Random Forest")

##
## Evaluate results
##

library(ggplot2, quietly=TRUE)
library(plyr, quietly=TRUE)

print("Obtain the response from the classifyers...")
cls$dtpr <- predict(cls$dtfit, newdata=na.omit(cls$dt[cls$test, c(cls$input, cls$target)]))
cls$svmpr <- predict(cls$svmfit, newdata=na.omit(cls$dt[cls$test, c(cls$input, cls$target)]))
cls$rfpr <- predict(cls$rffit, newdata=na.omit(cls$dt[cls$test, c(cls$input, cls$target)]))
```

```
## plot tree
cat(paste("Saving a plot of the decision Tree \n"))
jpeg('pop_dt_plot.jpg')
plot(cls$dtfit, uniform=TRUE,
     main="Classification Tree for popularity")
text(cls$dtfit, use.n=TRUE, all=TRUE, cex=.8)
dev.off()

## Plot count of popularity distribution

## Plot count of genre distribution
print("Creating a popularity distribution")
jpeg('popularity_distribution.jpg')
g1<-ggplot(cls$dt, aes(x=factor(1), fill=factor(popular))) + geom_bar(width = 1)
plot(g1 + coord_polar(theta="y"))
dev.off()
print("Saved popular distribution plot")
```

A.2.2 Song year prediction and plots

```
##
## Thomas Astad Sve
##
##
## Create new environment, classification
cls <- new.env()

## Predict year
cls$dt <- read.csv(file = "../datasets/msd_year.csv", header = TRUE)

## Split into train, validate and test set
val <- 0.15
test <- 0.15
train <- 0.7
cls$nrow <- nrow(cls$dt)
print(paste("Number of rows: ", cls$nrow, "\n"))

cls$sample <- cls$train <- sample(cls$nrow, train*cls$nrow)
cls$validate <- sample(setdiff(seq_len(cls$nrow), cls$train), val*cls$nrow)
cls$test <- setdiff(setdiff(seq_len(cls$nrow), cls$train), test*cls$nrow)

## Choose variables
cls$input <- c("loudness", "tempo", "time_signature", "key", "mode", "duration")
cls$numeric <- c("loudness", "tempo", "time_signature", "key", "mode", "duration")
cls$target <- "decade"
cls$ident   <- "track_id"

##
## Statistical distribution of the data set
##
library(ggplot2)
library(plyr)

print("Creating a year distribution pie")
jpeg('year_distribution.jpg')
g1<-ggplot(cls$dt, aes(x=factor(1), fill=factor(decade))) + geom_bar(width = 1)
plot(g1 + coord_polar(theta="y"))
dev.off()
print("Saved year distribution plot")

## Plot correlation between loudness and year
print("Creating a correlation plot between loudness and year")
jpeg('loudness_year.jpg')
g<-ggplot(cls$dt, aes(x=decade, y=loudness))
plot(g+geom_violin(alpha=0.5, color="gray")+geom_jitter(alpha=0.5, aes(color=decade),
                                    position = position_jitter(width = 0.1))+coord_flip())

dev.off()
print("Saved loudness and year correlation")

## Plot correlation between duration and year
print("Creating a correlation plot between duration and year")
jpeg('duration_year.jpg')
g<-ggplot(cls$dt, aes(x=decade, y=duration))

plot(g+geom_violin(alpha=0.5, color="gray")+geom_jitter(alpha=0.5, aes(color=decade),
                                    position = position_jitter(width = 0.1))+coord_flip())
dev.off()
print("Saved duration and year correlation")


##
## Do predictive analysis
##

library(rpart, quietly=TRUE)
## Build the Decision Tree model.
print("Classifying a decision tree")
cls$dtfit <- rpart(decade ~ .,
```

```
    data=cls$dt[cls$train, c(cls$input, cls$target)],
    method="class")


##
## SVM
##

library(e1071, quietly=TRUE)
print("Classifying using SVM...")
cls$svmfit <- svm(decade ~ .,
                  data=cls$dt[cls$train, c(cls$input, cls$target)])
print("Finished classifying using SVM")

##
## Random Forest
##

library(randomForest, quietly=TRUE)
print("Classifying using Random Forest...")
cls$rffit <- randomForest::randomForest(decade ~ .,
                                        data=cls$dt[cls$train, c(cls$input, cls$target)],
                                        ntree=500,
                                        mtry=2,
                                        importance=TRUE,
                                        na.action=randomForest::na.roughfix,
                                        replace=FALSE)

print("Finished classifying using Random Forest")

##
## Evaluate the results
##

print("Obtain the response from the classifyers...")
cls$dtpr <- predict(cls$dtfit, newdata=na.omit(cls$dt[cls$test, c(cls$input, cls$target)]))
cls$svmpr <- predict(cls$svmfit, newdata=na.omit(cls$dt[cls$test, c(cls$input, cls$target)]))
cls$rfpr <- predict(cls$rffit, newdata=na.omit(cls$dt[cls$test, c(cls$input, cls$target)]))


## Plot prediction and observation line on test-set
obs <- subset(cls$dt[cls$test, c(cls$input, cls$target)], select=cls$target)
```

A.2.3 Song genre prediction and plots

```
##
## Thomas Astad Sve
##
##

## Create new environment, classification
cls <- new.env()

##
## Load dataset
##

## Predict popularity
#cls$dt <- read.csv("msd_mean_pop.csv")

## Predict genre
cls$dt <- read.table(file = "../datasets/msd_genre_dataset.txt", header = TRUE, sep = ",", comment = "#", quote=NULL, fill=TRUE)

## Build the training/validate/test datasets.
## Split into 70/15/15 train/test/val
val <- 0.15
test <- 0.15
train <- 0.7

cls$nrow <- nrow(cls$dt)
cat(paste("Number of rows: ", cls$nrow, "\n"))

cls$sample <- cls$train <- sample(cls$nrow, train*cls$nrow)
cls$validate <- sample(setdiff(seq_len(cls$nrow), cls$train), val*cls$nrow)
cls$test <- setdiff(setdiff(seq_len(cls$nrow), cls$train), test*cls$nrow)

## The following variable selections have been noted.

cls$input <- c("loudness", "tempo", "time_signature", "key", "mode", "duration")
cls$numeric <- c("loudness", "tempo", "time_signature", "key", "mode", "duration")

cls$target <- "genre"
cls$ident  <- "track_id"

pcme <- function(actual, cl)
{
  x <- table(actual, cl)
  nc <- nrow(x) # Number of classes.
  nv <- length(actual) - sum(is.na(actual) | is.na(cl)) # Number of values.
  tbl <- cbind(x/nv,
```

```
                Error=sapply(1:nc,
                  function(r) round(sum(x[r,-r])/sum(x[r,]), 2)))
  names(attr(tbl, "dimnames")) <- c("Actual", "Predicted")
  return(tbl)
}

library(ggplot2, quietly=TRUE)
library(plyr, quietly=TRUE)

##
## Decision Tree
##
library(rpart, quietly=TRUE)
print("Classifying using decision tree...")
cls$dtfit <- rpart(genre ~ .,
                   data=cls$dt[cls$train, c(cls$input, cls$target)],
                   method="class")

print("Finished classifying using Decision Tree")

# Evaluate Result for Decision Tree

cls$dtpr <- predict(cls$dtfit, newdata=cls$dt[cls$test, c(cls$input, cls$target)], type="class")

print(length(cls$dt))
print(length(cls$test))
print(length(cls$dt[cls$test, c(cls$input, cls$target)]))
print(length(cls$dtpr))

perdt <- pcme(cls$dt[cls$test, c(cls$input, cls$target)]$genre, cls$dtpr)
round(perdt, 2)
cat(100*round(1-sum(diag(perdt), na.rm=TRUE), 2))

errormatdt <- table(cls$dt[cls$test, c(cls$input, cls$target)]$genre, cls$dtpr,
                    useNA="ifany",
                    dnn=c("Actual", "Predicted"))
write.table(errormatdt, file = "genre_error_matrix_dt.txt")

##
## Random Forest
##

library(randomForest, quietly=TRUE)
print("Classifying using Random Forest...")
cls$rffit <- randomForest::randomForest(genre ~ .,
                                        data=cls$dt[cls$train, c(cls$input, cls$target)],
                                        ntree=500,
                                        mtry=2,
                                        importance=TRUE,
                                        na.action=randomForest::na.roughfix,
                                        replace=FALSE)

print("Finished classifying using Random Forest")

## Evaluate results for random Forest
cls$rfpr <- predict(cls$rffit, newdata=cls$dt[cls$test, c(cls$input, cls$target)], type="class")


## Plot Error Matrix
errormatrf <- table(cls$dt[cls$test, c(cls$input, cls$target)]$genre, cls$rfpr,
                    useNA="ifany",
                    dnn=c("Actual", "Predicted"))

write.table(errormatrf, file = "genre_error_matrix_rf.txt")
perrf <- pcme(cls$dt[cls$test, c(cls$input, cls$target)]$genre, cls$rfpr)
round(perdt, 2)
cat(100*round(1-sum(diag(perrf), na.rm=TRUE), 2))

##
## SVM
##

library(e1071, quietly=TRUE)
print("Classifying using SVM...")
cls$svmfit <- svm(genre ~ .,
                  data=cls$dt[cls$train, c(cls$input, cls$target)])

print("Finished classifying using SVM")

## Evaluate Results for SVM
cls$svmpr <- predict(cls$svmfit, newdata=cls$dt[cls$test, c(cls$input, cls$target)], type="class")
errormatsvm <- table(cls$dt[cls$test, c(cls$input, cls$target)]$genre, cls$svmpr,
                     useNA="ifany",
                     dnn=c("Actual", "Predicted"))
write.table(errormatsvm, file = "genre_error_matrix_svm.txt")
persvm <- pcme(cls$dt[cls$test, c(cls$input, cls$target)]$genre, cls$svmpr)
round(persvm, 2)
cat(100*round(1-sum(diag(persvm), na.rm=TRUE), 2))

##
## Make plots
##
```

```
## Plot tree
print("Saving a plot of the decision Tree...")
jpeg('genre_dt_plot.jpg')
plot(cls$dtfit, uniform=TRUE,
     main="Classification Tree for genre")
text(cls$dtfit, use.n=TRUE, all=TRUE, cex=.8)
dev.off()

## Generate a Confusion Matrix
print("Saving a confusion Matrix of the decision Tree...")
jpeg('genre_dt_confusion_matrix.jpg')
plot(cls$dtfit, uniform=TRUE,
     main="Classification Tree for genre")
text(cls$dtfit, use.n=TRUE, all=TRUE, cex=.8)
dev.off()

## Plot count of genre distribution
print("Creating a genre distribution pie")
jpeg('genre_distribution.jpg')
g1<-ggplot(cls$dt, aes(x=factor(1), fill=factor(genre))) + geom_bar(width = 1)
plot(g1 + coord_polar(theta="y"))
dev.off()
print("Saved year distribution plot")


## Plot relation between loudness and genre
print("Plot showing relation between loudness and genre")
jpeg('genre_loudness.jpg')
g<-ggplot(cls$dt, aes(x=genre, y=loudness))
plot(g+geom_violin(alpha=0.5, color="gray")+geom_jitter(alpha=0.5, aes(color=genre),
                                    position = position_jitter(width = 0.1))+coord_flip())
dev.off()
print("Saved loudness and genre relation")
```