Predicting Hit Songs

Jakob Brandt & Emil Lanzén

Autumn 2021

Abstract

In this project, three classification models, the Logistic regression, the support vector machine and the Random forest classifier, are compared in terms of their ability to predict whether a song is a hit or not. The comparison is done on a data set that consists of roughly equally many hit songs and non-hit songs. The evaluation of the models is conducted using three evaluation metrics; accuracy, sensitivity and specificity. The results show that the models perform somewhat similar, except for how well the models predict the non-hit songs. The logistic regression performs substantially worse in this regard compared the machine learning models. However, it is difficult to determine to what extent these results generalize in terms of predicting observations that are not contained in the data set.

Contents

1 Introduction

The change in digital music availability and the technological advancements have greatly impacted the way people listen to music. However, the expectations and demand on technology have changed along with it. People today expect to be able to create their own playlists, search for music belonging to a specific genre, but also to receive new recommendations and custom playlists based on the music they listen to. This is all made possible by the advancements in *Music Information Retrieval* and the implementation of *Music Recommender Systems* (Kaminskas, Ricci, 2012).

As a result of the technological advancements the question regarding what makes some songs more successful than others on the global music scene naturally arises. While one could argue that its meaningless to speak about success in arts, we simply mean popular songs that reach some sort of top charting position. Furthermore, while difficult to address this question in a meaningful manner, the answer to the question would arguably make industry actors quite wealthy. One could argue that for actors involved in the global music industry, this qualifies as a million dollar question.

Enter *Hit Song Science*, one active research topic within Music Information Retrieval whose objective is to predict the success of songs, measured by the chart-position on various top lists. The ability to predict a song's success is beneficial for the musicians and labels in terms of a potential increase in revenue and to more easily reach out to a larger audience (Middlebrook Sheik, 2019). Furthermore, it is important to identify the underlying cause of success in the industry where the inequality has increased and number of songs reaching the top lists have decreased (Kim, Oh, 2021).

Although challenging to effectively predict the success of songs, as Salganik et al. (2006) shows, the technological advancements in machine learning and Music Information Retrieval, such as information on acoustic attributes, can be used in order to predict the success of songs more accurately (Kim, Oh, 2021). Also, while industry actors certainly mobilize a mass of computational power and predictive models, a question that arises is how some standard learning models with the help of some public, larger set of data could accurately predict the success of songs. While we doubt that our models have the same edge compared to those of industry actors with their access to a presumably wide range of variables that account for many things outside the sheer quality of the songs (e.g. marketing related variables), we would like to see how good the presumably top quality song-related variables in conjunction with some well-known learning models are at correctly classifying hit songs. Thus, the purpose of this project is to compare three classification models in their ability to predict the success of songs exclusively using variables related to acoustic attributes of the songs. The research question is as follows:

Are there any differences between the Logistic Regression, the Support Vector Machine and the Random Forest Classifier in their ability to predict the success of songs using variables related to musical properties of songs?

The rest of the report is organized as follows. Section 2 introduces and explains the data used in the project. Section 3 gives a theoretical background of the methods used throughout the project. Section 4 describes the package used in R for facilitating the analysis. Section 5 presents the results, and lastly Section 6 discusses the findings.

2 Data

The data set originates from kaggle.com and consists of 34740 songs. Around half the songs, 17425, are hit songs from any of the Billboard, Shazam or Spotify top 100 charting lists. The remaining 17315 songs are non-hit songs from Spotify. There are no missing values. While the original data set consists of 17 variables, 7 of these were removed. The reason for this is as stated that we intend to limit our focus and only look at how acoustic attributes, from here on referred to as *acoustic variables*, help explain why some songs become hits. Acoustic variables indicate the level of some musical property of a song. For example, the variable *danceability* in our data set indicates how dance friendly a song is. Since we deliberately limit the set of allowed variables it is true that some of the removed variables would presumably contribute to predictive power. However, a great deal of the removed variables are nonsensical in terms of adding predictive power (e.g. variables indicating song id, etc.). The variables and the binary response variable *On Chart* can be seen in Table 1.

Table 1: Variables and intervals

On Chart	*Energy*	*Acousticness*	*Danceability*	*Instrumentalness*
0 *or* 1	[0, 1]	[0, 1]	[0, 1]	[0, 1]
Liveness	*Loudness*	*Speechiness*	*Valence*	*Tempo*
[0, 1]	[-60, 3]	[0, 1]	[0, 1]	[0, 250]

Apart from the names of the variables, Table 1 shows the intervals of the variable values. Most of the values are in intervals between 0 and 1 apart from *Loudness* and *Tempo*. The intervals are based on the maximum and minimum values of the variables, so note that the intervals presented are empirical and not theoretical. According to the description of the data set on kaggle.com, all variables except the aforementioned variables range between 0 and 1. However, the variables *Loudness* and *Tempo* are according to the description float variables that typically range between -60 and 0 and between 50 and 150. The combination of the facts that these variables typically but not necessarily range between these values with that some variables deviate from the range extremely (especially true for *Tempo*) makes it hard to conclude whether some values are incorrect or not. Furthermore, we cannot generally tell how accurate the variable values are since we lack knowledge of the methods used to calculate them. In other words, we do not know how much the values deviate from their true value, if they ever do that.

Looking at Table 2, we can see the correlations between the explanatory variables and the response variables. The Point-biserial correlation coefficient is used instead of the Pearson correlation coefficient since the response variable is categorical and not continuous. The interpretation is still that a 1 or a -1 represents a perfect positive and a perfect negative correlation, respectively.

Table 2: Correlations between response variable and explanatory variables

Energy	*Acousticness*	*Danceability*	*Instrumentalness*	*Liveness*
0.2272	-0.2142	0.0977	-0.4638	-0.0357
Loudness	*Speechiness*	*Valence*	*Tempo*	
0.3021	-0.1830	0.2757	0.0417	

From Table 2 we can see that there are some very low levels of correlations for some variables, while for other variables the correlations are somewhat stronger. However, all correlation coefficients are within ± 0.5, indicating that the relationships in the data are not very strong. Apart from the numerical aids in determining the relationships in the data, Figure 1 shows the box plots of the variables colorized by category. As seen in the figure there are many cases of severe overlapping between the classes. This raises the possibility that the classification models are not going to be able to classify the observations well. However, there are also some cases of non-overlapping for some variables, which together with the large number of observations complicates making qualified guesses using the plot alone, since the models may be able to sufficiently recognize patterns in the data.

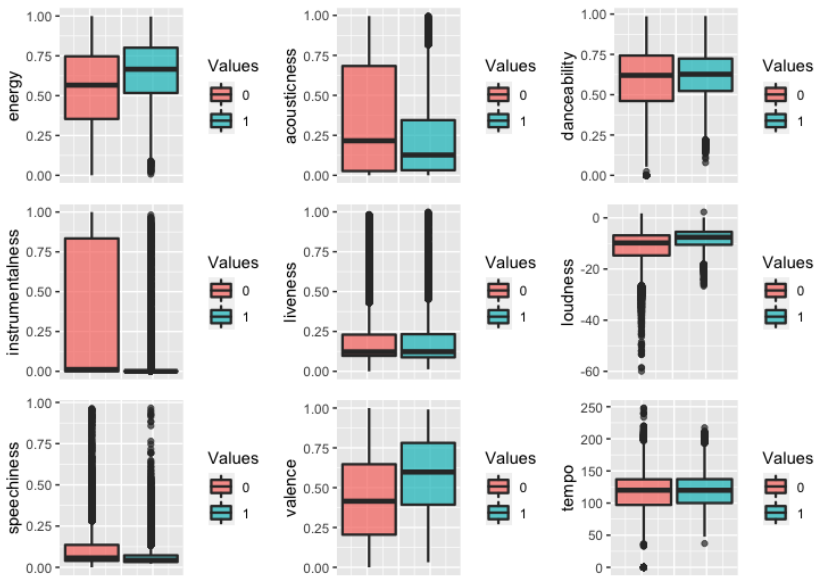

Figure 1: Boxplots of the explanatory variables. Red boxplots indicates observations that are non-hit song class and vice versa for the turquoise boxplots.

The data set is unique as it combines hit songs and non-hit songs. One of the strengths of the data set is that it matches the number of hit songs to the number of non-hit songs, which relieves us from the problem of imbalanced classes (more on this later). This relates to that fact that it may facilitate the ability of the models to correctly classify the non-hit songs from the hit songs, as if the data set reflected a pure random sampling of songs, the number of hit songs would be very small. However, a weakness is that we do not know what sampling method was used for selecting all songs. For example, the non-hit songs all come from Spotify, but there is no detailed explanation of how these were sampled.

3 Methods

In this section the statistical methods and analyses are presented. This includes the learning models used, the split of data into training and test set for model evaluation and also a description of the evaluation metrics used.

3.1 Learning Models

In this subsection the theory behind the learning models is presented.

3.1.1 Logistic regression

In addition to the machine learning models used in this research, the Logistic regression is used both in terms of model evaluation and comparison to the other models, but also in order to make inference and interpretation of relationships easier. Hair et al. (2010) states that the reason why Logistic regression is often preferred by many researchers within the domain of classification is because of its simplicity and interpretability. This also holds for this project where the Logistic regression is included as a benchmark to the other models as well as for doing inference. Multiple logistic regression is the corresponding model when there are more than one explanatory variable, and is given by the equation where

$$P(Y_i|X_i, .., X_p) = \frac{e^{\beta_0+\beta_1 X_1+...+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+...+\beta_p X_p}}. \tag{1}$$

Here $X = (X_1 + ... + X_p)$ are the explanatory variables used in to model to predict the response variable Y. The formula can equivalently be expressed as

$$\log(\frac{p(Y_i|X_i, ..., X_p)}{1-p(Y_i|X_i, ..., X_p)} = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p. \tag{2}$$

The left-hand side of equation 2 is the *logit function* of p, which is linear in the explanatory variables. This means that an estimated multiple logistic regression model can be used to predict the probability of the class belonging of a given observation. The task is to determine the probability threshold for when a given observation should be classified as positive or negative, in order to then correctly classify new observations which the model has not been trained on. The *test error rate* is the number of incorrect classifications of new observations. James et al. (2013) states that the test error rate can be reduced using the *Bayes classifier*. It assigns a given observation to the class which it most likely belongs to. An observation with explanatory vectors $x_1, ..., x_p$ should be assigned to class j for which

$$p(Y = j|X_1 = x_1, ..., X_p = x_p) \tag{3}$$

is the largest. In the case where the response variables is binary, a new observation should be assigned to class 1 if

$$p(Y = 1|X_1 = x_1, ..., X_p = x_p) > 0.5 \tag{4}$$

and to class 0 otherwise.

3.1.2 Support vector machines

Since the introduction in the 1990s, the *Support vector machines* (SVM) have proven to perform well in a large variety of areas. The SVM consist of a set of classifiers and are derived from the maximal margin classifier which classifies using a separating hyperplane (James et al., 2013). If the feature space is 2-dimensional then the hyperplane is 1-dimensional, which is a straight line. The side of the hyperplane for which an observation $x = (x_1, ..., x_p)$ lie is dictated by if

$$\beta_0 + \sum_{i=1}^{p} \beta_i x_i > 0, \quad \text{or} \quad \beta_0 + \sum_{i=1}^{p} \beta_i x_i < 0. \tag{5}$$

If the hyperplane correctly classifies the observations, then it can be used as a classifier. In the case where a separated hyperplane can in fact be used to classify a data set, then there are an infinite number of such hyperplanes. The maximal margin classifier selects the hyperplane that lies the farthest from the training observations. There is a subset of training observations, called *support* *vectors*, that lie on the margin of the hyperplane and therefore affect its shape.

However, the separating hyperplane needs to exist in order for the maximal margin classifier to work. But this is not always the case, since classes often cannot be separated by linear class boundaries. In order to cope with this problem one can use a *soft margin* which correctly classifies most of the observations, and is known as the *support vector classifier* which is a generalization of the maximal margin classifier for the non-separable case. The support vector classifier is more robust to the training observations and can correctly classify training observations more easily. It classifies based on which side of the hyperplane the observations lie, so that most of the observations are correctly classified. The support vector classifier aims to solve the problem

$$\begin{gathered}\max_{\beta_0,\beta_1,\ldots,\beta_p,\ \epsilon_1,\ldots,\epsilon_n,\ M} M \\ \text{subject to} \sum_{\mathrm{j}=1}^{\mathrm{p}} \beta_{\mathrm{j}}^2 = 1, \\ \mathrm{y_i}(\beta_0 + \beta_1 \mathrm{x_{i1}} + \beta_2 \mathrm{x_{i2}} + \ldots + \beta_{\mathrm{p}} \mathrm{x_{ip}}) \geq \mathrm{M}(1 - \epsilon_{\mathrm{i}}), \\ \epsilon_{\mathrm{i}} \geq 0, \ \sum_{\mathrm{i}=1}^{\mathrm{n}} \epsilon_{\mathrm{i}} \leq \mathrm{C},\end{gathered} \tag{6}$$

where M is the width of the margin, the quantity we seek to maximize. C is a non-negative parameter that regulates to the extent of which the hyperplane misclassifies observations and the magnitude of the margins. C is chosen using cross-validation and adjusts the balance between the model's variance and bias. An increase in the value of C reduces the risk of model overfitting and variance, at the cost of added bias.

The support vector machines is an enhancement of the support vector classifier in the way that it uses *kernals* to accommodate non-linear boundaries, by expanding the feature space. The SVM originates from the optimization problem in Equation 6, and the solution is proven to only incorporate the *inner products* of the observations, and is given by

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}. \tag{7}$$

The linear support vector classifier can be expressed as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle, \tag{8}$$

where α_i are parameters, one for every observation. The computation of the inner product of every new point x and each of the existing training points x_i is required in order to estimate the model and its parameters. However, the parameters are nonzero only for the support vectors. The kernel function has the purpose of measuring the similarity between two observations. It does this by replacing every inner product in Equation 8 with a generalization. One commonly used kernel is the radial kernel

$$K(x_i, x_{i'}) = exp\left(-\gamma + \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right), \tag{9}$$

where γ is a positive constant selected by cross-validation.

3.1.3 Random forest

The *Random forest classifier* originates from classification trees, which are basically decision trees used to predict qualitative responses. James et al. (2013) explains that decision trees are constructed by dividing the predictor space $X_1, ..., X_p$ into J non-intersecting regions, $R_1, ..., R_j$. The splitting of the predictor space is done by so-called *recursive binary splitting.* The principle is that it splits the predictor space repeatedly based on the largest reduction of the classification error. It divides the predictor space according to

$$\{X | X_j < s\} \text{ and } \{X | X_j \geq s\}, \tag{10}$$

where X_j is the predictor space and s is the cutpoint. In the classification setting one commonly used measure for splitting is the *Gini index*

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}), \tag{11}$$

where $\hat{p}_{mk}$ represents the proportion of training observations that belong to the kth class in the mth region.

Using decision trees in the domain of classification is expected to result in model overfitting. One way to cope with this problem is to use a method that incorporates bootstrapping, called *bagging*. It generates bootstrap samples from the training set, and the model is then trained on every replicate in order to obtain B classification models, which are then averaged as follows

$$\hat{f}_{bag}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x). \tag{12}$$

Averaging the B trees results in lower variance compared to the variance of the individual trees. Predicting an observation is simply done by keeping track of the class prediction by the B trees and choose according to the most frequently occurring class prediction.

Random forest is an enhanced version of bagged trees with the purpose of reducing the variance of the model. Random forest creates less correlated models by only taking a subset of the predictors into account at each split. This reduces the variance, since highly correlated models do not lead to a substantial variance reduction even when averaging the classification models. At each split m predictors are randomly chosen, where the number of chosen predictors usually equal the square root of the total number of predictors in the data set, such that

$$m \approx \sqrt{p}. \tag{13}$$

This reduces the correlation among the trees considerably at the cost of a minor increase in bias.

3.2 Training and test sets

The observations in the data set were randomly split into a training and test set. This was done so that 70 percent of the observations were included in the training set, and the remaining 30 percent were included in the test set. This is equivalent to approximately 24 300 observations in the training set and 10 400 observations in the test set. The reason for partitioning the data into two subsets is to better capture the true performance of the models. In order to get the out-of-sample error, which is the error when classifying new observations, the models have to be evaluated on observations which the models have not been trained on. Furthermore, there was no a priori reason behind the choice of the 70 percent - 30 percent split other than it seemed reasonable and that prior experiments using this splitting rule has yielded good results.

3.3 Model selection

The radial kernel was selected for the SVM. 5-fold cross-validation was implemented in order to determine the value of the hyperparameters in the model. This is done by testing the model with different hyperparameters and calculating the error. The 5-fold cross-validation splits the training set into 5 parts, four of which are training sets and the remaining one is used for calculating the error. This procedure is then repeated for the different subsets, where one of the subsets is used to calculate the in-sample error and the rest are used for training the model. The reason for the choice of 5 folds is that while accurate values of the hyperparameters are desired, the computational time required for LOOCV seemed unreasonable. 5-fold and 10-fold cross-validation also seems to perform very similar to LOOCV (James et al., 2013), and we thus chose 5 folds since the method takes the least amount of time compared to the aforementioned choices.

The weighted in-sample error of the 5 in-sample errors are a good estimate of the out-of-sample error. The SVM has two hyperparameters that cannot be determined by minimizing the in-sample error, therefore the cross-validation was done for the different values of the parameters. This results in several values of the hyperparameters, where the ones that result in the lowest cross-validation error is selected for the final model. The procedure is illustrated in Table 3.

Table 3: 5-fold cross-validation

Train	Train	Train	Train	Validation
Train	Train	Train	Validation	Train
Train	Train	Validation	Train	Train
Train	Validation	Train	Train	Train
Validation	Train	Train	Train	Train

The Random forest has two hyperparameters; the number of trees and the number of explanatory variables included in each split. The number of trees were set to 500, and the number of variables included at each split was varied using cross-validation in order to calculate the classification error for each number of the included variables. However, Oshiro et al. (2012) mentions the fact that the literature does not provide information or directions about how many trees should be chosen. It is also stated that there are uncertainties regarding whether there is an optimal number of trees and that additional trees would only result in an increase in the computational time required.

3.4 Evaluation metrics

In this subsection, a description of the evaluation metrics is presented. The chosen evaluation metrics are commonly used when evaluating the performance of a classification model.

3.4.1 Confusion matrix

Confusion matrices are suitable when presenting and evaluating the performance of a classification model. In the case where the response variable is binary the confusion matrix is a 2x2 table that contains the output of a classifier in terms of correct and incorrect classifications. On the y-axis lies the predicted values and on the x-axis lies the actual values. An observation that the model predicts correctly as being positive is called *True positive*, whist an observation that the model predicts as being positive when it in fact is negative is called *False positive*. It is called *True negative* when the model correctly predicts that an observation is negative, and *False negative* when the observation is positive.

Table 4: Confusion matrix

		Actual values	
		0	1
Predicted values	0	*True negative*	*False negative*
	1	*False positive*	*True positive*

3.4.2 Accuracy

One of the most commonly used evaluation metrics is accuracy. It simply calculates the ratio between the number of correct classifications and the total number of classifications.

3.4.3 Sensitivity and specificity

Sensitivity and specificity serve the purpose of measuring how classification models predict positive observations in relation to negative observations. Suppose that two classifications models perform equally in terms of accuracy, this does not necessarily mean that the models classify positive and negative observations equally. One of the models might tend to classify positive observations as being negative, whilst the other model might be biased in the sense that it classifies negative observations as being positive. This can therefore be determined by evaluating the models using sensitivity and specificity. Sensitivity is a measure of how well positive observation are correctly classified

$$Sensitivity = \frac{True\ positives}{True\ positives + False\ negatives}, \tag{14}$$

whilst specificity measures the degree of which negative observations are correctly classified

$$Specificity = \frac{True\ negatives}{True\ negatives + False\ positives}. \tag{15}$$

4 Implementation

The package used in R (R Core Team, 2020) is called *Caret* (Kuhn, Max, 2021), which is short for Classification and Regression Training. It aims at greatly facilitating the processes involved in machine learning, both in classification and regression. The package introduces functions for data splitting, pre-processing, feature selection, model tuning and variable importance estimation (Kuhn, Max, 2021), although the tools that are relevant in this project are the data splitting and model tuning functions.

The main benefits of using the Caret package is that it greatly facilitates the process in implementing code for the aforementioned machine learning processes. However, as a part of this report, we would like to test if the results from running a classification model using Caret yields reasonable results using a simulation study. The simulation design is quite simple where we start by simulating 1000 observations from four different probability density functions with a binary response variable. The parameters of the explanatory variables will differ by a small amount according to if the value of the response variable is a 0 or a 1 so that there are some levels of correlations between the response and the explanatory variables. Table 5 shows the variables used in the simulation study together with the parameter values according to the value of the response variable.

Table 5: Simulation variables

y	x_1	x_2	x_3
$\sim Be(p)$	$\sim N(\mu, \sigma^2)$	$\sim pois(\lambda)$	$\sim exp(\lambda)$
$X = 1$	$\sim N(1, 1)$	$\sim pois(17)$	$\sim exp(5)$
$X = 0$	$\sim N(0, 1)$	$\sim pois(15)$	$\sim exp(8)$

If we were to set p to 0.5, given that there are some correlations between the explanatory variables and the response variable, we would expect that the learning model would perform somewhat good in terms of accuracy and sensitivity. It would probably categorize the positives and the negatives correctly in many cases, but also classify some deal of the observations incorrectly. Now, if we instead were to set p to 0.95 without changing any other parameters, we would instead expect the learning model to incorrectly classify a great deal of the observations belonging to the category with 5 % of the observations as if they belonged to the category holding 95 % of the observations. Another way of phrasing this would be that the model classifies most minority class observations as belonging to the majority class, and this problem is often referred to as the class imbalance problem (Krawczyk, 2016). The reason for this is often that there is some extent of overlapping in the data between the classes, and learning models have a somewhat hard time accounting for this.

However, for the sake of our simulation, this allows us to test if the results of our learner gives reasonable results. By comparing metrics related to confusion matrices, we can compare how the models classify observations when we change the imbalance of the classes. By setting p to 0.5 for one data set and 0.95 for another (with the same parameter values for the explanatory variables as seen in Table 5), we can see how, for example the SVM, deals with the two data sets in its effort to classify the observations correctly. We would expect that for the class with a 50-50 distribution in the classes, the accuracy and specificity should be somewhat good but not perfect. However, for the class with the 5-95 distribution, we would expect a very high accuracy (at least higher than 0.9) and a very low specificity (below 0.1). This is conditioned on the fact that, after all, there should exist some correlations in the data between the response and the explanatory variables. Figure 2 displays the densities of the explanatory variables depending on the class as well as the distribution of the classes. In the figure we can also see that there are some correlation between the variables, which also depends on the distribution of the classes.

For the correlations between categorical and continuous variables (X_1 and X_3),

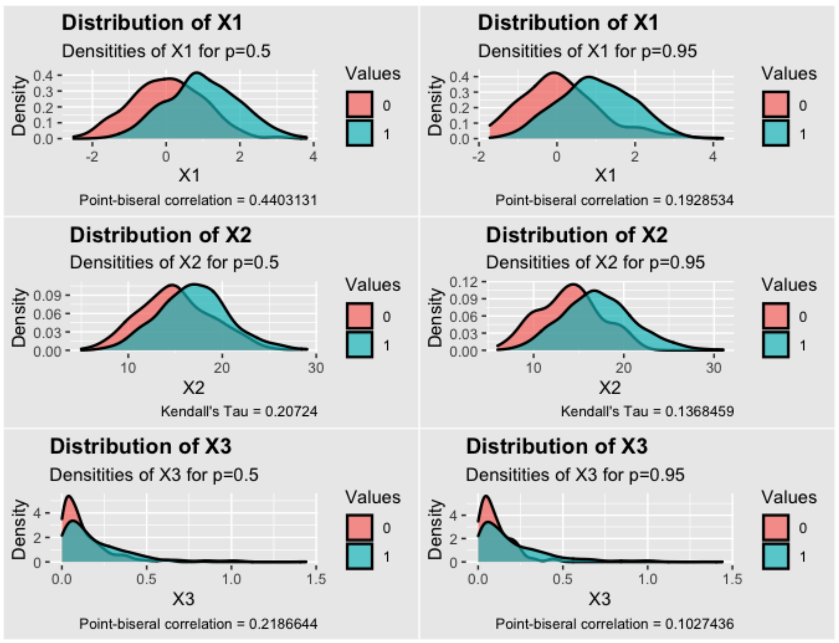

Figure 2: Distributions of explanatory variables

the Point-biserial correlation coefficient was used, and for the correlation between a categorical and a discrete variable (X_2), Kendall's Tau was used. The idea of the simulation is to fit an SVM model to the data sets with the differing proportion of positives and negatives using the procedure of data splitting and 5-fold cross-validation with the help of the Caret package. To account for sampling variability, a total of 2000 data sets were generated with variables and associated parameters corresponding to those of Table 5. In half of the sets, the parameter of the response variable was set to 0.5 and for the remaining it was set to 0.95. An SVM was fitted to every data set and the accuracy and specificity were then calculated as functions of each produced confusion matrix. The average accuracy and specificity from the simulated data sets can be seen in Table 6. As seen in Table 6 (a), we can tell that the SVM for the data where $p = 0.5$ had an average accuracy and specificity that were close to each other, meaning the model had some misclassifications but that on average the model could do more correct classifications than incorrect ones. Also an expected result is the fact that when $p = 0.95$, as seen in Table 6 (b), the accuracy was very high while the specificity was ridiculously low, meaning a very high number of observations belonging to the minority class were classified as if they belonged to the majority class.

Table 6: Simulation output

(a) p = 0.5

	SVM
Accuracy (mean)	0.7177
Sensitivity (mean)	0.7186

(b) p = 0.95

	SVM
Accuracy (mean)	0.9513
Specificity (mean)	0.0291

Figure 3, which shows the means and 95 % confidence intervals of the metrics from each simulated data set, tells a similar story to that of Table 6. In short, our expectations of the outcome of the simulation study were met by the output. The results seem reasonable and conditioned on this it seems like the Caret package gives reasonable results.

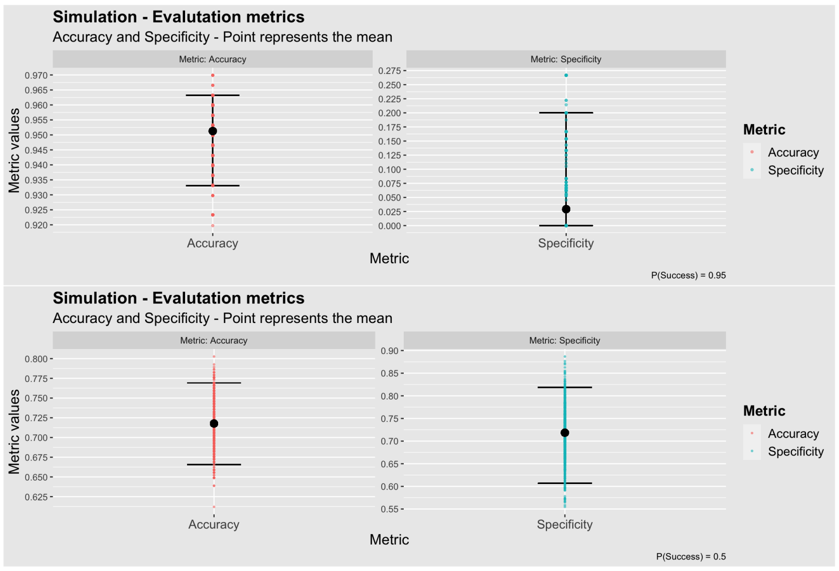

Figure 3: Simulation - 95 percent confidence intervals of the evaluation metrics

5 Results

The accuracy, sensitivity and specificity of the classification models are presented in Table 7. Comparing the accuracy of the three models, the results are somewhat similar, especially for the SVM and Random Forest. The logistic regression performs worse than the other two in terms of accuracy, whilst Random Forest just outperforms the SVM. In terms of sensitivity, the models are very similar in their performance, with support vector machines barely outperforming the other two models. In terms of specificity, the differences are the greatest. The logistic regression performs substantially worse than the SVM and Random forest, with Random forest having the highest specificity. Although the logistic regression performed worse than the SVM and the random forest in most cases, the results show that for this particular data set the machine learning models are not that superior to the logistic regression, since the marginal differences are not that large.

Table 7: Evaluation metrics of the models

	Logistic regression	SVM	Random forest
Accuracy	0.7538	0.7911	0.8016
Sensitivity	0.8445	0.8659	0.8439
Specificity	0.6625	0.7158	0.7590

As seen in Table 8 the parameter estimates from the Logistic regression are all significant on (at least a 5 percent significance level) different levels except for the *Energy* variable. The implication is that the energy-level of a song does not seem to determine whether a song is a hit or not. Furthermore, our results suggest that *Instrumentalness* and *Speechiness* have the largest effect on the logarithm of the odds of the song being a hit or not. With regards to the significance of the parameters, instead of exposure to the multiple testing

Table 8: Parameter estimation using logistic regression

Intercept	*Energy*	*Acousticness*	*Danceability*	*Instrumentalness*
1.8204***	-0.2134	-1.2578***	-1.3091***	-4.3639***
Liveness	*Loudness*	*Speechiness*	*Valence*	*Tempo*
$-0.1830^{\cdot}$	0.0395***	-5.4910***	2.1020***	-0.0012*
Signif. codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'

problem by testing each individual parameter estimate with the Wald test, we conducted a joint Likelihood Ratio-test (LR-test). Looking at Table 9, the p-value from the LR-test indicates a very low probability of a Type I error. Thus, its safe to say that we can conclude that at least one of the explanatory variables influences the odds of a song being a hit.

Table 9: LR-test - logistic regression

LR-test
$Pr(> \chi^2) < 0.0001$

6 Conclusion

The results suggest that there are some differences of the models in their ability to predict the success of songs using only acoustic variables. However, in many cases the metrics stemming from the values of the confusion matrices were minor in their differences. Although it's hard to tell exactly how well the models would generalize in their ability to classify new observations outside of this particular data set, the deployment of training sets and evaluation using the test sets tells us that the models may do quite a good job at this.

Although some confidence in the ability of our models could be derived from the results, something that speaks against this is the fact that the data set with a very high probability does not reflect the actual population of songs. We simply base this on the fact that most produced songs do not become hit songs, and in our data set half the songs are hits. We can however speculate that the models may be better fitted for classification using this kind of data set, since if the data set would be reflective of the actual population of songs, we would have to deal with the problem of imbalanced classes. As a result, which is also true for data where the correlations are not perfect, the models could have had a very hard time distinguishing the positives from the negatives. Another drawback of the project stems from the fact that we lack knowledge of the sampling method for the non hit songs, as touched upon earlier in the data Section.

As stated during the exploratory data analysis, there were cases of severe overlapping in the data. However, mainly because of the large number of observations, we thought that we might be able to see some significant parameter estimates from the Logistic regression model. Judging from the results, the large number of observations seems to have had an effect on this. Many of the estimates were significant, but since the values in many cases are so small, it begs the question how important these really are in predicting the success of songs.

Apart from these eventual drawbacks, the results of the confusion matrices suggest that we generally may be able to predict the success of songs using statistical methods. The intentional limiting of the data set to only include the acoustic variables certainly influenced the predictive power of the models. However, this begs the question how powerful a predictive model with the use of more explanatory variables accounting for other things than sheer musical properties could be.

Bibliography

James, G; Witten, D; Hastie, T; Tibshirani, R. 2013. *An Introduction to Statistical Learning.* 8th edition. New York: Springer.

Kaminskas, I; Ricci, F. 2012. *Contextual music information retrieval and recommendation: State of the art and challenges.* Computer Science Review 6 (2012) 89-119.

Kim, S.T; Oh, J.H. 2021. *Decision Support Systems.* Decision Support Systems 145 (2021) 113535.

Krawczyk, B. 2016. *Learning from imbalanced data: open challenges and future directions.* Progress in Artificial Intelligence 5(4). 221-232.

Kuhn, M. 2021. Caret: Classification and Regression Training. R package version 6.0-88. https://CRAN.R-project.org/package=caret.

Middlebrook, K; Sheik, K. 2019. *Song Hit Prediction: Predicting Billboard Hits using Spotify Data..*

Oshiro, T.M; Perez, P.S; Baranauskas, J.A. 2012. *How Many Trees in a Random Forest?.* Machine Learning and Data Mining in Pattern Recognition (2012) Volume 7376 154-168.

R Core Team. 2020. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Salganik, M.J.; Dodds, P.S.; Watts, D.J. 2006. *Experimental Study of Inequality and Unpredictability in an Artificial Cultural Market.* Science 311 (2006) 5762, 854-856.

7 Appendix

Table 10: Confusion matrix - Logistic regression

		Actual values	
		0	1
Predicted values	0	3441	813
	1	1753	4414

Table 11: Confusion matrix - SVM

		Actual values	
		0	1
Predicted values	0	3718	701
	1	1476	4526

Table 12: Confusion matrix - Random forest

		Actual values	
		0	1
Predicted values	0	3942	816
	1	1252	4411