SONG POPULARITY PREDICTION

PREDICT WHICH SONGS WILL MAKE IT TO THE TOP 10

12/06/2017

Group 2: Lane Birkenfeld, Naman Varshney, Pragya Jha, Qing Liu

TABLE OF CONTENTS

INTRODUCTION

Our objective was to predict which acoustic features or rather, which combination of features, makes a song popular. The input for our algorithm is a list of song characteristics. We then use a number of machine learning algorithms to find out the trends of popular songs.

The ability to make accurate predictions of popularity of songs has huge implications for the music industry. It would be interesting to find out the popularity of a song before its release. And although it is true that there are several external factors such as promotions and marketing that can affect the popularity of any song, we are not going to consider them for our project.

Besides providing the ability to predict hit songs, creating an accurate model could provide great insight for the production of songs. Song producers could possibly determine what level of acoustic feature is most popular as well as how important each feature could be to producing a hit song.

RELATED WORK

Hit Song Science, as it is famously known as, has its dedicated Wikipedia page. It concerns the possibility of predicting whether a song will be a hit, prior to its distribution using automated means such as machine learning software. The paper can be found out at this link.

In this project, they have used Decision trees, Naïve Bayes, Logistic Regression and Support Vector Machines with K-fold cross validation. They achieved a maximum accuracy of 81% with logistic regression.

DATA DESCRIPTION

For our project, we wanted to pick an objective that was interesting and achievable. We searched through several different databases looking for a set of data that met several criteria. Chief among those criteria were understandability, parameters of size and predictors, and cleanliness.

Our first concern was understandability. We wanted a data set that related to something we all knew. Our backgrounds were fairly different so the data set was most likely going to have to be something basic. Besides this, we wanted the data to be interesting. To sum it up, we were going to be giving a presentation to the entire class and the professor based on this data, and if we were going to have to struggle greatly to understand or take interest in our data, we couldn't expect our audience to.

Our next concern, parameters of size and predictors, was at the front of our minds. We knew the difference between a doable objective and an unachievable objective could easily be the size of the data we were working with. Besides this, our professor had stressed this point several times when speaking about our projects.

And finally, the cleanliness of the data was a major concern for our group. It would have been easy to spend an inordinate amount of time on the data simply cleaning it. Members of our group had worked in jobs that largely consisted of cleaning data, and we were convinced it would be wise to avoid choosing a set of data that would require a high amount of preparation.

We settled on a data set that seemed to meet all of the standards we had and proceeded with our new objective, to predict whether a song will reach a spot in the top 10 of the Billboard Hot 100 Chart.

Our dataset songs.csv consists of all songs which made it to the Top 10 of the Billboard Hot 100 Chart from 1990-2010 plus a sample of additional songs that didn't make the Top 10. This data comes from three sources: Wikipedia, Billboard.com, and EchoNest. The set contains 39

variables with 7574 records. The ratio of records in the data that had been in the top ten to those that hadn't was 1119 to 6455, or roughly 15 percent of the records were top 10 songs.

The variables included in the dataset either describe the artist or the song, or they are associated with the following song attributes: time signature, loudness, key, pitch, tempo, and timbre.

We were able to understand or mostly understand nearly all of the variables with minimal effort, except timbre. Timbre was broken into 24 different variables starting with "timbre_0_min", "timbre_0_max", and extending to "timbre_11_min", and "timbre_11_max".

The answer to what exactly these values meant was found in the EchoNest Analyzer Documentation [1] which reads as follows: "Timbre is the quality of a musical note or sound that distinguishes different types of musical instruments, or voices. It is a complex notion also referred to as sound color, texture, or tone quality, and is derived from the shape of a segment's spectro-temporal surface, independently of pitch and loudness. The Echo Nest Analyzer's timbre feature is a vector that includes 12 unbounded values roughly centered around 0."

We filtered these values to only include the numerical data we would be using to predict (all of the acoustic feature variables and no song and artist identification variables) as well as the binary variable we would be predicting, top 10.

DATA REPRESENTATION – USING TABLEAU

We did a general analysis in Tableau, which gave us an understanding about the correlation between each variable and dependent variable.

The below charts are examples what Tableau presented us. The x-axis represents how many times a song was in top 10 during the 20 years. The y-axis is the average pitch of a song or average key of a song. We can conclude from these charts that the pitch of a song is much more correlated to the top 10 than the key of a song. For example, most of the songs which got into top 10 once are located under pitch 0.05. That means if a song's pitch is above 0.05, we can predict that it has very low probability of being in top 10. However, we cannot predict the probability of whether a song will be in top 10 with the distribution of "key" value, since the keys of Top 10 are scattered everywhere.

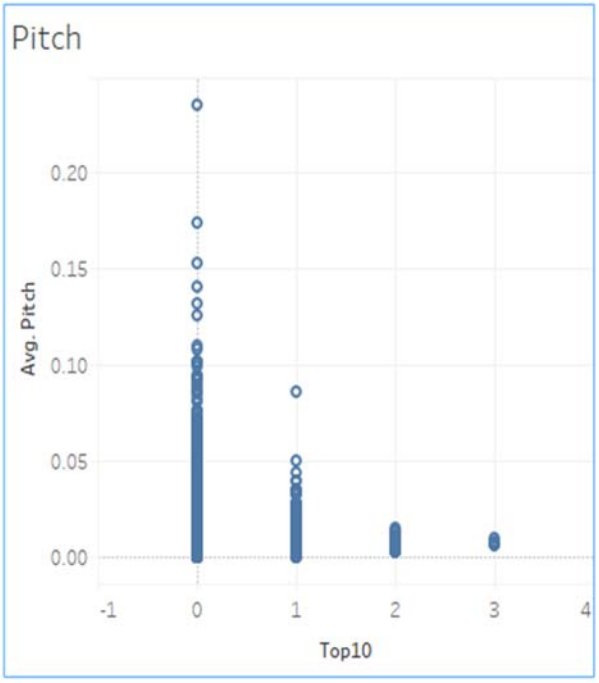

Figure 1 Relationship between songs that made to top10 and Avg. Pitch

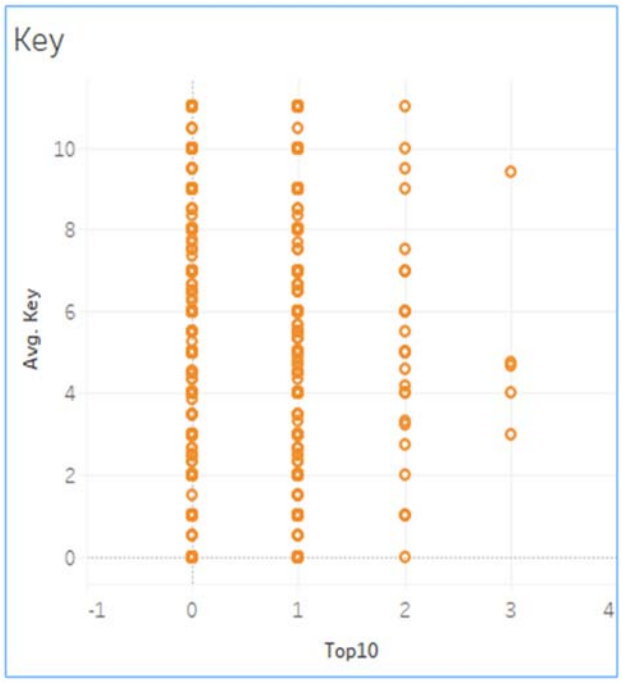

Figure 2 Relationship between songs that made to top10 and Avg. Key

The below chart is the trend analysis about the change of people's taste during these 20 years. They are almost all quite stable, which means people didn't change their taste a lot so the standard is very similar as time goes on. With this, we can conclude that the time is not a big issue in our project analysis, at least in regard to most of the variables.

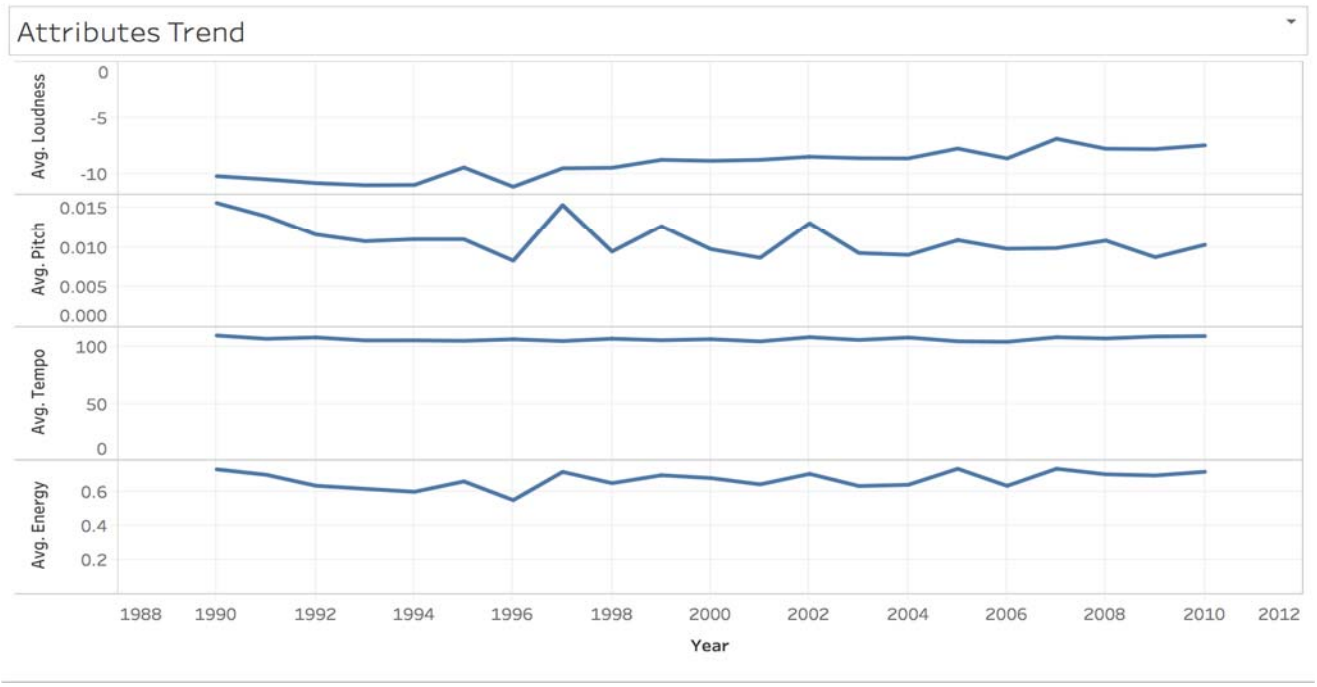

Figure 3 Chart depicting trends of certain key factors for popular songs over year

ROADBLOCK

The exception to what was stated in the last section is average loudness. Judging by the charts above the taste in music changes after sometime for this variable. Like the chart description said, the trends are stable, but average loudness seems to be trending upward. We were worried that trying to account for this would make this project a trend analysis project instead of a predictive classification one. As this was beyond the scope of our class, we stuck to classification techniques.

CORRELATION

In any model requiring the prediction of the value or class of a response variable or a dependent variable, we usually choose the predictor variables that are independent of each other. There is strong reason for this. When we have predictors that are not independent, change in the value of a predictor might affect other correlated predictors as well as our response variable. This leads to an erroneous explanation of the predictive ability of this predictor on our response variable. If bad enough, this correlation might lead to the ultimately wrong prediction.

To analyze the correlation between all the predictors, we calculated correlation coefficient for all pairs of predictors and then sorted the coefficient values in descending order. Upon doing this, we found out one particularly high correlation, the one between **Loudness** and **Energy.** This correlation had a coefficient of **0.74.** The coefficient takes values from -1 to 1, -1 suggesting high negative correlation and 1 suggesting a high positive correlation.

We will have to keep this correlation in mind, and reevaluate it when we later apply subset selection or remove unimportant predictors. Ideally we would expect our final set of predictors does not contain both of these predictors.

PREDICTIVE MODELS USED

In this project, we were classifying whether a song will make it to the top 10. This attribute is represented in our data as a factor with two classes 0 and 1. 0 indicating that the song did not make it to the top and 1 indicating otherwise. Hence, this is a binary classification technique and for this we used the following predictive models:

1. Logistic Regression
2. Random Forest
3. Support Vector Machines

We have tuned all the models on their specific parameters in order to get best results. A detailed explanation of each model and their results is presented in Application section of this report.

VALIDATION TECHNIQUE USED

Validation is an important aspect of any project, be it coding, designing a building, or anything in between. It helps in determining whether the work done on a project is yielding the intended result. Validation is often confused with verification; however, validation defines whether the project was done the right way whereas verification defines whether what was built was the correct object.

In predictive analytics, there are few types of validation such as:

1. Hold-Out Validation
2. Leave one out cross validation
3. K-fold cross validation

In this project, we have used Hold-out validation and split the data into a training set consisting of 60% of all the observations, and a testing set "SongsTest", consisting of the remaining 40%. We chose hold-out validation because it works well with large amounts of data like the amount we had.

Using this split, we are calculating three values for our testing data set:

1. **Accuracy:** Accuracy is often the starting point for analyzing the quality of a predictive model, as well as an obvious criterion for prediction. Accuracy measures the ratio of correct predictions to the total number of cases evaluated. It may seem obvious that the ratio of correct predictions to cases should be a key metric, but a predictive model may have high accuracy and still be useless. [2]

2. **Sensitivity:** Sensitivity is known as the true positive rate. It measures the portion of observed positives that were predicted to be positive. In other words, of all the transactions that were truly fraudulent, what percentage did our model find. [3]
3. **Specificity:** Specificity is the true negative rate. It measures the portion of observed negatives that were predicted to be negatives. In other words, of all the transactions that were legitimate, what percentage did our model predict to be so. [3]

We were more concerned about the songs that are in top 10 to be predicted right, than we were about songs that are not in the top 10 and still predicted as being in top 10. Hence, False negative rate or 1-Sensitivity was a good metric for us to use as we want the lowest FNR value or highest Sensitivity.

A good model should always have a balance of both Sensitivity and Specificity. We will see in the Application section how these values can differ from each other given a particular accuracy.

APPLICATION

In this part we will be discussing the application of the models and their outcomes. We tuned the different parameters associated with each model, and we compared results with every other version of the model in the Model Selection section, to find out the best fit model.

We started our analysis with Logistic Regression.

LOGISTIC REGRESSION

Logistic Regression is used only when the outcome is dichotomous, i.e. the response variable can take only two values. In our case the value we were predicting could take only "0" or "1", indicating if the song made it to the top 10. Logistic regression models generally require limited computing power and are less prone to overfitting than other models. This type of model provides us with probabilities of being in one of the categories. Our work is to find a particular threshold probability at which the model's metrics are best.

FULL MODEL

The first step to every predictive analytics model assessment is to create a baseline using all predictors in the model. We did exactly this and created a model using training data with all the predictors. The next step was to predict the values for the testing data and find the probability threshold value. Upon running a "FOR" loop for every threshold probability from 0.1 to 0.99 with the steps of 0.01, we found that threshold of 0.5 is optimal. This threshold gives out the following accuracy, sensitivity, and specificity value. We will use these values to compare to our other models:

1. **Accuracy:** 0.87
2. **Specificity:** 0.60
3. **Sensitivity:** 0.88

BACKWARD SUBSET SELECTION

By looking at the summary of the above model and the p values of predictors, we observed that some of the predictors had a low "pvalue" or in other words, low predictive ability. There is no use in keeping these predictors since they will negatively affect our analysis. After all, using predictors that have no relationship with the response tends to cause a deterioration in the test error rate. This is because such predictors cause an increase in variance without a corresponding decrease in bias. Removing such predictors may yield an improvement. In this case, we can follow two popular approaches, forward and backward subset selection. Both of these techniques are used to remove the predictors with low predictive ability.

We used backward subset selection which removed "energy", confirming our suspicions from our correlation analysis. Remember in the correlation analysis, we found a correlation with loudness and energy. Our conclusion was both of these should not be included together in a model. The subset selection confirmed this theory, and we refit the logistic regression model, finding better results in terms of specificity and only slight deterioration in terms of accuracy and sensitivity. Here Are the values for this model:

1. **Accuracy:** 0.86
2. **Specificity:** 1.00
3. **Sensitivity:** 0.86

RANDOM FOREST

Next, we tried random forests or random decision forests. These models are an ensemble learning method for classification, regression, and other tasks. They operate by constructing a multitude of decision trees during the training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. [4]

Just like with the Logistic Regression model, this model was trained on the same train data set and tested on the same test data set.

The advantages of this model were:

1. It takes care of the interaction. Given so many predictors, most of which we still did not completely understand, this was very useful. These predictors might have interactions with each other which we would not realize while creating any model.
2. It gives an importance value for each predictor, making it easy to identify and remove the unimportant ones.

Random Forest has two parameters on which we can tune the mode:

1. mtry: number of variables it takes to create each decision tree
2. ntree: number of trees it will create to create the forest

BAGGING

We started off with bagging, which is simply a special case of a random forest in which mtry is equal to the number of predictors, for our data 34.

We tried changing the number of trees grown by random forest by changing the ntree argument and finally decided upon keeping it as 1000. Following are the values for our model after bagging i.e. mtry =34:

1. **Accuracy:** 0.87
2. **Specificity:** 0.67
3. **Sensitivity:** 0.88

MTRY TUNING

Mtry parameter is a tunable parameter. In the above bagging example, we set it to 34. But next we used tuning to try to optimize the mtry like we did for the threshold value in the Logistic Regression model. We varied the value of mtry from 1 to 34, and upon doing that, we found the ideal mtry to be 9. With this change, we saw the accuracy and sensitivity remain the same, but we did see an improvement in specificity. Following are the metric values:

1. **Accuracy:** 0.87
2. **Specificity:** 0.69
3. **Sensitivity:** 0.88

IMPORTANT PREDICTORS

One great thing about random forest is that it gives the importance value of each predictor, making it easier for us to remove the ones with low predictive power or the unimportant ones.

Following is the Importance-Variable plot, for mtry = 9, and ntree = 1000

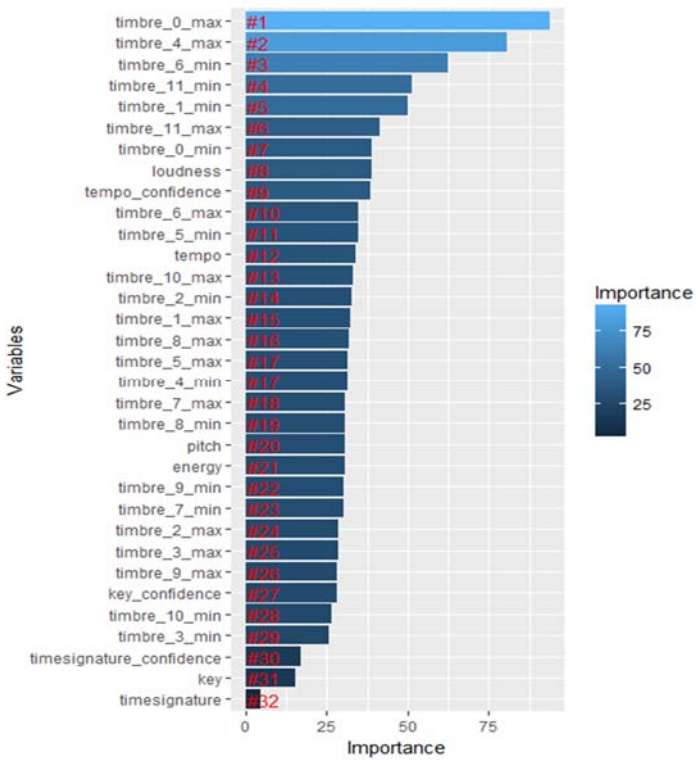

Figure 4 Impotance-Variable plot from Random Forest

It is easy to see from the above plot, that many predictors with less importance are within a similar range. To choose between them was a nightmare, so we ran different models, removing one variable at a time and noting down the accuracy, sensitivity, and specificity values for each. With this we got the best result by using the top 26 variables. This approach for removing predictors is not the most efficient approach, but considering our fairly low number of predictors, this method seemed easier to do. The following is our results:

1. **Accuracy:** 0.87
2. **Specificity:** 0.69
3. **Sensitivity:** 0.88

SUPPORT VECTOR MACHINES (SVM)

The last model on our list was SVM. For SVM, its algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space. These points are 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. SVMs can efficiently perform a non-linear classification using what is called a kernel. This technique uses kernels by implicitly mapping their inputs into high-dimensional feature spaces. [6] The classifier finds a linear decision rule in a higher dimension space which creates a non-linear decision boundary in the original space.

SVM can use many types of kernels, but we used just three; Linear, Radial and Polynomial. A kernel is an efficient means of enlarging the feature space to include the polynomial terms just discussed. We ran an SVM for each kernel and their respective tuning parameters. We then reported their results.

LINEAR KERNEL

Linear kernel of SVM can be tuned on cost parameter. The cost parameter specifies the cost of violation to the margin where margin is the area on both side of the classifier line that contains the support vectors.

After tuning the model with cost parameter, we got the best results at a cost of 0.1. Though our key metric, sensitivity, is particularly high, specificity is 0. This means that our classifier is not classifying the true negatives at all, which is not a good model. The reported values are as follows:

1. **Accuracy:** 0.85
2. **Specificity:** 0.00

3. **Sensitivity:** 1.00

RADIAL KERNEL

Next, we tried the Radial kernel and tuned it with a Gamma Value, giving us a gamma of 0.5 as our best value. This produced almost the same result as with the linear kernel. Here are the following metric values:

1. **Accuracy:** 0.86
2. **Specificity:** 0.00
3. **Sensitivity:** 1.00

POLYNOMIAL KERNEL

The model of our entire project was a SVM model with Polynomial kernel tuned with degree parameter. A degree parameter simply defines the degree of the classifier equation, like quadratic, cubic, etc. We found a degree equal to 2 as the optimal degree, but It had a disappointing result, similar to the above two. Below are the metric values:

1. **Accuracy:** 0.86
2. **Specificity:** 0.08
3. **Sensitivity:** 0.99

The next and final step of our analysis was to choose the best model for our objective.

MODEL SELECTION

After running each model, we reported the accuracy, sensitivity and specificity values. Below is a consolidated table of this data, with the green background tuple indicating the best model.

Model	Accuracy	Specificity	Sensitivity
LogFull Th 0.5	0.87	0.60	0.88
LogRed Th 0.5	0.86	1.00	0.86
RF Bag	0.87	0.67	0.88
RF mtry 9	0.87	0.69	0.87
RFRed Top 26 mtry 9	0.87	0.69	0.88
SVM Lin Cost 0.1	0.85	0.00	1.00
SVM Rad gamma 0.5	0.86	0.00	1.00
SVM Poly deg 2	0.86	0.08	0.99

As can be seen from the table above, the accuracy for all of the models is across the board not that different. Hence, choosing accuracy as our key metric would have been a bad choice. As was discussed in the Validation section, our main focus was on sensitivity. Given this, we can focus on the high sensitivity values such as 1. In these cases, the relative specificity values are low showing that the model is classifying all songs to be in top 10. These models would still be a bad choice. The only model with the high values for both specificity and sensitivity is logistic

regression with reduced model and threshold of 0.5. This can also be seen from the tableau chart on the below.

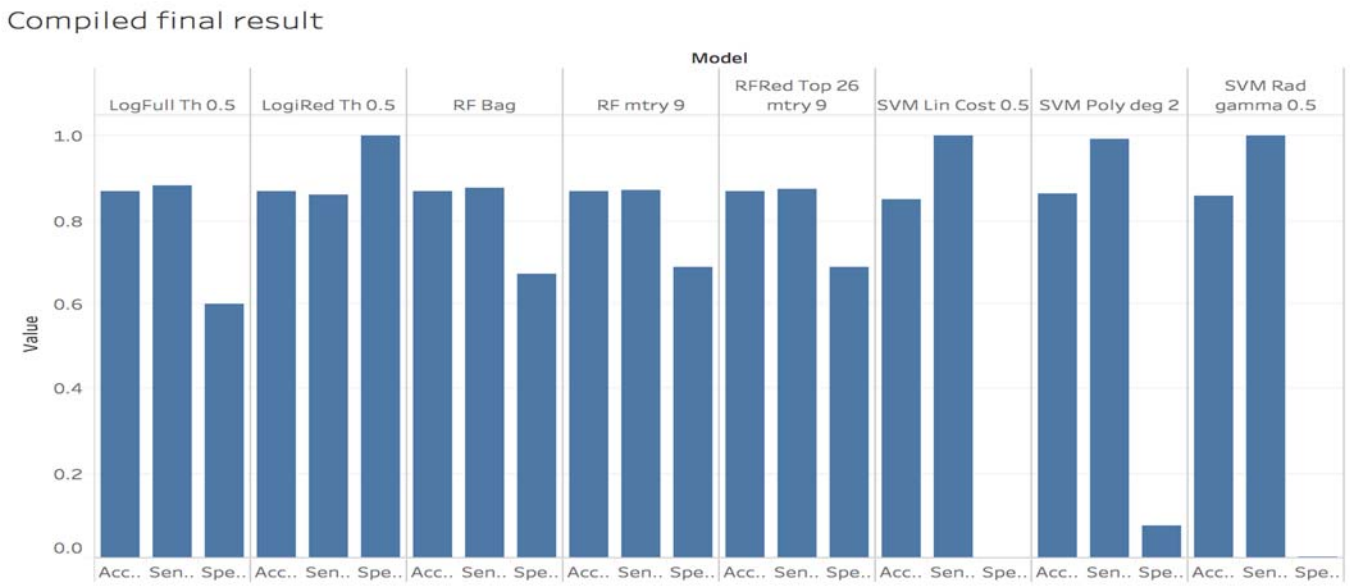

Figure 5 Chart with all our metric values for all models

Dictionary for both table and chart:

1. **LogFull Th0.5:** Logistic Regression complete model with threshold 0.5
2. **LogRed Th0.5:** Logistic Regression reduced model with threshold 0.5
3. **RF Bag:** Random forest with bagging
4. **RF mtry 9:** Random forest with mtry = 9
5. **RFRed Top 26 mtry 9**: Random forest reduced to top 26 variables and mtry = 9
6. **SVM Lin Cost 0.5**: SVM with linear kernel and Cost = 0.5
7. **SVM Poly deg 2:** SVM with polynomial kernel and degree = 2
8. **SVM Rad gamma 0.5:** SVM with radial kernel and gamma =0.5

IF TIME PERMITTED

The subset selection methods that we used in our models involved using a subset of the predictors. Another option would have been fitting our models containing all predictors using techniques such as Ridge Regression and Lasso that constrain the coefficient estimates. Lasso is less flexible and more interpretable than linear regression. This is because in the final model, the response variable is only related to the predictors with nonzero coefficient estimates. We could have also tried Bayes Classifier, which produces the lowest possible test error rate. We also could have found out the optimal k value and ran the KNN model. Another technique we could have used would be to have tuned the number of trees, the shrinkage parameter, and the number of splits in each tree. From this we could have tried out all possible types of random forest models. Finally, we could have tried making a weighted average of all the possible combinations of models to get the desired test error rate. It is generally assumed that the combination of models, which is also called Ensemble Averaging, works better than individual models. In brief, we could have tried a variety of models by using all different types of parameter tunings and could have likely surpassed our model's metrics.

Apart from making the perfect model, there are a variety of extensions that could be made to the existing model. One extension would be to include popular song revenues. Another would be to find out how music genre and singers' popularity change over time, which one might suspect would lead to more accurate predictions. Finally, perhaps the biggest improvement that could be made would be to include other important factors when making popularity decisions such as subject and artist notoriety. All of these extensions could have been valuable but would have required huge amounts of added data.

REFRENCES

[1]
https://web.archive.org/web/20160528174915/http://developer.echonest.com/docs/v4/_static/AnalyzeDocumentation.pdf

[2]
https://en.wikipedia.org/wiki/Accuracy_paradox

[3]
https://www.theanalysisfactor.com/sensitivity-and-specificity

[4]
https://en.wikipedia.org/wiki/Random_forest

[5]
https://en.wikipedia.org/wiki/Support_vector_machine