CAPSTONE PROJECT 1
ANALYZING AND PREDICTING SONGS

Abigail Hipolito

October 31, 2017

TABLE OF CONTENTS

1. Introduction

1.1 About

If there's one thing I can't live without, it's not my phone or my laptop or my car – it's music. I love music and I love getting lost in it. My inspiration for this project is to find out what it is about a song that I enjoy so much.

1.2. Problems/Goals

- What features of a song do I like the most/least?
- How does music I like compare to music I don't like and to today's top hits?
- Create a predictive model on whether I will like or dislike a song.

1.3. Client

- Music listeners can gain insight on what they listen to and how their taste in music compares to music they don't like and to today's top hits.
- Music listeners can use the model to predict whether or not they will like a song.

2. Data Cleaning

2.1. Dataset

The dataset includes three Excel files:

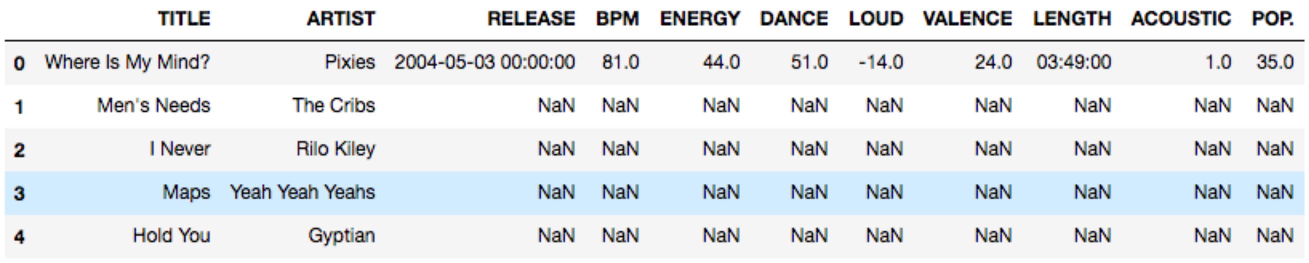
1. Liked playlist – (630 rows, 11 columns)
2. Disliked playlist – (537 rows, 11 columns)
3. Today's Top Hits playlist – (125 rows, 11 columns)

2.2. Data Acquisition

Using Spotify, I compiled the 'Liked' and 'Disliked' playlists. I put together 'Today's Top Hits' playlist from three of Spotify's playlists: 'Global Top 50', 'United States Top 50', and 'United States Viral 50'. There were some overlap between Spotify's three playlists so that's why there are only 125 songs and not 150 songs in 'Today's Top Hits' playlist.

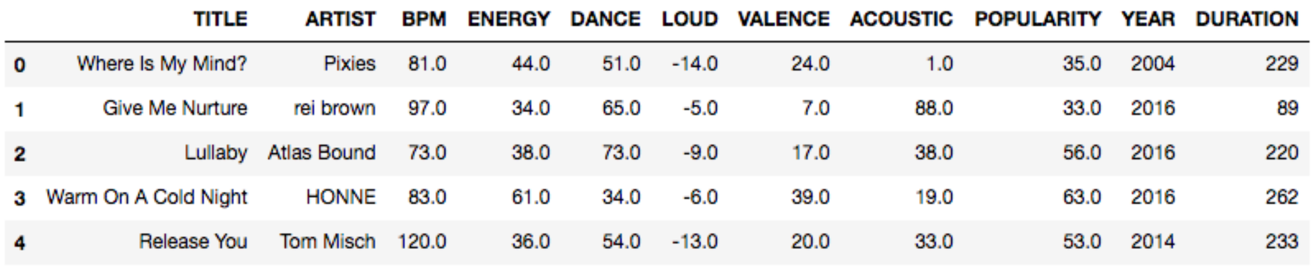
After creating these three playlists, I used a web app from Echo Nest called Sort Your Music that pulls the audio features of each song from Spotify. Sort Your Music pulls 9 audio features from Spotify and generates a neat table of songs and the values of their audio features.

Before Data Wrangling:



	TITLE	ARTIST	RELEASE	BPM	ENERGY	DANCE	LOUD	VALENCE	LENGTH	ACOUSTIC	POP.
0	Where Is My Mind?	Pixies	2004-05-03 00:00:00	81.0	44.0	51.0	-14.0	24.0	03:49:00	1.0	35.0
1	Men's Needs	The Cribs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	I Never	Rilo Kiley	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	Maps	Yeah Yeah Yeahs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	Hold You	Gyptian	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

After Data Wrangling:



	TITLE	ARTIST	BPM	ENERGY	DANCE	LOUD	VALENCE	ACOUSTIC	POPULARITY	YEAR	DURATION
0	Where Is My Mind?	Pixies	81.0	44.0	51.0	-14.0	24.0	1.0	35.0	2004	229
1	Give Me Nurture	rei brown	97.0	34.0	65.0	-5.0	7.0	88.0	33.0	2016	89
2	Lullaby	Atlas Bound	73.0	38.0	73.0	-9.0	17.0	38.0	56.0	2016	220
3	Warm On A Cold Night	HONNE	83.0	61.0	34.0	-6.0	39.0	19.0	63.0	2016	262
4	Release You	Tom Misch	120.0	36.0	54.0	-13.0	20.0	33.0	53.0	2014	233

Description of each Audio Feature:

1. **Beats Per Minute (BPM)** - The tempo of the song.
2. **Energy** - The energy of a song - the higher the value, the more energetic the song
3. **Dance** - The higher the value, the easier it is to dance to this song.
4. **Loud** - The higher the value, the louder the song.
5. **Valence** - The higher the value, the more positive mood for the song.
6. **Acoustic** - The higher the value the more acoustic the song is.
7. **Popularity** - The higher the value the more popular the song is.
8. **Year** – Release year of the song.
9. **Duration** – Length of song in seconds.

2.3. Data Wrangling

2.3.1. Dropping Null Values

I removed entries with null values from all three playlists:

- 'Liked' playlist: removed 10 rows with null values (from 630 to 620 rows)
- 'Disliked' playlist: removed 8 rows with null values (from 537 to 529 rows)
- 'Today's Top Hit' playlist: removed 1 row with null values (from 125 to 124 rows)

2.3.2. Changing Column Names

I changed some of the column names for simplicity and readability:

- From ‘RELEASE’ to ‘YEAR’
- From ‘LENGTH’ to ‘DURATION’
- From ‘POP.’ to ‘POPULARITY’

2.3.3. Changing Data Types

I changed the data types of two columns for easier analysis:

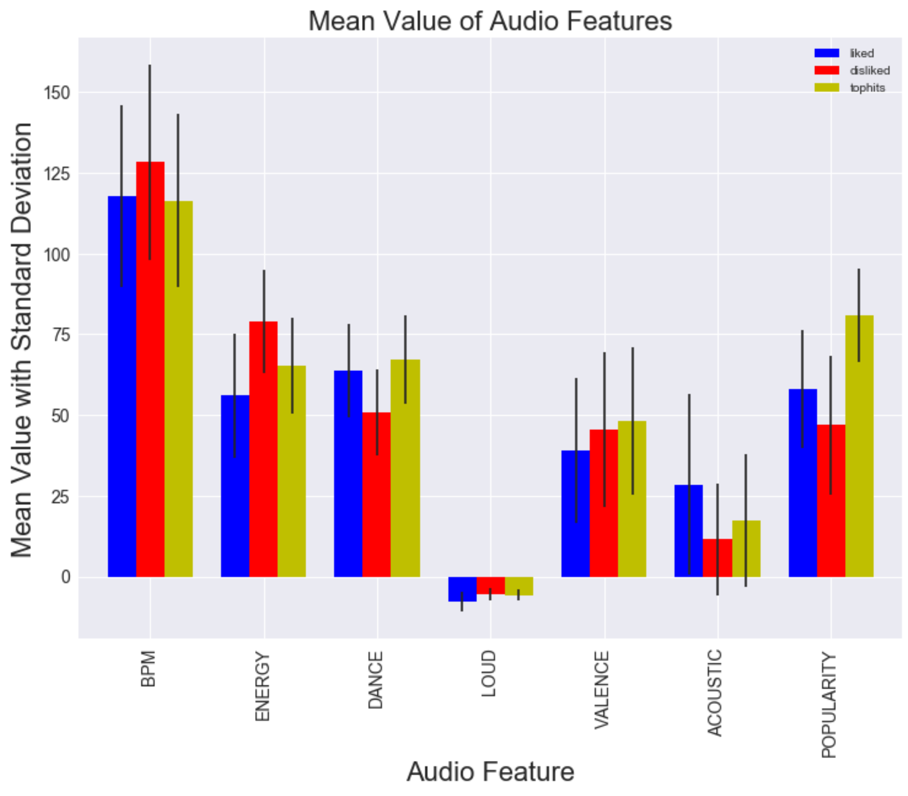
- In ‘RELEASE’ column, from object type to numeric type
- In ‘LENGTH’ column, from object type to numeric type

2.3.4. Date/Time Conversions

- In the ‘RELEASE’ column, I parsed just the year part from the YEAR-MONTH-DAY HOUR:MIN:SEC format.
- In the ‘LENGTH’ column, I converted the entries from a MIN:SEC:MSEC format to just SECONDS for easier analysis.

3. Exploratory Data Analysis

3.1. Means and Standard Deviations

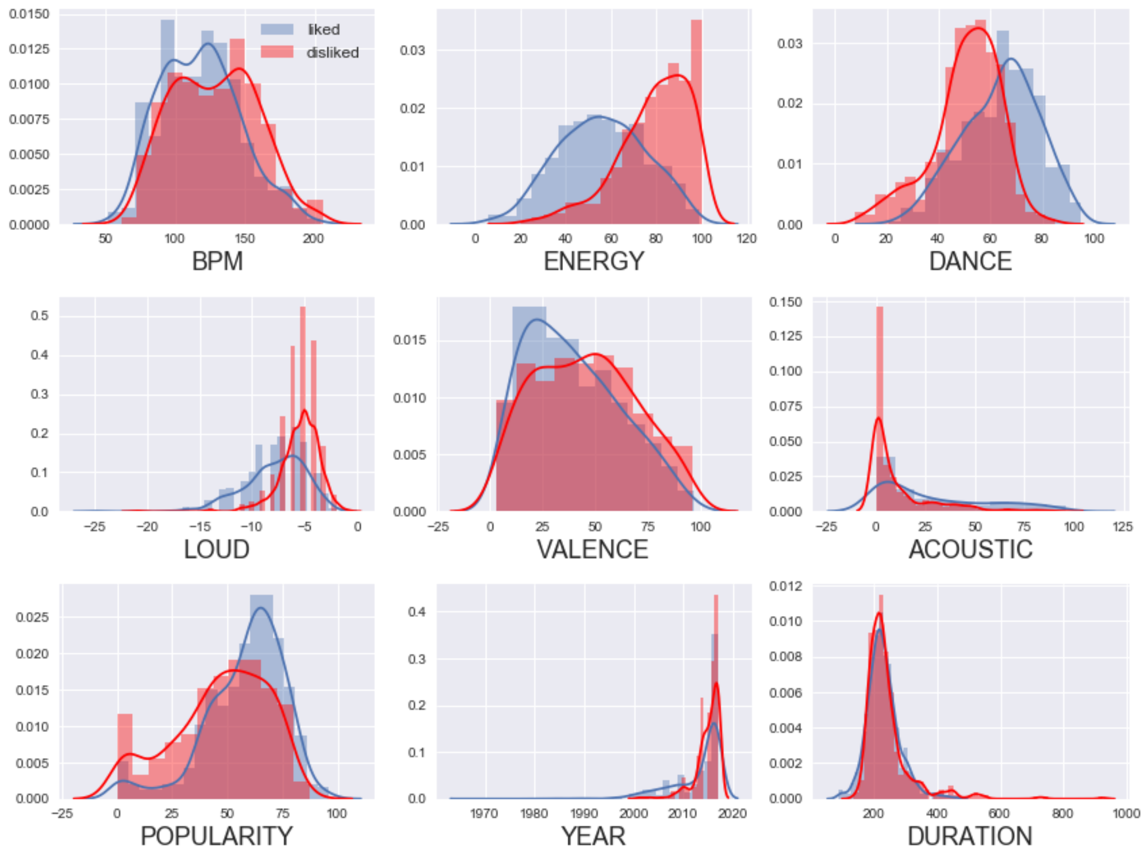


The first thing I looked to find out is the similarities and differences of each playlist. Using the mean values and standard deviations for each audio feature in my Liked, Disliked, and Today's Top Hits playlists, I can see a general comparison.

From the plot above, I see clear distinctions between each playlist. In general, I seem to like songs that are **lower in ENERGY, lower in VALENCE, and are more ACOUSTIC**.

The error bars do show overlap so I can conclude that my data has high variability. The larger and overlapping error bars can also signify less reliability in the means of my data; however, this does not indicate that the data is not valid.

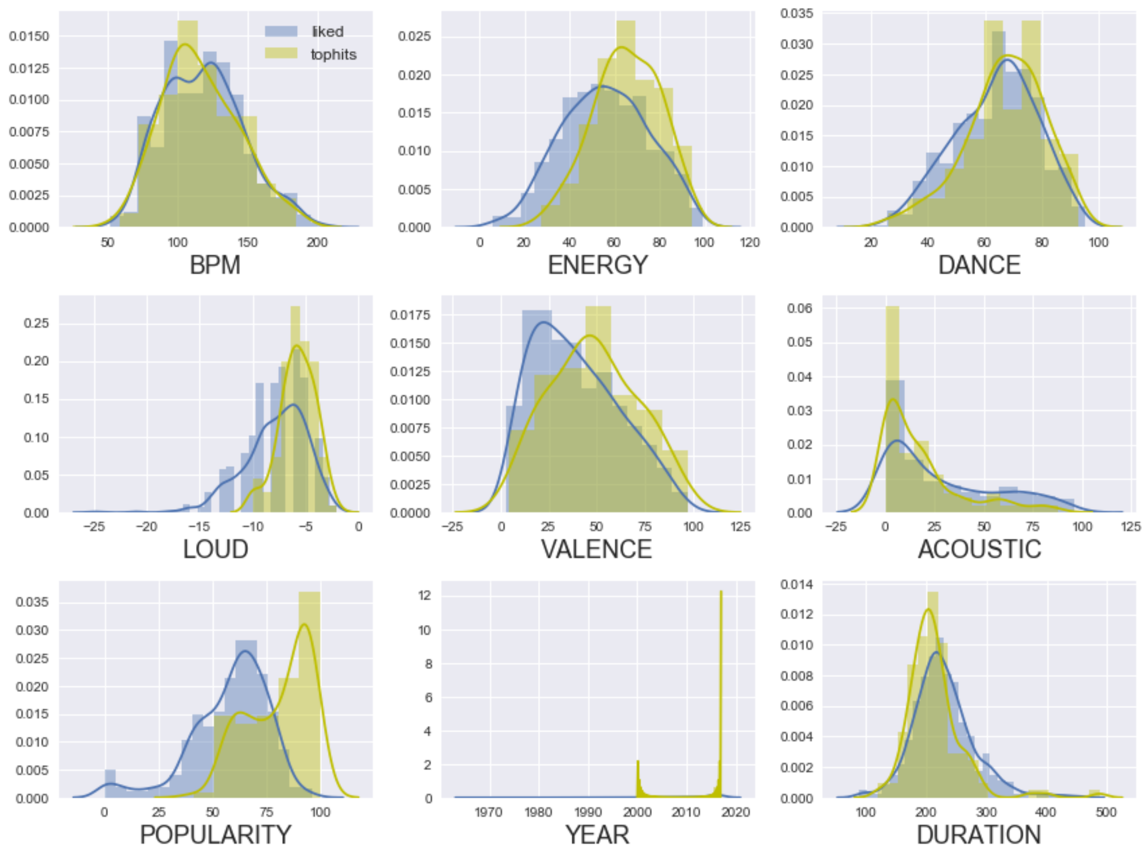
3.2. Distributions: Liked vs. Disliked Songs



Next, I compared the distributions of my Liked and Disliked songs. Looking at the distributions of each feature, there are clear distinctions between the two playlists.

I seem to **dislike songs that have a higher BPM, are higher in ENERGY, have lower DANCEABILITY, are LOUDER, have more VALENCE, are less ACOUSTIC, and are less POPULAR**. The YEAR and DURATION distributions are the only features that do not show much difference between the two playlists.

3.3. Distributions: Liked vs. Today's Top Hit Songs



Next, I compared the distributions of my Liked songs and Today's Top Hits. **Today's Top Hit songs are higher in ENERGY, are LOUDER, have more VALENCE, and are less ACOUSTIC.** Expectedly, **POPULARITY is higher and RELEASE YEAR is more recent for Today's Top Hits**. It's also interesting to note that DURATION is generally lower in today's top hits, probably because songs played on the radio are usually shorter.

4. Machine Learning

I use supervised learning, classification algorithms to make prediction models on whether I like or dislike a song. The 3 models I use are: k-Nearest Neighbor Classifier, Logistic Regression, and Random Forest Classifier.

4.1. Data Pre-Processing

First, I add a 'TARGET' column on the Liked and Disliked playlists: 1 for liked songs and 0 for disliked songs. I then combine the two playlists.

For the predictive model, I am only concerned with the Liked and Disliked playlists. In my Liked vs. Disliked Songs analysis above, YEAR and DURATION do not show much variation between the two classes; therefore, these features would not be helpful in the model.

The features useful for predictive modeling are:

10. **Beats Per Minute (BPM)** - The tempo of the song.
11. **Energy** - The energy of a song - the higher the value, the more energetic the song
12. **Dance** - The higher the value, the easier it is to dance to this song.
13. **Loud** - The higher the value, the louder the song.
14. **Valence** - The higher the value, the more positive mood for the song.
15. **Acoustic** - The higher the value the more acoustic the song is.
16. **Popularity** - The higher the value the more popular the song is.

Now that I have the features and target variable, I split the dataset in two parts, X and y:

```
X = combined[['BPM','ENERGY','DANCE','LOUD','VALENCE','ACOUSTIC','POPULARITY']]
y = combined['TARGET']
```

Now that I have X and y, I split them into training and testing tests, using 75% as training sets and 25% as testing sets.

4.2. Model the Data – Imbalanced Data

Since the data is imbalanced (619 majority class, 529 minority class), **resampling** (oversampling the minority class) and **SMOTE** (synthetic minority oversampling technique) are used to test out which sampling method is best for modeling.

The **area under the receiver operating characteristic curve (ROC AUC)** is used to score the accuracy of each sampling method. An area of 100% represents a perfect test and an area of 50% represents random guessing. The benchmark for this test is to beat random guessing. As shown below, the **resampled data** yields the highest scores across all three classifiers. So from this point on, the resampled data will be used on the models.

	ROC AUC*		
	Imbalanced	**Resampled**	**SMOTE**
k-Nearest Neighbor	85.43%	88.79%	86.90%
Logistic Regression	87.18%	89.90%	89.87%
Random Forest	85.07%	92.46%	91.44%

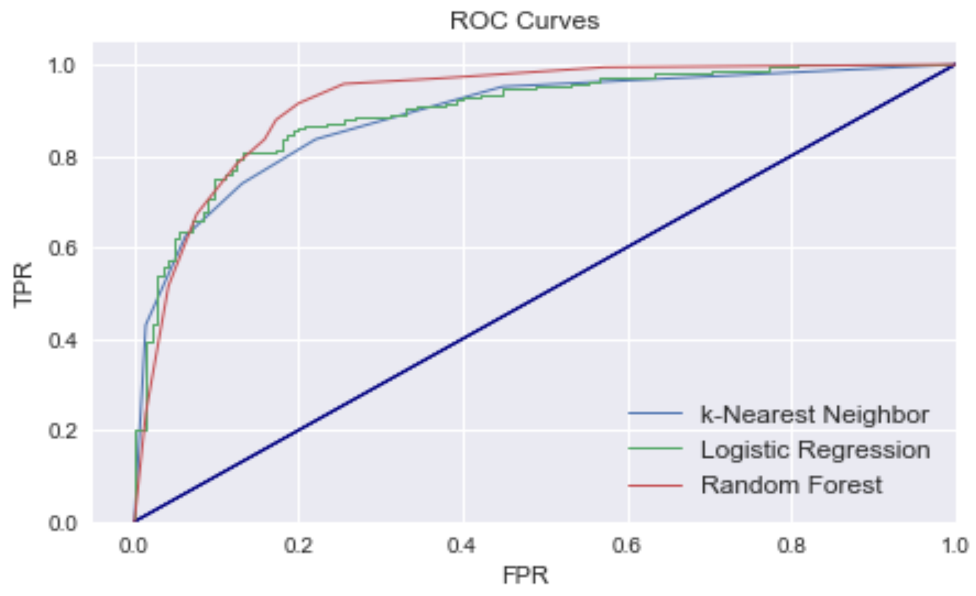
*Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores

4.3. Model the Data – Resampled Data

Now that I know that using the resampled data results in the highest ROC AUC scores, I plug in the resampled data into the three models to get more metrics. Below are the results:

	Resampled Data*			
	ROC AUC	**Accuracy**	**Precision**	**Recall**
k-Nearest Neighbor	88.79%	80.00%	86.52%	73.94%
Logistic Regression	89.90%	81.61%	83.75%	81.21%
Random Forest	91.94%	83.87%	85.71%	83.64%

*Metrics using Resampled Data



Overall, the three models performed fairly well, with ROC AUC scores well above the 50% benchmark. However, the **Random Forest classifier** resulted in the highest metrics, highlighted in green. With an ROC AUC score of 91.94% and an accuracy score of 83.87%, the Random Forest classifier performed very well on the test set. Looking at the ROC Curves, the Random Forest classifier is closest to 1.0 (perfect score).

On the next section, hyperparameter tuning will be performed on the Random Forest classifier to see if the model can be improved.

4.4. Hyperparameter Tuning - Random Forest Classifier

The parameters tuned are: `n_estimators, min_samples_leaf, and max_features`:

- `n_estimators: np.arange(10,200,10)`
- `min_samples_leaf: np.arange(1,100,10)`
- `max_features: ['auto', 'sqrt', 'log2']`

After using `GridSearchCV` to tune the parameters, the optimized parameters are as follows:

- `n_estimators: 80`
- `min_samples_leaf: 1`
- `max_features: 'sqrt'`

After using the new optimized parameters, I compare the results of the Random Forest model using default parameters and using optimized parameters. There is a definite improvement when using the optimized parameters, especially in Recall, with a 2.42% improvement.

	Random Forest			
	ROC AUC	**Test Accuracy**	**Precision**	**Recall**
Default	91.94%	83.87%	85.71%	83.64%
Optimized	93.38%	85.16%	86.06%	86.06%
Difference	1.44%	1.29%	0.35%	2.42%

5. Conclusion

5.1. Summary

The goals of this project were to find out what features of a song I like/dislike and to predict whether I like or dislike a song. Through exploratory data analysis and machine learning, these goals were accomplished.

After doing some exploratory data analysis, I found that I like songs that are lower in ENERGY, lower in VALENCE, and are more ACOUSTIC and I dislike songs that have a higher BPM, have lower DANCEABILITY, are LOUDER, have more VALENCE, and are less POPULAR.

After trying three different models to predict whether I will like or dislike a song, the best performing model is Random Forest with hypertuned parameters. Overall, I am excited about the results and believe the model can be useful in predicting whether or not you will like a song.