Music To My Wallet

Harish Chandramohan, Aaron Devaney, Alex Hall, Anne Lam, Andrew Vignali

Hypothesis/Questions to Answer

1. What parameters of a song best defines its popularity?

2. Can we build a model that will predict a song's popularity?

3. Can we use the popularity score to project revenue?

Data Exploration

Dataset: Spotify Dataset, 1921-2020, 160k songs

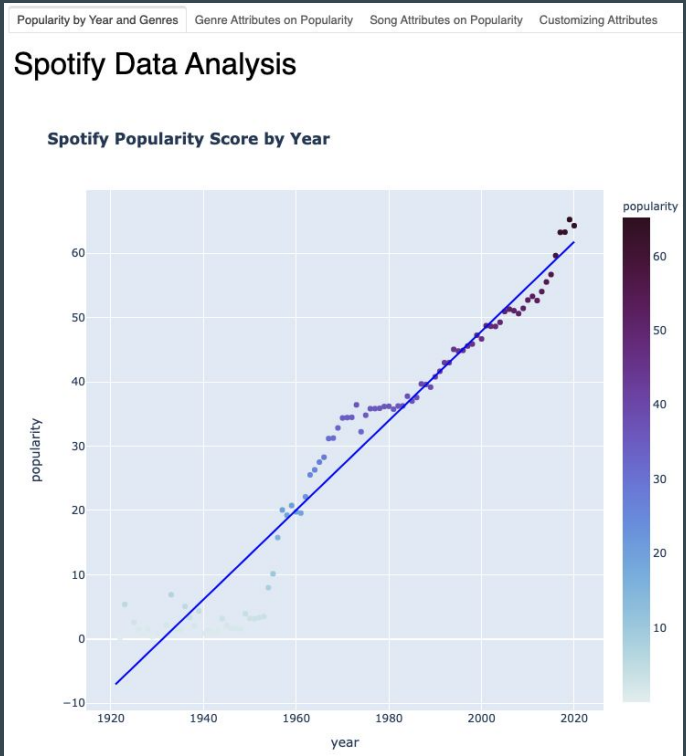
Link: https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks

Parameters:

- Year
- Valence (float) - measure describing the musical positiveness of a track
- Acousticness (float) - confidence measure of whether the track is acoustic
- Danceability (float) - describes how suitable a song is for dancing
- Duration (integer) - how long a song is in milliseconds
- Energy (float) - perceptual measure of intensity and activity
- Explicit (dummy) - 0 if contains no explicit language, 1 if present
- Instrumentalness (float) - predicts whether a song uses no words
- Liveness (float) - predicts presence of audience in recording
- Loudness (float) - overall loudness in decibels
- Mode (dummy) - indicates modality of a track, 0 if minor, 1 if major
- Speechiness (float) - detects the presence of spoken words in a track
- Tempo (float) - overall estimated tempo of track in BPM

Prediction (Y Value): Popularity Score ranging from 0 to 100

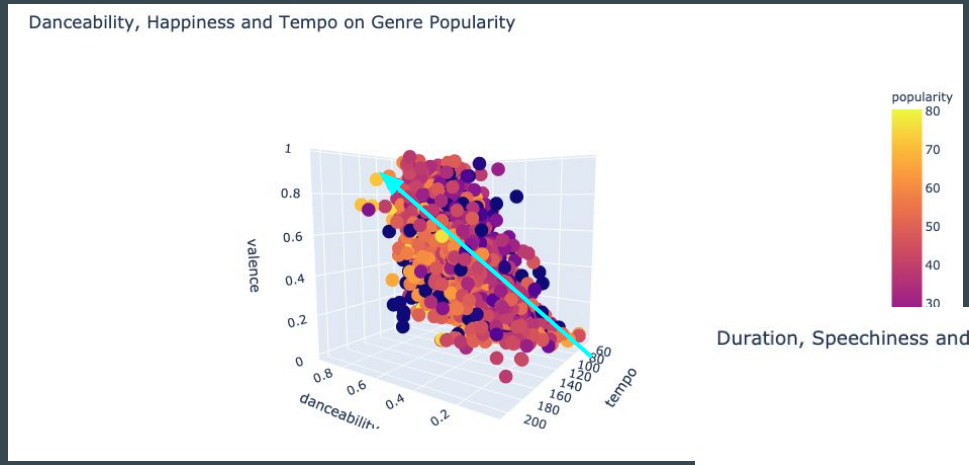
Data Analysis & Visualizations



	valence	year	acousticness	artists	danceability	duration_ms	energy	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
0	0.0594	1921	0.982	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0.665	-20.096	1	Piano Concerto No. 3 in D Minor, Op. 30: III. ...	4	1921	0.0366	80.954
1	0.9630	1921	0.732	['Dennis Day']	0.819	180533	0.341	0.160	-12.441	1	Clancy Lowered the Boom	5	1921	0.4150	60.936
2	0.0394	1921	0.961	['KHP Kridhamardawa Karaton Ngayogyakarta Hadi...	0.328	500062	0.166	0.101	-14.850	1	Gati Bali	5	1921	0.0339	110.339
3	0.1650	1921	0.967	['Frank Parker']	0.275	210000	0.309	0.381	-9.316	1	Danny Boy	3	1921	0.0354	100.109
4	0.2530	1921	0.957	['Phil Regan']	0.418	166693	0.193	0.229	-10.096	1	When Irish Eyes Are Smiling	2	1921	0.0380	101.665

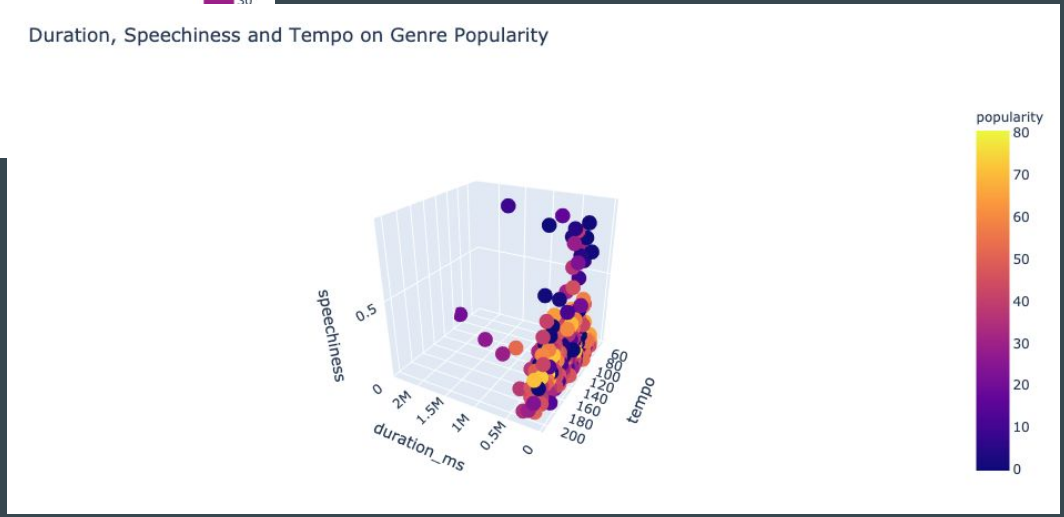
- The Spotify data seem to have an **recency bias effect** as popularity scores tend to trend higher in the most recent years as people tend to listen to music produced more recently.
- We ended up filtering the data more by **only using songs produced after 1960s** (120K data points)

3D-Scatter Plots for Broad View

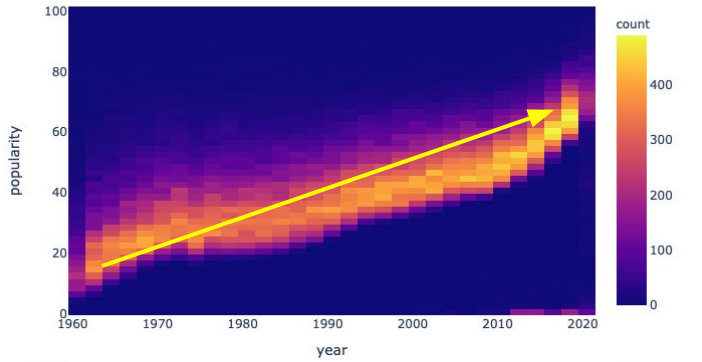


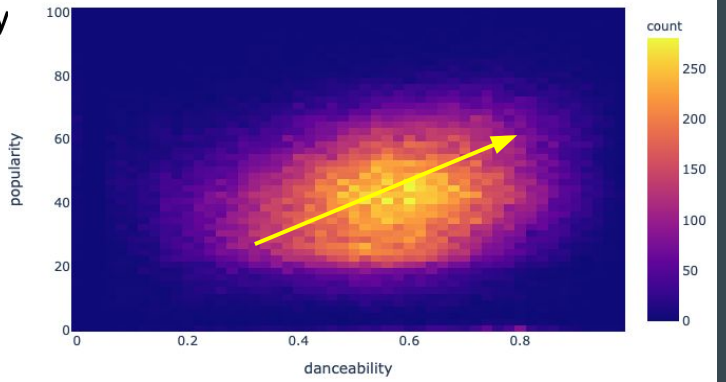
- Found that danceability and duration seems to have an impact on popularity
- Also found that parameters like tempo and valence didn't have much of an impact

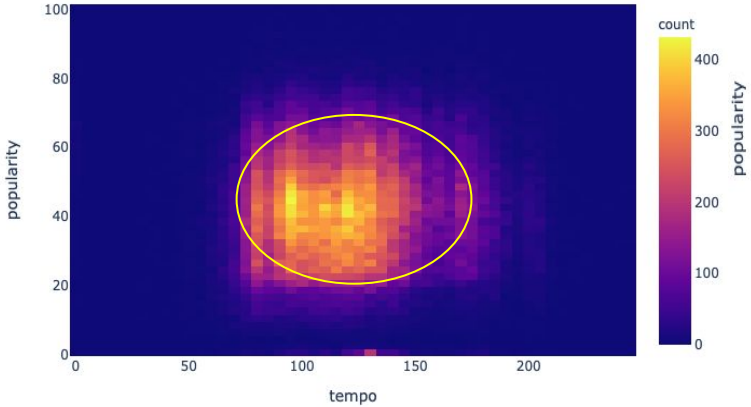
- Used 3D plots to get a broad view of quick relationships and trends to see which parameters to focus on more in our model

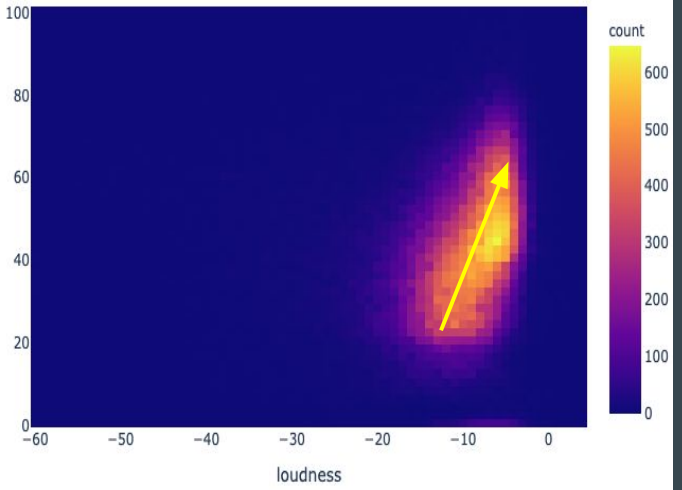


View of Song Attributes Contribution on Popularity

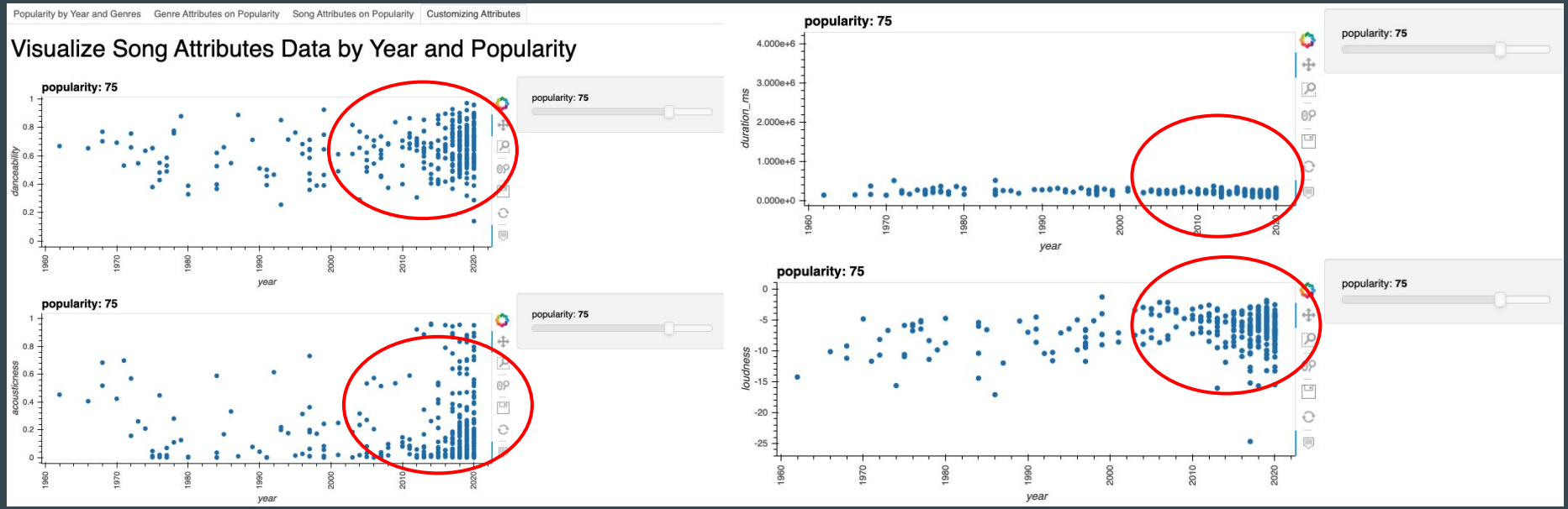








Visualizing the Parameters by Popularity Score



Modeling

Data

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
0	0.0594	1921	0.982	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0	4BJqT0PrAfrxzMOxytFOIz	0.878000	10	0.665	-20.096	1	Piano Concerto No. 3 in D Minor, Op. 30: III. ...	4	1921	0.0366	80.954
1	0.9630	1921	0.732	['Dennis Day']	0.819	180533	0.341	0	7xPhfUan2yNtyFG0cUWkt8	0.000000	7	0.160	-12.441	1	Clancy Lowered the Boom	5	1921	0.4150	60.936
2	0.0394	1921	0.961	['KHP Kridhamardawa Karaton Ngayogyakarta Hadi...	0.328	500062	0.166	0	1o6I8BglA6ylDMrIELygv1	0.913000	3	0.101	-14.850	1	Gati Bali	5	1921	0.0339	110.339
3	0.1650	1921	0.967	['Frank Parker']	0.275	210000	0.309	0	3ftBPsC5vPBKxYSee08FDH	0.000028	5	0.381	-9.316	1	Danny Boy	3	1921	0.0354	100.109
4	0.2530	1921	0.957	['Phil Regan']	0.418	166693	0.193	0	4d6HGyGT8e121BsdKmw9v6	0.000002	3	0.229	-10.096	1	When Irish Eyes Are Smiling	2	1921	0.0380	101.665

Data Preparation

```
#Use only essential columns
cleanData = data_1960_df[['valence', 'year', 'acousticness', 'danceability',
       'duration_ms', 'energy', 'explicit', 'instrumentalness', 'key',
       'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'popularity']]
```

```
#Remove songs created before 1960
year_limit=1960
data_1960_df = data[data['year']>year_limit]
```

```
#Split data into X and y
X = cleanData.iloc[:, 0:14].values
y = cleanData.iloc[:, 14].values
```

```
#Scale X with Standard Scaler
scaler = StandardScaler().fit(X)
X = scaler.transform(X)
```

Modeling

Data Processing

```
#Initalize Nueral Networks
nn = Sequential()

# Hidden layer
nn.add(Dense(units=64, input_dim=14, activation="relu"))

# Second hidden layer
nn.add(Dense(units=32, activation="relu"))

# third hidden layer
nn.add(Dense(units=16, activation="relu"))

# fouth hidden layer
nn.add(Dense(units=8, activation="relu"))

# Output layer
nn.add(Dense(units=1, activation="linear"))
```

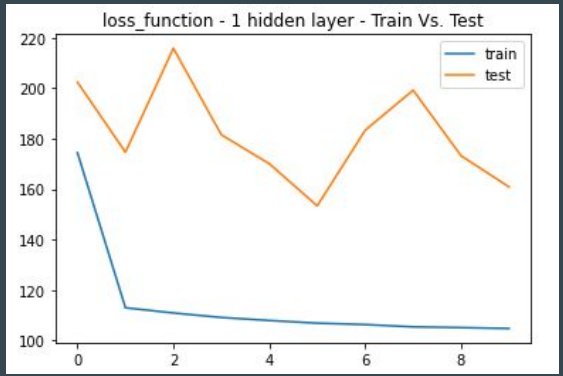
```
# Compile the model
nn.compile(loss="mean_squared_error", optimizer="adam", metrics=["mse"])
```

```
# Fit the model
model_1 = nn.fit(X, y, validation_split=0.3, epochs=10)
```

```
Epoch 1/10
2601/2601 [==============================] - 5s 2ms/step - loss: 362.8113
Epoch 2/10
2601/2601 [==============================] - 5s 2ms/step - loss: 114.6174
Epoch 3/10
2601/2601 [==============================] - 5s 2ms/step - loss: 112.3917
Epoch 4/10
2601/2601 [==============================] - 5s 2ms/step - loss: 110.3810
Epoch 5/10
2601/2601 [==============================] - 4s 2ms/step - loss: 109.0327
Epoch 6/10
2601/2601 [==============================] - 5s 2ms/step - loss: 107.1948
Epoch 7/10
2601/2601 [==============================] - 5s 2ms/step - loss: 106.1605
Epoch 8/10
2601/2601 [==============================] - 5s 2ms/step - loss: 105.5710
Epoch 9/10
2601/2601 [==============================] - 5s 2ms/step - loss: 104.9903
Epoch 10/10
2601/2601 [==============================] - 5s 2ms/step - loss: 104.2273
```

Modeling

Results



actual	predicted
72	61.326263
68	65.772156
76	67.252808
70	64.317543
74	70.672508

```
#Statistics on results
from scipy.stats import ttest_ind

ttest_ind(df['actual'], df['predicted'])
```

```
Ttest_indResult(statistic=-47.58512250813055, pvalue=0.0)
```

Model Comparison - Classification Model Types

We ran our train and test data across four different classification model types to compare our results.

1) Deeplearning
2) Gradient Boosting Model
3) Decision Tree
4) Random Forest

```
Accuracy Score for Deeplearning Model: 0.040639192598822 54
Accuracy Score for Gradient Boosting Model: 0.008847771236333053
Accuracy Score for Decision Tree Model: 0.03875525651808242
Accuracy Score for Random Forest Model: 0.04827586206896552
```

Our most accurate model is Random Forest

Model Comparison - Scaler Types

RobustScaler - Data is robust to outliers, removes the median and scales data according to quantile ranges. The output statistics are based off percentiles and therefore not influenced by large outliers

QuantileTransformer - This method transforms the features to follow a normal distribution, therefore spreading out the most frequent values

```
The accuracy score using RobustScaler is 0.0148444070647603 03
The accuracy score using StandardScaler is 0.035096719932716 57
The accuracy score using MinMaxScaler is 0.034373423044575274
The accuracy score using QuantileTransformer is 0.0055172413793 10344
```

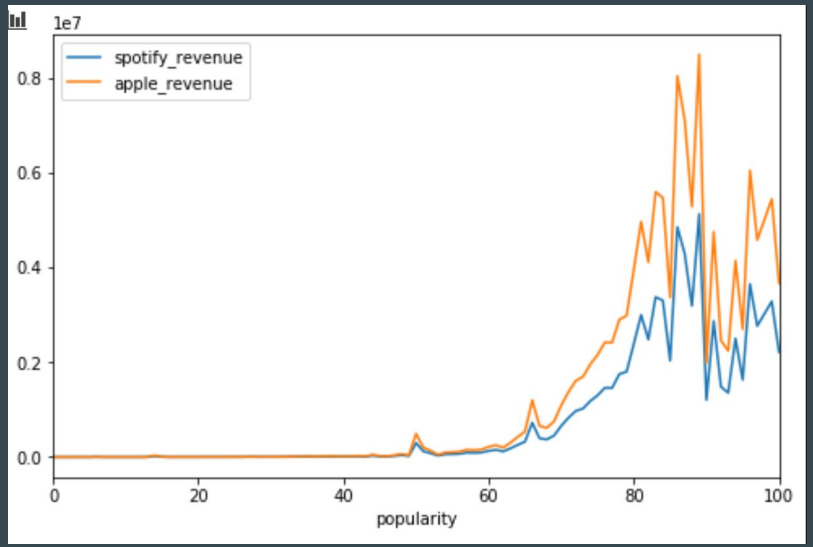
Clearly, the most accurate scaling type on our model was StandardScaler..

Data Collection for Popularity and Revenue Analysis

There was no data we could find about how many times each song had been played on Spotify. This data had to be manually collected, so we chose a sample of 5 songs from each popularity score category. These songs were then grouped by popularity to get the mean play count.

Spotify pay-per-stream: up to $0.00437

Apple pay-per-stream: up to $0.00735



Relationship between Popularity Rank and Revenue

- Some songs that have well-known artists but a lesser play count still seem to receive higher popularity scores. Ex: Wabash Cannonball by Johnny Cash - play count 521,601

- Holiday Music has a much greater # of plays than other songs with the same popularity score. Ex: Let it Snow! Let it Snow! Let it Snow!

- Multiple songs with the same title (cover music, featuring other artists.) This was most prevalent with classical music and seasonal or holiday music

- Multiples of the same songs repeated in different albums/collections - many have different play counts

Revenue Prediction Widget

- Created a widget that uses our deep learning model to make revenue projections based on song popularity
- Key audio features can be adjusted by sliders and the function can be executed once all of the features are set
    - Our model uses all the audio feature to make its predictions; however, instead of requiring the user to input all the feature, we selected what we considered the most important features as the sliders
- A musician or record producer could potentially use this tool to fine tune audio features of a song to produce maximum revenue from streaming on Spotify

Demonstration