COGS 108 - Final Project

Overview

Our project was to look at the top songs of 2017 and see what attributes play a significant role in the placement of those songs on Spotify Charts. We looked at 14 attributes that were determined and calculated by Spotify. Our focus was to compare both the differences between each attribute and the number of streams per song. This was to see if we would be able to determine the chances of a song charting.

Names

- James Zeng
- Eunice Chan
- Raju Ivaturi
- Gen Barcenas
- Isabel Li
- Sarah Cho

Group Members IDs

- A14251323
- A13516475
- A13529994
- A15753315
- A14542428
- A15074167

Research Question

What factors make a song chart the Top 10 Spots on the Spotify Top chart in 2017? Factors included danceability, speechiness, valence, and other musical attributes that may influence a song's ability to gain popularity through streams.

Background and Prior Work

Spotify is a company that provides music streaming services for their subscribers. They gather a large collection of songs containing millions of tracks. Spotify works both as a radio and personal music collection. Spotify offers both free and premium versions of its services. The free version

includes ads every thirty minutes, has a limited amount of skips, and subscribers are unable to listen to songs offline. The paid version is ad free, unlimited amounts of skips, and playlists can be saved to listen to offline. Spotify gains access to music through contracts with artists; however, because it is dependent upon if an artist agrees with the contract, some music is not available on Spotify, like Jay Z. Also, Spotify works mostly with more established artists, so it doesn't always have music that is published independently.

Spotify's Top 200 Chart is made through counting the number of streams of each song. A stream is counted when a song has been played for over 30 seconds. The chart is updated daily, but they also do keep track of the numbers for a week as well.

This question was of interest to our group because we are young college students that enjoy music and utilize Spotify. We were influenced to use this approach because of a past team member's recollection of a data visualization project involving Kanye West and his lyrics. We also wanted to use data that was relevant to our daily lives and we thought exploring pop culture and trends would be the best way to do so. We also all marked "music" on the course survey.

A prior study we found looked at was one on Billboard Top 200 songs and they looked at the chords, structure, instrumentation, and timing of popular songs. From that they had they analyzed the parts of the music to understand what made it a popular charting song.

Another prior study we looked at was one on Kaggle: This link is a composition of data that analyzes top Spotify songs of 2017. The dataset contains the daily ranking of the 200 most listened to songs from 53 countries from 2017 to 2018. The data directly correlates to our research project because our team is also analyzing different factors that contribute to a songs popularity. It contains more than 2 million rows, which comprises 6629 artists, 18598 songs for a total count of one hundred five billion streams count. Variables included with the dataset are song name, artists, danceability, energy, key, loudness, speechiness, etc. These all take into account different factors of a song, similar to the second link, and illustrate the common factors that top songs share.

References (include links):

- 1) https://www.spotify.com/us/about-us/contact/ (https://www.spotify.com/us/about-us/contact/)
- 2) http://ddmal.music.mcgill.ca/research/billboard (http://ddmal.music.mcgill.ca/research/billboard)

Hypothesis

Based on our background knowledge of currently trending artists in America, we hypothesize a strong positive correlation between the ranking of a song on Spotify's Top 200 chart and danceability, valence, speechiness. We chose these three attributes because they each are unique attributes and seemed unlikely to have any correlation with one another. We hypothesize a positive correlation because based off of our own knowledge and experience on music trends, happy-sounding, rap-like steady songs tend to chart well.

Dataset(s)

We found a dataset from Kaggle which consists of the top 100 songs from Spotify in 2017. We wanted this dataset because it contained exactly what we were looking for: attributes of popular songs. For each song it contains its: id, name, artists, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration_ms, and time_signature. We are hoping to see some factor in these audio features that contributed to the song's success.

- Dataset Name: Top Tracks of 2017
- Link to the dataset: https://www.kaggle.com/nadintamer/top-tracks-of-2017 (https://www.kaggle.com/nadintamer/top-tracks-of-2017)
- Number of observations: 100

This dataset consists of the track name and artists of the top 100 songs of 2017 played on Spotify. It also contains a number of audio features associated with each song.

- Name is the title of the song.
- Artists are the main artist of that song. This doesn't always include the artists that were featured because it is dependent on what an artist decides to put on Spotify.
- Danceability is how suitable a song is for dancing. This includes a combination of the regularity of the beat, the song tempo, and rhythm / beat pattern of the song.
- Energy is the intensity of the song which is measured by the wide range of sound volumes. High energy songs are characterized as "fast, loud, noisy" while low energy songs are characterized as "slow, quiet".
- Key is the key signature that the song was written in. There are 11 different keys and 0 = C, 1 = C♯/D♭, 2 = D, etc.
- Loudness is the overall loudness of a track in decibels and is averaged across the track.
- Mode indicates if the song is written in a minor or major key.
- Speechiness is the presence or absence of verbal speech or communication. For example, talk show-like speech is closer to 1.0 on the scale of 0 to 1.
- Acousticness is a measure of the amount of electricity used in a song. If a song is less electrically amplified it scores higher on a scale from 0 to 1.
- Instrumentalness measures the lack of vocals; a score closer to 1.0 means that there is no vocal content. Liveness is measuring the amount of presence there is of an audience in the recording. If a song was performed live, the liveliness value would be higher than a song recorded in a studio.
- Valence is a measure of the positivity of the sounds. The higher the rating the more positive / happy it sounds.
- Tempo is the number of beats per minute or the pace of the song.
- ID is the unique Spotify URL of the song.
- Duration_ms is the length of the song in milliseconds.
- Time_signature is a number specifying how many beats are in each bar.

After finding this dataset, we felt that this dataset was inherently limited because we focus only on the most popular songs in 2017. This means that there may not much variance between them. All these songs may have somewhat of the same factors, which led to them being popular in the first place. Therefore, we decided to add all the songs that were in the top 200 at any point in 2017 to try to include more variance between the songs we analyze.

We went to spotifycharts.com and pulled the top 200 United States songs for each week in 2017, for more songs to include in our dataset.

- Dataset Name: Spotify Charts Top 200 Songs for each Week in 2017
- Link to the dataset: https://spotifycharts.com/regional/us/weekly/ (https://spotifycharts.com/regional/us/weekly/) (weeks in 2017 dropdown menu)
- Number of observations: 200 (each)

This dataset consists of the the position, track name, artists, and # of streams of the top 200 songs of the week specified. We felt that this dataset represents trending songs during particular weeks in 2017.

- Position is the song's popularity rank during that week.
- Track name is the name of the song.
- Artists are the main artist of that song. This doesn't always include the artists that were featured because it is dependent on what an artist decides to put on Spotify.
- Streams is the number of plays that the song had that week.

We combined the datasets by pulling chart data from spotifycharts.com for each week in 2017 that the song was on the top 200 and added it to the Kaggle dataset. We first included a 'streams' column because we wanted to keep track of how popular a song was. We added it by first checking if the song already exists in the dataset. If it did, we would just add the stream count to that song entry. If it didn't, we would append the track entry onto the Kaggle dataset, with its stream count. We kept doing this until there were no more songs to analyze in 2017 from spotifycharts.com. Because spotify does not include the song features, we also went to Spotify's API and pulled the song features and filled each song in our dataset with its corresponding features. We essentially created a dataset featuring all the songs that were trending at any point in 2017.

Setup

In [24]:
```
# for dataframes, and easy tables
import pandas as pd
# for its variety of data types (for dataframe columns)
import numpy as np
# to ping requests from spotifycharts and get the chart content
import requests
# in order to read strings as file streams
# (so we can use pd.read_csv easily)
import io
# python wrapper for the Spotify API
import spotipy
# so spotify knows who is pulling from their API
from spotipy.oauth2 import SpotifyClientCredentials
# data visualization
import matplotlib.pyplot as plt
# more data visualization
import seaborn as sns
# OLS regression
import statsmodels.api as sm
from sklearn import linear_model
```

Data Cleaning

We first dropped columns we felt were completely irrelevant to our hypothesis. The attributes that we ended up dropping were ID, mode, key, duration_ms, instrumentalness, liveness, and acousticness. The Kaggle dataset described ID as being the Spotify ID of the track. A track's ID is important because we can find all the music features corresponding to the track using Spotify's API. However, we found that the ID's were broken and some did not refer to the song that they were supposed to. Mode, key, duration_ms, instrumentalness, liveness, and acoustic-ness were all attributes that we felt had no correlation with how popular a song is. A track that is in a different key, has different duration, was acoustic, or contained a possible audience should also not have any correlation with how popular a song is.

The Kaggle dataset and the Spotify Charts datasets had small differences in the column names which made merging slightly harder. In order to remedy this, we adjusted the track name and artists column to match the Kaggle datasets column names.

After building our dataset, we noticed that spotifycharts.com contained a couple NaN values in their charts on 2017-07-14 and 2017-11-03. This would skew our chart, because we have no idea what these songs are and which entry to add its corresponding number of streams to. So, there would be weird NaN entries in our data. We felt that it was best to just omit adding the data from these two charts, because we were concerned about the credibility of a dataset with NaN values. Because this is just a small amount of data, our resulting dataset should not be greatly affected by this.

There were also some songs that did not have the exact same name in the Kaggle dataset when looking at the spotifycharts.com datasets. Songs such as "I'm the One" and "Wild Thoughts" were listed as such in the Kaggle dataset, but in Spotify Charts, the listing of the song were ""I'm the One (feat. Justin Bieber, Quavo, Chance the Rapper & Lil Wayne)" and "Wild Thoughts (feat. Rihanna & Bryson Tiller)". We think the problem is that the Kaggle dataset decided to make some simplifications to make the chart more readable, but it ultimately caused some problems when trying to automate the counting of streams over 2017. It created multiple entries of the same songs, with varying stream counts. As a result, we changed all the names of the Kaggle dataset in the beginning to match with the names that will show up in Spotify Charts.

One song in particular also caused a few problems. Along with the problem listed above, "I'm the One" had a different name for one of the datasets pulled from Spotify Charts. For that one chart from the website, it had decided to change to a name without all featuring artists. Although it doesn't make a large difference, we located the duplicate song entry and combined the stream counts.

All that remains in our dataset is the track name, artists, danceability, energy, loudness, speechiness, valence, tempo, and streams.

The following snippets of code is shown to demonstrate these data cleaning steps taken.

In [ ]:
```
# the commented line will rename each returned DataFrame
#   from spotifycharts to match the columns on Kaggle.
# we do this to make such no duplicate columns with the
#   same values appear
def get_chart(start):
    end = start + pd.DateOffset(weeks=1)
    date = f'{start.date()}--{end.date()}'
    url = f'https://spotifycharts.com/regional/us/weekly/{date}/download'
    res = requests.get(url)
    if res.status_code == 404:
        return []
    csv_file = io.StringIO(res.content.decode('utf-8'))
    df = pd.read_csv(csv_file)
    df.columns = df.iloc[0]
    df = df[1:]
    df.reset_index(drop=True, inplace=True)
    df.Position = df.Position.astype(np.int64)
    df.Streams = df.Streams.astype(np.int64)
    df.columns = df.columns.str.strip().str.lower().str.replace(' ', '_').s
    # the above line renames first to easy accessing variables
    df.rename(columns={'track_name': 'name', 'artist': 'artists'}, inplace=
    # the above line renames columns
    return df
```

In [4]:
```
# renaming the songs to what Spotify has them listed as
main = pd.read_csv('featuresdf.csv')
main = main.drop(columns=['id', 'time_signature', 'mode', 'key', 'duration_
# drop the initial columns from Kaggle
main.at[4, 'name'] = "I'm the One (feat. Justin Bieber, Quavo, Chance the R
main.at[32, 'name'] = "Wild Thoughts (feat. Rihanna & Bryson Tiller)"
main.at[33, 'name'] = "Slide (feat. Frank Ocean & Migos)"
main.at[42, 'name'] = "Feels (feat. Pharrell Williams, Katy Perry & Big Sea
main.at[47, 'name'] = "Chantaje (feat. Maluma)"
main.at[65, 'name'] = "Don't Wanna Know"
main.at[74, 'name'] = "Cold"
main['streams'] = 0
```

In [ ]:
```
# will not work independently

# if statement: data cleaning
# len(new) != 0 ensures that get_chart returns something
#   that actually contains something
# get_chart will return [] if the url returns 404
# ~new.isna().any().any() ensures that when chart data comes
#   and NaN values are found at all, we discard the chart
# isna() checks every value in table if it's null
# .any() will see if there exists any True values in the large table.
# .any() on the returned list will return one single boolean variable.
while start != end:
    print (start)
    new = get_chart(start)
    if (len(new) != 0) & (~new.isna().any().any()):
        # gap in late may / early june, and NaN (broken)
        new = new.drop(columns=['position'])
        # unnecessary column (cleaned)
        add_plays(big_data, new)
        new = new[~(new['name'].isin(big_data['name']))]
        big_data = big_data.append(new, sort=False, ignore_index=True)
    start = start + pd.DateOffset(weeks=1)
```

In [ ]:
```
# This is data cleaning in a sense because when we combined
#   the two datasets, we will have a lot of NaN values.
# No audio features from spotifycharts, so we have to fill in
#   the audio features of each track ourselves
def get_features(url):
    features = sp.audio_features(url)
    d = features[0]['danceability']
    e = features[0]['energy']
    l = features[0]['loudness']
    s = features[0]['speechiness']
    v = features[0]['valence']
    t = features[0]['tempo']
    return [d,e,l,s,v,t]
```

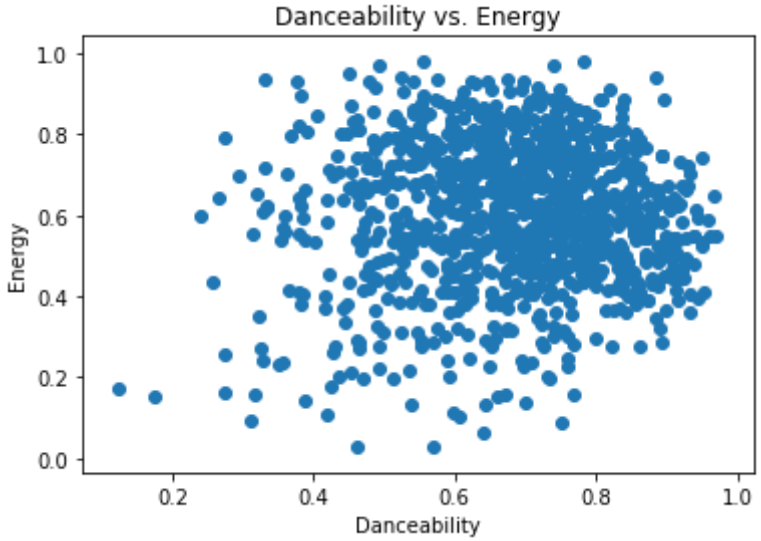
Data Analysis & Results

We were able to graph scatter plots of every musical attribute against streams, and this provided us useful visualizations of the correlations between each relevant relationship.

In [26]:
```
df = pd.read_csv('song.csv')
```

In [35]:
```
plt.title('Danceability vs. Energy')
plt.xlabel('Danceability')
plt.ylabel('Energy')
plt.scatter(df.danceability, df.energy)
```

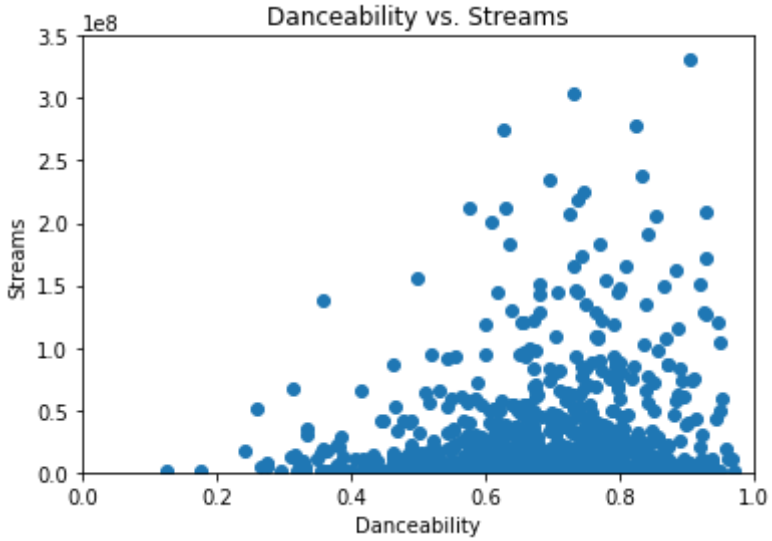
Out[35]: <matplotlib.collections.PathCollection at 0x117aa7f98>



Higher danceability can generally be correlated with high energy, where danceability is defined as how suitable a song is for dancing and energy is described as the intensity of a song's sound volumes. While higher danceability is typically associated with a steady beat, this scatterplot demonstrates the positive correlation between danceability and energy. The fluctuation and magnitude of a song's sound can also determine that same song's danceability.

In [27]:
```
plt.axis([0, 1.0, 0, 350000000])
plt.title('Danceability vs. Streams')
plt.xlabel('Danceability')
plt.ylabel('Streams')
plt.scatter(df.danceability, df.streams)
```

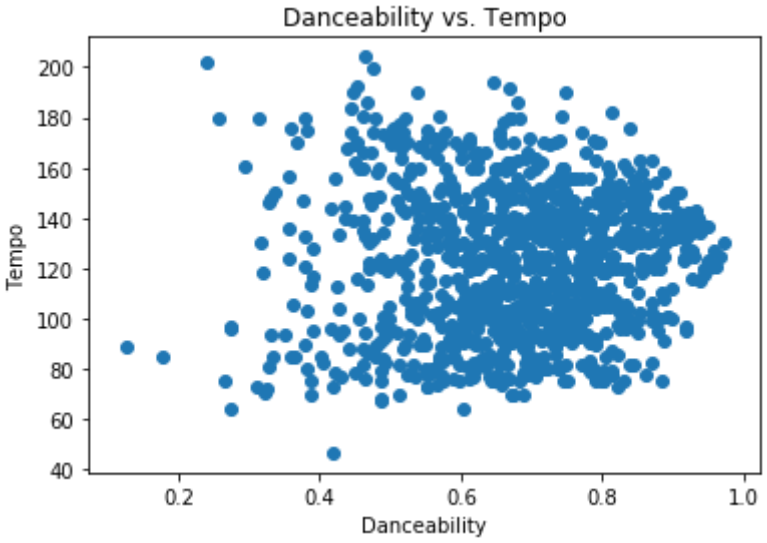
Out[27]: <matplotlib.collections.PathCollection at 0x117194128>



Danceability is the suitability of a song for dancing. This scatterplot demonstrates that the higher the danceability of a song is the more streams it gets.
From this, we saw that danceability and energy had the clearest positive correlation with streams.

In [37]:
```
plt.title('Danceability vs. Tempo')
plt.xlabel('Danceability')
plt.ylabel('Tempo')
plt.scatter(df.danceability, df.tempo)
```

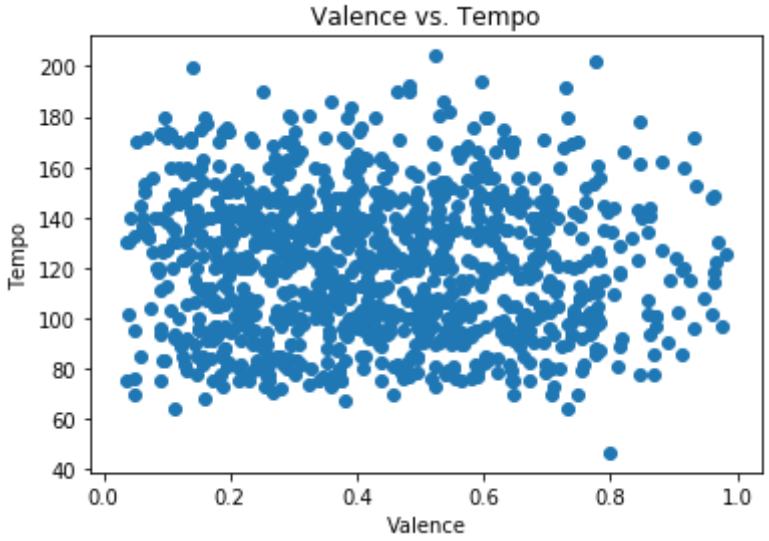
Out[37]: `<matplotlib.collections.PathCollection at 0x117b98e48>`



Higher danceability can be correlated with a certain tempo as many popular songs share similar tempos, with danceability being the suitability of a song for dancing and tempo being the beat pattern of the song. As the scatter plot demonstrates, there is a positive correlation between higher danceability and tempos between 80 and 160 beats per minute.

In [38]:
```
plt.title('Valence vs. Tempo')
plt.xlabel('Valence')
plt.ylabel('Tempo')
plt.scatter(df.valence, df.tempo)
```

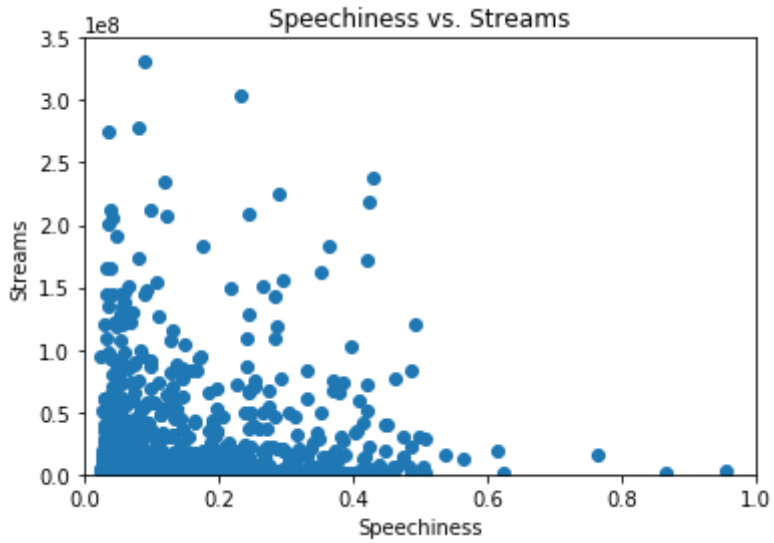
Out[38]: <matplotlib.collections.PathCollection at 0x117c89780>



Valence measures the positiveness of a song while tempo looks at the rhythm of a song. The scatterplot shows that valence and tempo doesn't have a correlation. Because valence includes both sad and happy songs, it is understandable while they generally don't have a correlation because sad songs usually don't have the same rhythm as happy songs.

In [40]:
```
plt.axis([0, 1.0, 0, 350000000])
plt.title('Speechiness vs. Streams')
plt.xlabel('Speechiness')
plt.ylabel('Streams')
plt.scatter(df.speechiness, df.streams)
```

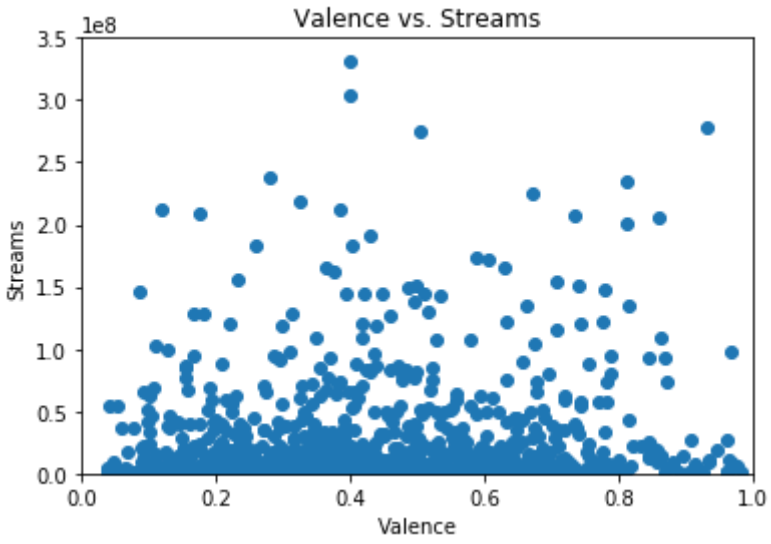
Out[40]: <matplotlib.collections.PathCollection at 0x117e207f0>



Speechiness is the presence or absence of verbal speech or communication. This scatterplot shows that there are more streams for songs with less speechiness.

In [28]:
```
plt.axis([0, 1.0, 0, 350000000])
plt.title('Valence vs. Streams')
plt.xlabel('Valence')
plt.ylabel('Streams')
plt.scatter(df.valence, df.streams)
```

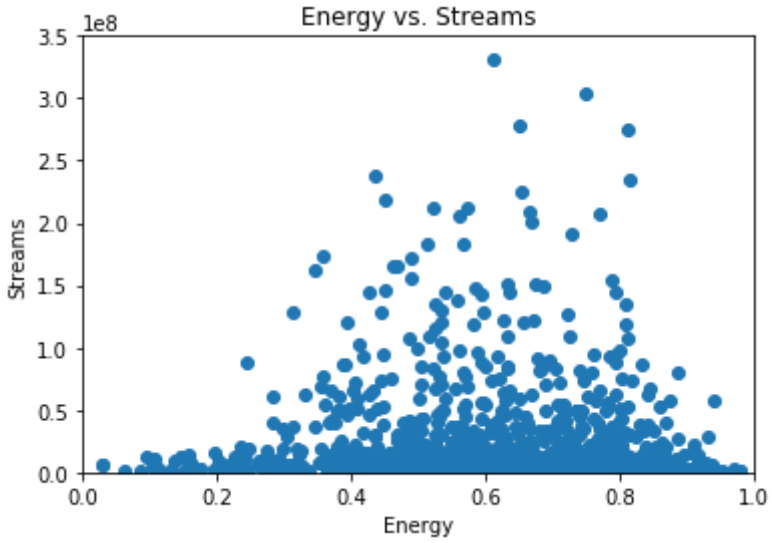
Out[28]: <matplotlib.collections.PathCollection at 0x1175555f8>



Valence is the positivity of a song. This scatterplot shows that valence does not really impact the number of streams a song gets as the points are spread out across the chart, demonstrating that both positive and negative valence songs can achieve a high number of streams.

In [30]:
```
plt.axis([0, 1.0, 0, 350000000])
plt.title('Energy vs. Streams')
plt.xlabel('Energy')
plt.ylabel('Streams')
plt.scatter(df.energy, df.streams)
```

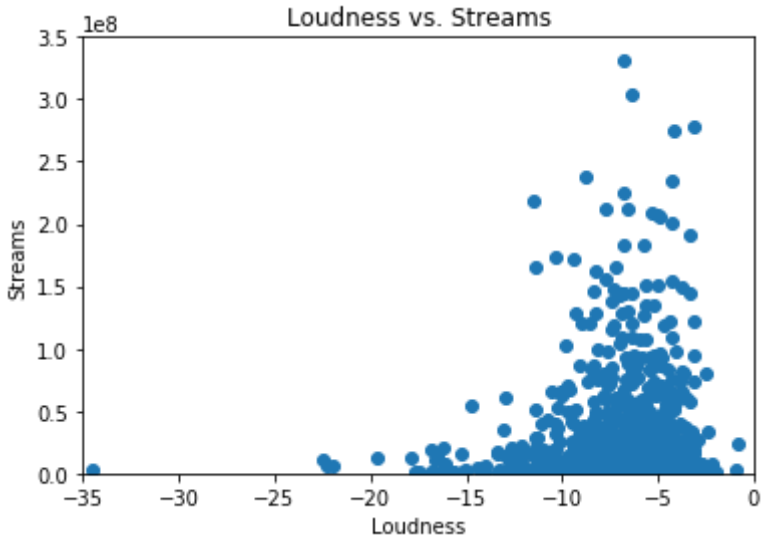
Out[30]: <matplotlib.collections.PathCollection at 0x1176e0588>



Energy is the intensity of a song's sound volume.This scatterplot demonstrates that if a song has more energy it will have more streams.

In [31]:
```
plt.axis([-35, 0, 0, 350000000])
plt.title('Loudness vs. Streams')
plt.xlabel('Loudness')
plt.ylabel('Streams')
plt.scatter(df.loudness, df.streams)
```

Out[31]: <matplotlib.collections.PathCollection at 0x117735da0>

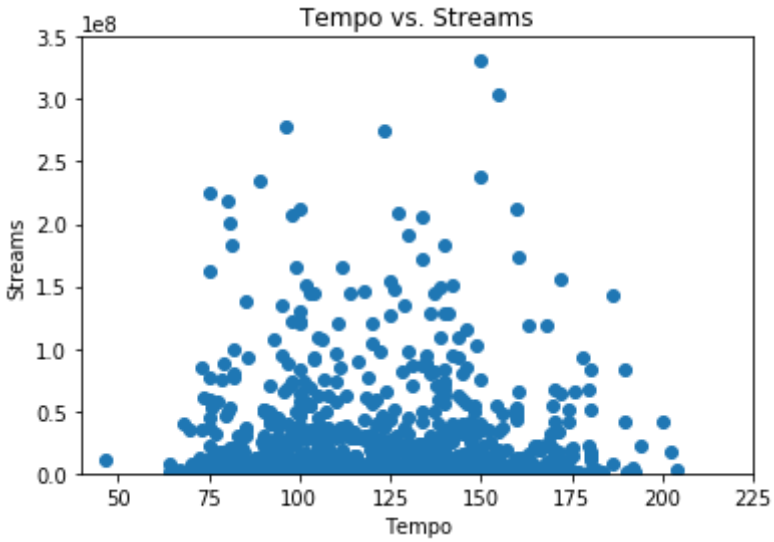


Loudness is the overall loudness of a song based on decibels. This scatterplot shows that there a preference for a certain range of loudness of between -10 decibels and -3 decibels. This range has

the most streams.

In [32]:
```
plt.axis([40, 225, 0, 350000000])
plt.title('Tempo vs. Streams')
plt.xlabel('Tempo')
plt.ylabel('Streams')
plt.scatter(df.tempo, df.streams)
```

Out[32]: <matplotlib.collections.PathCollection at 0x1177ffef0>



Tempo is the rhythm of a song. This scatterplot shows that the optimal tempo for a song is between 60 to 180 beats per minute. Anything outside this range won't garner as many streams.

We used linear regression to further understand the potential relationships between the music attributes as the explanatory variables and stream counts as the dependent. We began with single linear regression models, which demonstrate a single attribute's performance as the explanatory x along with number of streams as the dependent y, in an equation in the form of y = mx + B. When viewing the outputs of our regression functions, we were able to learn the slopes/coefficients of the explanatory variables and the p-values, allowing us to understand correlation and perform hypothesis tests on the data. Danceability and valence had high coefficients and low p-values, demonstrating high correlation and statistically significant effects on the total count of streams for a given song, respectively. Speechiness, however, had a very high p value and a relatively small coefficient comparatively, demonstrating its ambiguous relationship with number of streams of songs.

In [50]:
```
x = df.danceability
x = sm.add_constant(x)
y = df.streams
model = sm.OLS(y, x).fit()
model.summary()
```

```
/usr/local/lib/python3.7/site-packages/numpy/core/fromnumeric.py:2223: Fu
tureWarning: Method .ptp is deprecated and will be removed in a future ve
rsion. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Out[50]:

OLS Regression Results

Dep. Variable:	streams	**R-squared:**	0.022
Model:	OLS	**Adj. R-squared:**	0.021
Method:	Least Squares	**F-statistic:**	24.41
Date:	Wed, 12 Jun 2019	**Prob (F-statistic):**	9.02e-07
Time:	22:27:38	**Log-Likelihood:**	-20257.
No. Observations:	1070	**AIC:**	4.052e+04
Df Residuals:	1068	**BIC:**	4.053e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-5.618e+06	6.02e+06	-0.933	0.351	-1.74e+07	6.2e+06
danceability	4.291e+07	8.68e+06	4.941	0.000	2.59e+07	5.99e+07

Omnibus:	745.702	**Durbin-Watson:**	0.824
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	9011.724
Skew:	3.169	**Prob(JB):**	0.00
Kurtosis:	15.727	**Cond. No.**	10.3

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [51]:
```
x1 = df.speechiness
x1 = sm.add_constant(x1)
y = df.streams
model1 = sm.OLS(y, x1).fit()
model1.summary()
```

Out[51]:

OLS Regression Results

Dep. Variable:	streams	**R-squared:**	0.000
Model:	OLS	**Adj. R-squared:**	-0.001
Method:	Least Squares	**F-statistic:**	0.007058
Date:	Wed, 12 Jun 2019	**Prob (F-statistic):**	0.933
Time:	22:27:41	**Log-Likelihood:**	-20269.
No. Observations:	1070	**AIC:**	4.054e+04
Df Residuals:	1068	**BIC:**	4.055e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.34e+07	1.88e+06	12.465	0.000	1.97e+07	2.71e+07
speechiness	8.18e+05	9.74e+06	0.084	0.933	-1.83e+07	1.99e+07

Omnibus:	761.240	**Durbin-Watson:**	0.820
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	9480.693
Skew:	3.251	**Prob(JB):**	0.00
Kurtosis:	16.053	**Cond. No.**	7.96

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [22]:
```
x2 = df.valence
x2 = sm.add_constant(x2)
y = df.streams
model1 = sm.OLS(y, x2).fit()
model1.summary()
```

Out[22]:

OLS Regression Results

Dep. Variable:	streams	**R-squared:**	0.006
Model:	OLS	**Adj. R-squared:**	0.005
Method:	Least Squares	**F-statistic:**	6.054
Date:	Wed, 12 Jun 2019	**Prob (F-statistic):**	0.0140
Time:	21:29:44	**Log-Likelihood:**	-20266.
No. Observations:	1070	**AIC:**	4.054e+04
Df Residuals:	1068	**BIC:**	4.055e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.737e+07	2.79e+06	6.223	0.000	1.19e+07	2.28e+07
valence	1.42e+07	5.77e+06	2.460	0.014	2.88e+06	2.55e+07

Omnibus:	756.521	**Durbin-Watson:**	0.846
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	9329.460
Skew:	3.226	**Prob(JB):**	0.00
Kurtosis:	15.947	**Cond. No.**	5.54

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Still using a constant, we performed a multiple linear regression as well, comparing the effects of all 3 of our hypothesized effective music attributes as a collective, in the form of $y = \beta 0 + \beta 1x1 + \beta 2x2 + \cdots \beta kxk\ldots$. For this, we saw that danceability had a P-value far below 0.01, demonstrating highly statistically significant effect on streams. Valence had a p-value slightly greater than 0.01, demonstrating significant contribution to stream counts as well, while speechiness had a p-value of 0.416. Danceability and valence also sported large positive coefficients in the millions while speechiness had a negative coefficient in the millions.

In [25]:
```
df_mult = df[['valence', 'speechiness', 'danceability']].copy()
lm = linear_model.LinearRegression()
model2 = sm.OLS(y, df_mult).fit()
model2.summary()
```

Out[25]:

OLS Regression Results

Dep. Variable:	streams	**R-squared:**	0.267
Model:	OLS	**Adj. R-squared:**	0.265
Method:	Least Squares	**F-statistic:**	129.7
Date:	Wed, 12 Jun 2019	**Prob (F-statistic):**	1.29e-71
Time:	21:30:21	**Log-Likelihood:**	-20256.
No. Observations:	1070	**AIC:**	4.052e+04
Df Residuals:	1067	**BIC:**	4.053e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
valence	7.501e+06	5.66e+06	1.325	0.186	-3.61e+06	1.86e+07
speechiness	-7.957e+06	9.77e+06	-0.814	0.416	-2.71e+07	1.12e+07
danceability	3.198e+07	4.5e+06	7.100	0.000	2.31e+07	4.08e+07

Omnibus:	752.424	**Durbin-Watson:**	0.838
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	9239.019
Skew:	3.203	**Prob(JB):**	0.00
Kurtosis:	15.892	**Cond. No.**	6.85

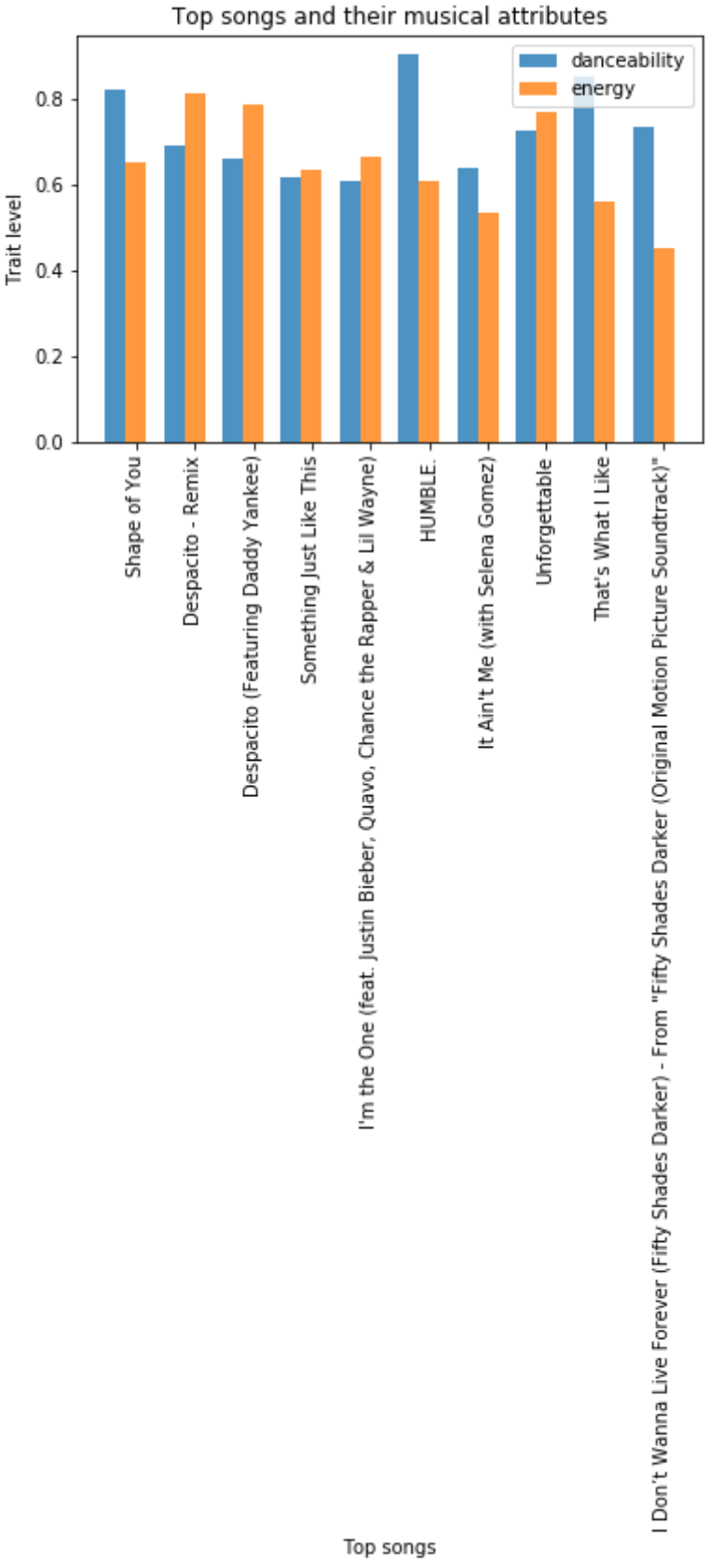
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

We feel these results demonstrate that speechiness performs oppositely when compared alongside categories that do strongly positively correlated to stream amounts, and that high amounts of danceability and valence result in high streaming, ‘hit’ songs. This is supported by the calculated p-values and coefficients.

We also chose to do bar graphs to see the relationship in these attributes.

In [43]:

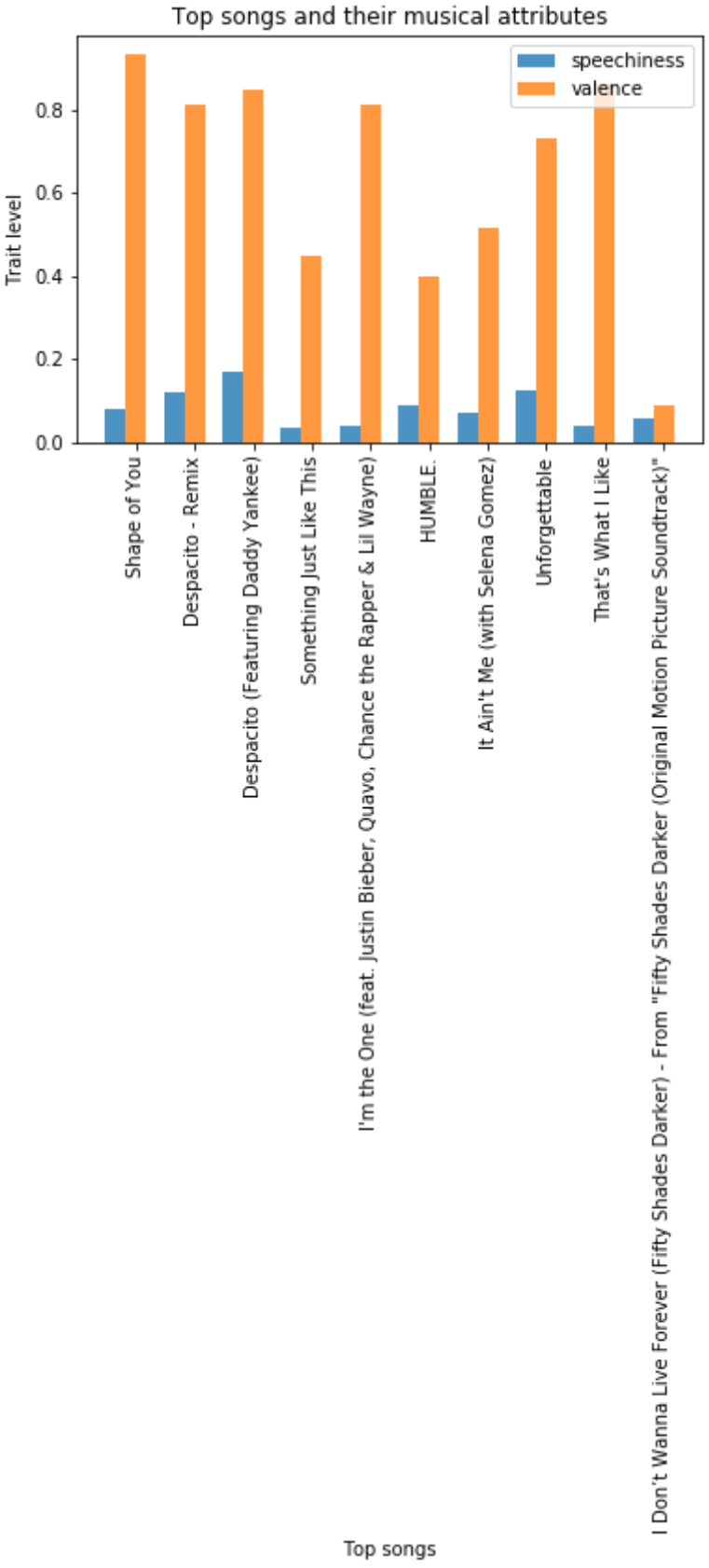
```
# data to plot
n_groups = 10

# create plot
fig, ax = plt.subplots()
index = np.arange(n_groups)
bar_width = 0.35
opacity = 0.8
rects1 = plt.bar(index, df['danceability'].head(10), bar_width,
alpha=opacity,
label='danceability')

rects2 = plt.bar(index + bar_width, df['energy'].head(10), bar_width,
alpha=opacity,
label='energy')

plt.xlabel('Top songs')
plt.ylabel('Trait level')
plt.title('Top songs and their musical attributes')
plt.xticks(index + bar_width, df['name'].head(10))
plt.legend()
plt.xticks(rotation=90)
plt.show()
```



This graph shows the relationship of the musical attribute danceability and energy with the Top 10 songs on the Spotify charts. As illustrated in the graph, the level of danceability and energy is high for most of the top songs on the Spotify charts.

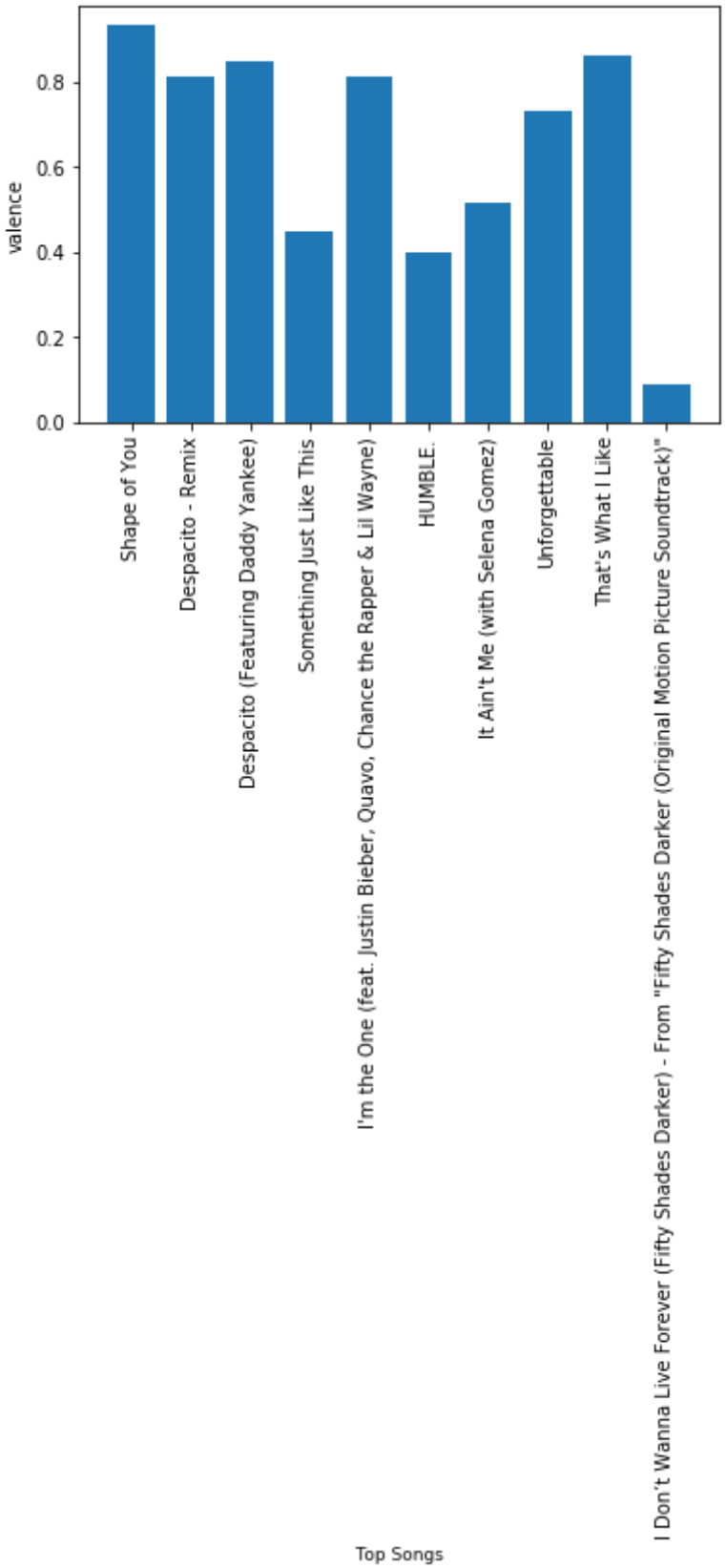
In [44]:

```
# data to plot
n_groups = 10

# create plot
fig, ax = plt.subplots()
index = np.arange(n_groups)
bar_width = 0.35
opacity = 0.8

rects1 = plt.bar(index, df['speechiness'].head(10), bar_width,
alpha=opacity,
label='speechiness')

rects2 = plt.bar(index + bar_width, df['valence'].head(10), bar_width,
alpha=opacity,
label='valence')

plt.xlabel('Top songs')
plt.ylabel('Trait level')
plt.title('Top songs and their musical attributes')
plt.xticks(index + bar_width, df['name'].head(10))
plt.legend()
plt.xticks(rotation=90)
plt.show()
```



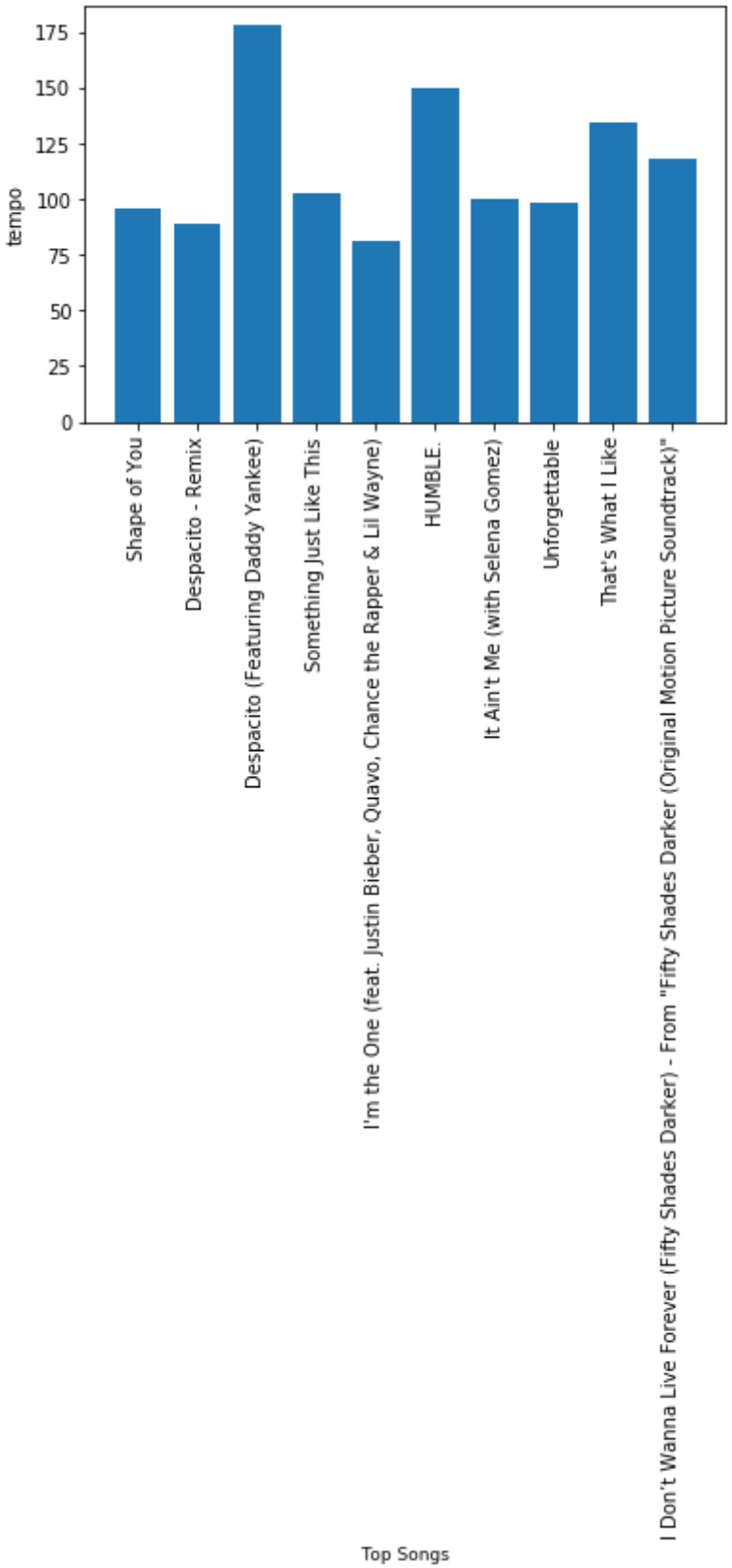
This graph shows the relationship of the musical attribute valence and speechiness with the Top 10 songs on the Spotify charts. As illustrated in the graph, the level of valence is high, however, the level of speechiness for the songs in the top charts remains low. This means that songs that are more instrumental with high valence tend to chart more than others.

In [47]:
```
plt.bar(df['name'].head(10), df['valence'].head(10))
plt.xlabel('Top Songs', fontsize=9, horizontalalignment='center')
plt.xticks(rotation=90)
plt.ylabel('valence')
plt.show()
```



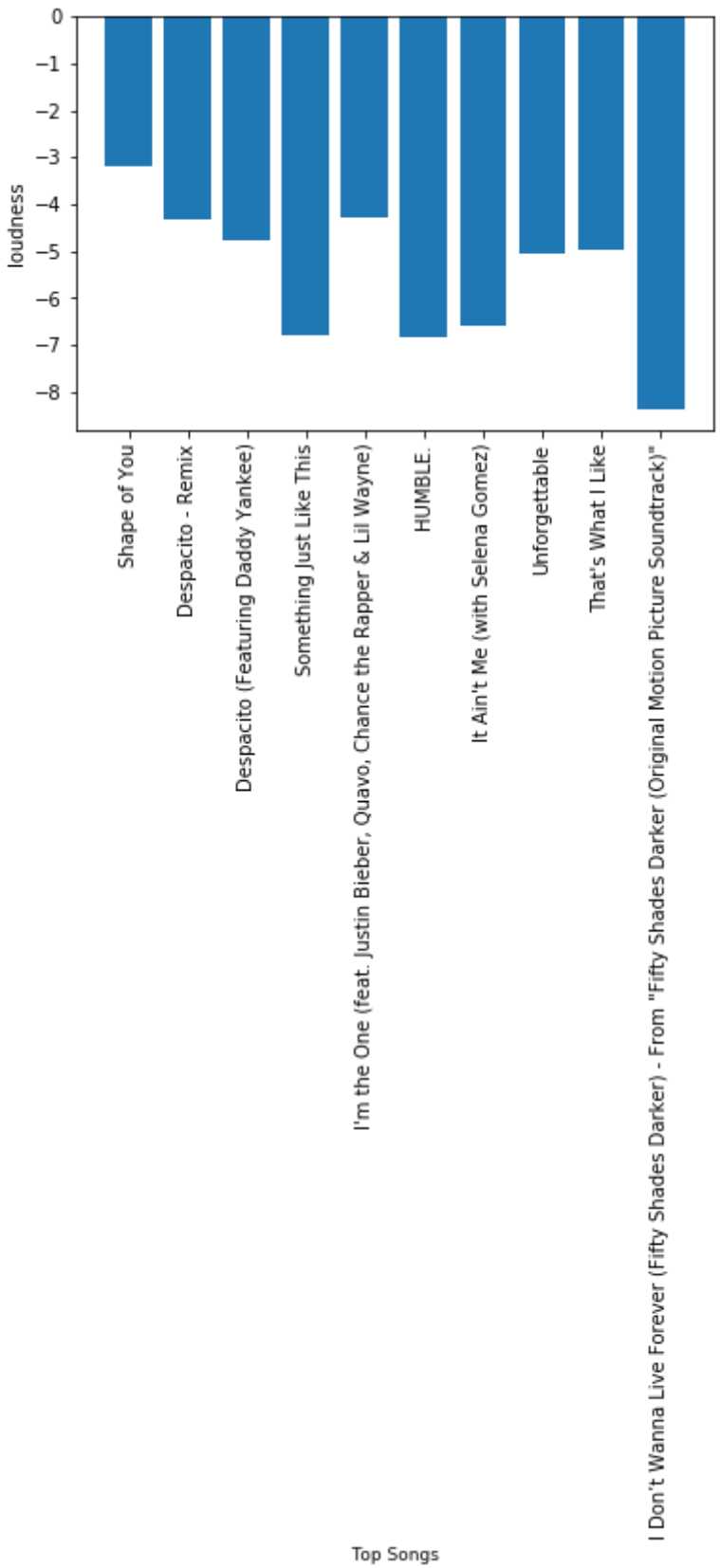
This graph shows that relationship of the musical attribute valence with the Top 10 songs on the Spotify charts. As shown in the graph, songs that chart high on Spotify have varying valence so it's difficult to have a definitive conclusion.

In [45]:
```
plt.bar(df['name'].head(10), df['tempo'].head(10))
plt.xlabel('Top Songs', fontsize=9, horizontalalignment='center')
plt.xticks(rotation=90)
plt.ylabel('tempo')
plt.show()
```



This graph shows the relationship of the musical attribute tempo with the Top 10 songs on the Spotify charts. As shown in the graph, songs that chart on Spotify have high tempo.

In [46]:
```
plt.bar(df['name'].head(10), df['loudness'].head(10))
plt.xlabel('Top Songs', fontsize=9, horizontalalignment='center')
plt.xticks(rotation=90)
plt.ylabel('loudness')
plt.show()
```



The graph shows that relationship of the musical attribute loudness with the Top 10 songs on the Spotify charts. As shown in the graph, songs that chart on Spotify have high levels of loudness.

Ethics & Privacy

We have permission to use Spotify Top 200 Chart data because it is accessible to the public. We didn't violate the terms of use in collecting the data because the terms of use are associated with how Spotify collects their information. The potential biases to our data is that it involves music and artists that are being streamed and/or associated with Spotify. Our data is only extracted from users who stream on Spotify as well. Therefore, this would exclude artists that promote exclusively on other platforms such as Tidal, Apple Music, or SoundCloud such as Jay Z. We chose to use these constraints because our dataset would be too big and therefore, unfeasible to manipulate for our project.

Additionally, since we constrained our data to only contain US streaming counts, we will only understand the relationship between musical factors in a song and it's amount of streams in the US. This prevents us from examining what factors make a song popular worldwide.

We also don't need to encrypt our data due to the fact that there is no personal information such as (email, users' names, etc.) tied to the Spotify Chart data. The data displays how often a song is streamed and doesn't include information that would reveal which users' streamed that particular song. The only information the chart data reveals is the song title, artist's name, and the number of streams per song. If an artist doesn't want their name to be associated with our dataset, we can remove all of the songs for that particular artist so that there is no reference to our name dataset.

After this project concludes, we will upload it to GitHub. There is no way to guarantee that the project can be completely removed and unavailable to the public once it has been posted online. Although there is no personal information attached to the Spotify Chart data, if an artist decides that they no longer want their song and name information associated with our dataset, we can only remove the references for future copies of the project.

Conclusion & Discussion

Based on the data set we analyzed through graph analysis and linear regression, our results confirms two of the three attributes in our hypthesis. As stated above, we hypothesize that songs with high valence, high speechiness, and high danceability songs will place in the top 10 charts of Spotify. This is true for the songs with high valence and high danceability; however, songs that place in the top 10 charts of Spotify have low speechiness. As a result, songs that do not contain a lot of lyrics tend to chart as long as their measure of valence and danceability remain high. Therefore, we conclude that songs that place in the top 10 charts have attributes that primarily express happy emotions and are danceable, however, our data set has its limitations.

One of the limitations of our project is the possible lack of variability in song attributes of popular songs. Since our data only includes the top 100 songs in the U.S. Spotify charts, the traits that make a song popular may not significantly differ across songs. This is why including an outside

factor-- such as social media activity-- can potentially narrow down what defining features influence whether or not songs gain popularity. Additionally, “one-hit wonders” act as outliers in the dataset and demonstrate a high potential of confounding variables notably affecting the ranking of a song on Spotify charts. Another key factor that could influence streaming charts could be the ongoing current events, such as the passing of an artist which could cause a song to be streamed more to celebrate the artists' work and legacy.

The key impact of this work on society is providing music producers and artists with a clearer idea of what technical characteristics of a song are linked to a greater likelihood of song popularity. Along with artists understanding what it would take to create a charting song, this project also benefits producers and others in the music industry that create songs. By breaking down the songs to their individual attributes, it is easy to determine a “recipe” that creates a hit, which producers can use for future projects. This project will also help Spotify users understand the general preferences of other users across all demographics within the United States.