1. Problem Statement

The objective of this project will be to create an iOS app to predict the popularity of a given song on Spotify. It is meant primarily for artists who want to gauge how popular their song will be before releasing it. Artists could also use the app to compare multiple mixes/remixes of the same song and decide which one to release.

2. Data Preprocessing

The dataset I will be using is called Spotify Audio Features and can be found on Kaggle: https://www.kaggle.com/tomigelo/spotify-audio-features#SpotifyAudioFeaturesNov2018.csv)

I changed directions in both the dataset and the project as I felt this dataset proposed more practical real-world applications. It contains a number of parameters for about 39 000 songs including their popularity on Spotify. These parameters include the song's id, artist and title, technical information such as the key, mode and tempo but also more characteristic information such as its 'danceability', 'accousticness', 'energy' and 'Instrumentalness'.

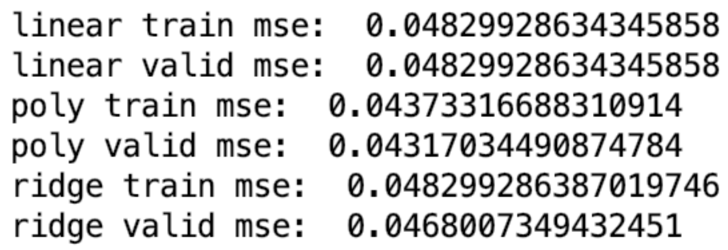
The features I will start by using are all the parameters except the track's name artist and id. Since my first objective for the app is for it to be able to predict a song's popularity with a user filling in the values for all the parameters. Since the dataset does not provide information about the artist's popularity, I will assume the artist's name is irrelevant to the popularity of a song for all intents and purposes. In the "Next Steps" section, I have highlighted how these 3 features which I have omitted could serve useful.

3. Machine Learning Model

The model I have chosen to implement my first iteration is polynomial regression. As the results in Section4 show this was the model that yielded the best results. The degree of the polynomial regression is 2. Since the dataset is large, I split it into 70% for training and 15% for each of the validation and test sets. It did not seem like my model was overfitting or underfitting since the MSE for the training set was similar to the MSE for the validation set.

To implement the polynomial regression, I used SciKit for simplicity and conciseness. Since there is little customization in polynomial regression, using the custom model we implemented in the previous assignment would not have offered many advantages.

4. Preliminary Results



```
linear train mse:  0.04829928634345858
linear valid mse:  0.04829928634345858
poly train mse:  0.04373316688310914
poly valid mse:  0.04317034490874784
ridge train mse:  0.048299286387019746
ridge valid mse:  0.0468007349432451
```

The above results show the MSE for the potential models I could use. These prove that polynomial regression yields the best results. From there, a similar procedure allowed me to determine that a degree of 2 and a 15,15,70 split are my best options. The MSE is given on the popularity score which is a percentage, it can thus be interpreted as a 4.3% error which is acceptable. Since the train and validation sets have similar MSEs we can see that this is not a case of under or over-fitting.

5. Next Steps

To make the app easier to use and for it to have more practical uses, the next step is for me to make it possible for a user to directly input a song (either a file or a recording) instead of the parameters. The app would then guess the parameters by itself and output the potential popularity.

To make this possible I would have to make use of the Spotify SDK for swift. It allows to play a song using its id. This would allow me to train a model using actual songs instead of the values of the parameters. I presume that a Neural Network would be my model of choice for such a problem. The Neural Network could either take the song as input and output the values of the parameters which would become the input of the regression which I currently have. It could also output directly the popularity of the song without the extra step of computing the parameters. I would need to test both approaches to evaluate which gives better results.

An obvious problem that remains with both approaches would be the size of my dataset (39 000 songs). A few solutions I can think of right now would be to either choose 10-15 seconds within the song, limit the size of my dataset or use PCA.