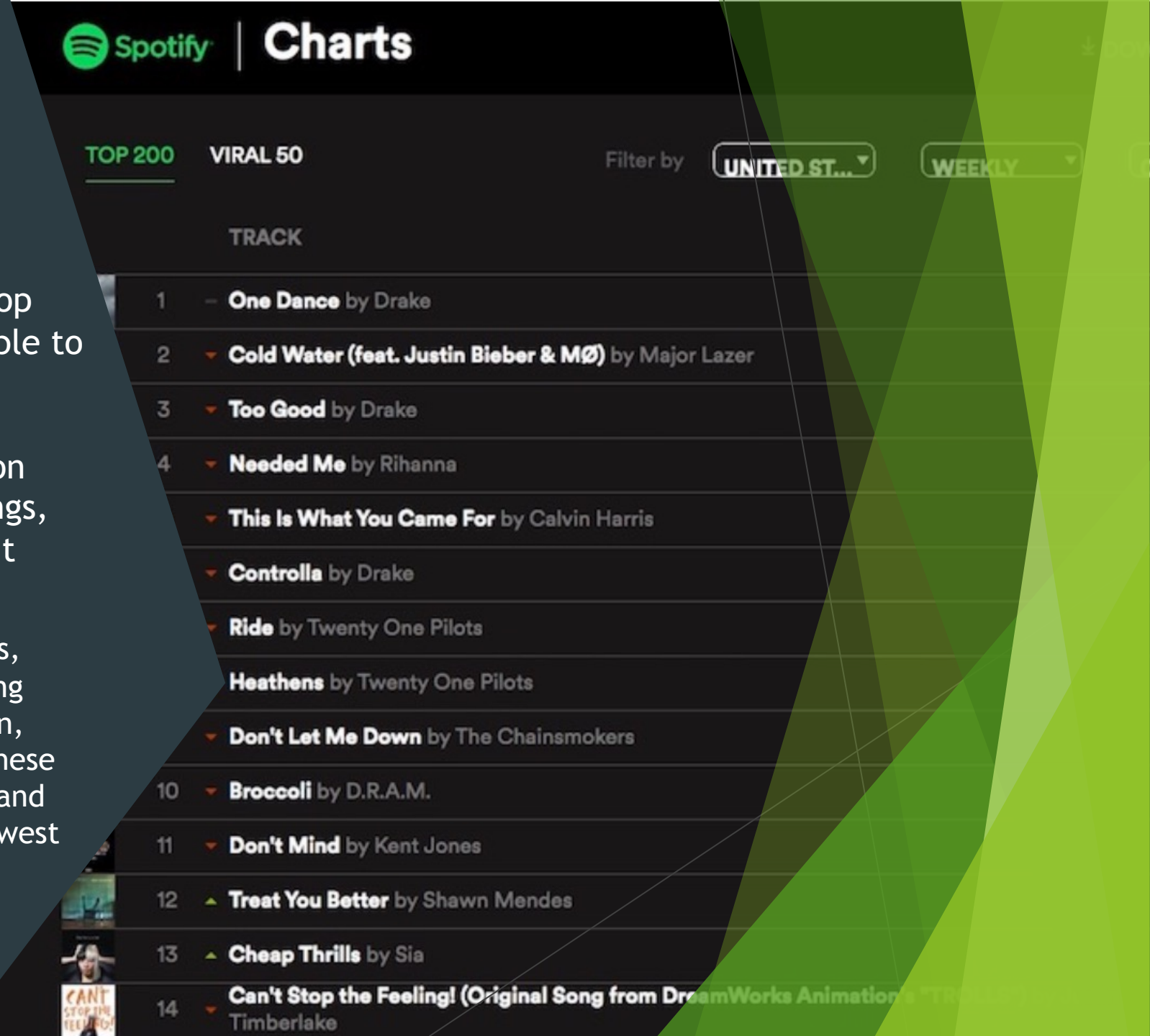
Predicting Spotify Charting Success

Hannah Hill, Jasmine Jones, Alexis Bloor

Overview

- Objectives: This project aims to develop machine learning models, which are able to predict two outcomes related to the popularity of a song on the platform Spotify: 1) whether the song in question will chart in the Spotify Top 200 rankings, and 2) if so, for how many weeks will it remain in the Top 200 chart.
  - These models will use features of songs, collected from the Spotify API, including tempo, loudness, danceability, duration, ect. to predict successful outcomes. These models could be used by music artists and producers to gauge how likely their newest records are to achieve Top 200 status.



Tasks

Retrieving and cleaning data

Sources:

- Randomly Selected Database of Spotify songs (10,000 per 26 genres) https://www.kaggle.com/zaheenhamidani/ultimate-spotify-tracks-db?select=SpotifyFeatures.csv
- Top 200 Spotify Chart 2020-2021 https://www.kaggle.com/sashankpillai/spotify-top-200-charts-20202021?select=spotify_dataset.c sv

Testing Song Predictions:

- Logistic Regression, K-Nearest Neighbors clustering, Random Forest Decision Trees

Web Deployment:

- HTML/Javascript Flask-based application that allows users to input music features and outputs the predictions of the models

Lots of songs in the 10000 per genre database were repeated in multiple genres. Additionally indexes (exa popularity) seemed to differ minorly between instances of songs. All duplicate artist + track name instance were dropped.

```
[18]
### Get columns to match exactly
print(top.columns)
db_dedup_match = db_sample.drop(columns = ['genre', 'mode', 'instrumentalness', 'time_signature']
db_dedup_match['top_200'] = 0
top_dedup_match = top_dedup.drop(columns = ['highest_charting_position', 'number_of_times_charted
                                                                                        Python
...  Index(['highest_charting_position', 'number_of_times_charted',
       'week_of_highest_charting', 'track_name', 'streams', 'artist_name',
       'artist_followers', 'track_id', 'genre_list', 'release_date',
       'weeks_charted', 'popularity', 'danceability', 'energy', 'loudness',
       'speechiness', 'acousticness', 'liveness', 'tempo', 'duration_ms',
       'valence', 'key', 'top_200'],
      dtype='object')
```

- Dropped duplicates
- Renamed column names in both csv's (top 200, top 10000 genres) to match names in both data sets
- Merged both data sets
- Binned similar keys together to avoid error

LINEAR REGRESSION

```
[178]
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train_scaled, y_train)
```

Python

```
...  LinearRegression()
```

- Errors from model testing, with Linear Regression (training and testing scores below 0.010)

```
[179]
training_score = model.score(X_train_scaled, y_train)
testing_score = model.score(X_test_scaled, y_test)


print(f"Training Score: {training_score}")
print(f"Testing Score: {testing_score}")
```

Python

```
...  Training Score: 0.091609254630768 41
     Testing Score: 0.070049374315216 52
```

Model Summary

```
summ = pd.DataFrame(
    {"models": ["Logistic Regression", "KNN", "Random Forest", "Deep Neural Net"],
     "train_accuracy": [lr_train_score, knn_train_score, rf_train_score, fit_model.history['accur
     "test_accuracy": [lr_test_score, knn_test_score, rf_test_score, model_accuracy]})

summ
```

[186] Python

...

	models	train_accuracy	test_accuracy
0	Logistic Regression	0.899172	0.898534
1	KNN	0.893529	0.869222
2	Random Forest	1.000000	0.923337
3	Deep Neural Net	0.928141	0.899662

- Random Forest model best predicts if a song will make the top 200 list on Spotify

Model Summary

```
summ = pd.DataFrame(
    {"models": ["Logistic Regression", "KNN", "Random Forest", "Deep N
     "train_accuracy": [lr_train_score, knn_train_score, rf_train_scor
     "test_accuracy": [lr_test_score, knn_test_score, rf_test_score, m

summ
```

	models	train_accuracy	test_accuracy
0	Logistic Regression	0.588083	0.624352
1	KNN	0.651986	0.645078
2	Random Forest	0.992228	0.629534
3	Deep Neural Net	0.582038	0.608808

- Random forest had the best training score, but the KNN had the best testing accuracy score
- Concluded KNN best predicts how long a long will hold its spot on the top 200 so

Spotify Charting Predictions

Use the sliders below to predict the success of your song on Spotify

Popularity

Popularity of the song, 100 being the most popular.

Enter value between: 0 and 100

Popularity

Acousticness

Acousticness refers to not using electrical amplification of musical instruments. A score of 1 is indicative of a song played entirely on acoustic instruments.

Enter value between: 0 and 1

Acousticness

Danceability

Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat