# Project Stage IV

Merging tables

## Team

Anna Chang
Sripradha Karkala
Simmi Pateriya

## Data Description

We used the songs and tracks tables from the movie-track-song dataset that was provided in the previous project stage. Each tuple in these tables describe a song entity. The original tables contained 961,593 tuples and 734,485 tuples for songs and tracks respectively. We reduced the size to 17,731 (*sample_A*) and 10,000 (*sample_B*) after downsampling.

## Merging process

The downsampled datasets (*sample_A* and *sample_B*) and a matcher table (*matches*.csv) files were generated as a result of the previous stage on data matching.
Schema of *sample_A*

| id | title | artist_name | year |
|---|---|---|---|
| 457 | Mango Tree | Angus & Julia Stone | 2006 |

Schema of *sample_B*

| id | title | year | episode | song | artists |
|---|---|---|---|---|---|
| 140954 | Lockie Leonard | 2007 | New and Improved (#2.1) | Mango Tree | angus stone+julia stone+angus |

Schema of the matcher table (*matches*)

| ltable_id | rtable_id | Ltable_title | ltable_artist_name | ltable_year | rtable_title | rtable_year | rtable_episode | rtable_song | rtable_artists | Lin. reg | Log reg | svm |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 457 | 140954 | Mango Tree | Angus & Julia Stone | 2006 | Lockie Leonard | 2007 | New and Improved (#2.1) | Mango Tree | angus stone + julia stone + angus | 1 | 1 | 1 |

Using the *ltable_id* and *rtable_id* from the matcher table, we merged the data from the downsized samples to generate a single merged table (*merged.csv*).

From the data matching stage we know that *artists_name* and *artists* from *sample_A* and *sample_B* respectively refer to the same real world entity. Similarly, *title* from *sample_A* and *song* from *sample_B* refer to the same real world entity. We pick the matcher for Logistic Regression as it has better accuracy.

The following rules were used for generating the merged table.

| Column | Table | Rule |
|---|---|---|
| id | Sample_A | song_id |
| id | Sample_B | track_id |
| title | Sample_B | title |
| artist_name, artists | Sample_A, Sample_B | Take the longer string, which includes more information on the artists. Since this is based on string length, missing values are taken care of. This column is stored as *artists* |
| title, song | Sample_A, Sample_B | Take the longer string, which includes more information about the song. Since this is based on the string length, missing values are taken care of. This column is stored as song. |
| year | Sample_A | year |

| | | |
|---|---|---|
| episode | Sample_B | episode |
| year | Sample_B | episode_year |

The merged table contains 1303 tuples. And the final schema of the merged tables looks as follows:

| song_id | track_id | title | artists | song | year | episode | episode_year |
|---|---|---|---|---|---|---|---|
| 457 | 140954 | Lockie Leonard | angus stone+julia stone+angus | Mango Tree | 2006 | New and Improved (#2.1) | 2007 |
| 1203 | 444229 | Frank Sinatra: A Man and His Music + Ella + Jobim | paul mann+stefan weiÃŒÃ™+frank sinatra | Put Your Dreams Away | 1958 | | 1967 |
| 1935 | 121170 | Intervention | scott klass+the davenports | Five Steps | 2000 | Adam (#9.2) | 2005 |
| 1935 | 121498 | Intervention | scott klass+the davenports | Five Steps | 2000 | Salina and Troy (#2.5) | 2005 |
| 4202 | 541635 | Ministry: Tapes of Wrath | ministry | The Land of Rape & Honey (Live) | 0 | | 2000 |
| 4432 | 208737 | Shameless | the high strung | The Luck You Got | 2005 | The Sins of My Caretaker (#3.5) | 2011 |

## Source code

```
import pandas as pd

'''Rules:
save id from songs table as song_id
save id from tracks table as track_id
save title from tracks table
save artists name from whichever is long
save song from tracks if there else title
from songs
save year from songs
save episode from tracks
save episode year from tracks
'''
def apply_rules(tup1, tup2):
    song_id = tup1['id'].iloc[0]
    track_id = tup2['id'].iloc[0]
    title = tup2['title'].iloc[0]
    tup2_artist = tup2['artists'].iloc[0]
    tup1_artist = tup1['artist_name'].iloc[0]
    if len(tup2_artist) >= len(tup1_artist):
       artists = tup2_artist
    else:
        artists = tup1_artist
    tup2_song = tup2['song'].iloc[0]
    tup1_song = tup1['title'].iloc[0]
    if len(str(tup2_song)) >= len(tup1_song):
        song = tup2_song
    else:
        song = tup1_song
    year = tup1['year'].iloc[0]
    episode = tup2['episode'].iloc[0]
    episode_year = tup2['year'].iloc[0]

    data = [{'song_id':song_id, 'track_id':track_id, 'title':title, \
           'artists':artists, 'song':song , \
            'year':year, 'episode':episode, 'episode_year':episode_year}]
    return data

def save_to_csv(merged):
       merged.to_csv('./merged.csv', index=False)

'''create a schema as union of both tables
For the true matches we found in Linear regression
find corresponding tables from songs and tracks table.
Apply rules to combine the common attributes.
Write the final merged table as merged.csv
```

```
'''
def merge_table(table1, table2, matcher):
    schema = ['song_id', 'track_id', 'title', 'artists', 'song', 'year',\
              'episode', 'episode_year']
    merged_table = pd.DataFrame(columns=schema)
    for index, match in matcher.iterrows():
        if match['logistic regression'] == 1:
                tup_table1 = table1.loc[table1.id == match['ltable_id']]
                tup_table2 = table2.loc[table2.id == match['rtable_id']]
                data = apply_rules(tup_table1, tup_table2)
                df = pd.DataFrame(data, columns=schema)
                merged_table = merged_table.append(df)
    save_to_csv(merged_table)

if __name__ == '__main__':
    '''Read downsampled files and matcher
    sampleA - downsampled song dataset
    sampleB - downsampled tracks dataset
    Matches - Our matcher's output
    '''
    table1 = pd.read_csv('./sampleA.csv', index_col=False)
    #if pandas version more than 0.17 use table1.sort_values(by=['id'], ascending=True)
    table1.sort('id', ascending=True, inplace=True)
    table2 = pd.read_csv('./sampleB.csv', index_col=False)
    table2.sort('id', ascending=True, inplace=True)
    matcher = pd.read_csv('./Matches.csv', index_col=False)
    matcher.sort('ltable_id', ascending=True, inplace=True)
    merge_table(table1, table2, matcher)
```