cuetessa

March 8, 2024

1 Song mood prediction using shallow models

2 Objective: Predicting valence of songs

2.0.1 What is valence?

Score used to objectively define the musical positiveness of a song, including lyrics, instruments and the audio itself.

2.0.2 What are shallow models?

Contrary to the commonly mentioned deep learning models, shallow models use lower ammount of features created from the raw data.

2.0.3 Why is this study relevant?

Valence prediciton can be used on music platforms to create playlists, suggest songs based on moods (e.g. for different moments of the day) and help in identifying similar songs.

Data from different sources were used in this study, so this could be used as a4 comparative study about data quality for training of shallow models.

3 Methodology

3.0.1 Decide what kind of data will be used

We opted to extract features from raw audio data, so valence scores can be predicted even for instrumental songs.

3.0.2 Find data on the internet

We used both the Spotify API for developers and the Dataset for Emotional Analysis of Music (DEAM).

Spotify API is widely available, it has valence and audio previews for any song on the platform.

DEAM has been used in scientific articles as a dataset for model training and it is a reliable source. Songs may not be commercial or famous.

3.0.3 Extract features

Feature extraction methods were found in the literature for raw audio data using the Librosa module. We tried to replicate methods by using:

- Chroma
- RMS Energy
- Spectral Centroid
- Rolloff
- Zero Cross Ratings
- Mel-frequency Ceptrum Coefficient

We took the means and variances of both these signals and their first order differentiation signals.

3.0.4 Analyze the data (EDA) and Train Models

```
Feature Data
x train shape:  (1196, 104)
x test shape (298, 104)
x valid shape (298, 104)

Target Data
y train shape:  (1196, 1)
y test shape (298, 1)
y valid shape (298, 1)
```

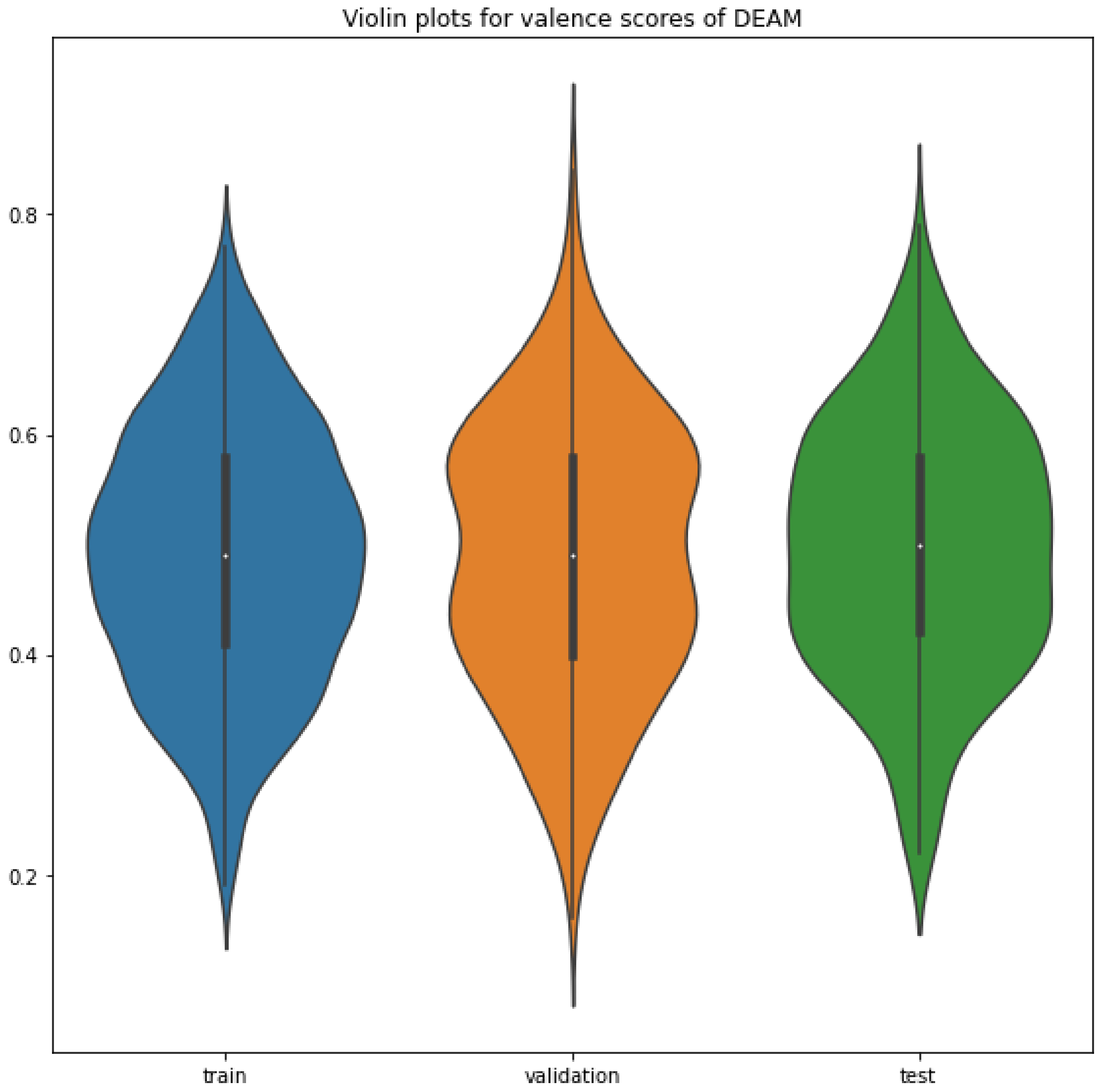
Below is a least of every feature considered for model training.

```
array(['chroma_mean', 'chroma_var', 'chroma_meandif', 'chroma_vardif',
       'rmse_mean', 'rmse_var', 'rmse_meandif', 'rmse_vardif',
       'spec_cent_mean', 'spec_cent_var', 'spec_cent_meandif',
       'spec_cent_vardif', 'spec_bw_mean', 'spec_bw_var',
       'spec_bw_meandif', 'spec_bw_vardif', 'rolloff_mean', 'rolloff_var',
       'rolloff_meandif', 'rolloff_vardif', 'zcr_mean', 'zcr_var',
       'zcr_meandif', 'zcr_vardif', 'mfcc0_mean', 'mfcc0_var',
       'mfcc0_meandif', 'mfcc0_vardif', 'mfcc1_mean', 'mfcc1_var',
       'mfcc1_meandif', 'mfcc1_vardif', 'mfcc2_mean', 'mfcc2_var',
       'mfcc2_meandif', 'mfcc2_vardif', 'mfcc3_mean', 'mfcc3_var',
       'mfcc3_meandif', 'mfcc3_vardif', 'mfcc4_mean', 'mfcc4_var',
       'mfcc4_meandif', 'mfcc4_vardif', 'mfcc5_mean', 'mfcc5_var',
       'mfcc5_meandif', 'mfcc5_vardif', 'mfcc6_mean', 'mfcc6_var',
       'mfcc6_meandif', 'mfcc6_vardif', 'mfcc7_mean', 'mfcc7_var',
       'mfcc7_meandif', 'mfcc7_vardif', 'mfcc8_mean', 'mfcc8_var',
       'mfcc8_meandif', 'mfcc8_vardif', 'mfcc9_mean', 'mfcc9_var',
       'mfcc9_meandif', 'mfcc9_vardif', 'mfcc10_mean', 'mfcc10_var',
       'mfcc10_meandif', 'mfcc10_vardif', 'mfcc11_mean', 'mfcc11_var',
       'mfcc11_meandif', 'mfcc11_vardif', 'mfcc12_mean', 'mfcc12_var',
       'mfcc12_meandif', 'mfcc12_vardif', 'mfcc13_mean', 'mfcc13_var',
       'mfcc13_meandif', 'mfcc13_vardif', 'mfcc14_mean', 'mfcc14_var',
       'mfcc14_meandif', 'mfcc14_vardif', 'mfcc15_mean', 'mfcc15_var',
       'mfcc15_meandif', 'mfcc15_vardif', 'mfcc16_mean', 'mfcc16_var',
       'mfcc16_meandif', 'mfcc16_vardif', 'mfcc17_mean', 'mfcc17_var',
       'mfcc17_meandif', 'mfcc17_vardif', 'mfcc18_mean', 'mfcc18_var',
       'mfcc18_meandif', 'mfcc18_vardif', 'mfcc19_mean', 'mfcc19_var',
       'mfcc19_meandif', 'mfcc19_vardif'], dtype=object)
```

These are the Pearon correlation values for every feature.

```
mfcc0_mean        0.564614
rolloff_mean      0.547272
spec_cent_mean    0.524815
spec_bw_mean      0.509111
chroma_vardif     0.411423
zcr_mean          0.384039
mfcc13_mean       0.363900
mfcc7_mean        0.355339
mfcc9_mean        0.349365
mfcc16_vardif     0.343538
mfcc15_mean       0.334761
mfcc17_vardif     0.332201
mfcc14_vardif     0.330344
mfcc17_mean       0.327830
mfcc15_vardif     0.326495
mfcc11_mean       0.321137
mfcc18_vardif     0.320691
mfcc13_vardif     0.312322
mfcc8_vardif      0.304462
mfcc12_vardif     0.302653
Name: valence_mean, dtype: float64
```

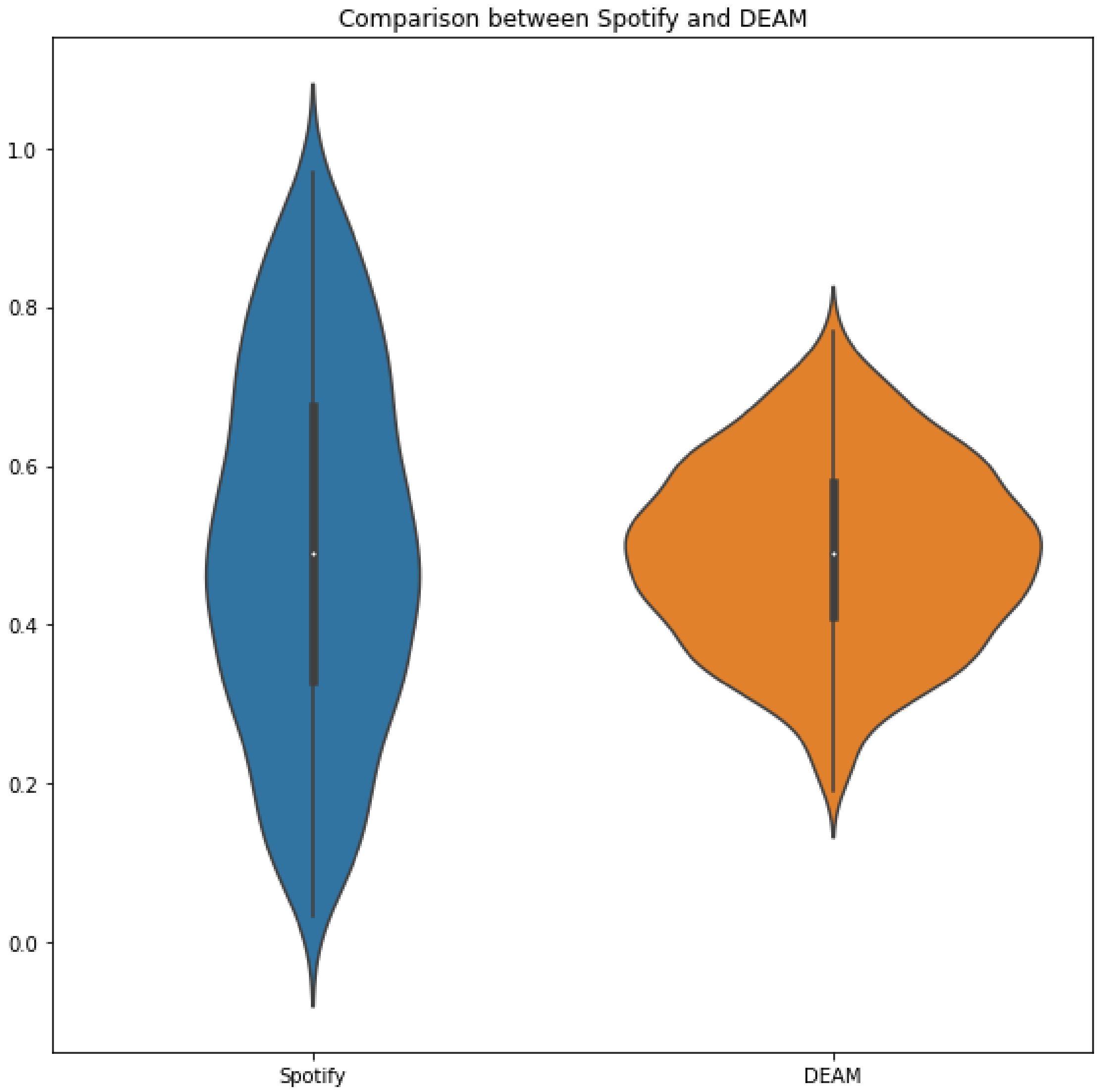
The graph below shows that data has been correctly distributed between training, testing and validation datasets.



```
Feature Data
x train shape:  (1084, 156)
x test shape (272, 156)
x valid shape (270, 156)

Target Data
y train shape:  (1084, 1)
y test shape (272, 1)
y valid shape (270, 1)
```

Since valence is prone to subjective error, it is valid to analyze distributions of both datasets. The probability distributions for both datasets are very different from each other. Spotify is reportedly considered less reliable.



We will try several models for machine learning training.

Model 0: Linear Regression Model

Model_0b: Linear Regression Model, Normalized Features

Model 1: XGBoost Regressor

Model 1b: XGboost Regressor, Normalized Features

Model 2: Support Vector Regressor

Model 2b: Support Vector Regressor, Normalized Data

Model 3: K-Nearest Neighbors Regressor

Model 3b: K-Nearest Neighbors Regressor, Normalized Data

Model 4: CatBoost Regressor

Model 4b: CatboostRegressor, Normalized Data

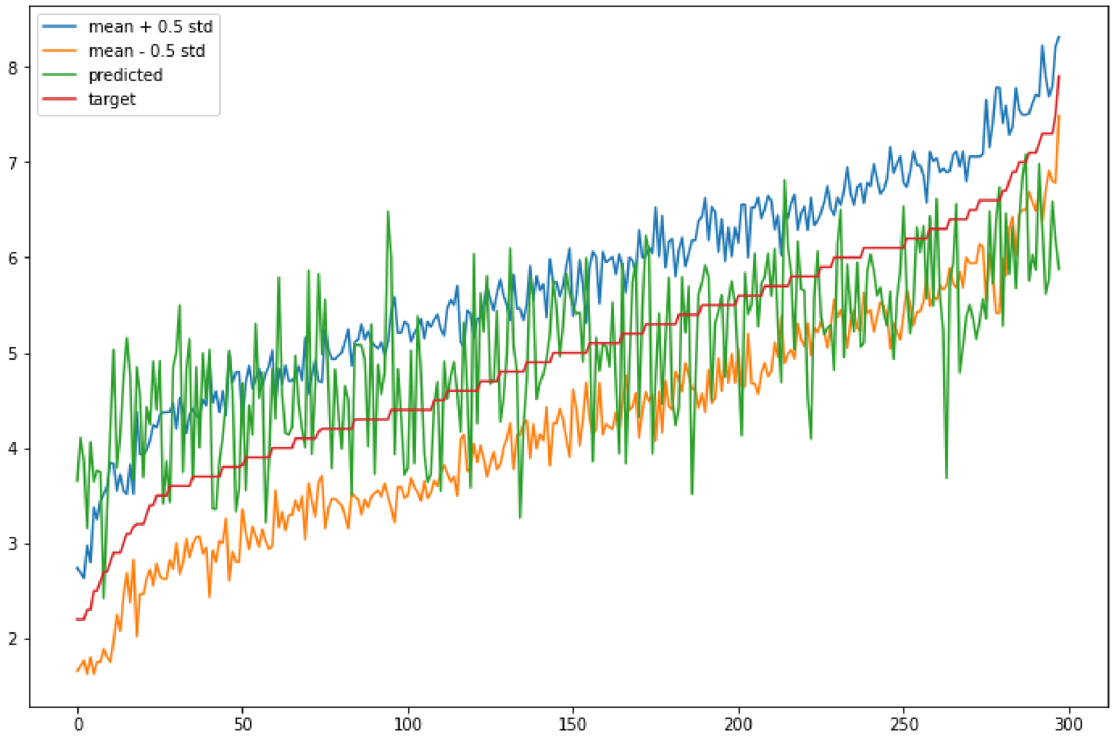
3.0.5 Performance Results and Selection of best Model

```
                                  model_description  \
8                                catboost regressor   
9          catboost regressor, normalized features   
0                                 linear regression   
1           linear regression, normalized features   
2                                 XGBoost regressor   
3           XGBoost regressor, normalized features   
7  k-nearest neighbors regressor, normalized feat…   
6                     k-nearest neighbors regressor   
4                          support vector regressor   
5    support vector regressor, normalized features   

                                              model      rmse       mae  
8  <catboost.core.CatBoostRegressor object at 0x7…  0.083495  0.068551  
9  <catboost.core.CatBoostRegressor object at 0x7…  0.083874  0.067704  
0                                LinearRegression()  0.087565  0.070914  
1                                LinearRegression()  0.087565  0.070914  
2                         XGBRegressor(seed=12345)  0.088296   0.07207  
3                         XGBRegressor(seed=12345)  0.089524   0.07306  
7                            KNeighborsRegressor()  0.091061  0.072954  
6                            KNeighborsRegressor()  0.107722  0.089935  
4                                             SVR()  0.122508   0.10229  
5                                             SVR()  0.122508   0.10229  
```

Prediction made by our final model:

```
<matplotlib.legend.Legend at 0x7fee6d2e1640>
```



Although the final predicted result of valence is not satisfactory when compared against the target, it is important to further consider the circumstances of this test. In fact, the audio files for the annotated dataset were not readily available, and spotify data was considered instead. Unfortunately, since the spotify data does not have a reported method of defining valence values, it is possible that they were using a model to assign valence values to their songs.

Unfortunately, considering the flat distribution of valence values of spotify when compared with the annotated dataset, this suggests that our training data is not reliable as a source of information. I.E. if the training dataset has random values assigned to each song, then the model is more likely to behave randomly.

Therefore, the main findings in this project are:

- Spotify data was not a reliable source of information for valence reference.
- Human annotated datasets are extremely important for audio related emotion trainig.
- Valence values tend to behave as a normal distribution.