# Project Ventoux, part 1: applying machine learning techniques to Beatport and Spotify data

For the last few months (March 2024 to now), I've been teaching myself python. As part of this process, I've been spending a lot of time with a Beatport and Spotify data setBeatport and Spotify music data set I found on Kaggle's dataset repository. In addition, I've supplmented with other data from around the web.

For part 1, I'm using a dataset I made that marries together Beatport chart rankingsa with Spotify and Beatport track data. Overall, I have Beatport track data for around ten million songs. When I filter for just the top songs overall and by genre (ranging from 500 to 1,300)tracks, depending on genre), it drops down to around 95,000.

For more detailed information on why I chose this project and how I approached it, please read this project overview on my website.

Questions? HMU at hello@uwsthoughts.com

## Some common ways I import and use data

Depends on where I am and what I have going on, I use any one of these to import and organize data. In a separate workflow, I keep files between the three sources synced.

### These always get called, regardless of underlying data source

In [3]:
```
#I know best practice is to use tian for pandas but I chose the name of a fa
import pandas as tian #named after a DC zoo panda
tian.set_option('display.max_columns', None) #invoking pandas setting I alwa

#I also diverged from numpy as lumpnump because thismakes me laugh and still
import numpy as lumpnump
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
from sklearn.metrics import root_mean_squared_error, mean_absolute_error, r2
from sklearn.model_selection import train_test_split
import copy
from scipy.stats import norm
import math
import os
```

### These are run when I use my Google Cloud Platform instance

In [4]:
```
#in case there are issues with using env variables
# !pip install certifi
# import os
# import certifi

# os.environ['SSL_CERT_FILE'] = certifi.where()

#for using gcp
# from google.cloud import storage as gcs
# from google.oauth2 import service_account as gsa
# from io import BytesIO
# import gcsfs
# from dotenv import load_dotenv

# load_dotenv()

# # System variables
# google_client_id = os.getenv('GOOGLE_CLIENT_ID')
# google_client_secret = os.getenv('GOOGLE_CLIENT_SECRET')
# google_project_id = os.getenv('GOOGLE_MUSIC_PROJECT')
# google_bucket = os.getenv('GOOGLE_PRIMARY_BUCKET')
# os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = os.getenv('GOOGLE_SERVICE_C

# # GCS setup details
# bucket_file_base = f"gs://{google_bucket}/"
# gcp_storage = gcs.Client()
# gcp_primary_bucket = gcp_storage.bucket(google_bucket)
# gcp_file_system = gcsfs.GCSFileSystem()

# gcp file example
# fact_bp_track_audio_audio_features_csv = f"gs://{google_bucket}/fact_bp_tr
```

## Importing files from my local computer

I've obfuscated most of the file path. The file names in the Notebook match file names in the Github repository.

In [5]:
```
top_tracks_stem = os.getenv('TOP_TRACKS_STEM')

toptracks_bpmeta_matrix_csv = f'{top_tracks_stem}/toptracks_bpmeta_matrix'
toptracks_bpmeta_matrix_df_csv = f'{top_tracks_stem}/toptracks_bpmeta_matrix

toptracks_bpmeta_matrix_df = tian.read_csv(toptracks_bpmeta_matrix_df_csv)
toptracks_bpmeta_matrix = lumpnump.genfromtxt(toptracks_bpmeta_matrix_csv, d
```

## for the music heads - the most controversial decision I made in the data prep phase

I had to do a lot of work around cleaning up "mix" because the values were inconsistent. This is prep work for a future phase and I didn't do anything with it here.

In [6]:
```
mix_name_changes = [
    ("Continuous DJ Mix", "Mixed"),
    ("Mix Cut", "Set Mixed"),
    ("Mixed", "Set Mixed"),
    ("Intro Mix", "Set Mixed"),
    ("Edit", "Radio Edit"),
    ("Main Mix", "Original Mix"),
    ("Album Version", "Album Mix"),
    ("Deep Mix", "Remix"),
    ("House Mix", "Remix"),
    ("Tribal Mix", "Remix"),
    ("Intro", "Set Mixed"),
    ("Edit Mix", "Radio Mix"),
    ("Bonus Track", "Album Mix"),
    ("continuous", "Set Mixed"),
    ("live", "Set Mixed"),
    ("remastered", "Remastered Mix"),
    ("orginal", "Original Mix"),
    ("ambient", "Ambient Mix"),
    ("chill", "Ambient Mix"),
    ("lounge", "Ambient Mix"),
    ("rework", "Remastered Mix"),
    ("remix", "Remix"),
    ("original", "Original Mix"),
    ("club", "Club Mix"),
    ("dub", "Dub Mix"),
    ("extended", "Extended Mix"),
    ("instrumental", "Instrumental Mix"),
    ("radio", "Radio Mix"),
    ("vip", "Remix"),
    ("album", "Album Mix"),
]

# for current_mix_value, utianated_mix_value in mix_name_changes:
#     fact_bpmeta_audio_df['mix'] = fact_bpmeta_audio_df['mix'].apply(lambda
```

# Defining cost, gradient, and gradient descent functions

## Cost function

In [7]:
```
def single_var_cost_function(x, y, w, b, lambdar):
    sample_size_m = x.shape[0]
    cost = 0.
    for i in range(sample_size_m):
        f_wb_i = lumpnump.dot(x[i], w) + b
        cost += (f_wb_i - y[i])**2
    cost = cost / (2 * sample_size_m)

    regularized_cost = 0
    for j in range(len(w)):
        regularized_cost += (w[j]**2)
```

```
    regularized_cost = (lambdar / (2 * sample_size_m)) * regularized_cost
    single_var_total_cost = cost + regularized_cost

    return single_var_total_cost
```

## Gradient function

In [8]:
```
def single_var_compute_gradient(x, y, w, b, lambdar):
    sample_size_m, n = x.shape
    single_var_djdw = lumpnump.zeros((n,))
    single_var_djdb = 0.
    for i in range(sample_size_m):
        cost_error = (lumpnump.dot(x[i], w) + b) - y[i]
        for j in range(n):
            single_var_djdw[j] += cost_error * x[i, j]
        single_var_djdb += cost_error
    single_var_djdw = single_var_djdw / sample_size_m
    single_var_djdb = single_var_djdb / sample_size_m
    for j in range(n):
        single_var_djdw[j] = single_var_djdw[j] + (lambdar / sample_size_m)

    return single_var_djdw, single_var_djdb
```

## Gradient descent

In [9]:
```
def single_var_gradient_descent(x, y, initial_w, initial_b, single_var_cost_
    sample_size_m = x.shape[0]
    j_history = []
    w_history = []
    w = copy.deepcopy(initial_w)
    b = initial_b

    for iteration in range(iterations):
        single_var_djdw, single_var_djdb = single_var_compute_gradient(x, y,
        w = w - learning_rate * single_var_djdw
        b = b - learning_rate * single_var_djdb
        if iteration < 100000:
            single_var_cost = single_var_cost_function(x, y, w, b, lambdar)
            j_history.append(single_var_cost)
            w_history.append(w)
        # if iteration % math.ceil(iterations / 10) == 0:
        #     print(f"Iteration {iteration:4}: Cost {float(j_history[-1]):8.

    return w, b, j_history, w_history
```

# Normalization functions

## z-score normalization + graping function

I didn't change these z-score functions too much after I plucked them out of some deeplearning.ai workbooks I had. I wanted to use them the way I had been taught but, in

the moment, didn't remember excactly how to code it.

In [10]:
```
def zscore_normalize(x):
    feature_mean = lumpnump.mean(x, axis=0)
    standard_deviation = lumpnump.std(x, axis=0)
    x_normalized = (x - feature_mean) / standard_deviation
    return x_normalized, feature_mean, standard_deviation

def norm_plot(ax, data):
    scale = (lumpnump.max(data) - lumpnump.min(data)) * 0.2
    x = lumpnump.linspace(lumpnump.min(data) - scale, lumpnump.max(data) + s
    _, bins, _ = ax.hist(data, bins=50, color="xkcd:azure", alpha=0.7)

    mu = lumpnump.mean(data)
    std = lumpnump.std(data)
    dist = norm.pdf(bins, loc=mu, scale=std)

    axr = ax.twinx()
    axr.plot(bins, dist, color="orangered", lw=2)
    axr.set_ylim(bottom=0)
    axr.axis('off')
```

## mean normalization + graphing function

In [11]:
```
def mean_normalization(x):
    feature_mean = lumpnump.mean(x, axis=0)
    feature_max = lumpnump.max(x)
    feature_min = lumpnump.min(x)
    x_mean_normalized = (x - feature_mean) / (feature_max - feature_min)
    return x_mean_normalized

#plots before and after in one swing
def mean_plot(data, title):
    normalized_data = mean_normalization(data)
    mean_values = lumpnump.mean(data, axis=0)

    plt.figure(figsize=(9, 4))

    # Before normalization
    plt.subplot(1, 2, 1)
    plt.plot(data)
    plt.plot([], [], label='Original Data')
    if lumpnump.ndim(mean_values) == 0:
        plt.axhline(y=mean_values, color='b', linestyle='--', label='Mean')
    else:
        for value in mean_values:
            plt.axhline(y=value, color='b', linestyle='--', label='Mean')
    plt.title(f'Before: {title}')
    plt.legend()

    # After normalization
    plt.subplot(1, 2, 2)
    plt.plot(normalized_data, color='green')
    plt.plot([], [], label='Normalized Data', color='green')
```

```
    plt.axhline(y=0, color='b', linestyle='--', label='Mean Normalized')
    plt.title(f'After: {title}')
    plt.legend()

    plt.tight_layout()
    plt.show()
```

## Log normalization function + graphing

In [12]:
```
def log_normalization(x):
    x_log = lumpnump.log(x + 1)
    return x_log

def log_plot(data, title):
    log_data = log_normalization(data)
    mean_values = lumpnump.mean(data)

    plt.figure(figsize=(9, 4))

    # Before normalization
    plt.subplot(1, 2, 1)
    plt.plot(data, label='Original Data')
    if lumpnump.ndim(mean_values) == 0:
        plt.axhline(y=mean_values, color='b', linestyle='--', label='Mean')
    else:
        for value in mean_values:
            plt.axhline(y=value, color='b', linestyle='--', label='Mean')
    plt.title(f'Before: {title}')
    plt.legend()
    plt.xlabel('Index')
    plt.ylabel('Value')

    # After normalization
    plt.subplot(1, 2, 2)
    plt.plot(log_data, color='green', label='Log Normalized')
    log_mean = lumpnump.mean(log_data)
    if lumpnump.ndim(log_mean) == 0:
        plt.axhline(y=log_mean, color='b', linestyle='--', label='Log Normal
    else:
        for value in log_mean:
            plt.axhline(y=value, color='b', linestyle='--', label='Log Norma
    plt.title(f'After: {title}')
    plt.legend()
    plt.xlabel('Index')
    plt.ylabel('Log Value')

    plt.tight_layout()
    plt.show()
```

# Preparing data for regression

## defining train/validate/split function

In [13]:
```
def training_split(data, train_size, validate_size, test_size):
    lumpnump.random.shuffle(data)
    x_train, x_temp = train_test_split(data, test_size=(validate_size + test
    x_validate, x_test = train_test_split(x_temp, test_size=test_size/(valid

    return x_train, x_validate, x_test
```

## Creating dictionary that maps field names to matrix indexes so it's easier to keep track of

In [14]:
```
toptracks_bpmeta_matrix_df_columns = toptracks_bpmeta_matrix_df.columns

toptracks_bpmeta_matrix_df_dict = {name: idx for idx, name in enumerate(topt
toptracks_bpmeta_matrix_df_dict
```

Out[14]:
```
{'isrc_numeric': 0,
 'release_id': 1,
 'release_date': 2,
 'release_year': 3,
 'release_month': 4,
 'label_id': 5,
 'track_id': 6,
 'genre_id_x': 7,
 'mix_id': 8,
 'is_remixed': 9,
 'duration': 10,
 'duration_ms': 11,
 'bpm': 12,
 'key_id': 13,
 'time_signature': 14,
 'mode': 15,
 'valence': 16,
 'danceability': 17,
 'energy': 18,
 'speechiness': 19,
 'loudness': 20,
 'liveness': 21,
 'instrumentalness': 22,
 'acousticness': 23,
 'year': 24,
 'genre_id_y': 25,
 'rank': 26,
 'beatport_track_id': 27,
 'points': 28}
```

## Establishing a subset of genres and labels to focus on

The labels and genres chosen were picked because they represent pretty different styles of music and should have different characteristics. Will they actually? We will soon

find out.

There are only around 35 total Beatport genres so expanding out to cover all of them for the next piece of work would be easy enough. However, there are over 78,000 labels in the beatport data and....I'm not graphing all 78,000. I picked 4 labels that effectively showcase all three genres. I have beef with melodic house and melodic techno being one genre so I picked Afterlife to cover melodic techno and Anjunadeep to cover melodic house.

## The label ID/name mappings were messy

I checked on the Beatport website and determined:

- The only Afterlife one that actually belongs to the famous melodic techno label is 56958. The others appear random and unrelated.
- all day I dream in waves doesn't appear related to All Day I Dream, the famous daytime party label
- Anjunadeep only has one ID, 1390. ID 804, Anjunabeats, is their sister trance label. Doesn't look like 42679, Anjuna Records, is related.

[12]:
```
label_ids_df['label_name'] = label_ids_df['label_name'].fillna('')
label_ids_df[label_ids_df['label_name'].str.contains('afterlife|anjuna|drumcode|all day I dream', case=False, regex=True)]
```

[12]:

| | label_id | label_name | label_url | updated_on |
|---|---|---|---|---|
| **177** | 1390 | Anjunadeep | beatport.com/label/anjunadeep/1390 | 2023-09-14 16:19:58 |
| **334** | 804 | Anjunabeats | beatport.com/label/anjunabeats/804 | 2023-09-14 16:20:00 |
| **14924** | 49754 | Afterlife Recordings | beatport.com/label/afterlife-recordings/49754 | 2023-09-14 20:27:31 |
| **17186** | 23038 | All Day I Dream | beatport.com/label/all-day-i-dream/23038 | 2023-09-14 21:47:02 |
| **17528** | 2027 | Drumcode | beatport.com/label/drumcode/2027 | 2023-09-14 22:26:35 |
| **21211** | 56958 | Afterlife Records | beatport.com/label/afterlife-records/56958 | 2023-09-15 00:12:44 |
| **36309** | 99284 | all day I dream in waves | beatport.com/label/all-day-i-dream-in-waves/99284 | 2023-09-15 10:10:58 |
| **41114** | 24577 | Afterlife | beatport.com/label/afterlife/24577 | 2023-09-15 14:23:08 |
| **42819** | 42679 | Anjuna Records | beatport.com/label/anjuna-records/42679 | 2023-09-15 16:10:56 |
| **53786** | 49510 | Afterlife Recordings LLC | beatport.com/label/afterlife-recordings-llc/49510 | 2023-09-16 04:24:35 |
| **57554** | 113538 | Anymax Limited/Afterlife/Interscope Records | beatport.com/label/anymax-limitedafterlifeinte... | 2023-09-16 09:56:23 |

## maping out the genre and label IDs I want to their names

This will make it easier to keep track of later

In [15]:
```
subset_genres = [6, 90, 93]
subset_genres_dict = {
    6: "Techno (Peak Time/Driving)",
    90: "Melodic House & Techno",
    93: "Organic House / Downtempo"
}

subset_labels = [1390, 23038, 2027, 56958]


label_ids_dict = {
    1390:'Ajunadeep',
    23038: 'All Day I Dream',
```

```
    2027: 'Drumcode',
    56958: 'Afterlife Records'
}

#core is just I know these IDs should remain stable and can be used with the
core_x_features = [16, 17, 18, 19, 20, 21, 22, 23]
core_y_target = 28
```

## Creating train, validate, and test data sets

I want to do work with all three types of nornmalization so I'm creating three copies of the dataset to make things easier to track.

In [16]:
```
#creating copies of matrix for each type of normalization
zscore_toptracks_matrix = toptracks_bpmeta_matrix
mean_normalized_toptracks_matrix = toptracks_bpmeta_matrix
log_nornmalized_toptracks_matrix = toptracks_bpmeta_matrix

#initial ratios to try
train_ratio = 0.6
validate_ratio = 0.2
test_ratio = 0.2

zscore_train, zscore_validate, zscore_test = training_split(zscore_toptracks
mean_train, mean_validate, mean_test = training_split(mean_normalized_toptra
log_train, log_validate, log_test = training_split(log_nornmalized_toptracks
```

# Scatter plots of selected genres and labels for different metrics against points

The feature_ids are based on the dictionary I created above. In the loop is a piece that grabs the human-readable names from the dictionary.

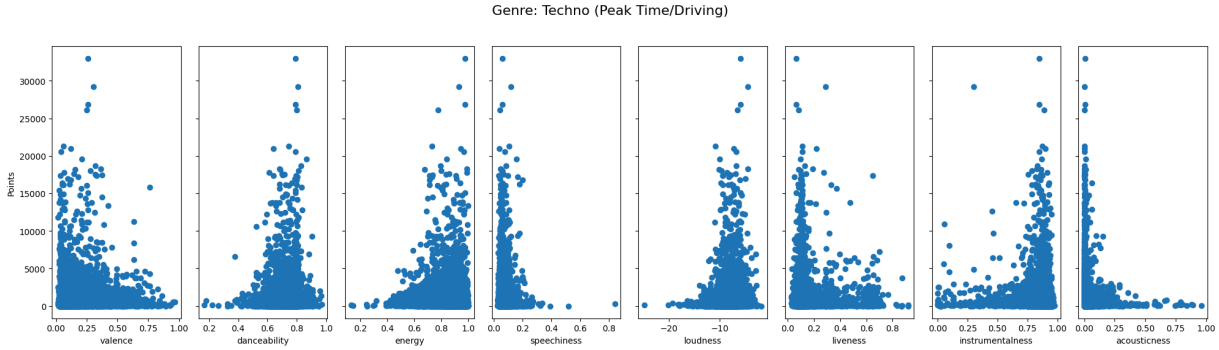
### By Genre

Based on the outputs, in future work I'll need to figure out a better approach for things like "speechiness" where there's intense clustering at one end. I have a suspicion, purely as a music friend, that there's been a lot of movement overtime within the narrow clusters.

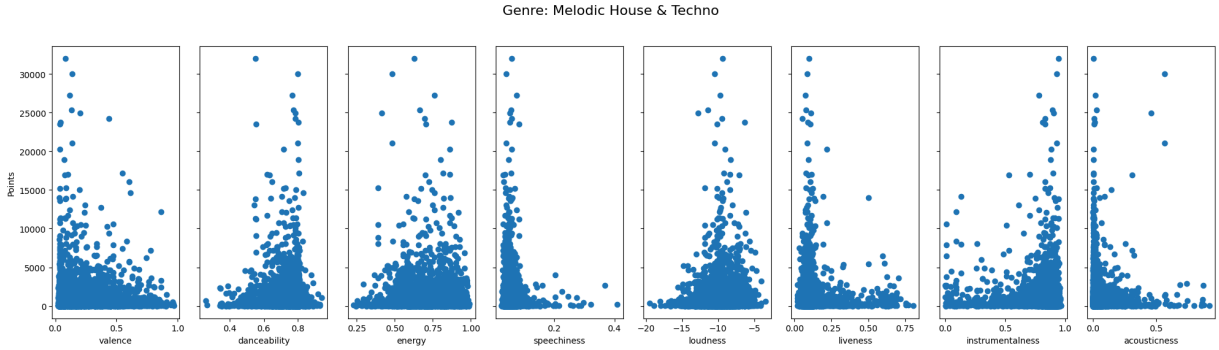
In [17]:
```
for genre_id in subset_genres:
    genre_matrix = toptracks_bpmeta_matrix[toptracks_bpmeta_matrix[:, toptra
    x_features = genre_matrix[:, core_x_features]
    y_points = genre_matrix[:, core_y_target]
    y_points = y_points.reshape(-1, 1)

    # Check shapes
    print(f"Genre: {subset_genres_dict[genre_id]}")
    print("x_features shape:", x_features.shape)
    print("y_points shape:", y_points.shape)
```

```
    fig, ax = plt.subplots(1, len(core_x_features), figsize=(20, 6), sharey=
    fig.suptitle(f'Genre: {subset_genres_dict[genre_id]}', fontsize=16)

    for i, feature_index in enumerate(core_x_features):
        ax[i].scatter(x_features[:, i], y_points)
        feature_name = list(toptracks_bpmeta_matrix_df_dict.keys())[list(top
        ax[i].set_xlabel(feature_name)
    ax[0].set_ylabel("Points")

    plt.tight_layout(rect=[0, 0.03, 1, 0.95])
    plt.show()
```

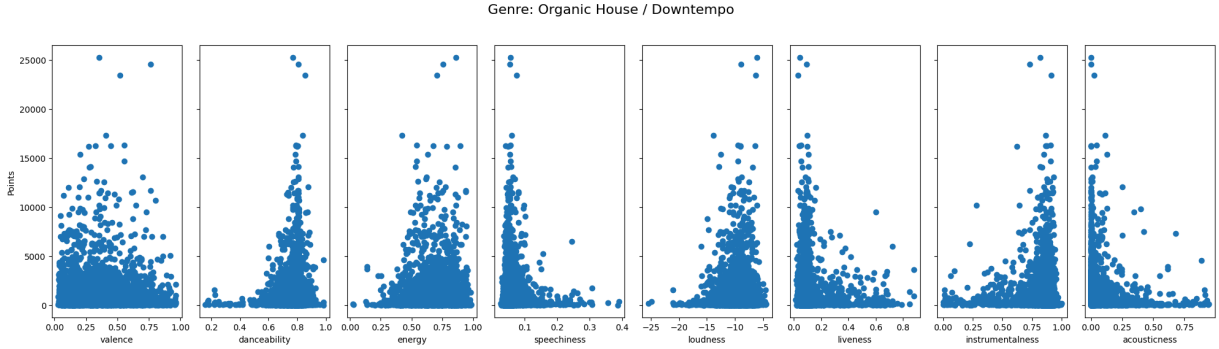
```
Genre: Techno (Peak Time/Driving)
x_features shape: (4857, 8)
y_points shape: (4857, 1)
```




```
Genre: Melodic House & Techno
x_features shape: (4075, 8)
y_points shape: (4075, 1)
```




```
Genre: Organic House / Downtempo
x_features shape: (3817, 8)
y_points shape: (3817, 1)
```

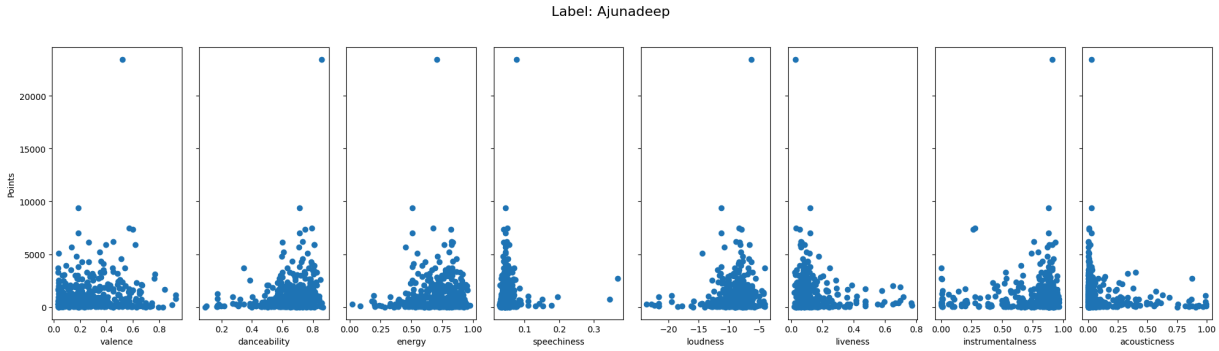



## By Label

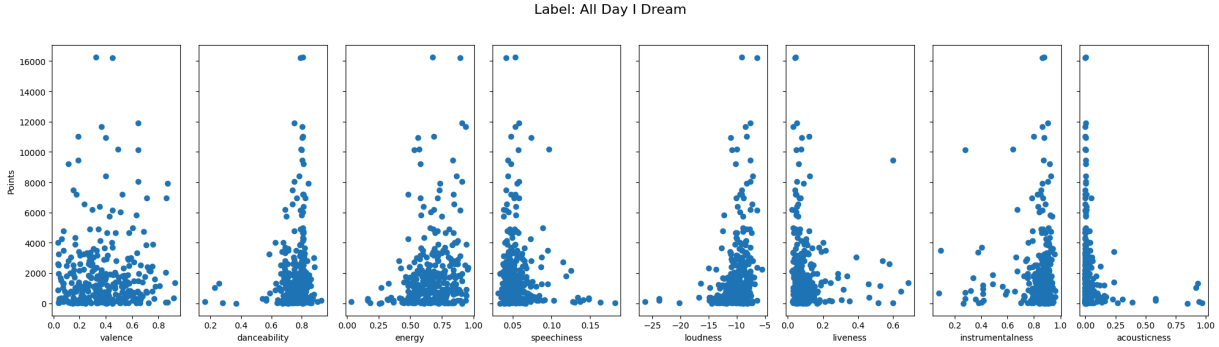
These are pretty small datasets once you filter for top songs and then label. Probably gonna cause me issues a little further down when it's time to run gradient descent.

In [18]:
```
for label_id in subset_labels:
    label_matrix = toptracks_bpmeta_matrix[toptracks_bpmeta_matrix[:, toptra
    x_features = label_matrix[:, core_x_features]
    y_points = label_matrix[:, core_y_target]
    y_points = y_points.reshape(-1, 1)

    # Check shapes
    print(f"Label: {label_ids_dict[label_id]}")
    print("x_features shape:", x_features.shape)
    print("y_points shape:", y_points.shape)

    fig, ax = plt.subplots(1, len(core_x_features), figsize=(20, 6), sharey=
    fig.suptitle(f'Label: {label_ids_dict[label_id]}', fontsize=16)

    for i, feature_index in enumerate(core_x_features):
        ax[i].scatter(x_features[:, i], y_points)
        feature_name = list(toptracks_bpmeta_matrix_df_dict.keys())[list(top
        ax[i].set_xlabel(feature_name)
    ax[0].set_ylabel("Points")

    plt.tight_layout(rect=[0, 0.03, 1, 0.95])
    plt.show()
```

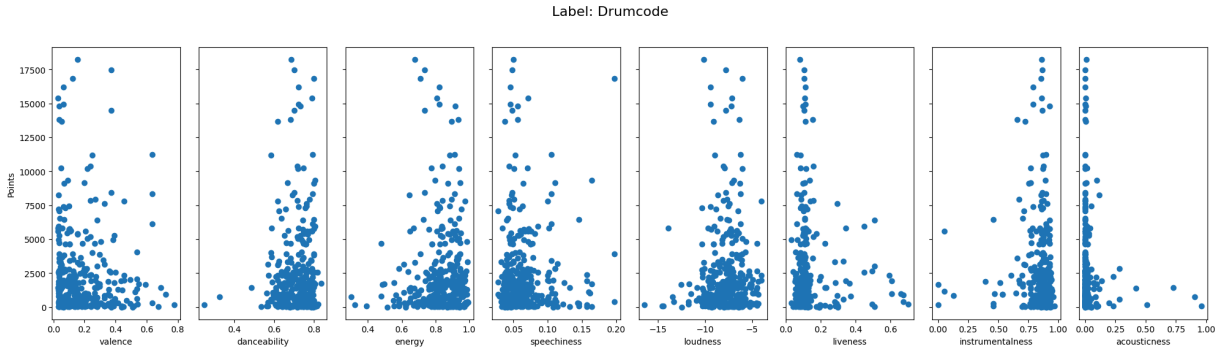
```
Label: Ajunadeep
x_features shape: (430, 8)
y_points shape: (430, 1)
```



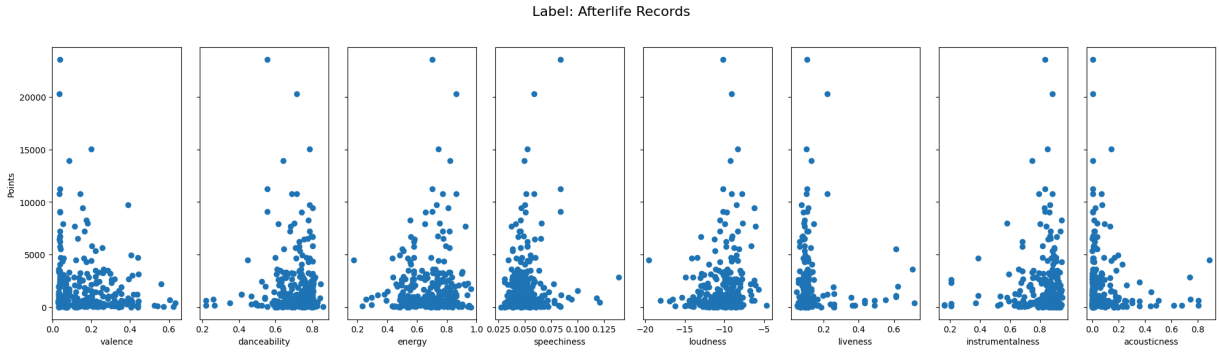

```
Label: All Day I Dream
x_features shape: (327, 8)
y_points shape: (327, 1)
```



```
Label: Drumcode
x_features shape: (338, 8)
y_points shape: (338, 1)
```




```
Label: Afterlife Records
x_features shape: (282, 8)
y_points shape: (282, 1)
```




# Running different normalization methods and reviewing the data

## Z-Score normalization

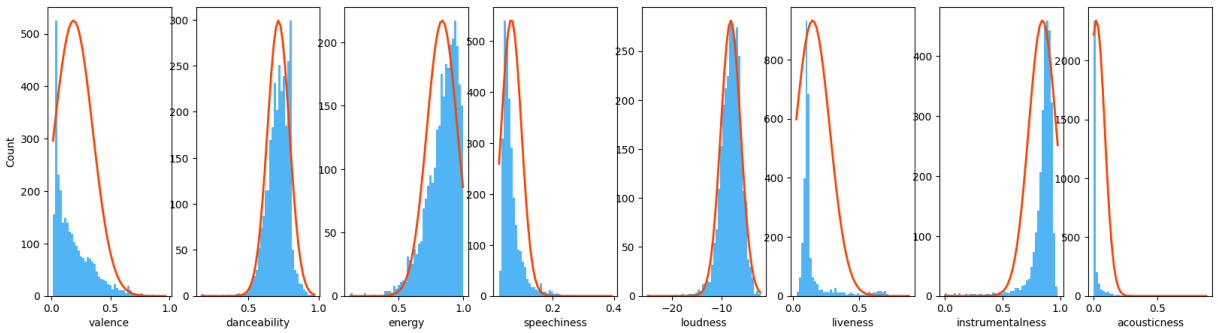
I got the idea for these before/after charts from deeplearning.ai and I just adapted it for my work. Kinda cool to see the before and after of what's going on.

### By Genre

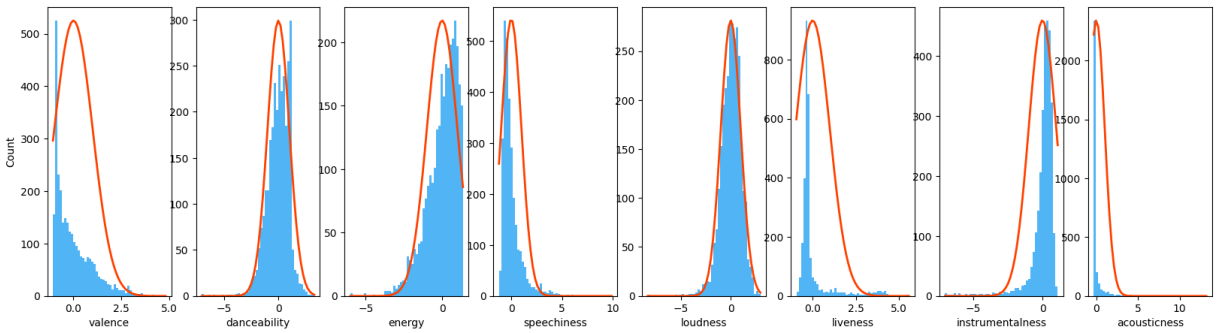
In [19]:
```
for genre_id in subset_genres:
    genre_matrix = zscore_train[zscore_train[:, toptracks_bpmeta_matrix_df_c
    x_features = genre_matrix[:, core_x_features]
    x_norm, feature_mean, standard_deviation = zscore_normalize(x_features)

    # Plot before normalization
    fig, ax = plt.subplots(1, len(core_x_features), figsize=(20, 5))
    for i in range(len(ax)):
        norm_plot(ax[i], x_features[:, i])
        ax[i].set_xlabel(list(toptracks_bpmeta_matrix_df_dict.keys())[list(t
    ax[0].set_ylabel("Count")
    fig.suptitle(f"Distribution of features before normalization for: {subse
    plt.show()
```

```
    # Plot after normalization
    fig, ax = plt.subplots(1, len(core_x_features), figsize=(20, 5))
    for i in range(len(ax)):
        norm_plot(ax[i], x_norm[:, i])
        ax[i].set_xlabel(list(toptracks_bpmeta_matrix_df_dict.keys())[list(t
    ax[0].set_ylabel("Count")
    fig.suptitle(f"Distribution of features after normalization for genre: {
    plt.show()
```

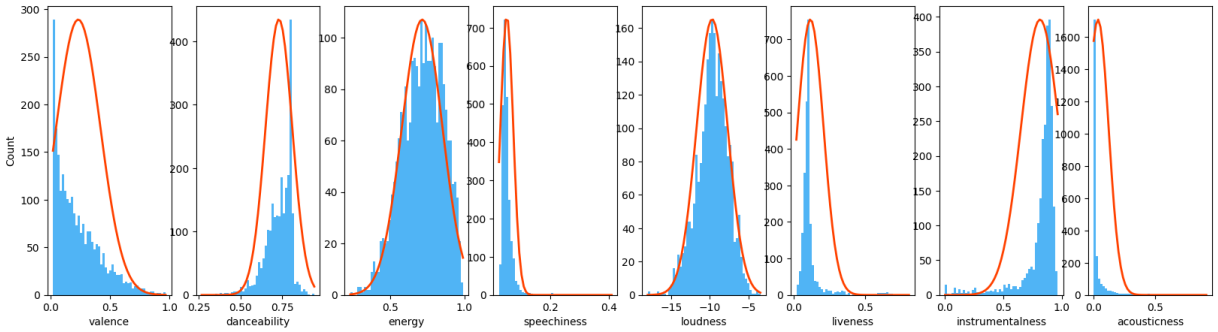
Distribution of features before normalization for: Techno (Peak Time/Driving)



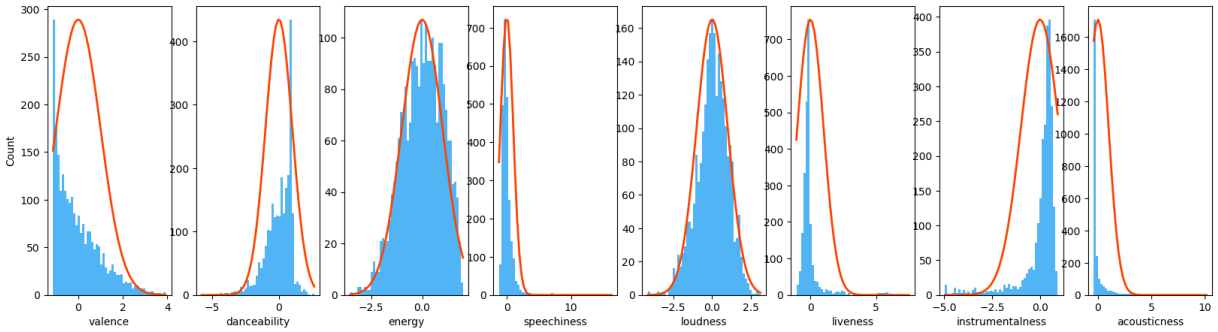

Distribution of features after normalization for genre: Techno (Peak Time/Driving)

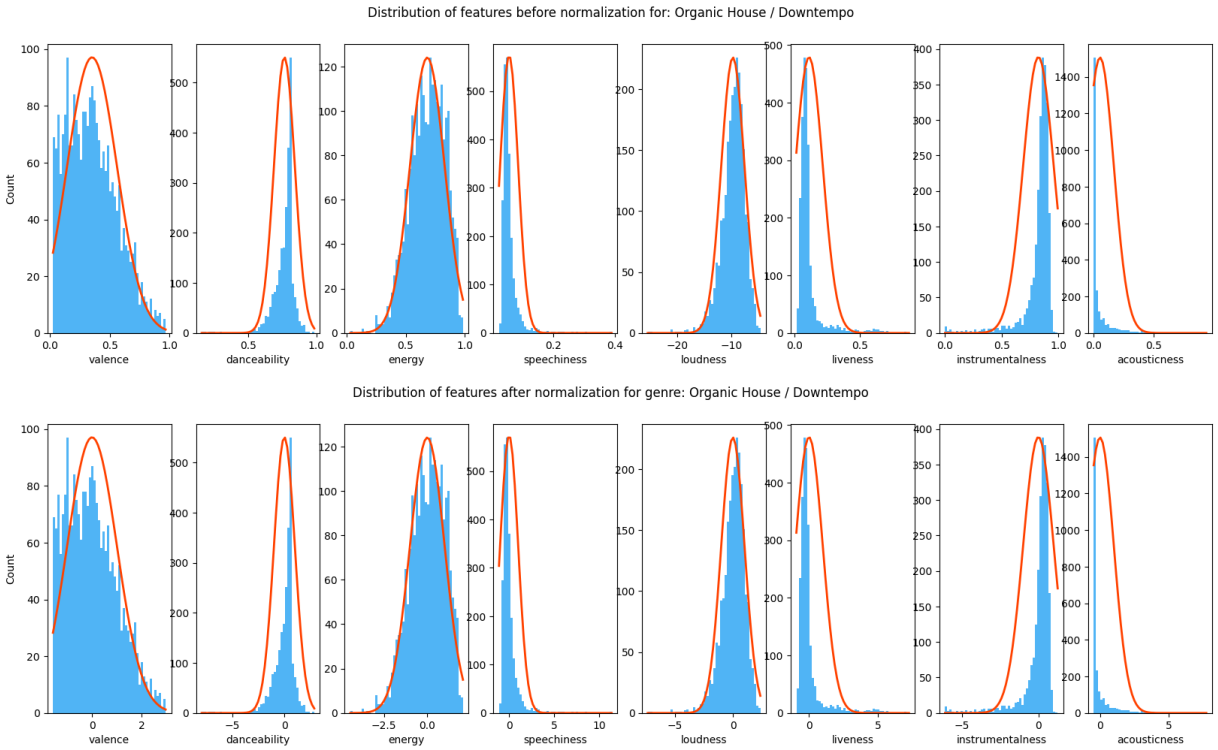



Distribution of features before normalization for: Melodic House & Techno




Distribution of features after normalization for genre: Melodic House & Techno

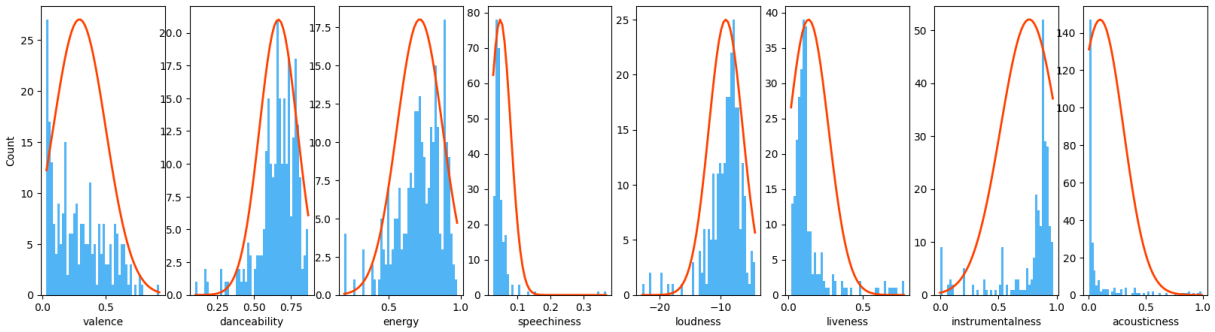




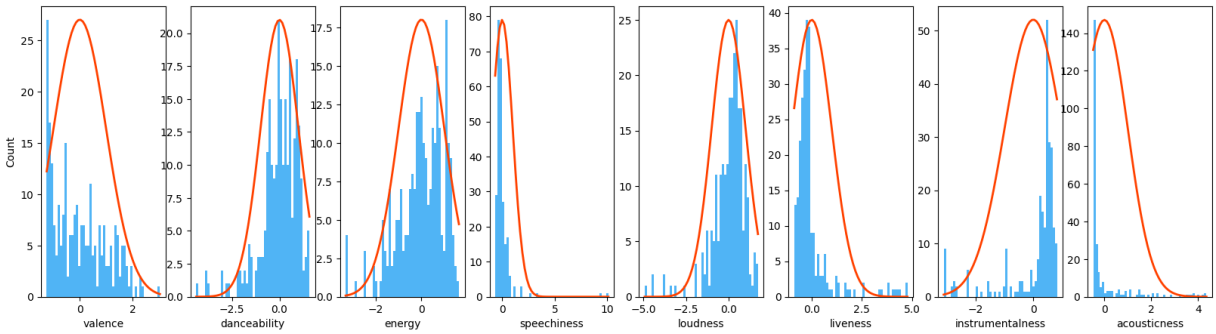

## By Label

In [20]:
```
for label_id in subset_labels:
    label_matrix = zscore_train[zscore_train[:, toptracks_bpmeta_matrix_df_c
    x_features = label_matrix[:, core_x_features]
    x_norm, feature_mean, standard_deviation = zscore_normalize(x_features)

    # Plot before normalization
    fig, ax = plt.subplots(1, len(core_x_features), figsize=(20, 5))
    for i in range(len(ax)):
        norm_plot(ax[i], x_features[:, i])
        ax[i].set_xlabel(list(toptracks_bpmeta_matrix_df_dict.keys())[list(t
    ax[0].set_ylabel("Count")
    fig.suptitle(f"Distribution of features before normalization for Label:
    plt.show()

    # Plot after normalization
    fig, ax = plt.subplots(1, len(core_x_features), figsize=(20, 5))
    for i in range(len(ax)):
        norm_plot(ax[i], x_norm[:, i])
        ax[i].set_xlabel(list(toptracks_bpmeta_matrix_df_dict.keys())[list(t
    ax[0].set_ylabel("Count")
    fig.suptitle(f"Distribution of features after normalization for label: {
    plt.show()
```

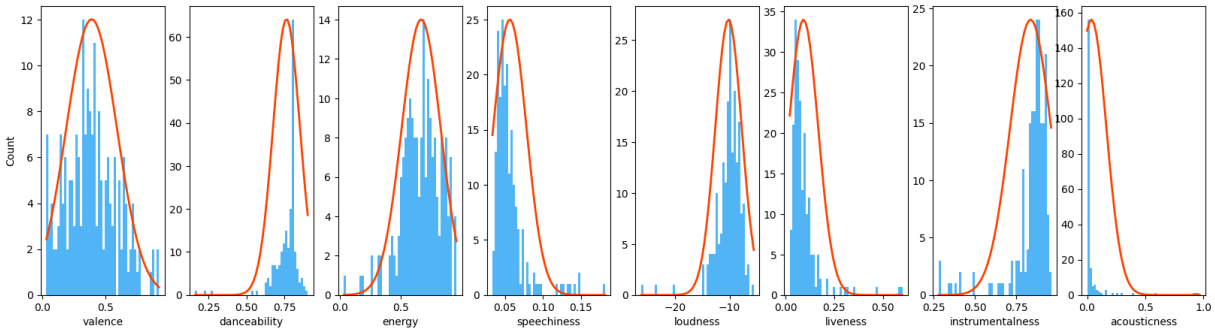
Distribution of features before normalization for Label: Ajunadeep



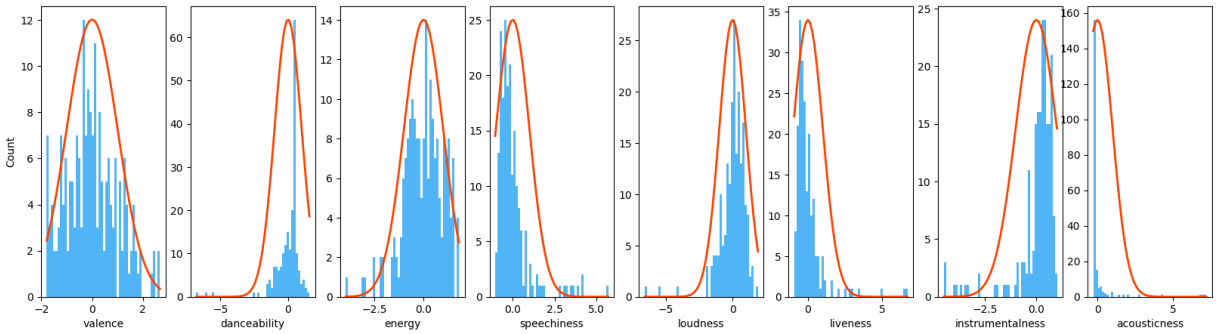

Distribution of features after normalization for label: Ajunadeep



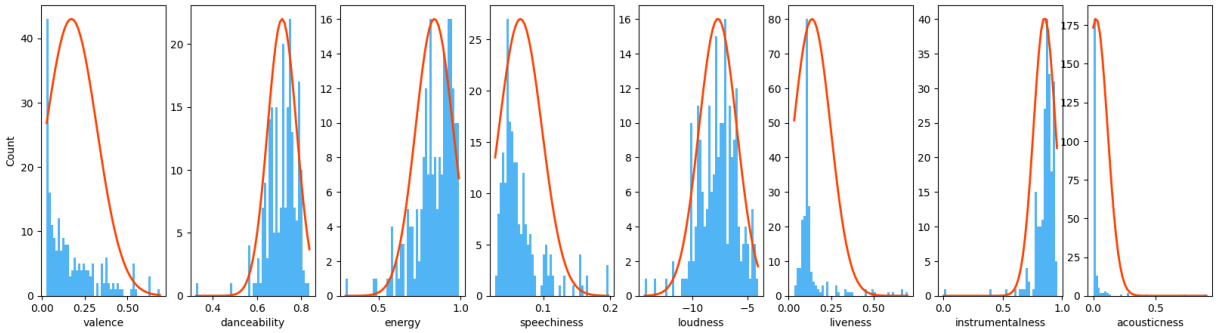

Distribution of features before normalization for Label: All Day I Dream

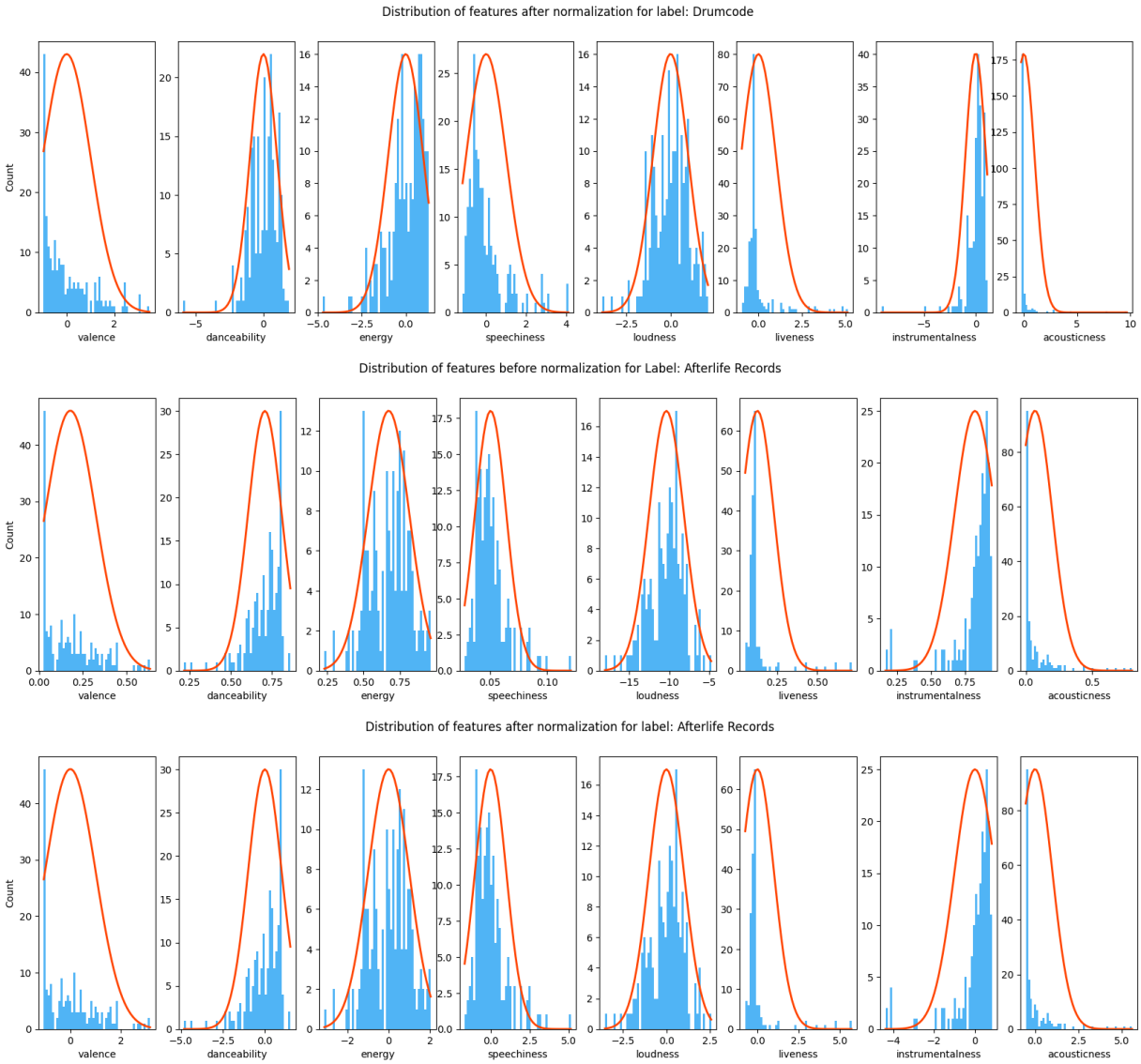



Distribution of features after normalization for label: All Day I Dream




Distribution of features before normalization for Label: Drumcode

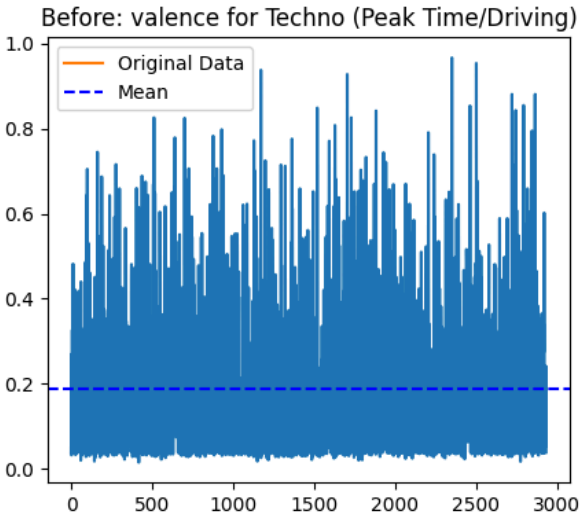


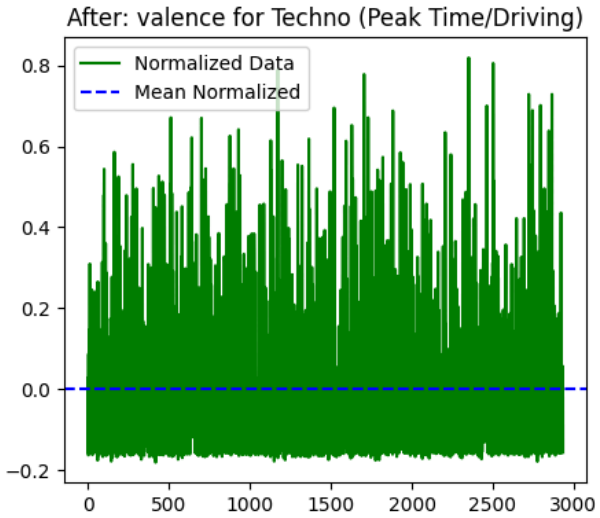


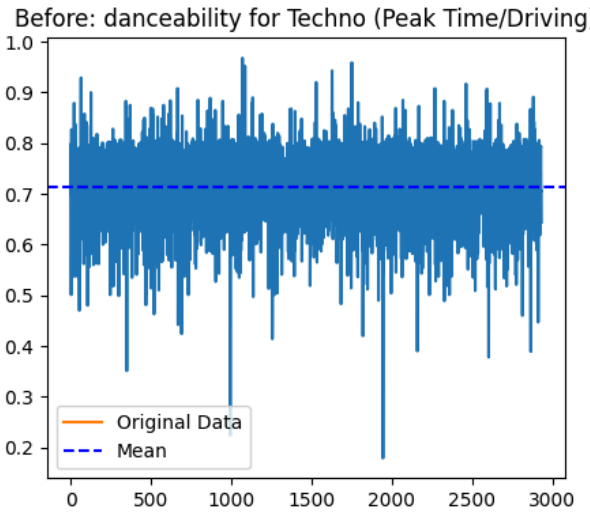

# Mean normalization

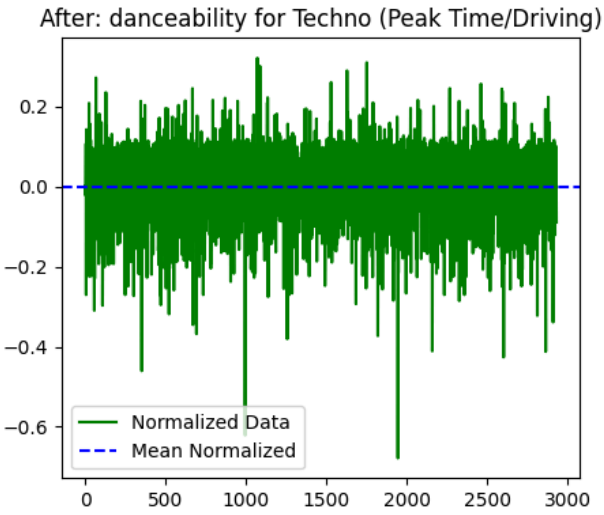
## By Genre

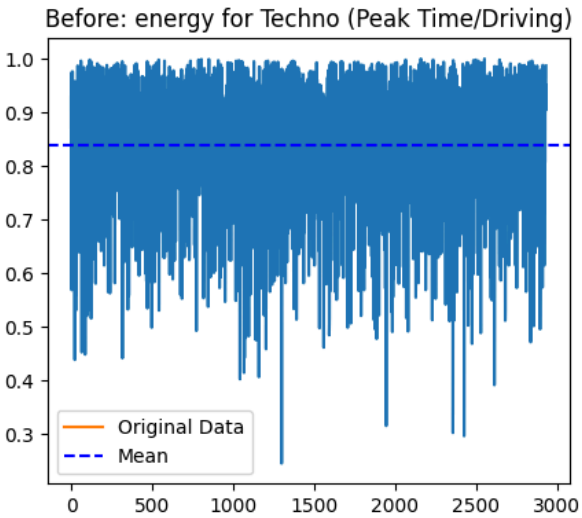
In [21]:
```
#by genre
for genre_id in subset_genres:
    genre_name = subset_genres_dict.get(genre_id, f"Genre {genre_id}")
    genre_matrix = mean_train[mean_train[:, toptracks_bpmeta_matrix_df_dict[
    x_features = genre_matrix[:, core_x_features]
    for i, feature_index in enumerate(core_x_features):
        feature = genre_matrix[:, feature_index]
        feature_name = list(toptracks_bpmeta_matrix_df_dict.keys())[list(top
        title = f"{feature_name} for {genre_name}"
        mean_plot(feature, title)
```

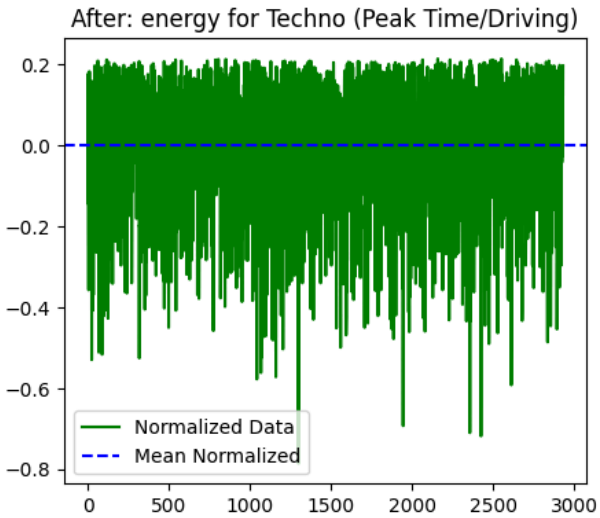

Before: valence for Techno (Peak Time/Driving)
Original Data
Mean

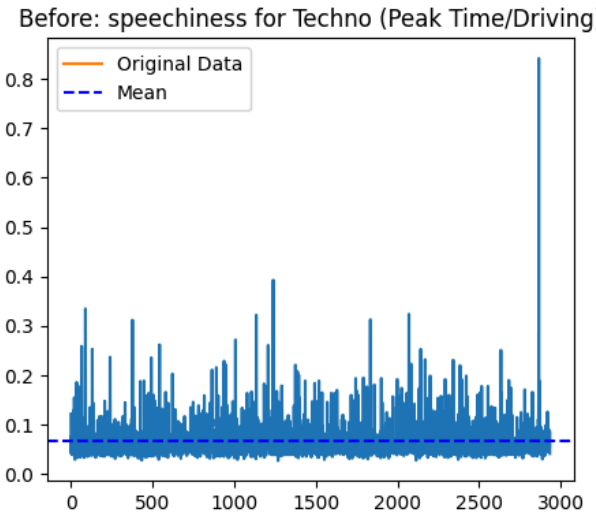

After: valence for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

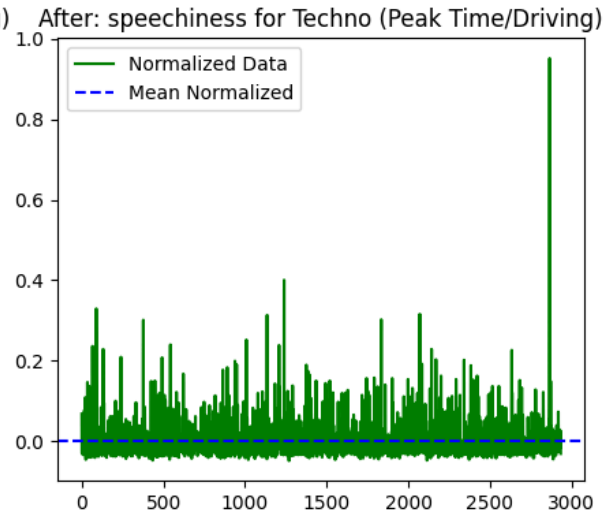

Before: danceability for Techno (Peak Time/Driving)
Original Data
Mean

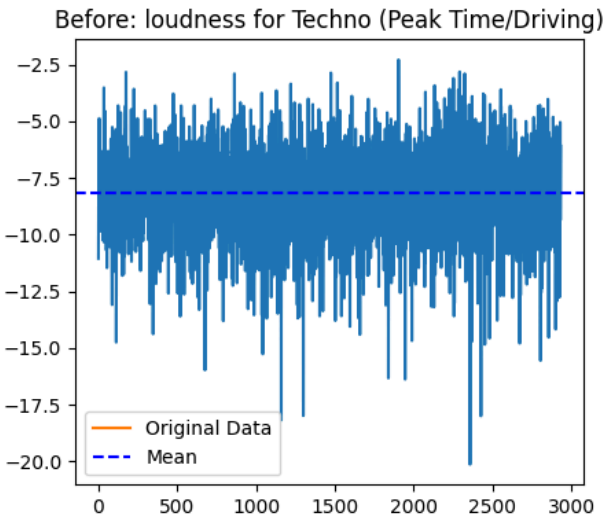

After: danceability for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

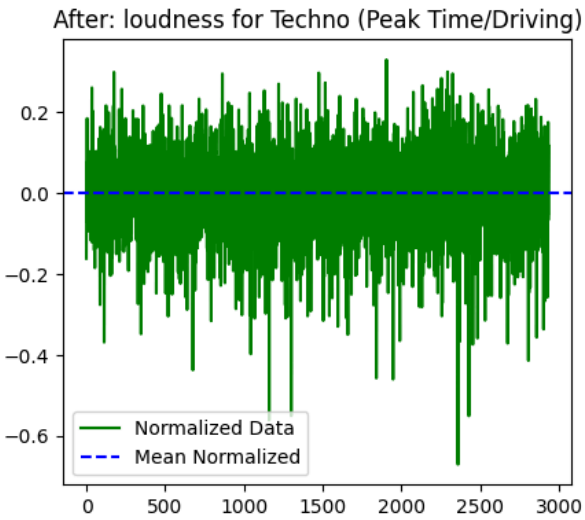

Before: energy for Techno (Peak Time/Driving)
Original Data
Mean

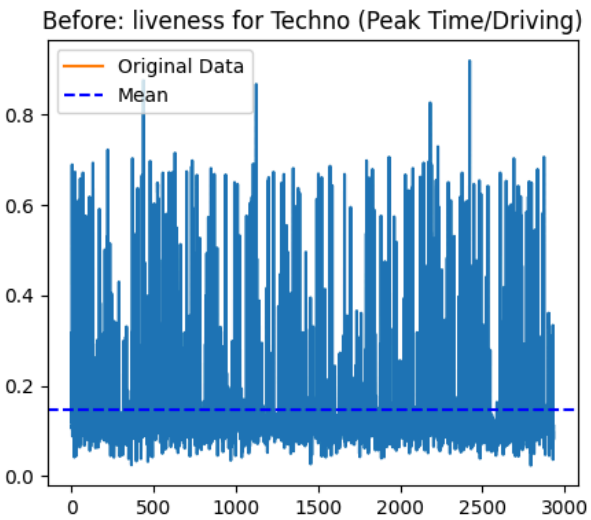

After: energy for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

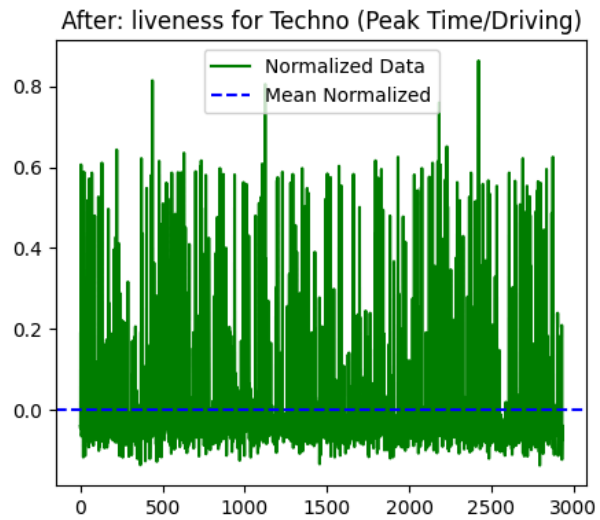

Before: speechiness for Techno (Peak Time/Driving)
Original Data
Mean

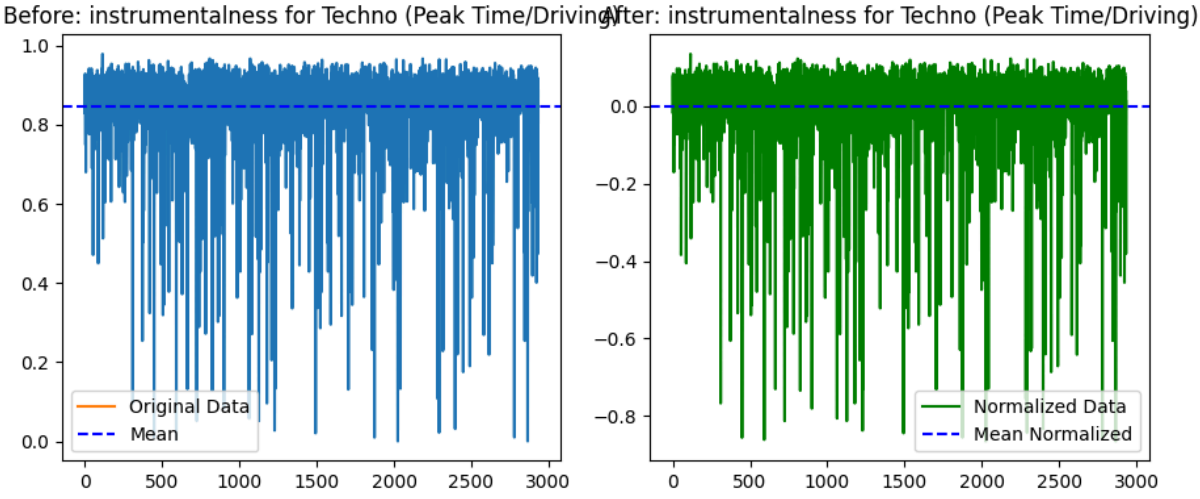

After: speechiness for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

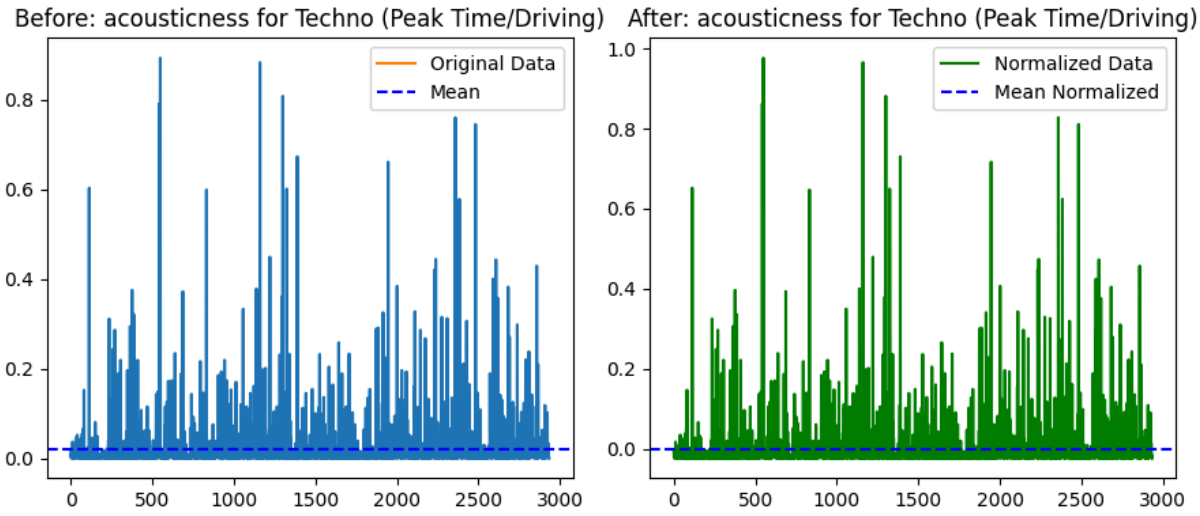

Before: loudness for Techno (Peak Time/Driving)
Original Data
Mean

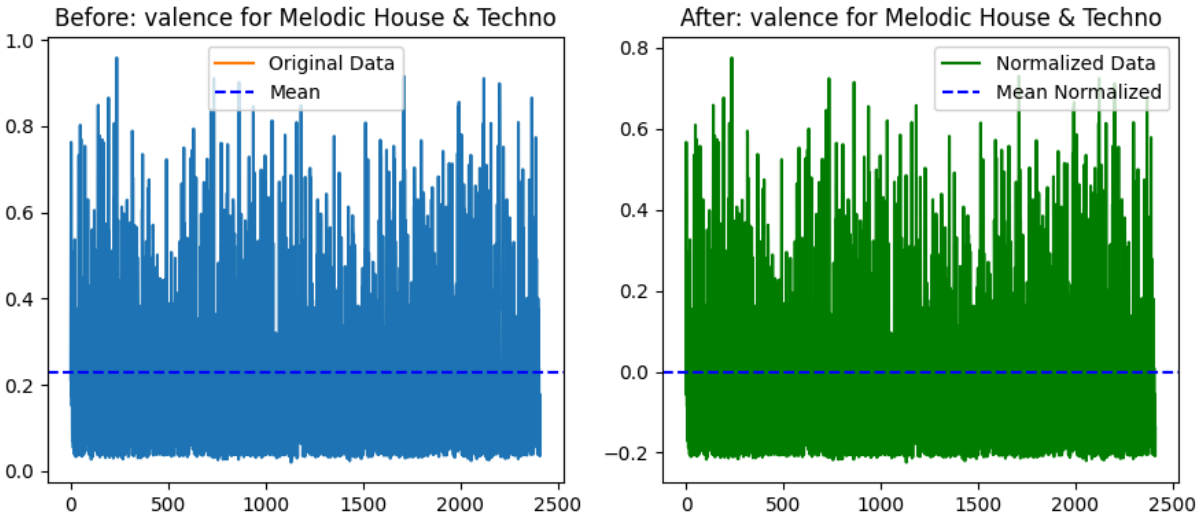

After: loudness for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

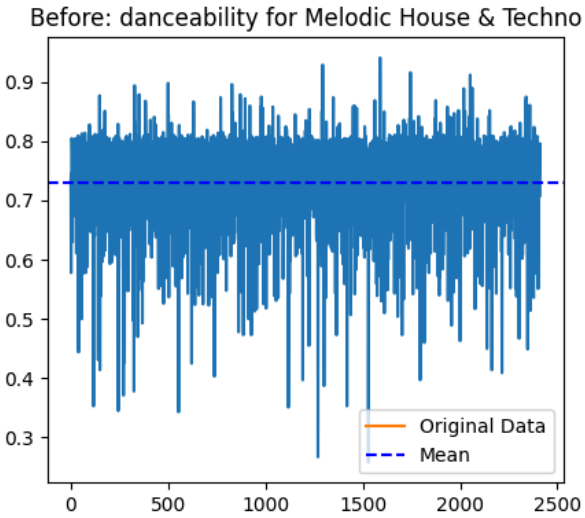

Before: liveness for Techno (Peak Time/Driving)
Original Data
Mean

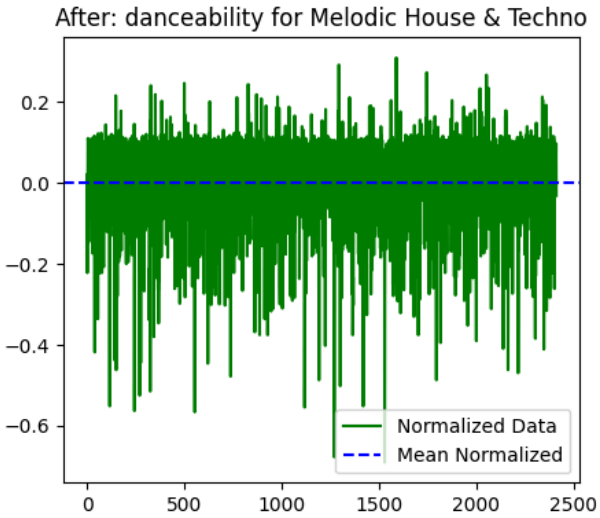

After: liveness for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

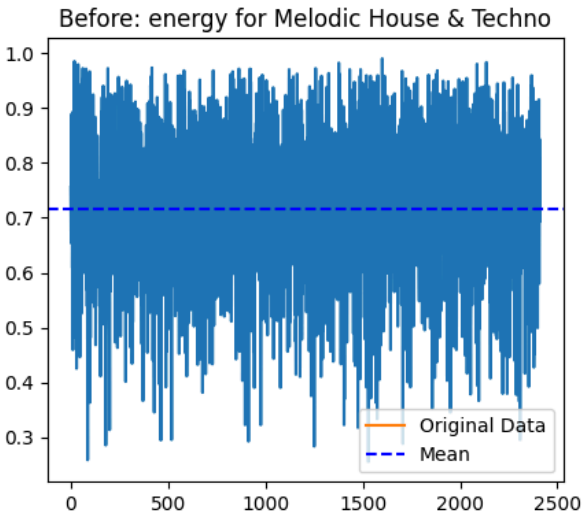

Before: instrumentalness for Techno (Peak Time/Driving)
Original Data
Mean
After: instrumentalness for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

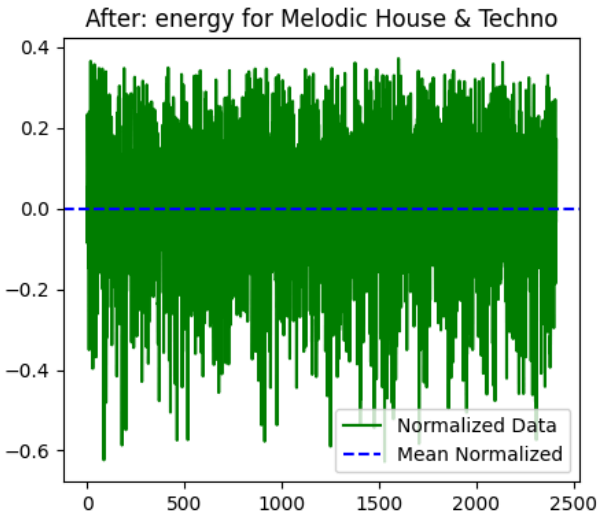

Before: acousticness for Techno (Peak Time/Driving)
Original Data
Mean
After: acousticness for Techno (Peak Time/Driving)
Normalized Data
Mean Normalized

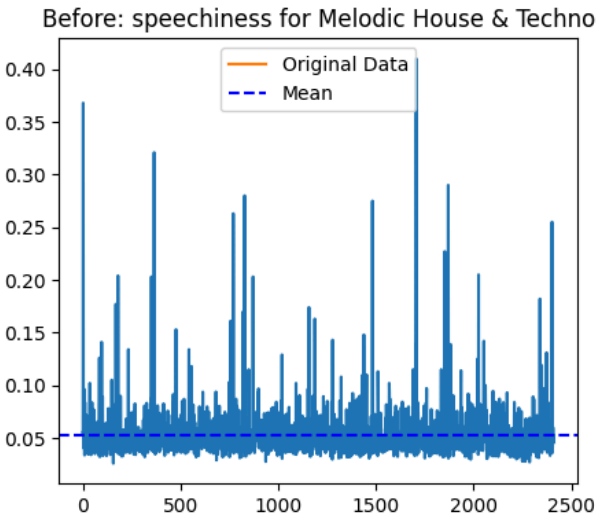

Before: valence for Melodic House & Techno
Original Data
Mean
After: valence for Melodic House & Techno
Normalized Data
Mean Normalized

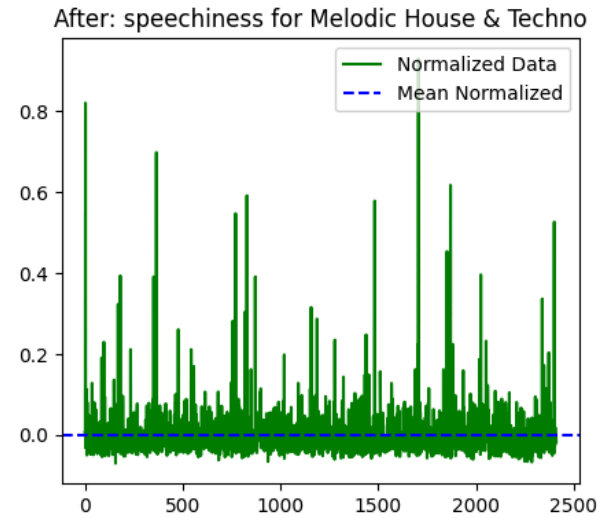

Before: danceability for Melodic House & Techno
Original Data
Mean

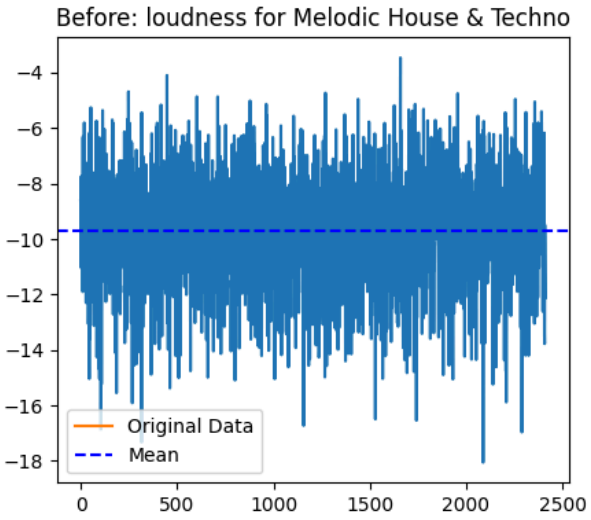

After: danceability for Melodic House & Techno
Normalized Data
Mean Normalized

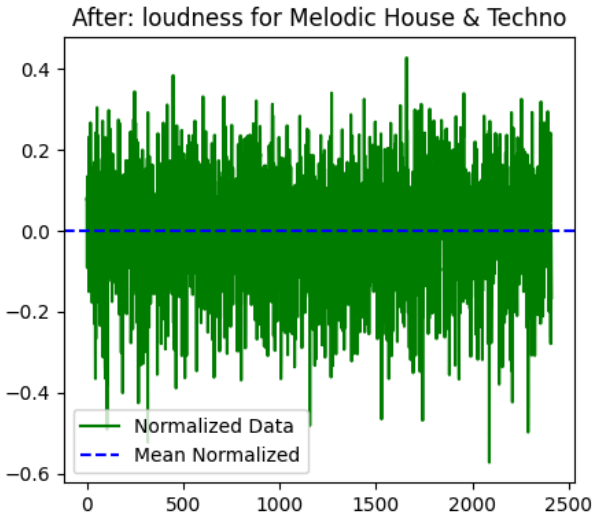

Before: energy for Melodic House & Techno
Original Data
Mean

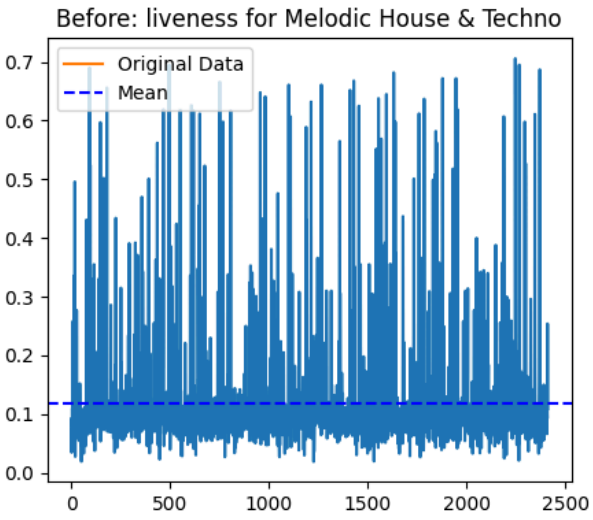

After: energy for Melodic House & Techno
Normalized Data
Mean Normalized

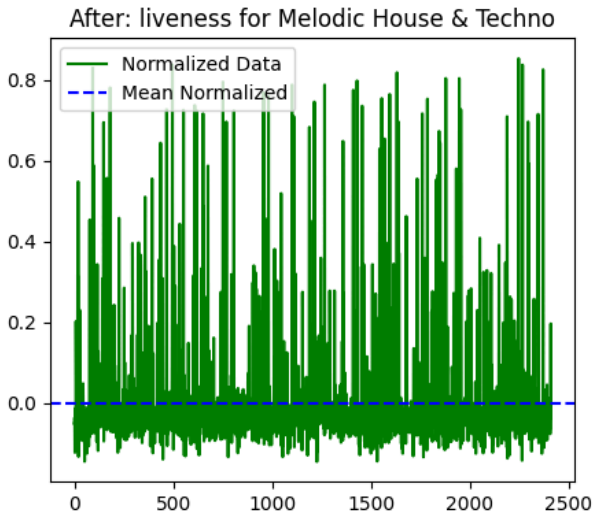

Before: speechiness for Melodic House & Techno
Original Data
Mean

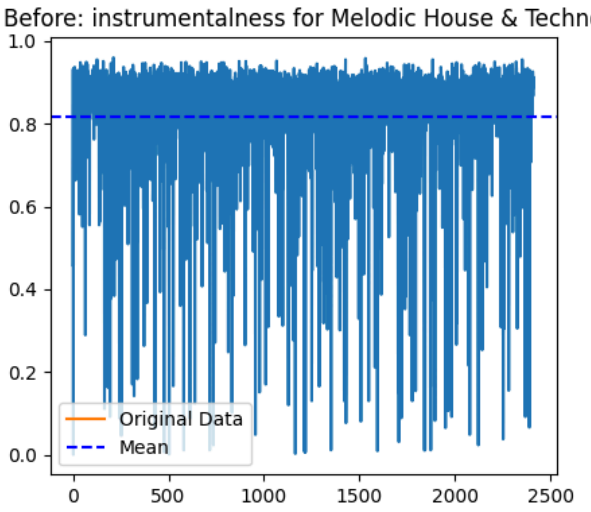

After: speechiness for Melodic House & Techno
Normalized Data
Mean Normalized

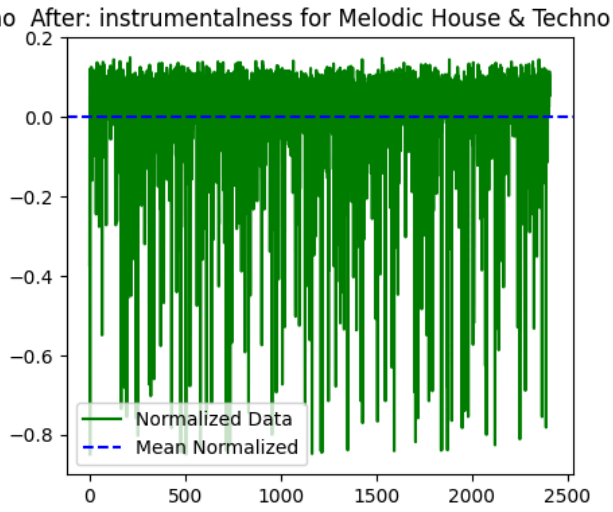

Before: loudness for Melodic House & Techno
Original Data
Mean

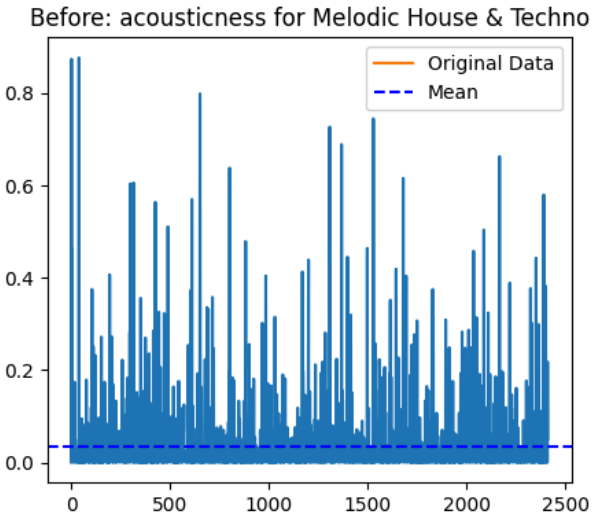

After: loudness for Melodic House & Techno
Normalized Data
Mean Normalized

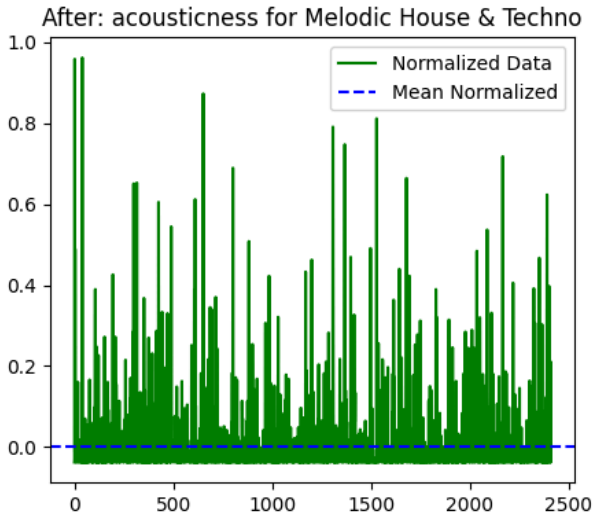

Before: liveness for Melodic House & Techno
Original Data
Mean

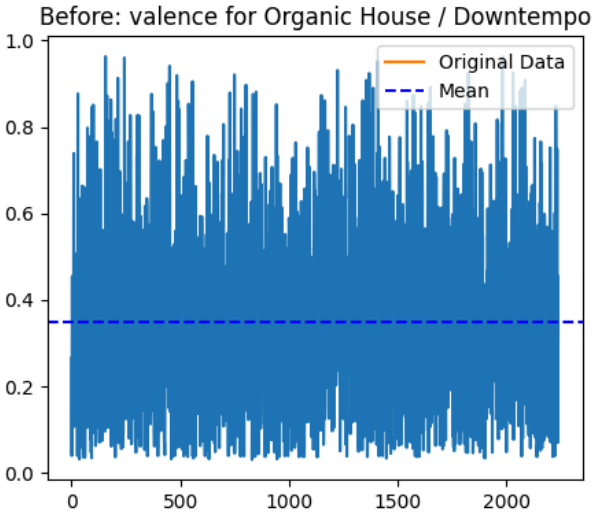

After: liveness for Melodic House & Techno
Normalized Data
Mean Normalized

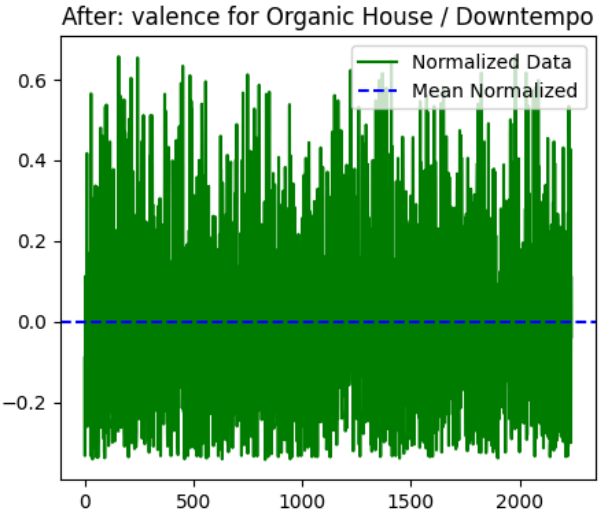

Before: instrumentalness for Melodic House & Techno
Original Data
Mean

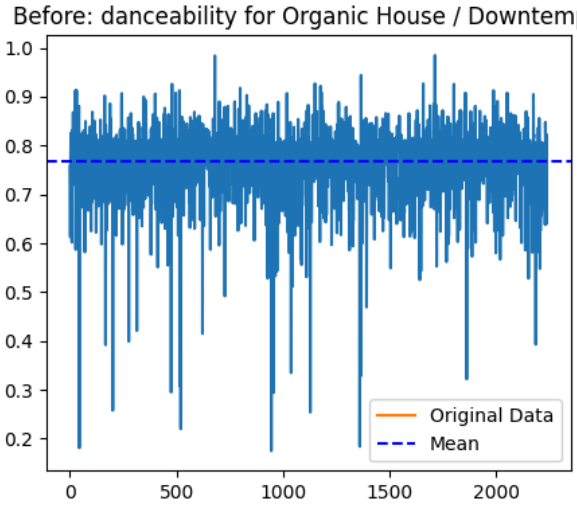

After: instrumentalness for Melodic House & Techno
Normalized Data
Mean Normalized

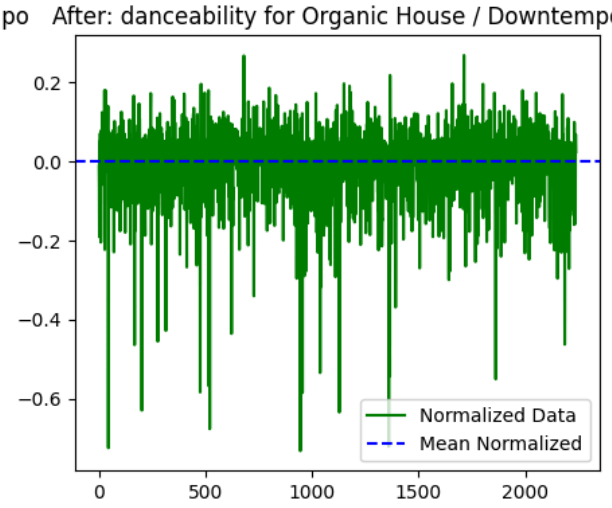

Before: acousticness for Melodic House & Techno
Original Data
Mean

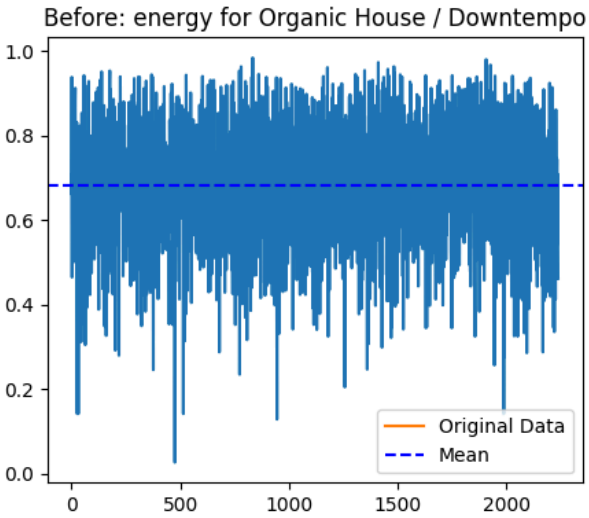

After: acousticness for Melodic House & Techno
Normalized Data
Mean Normalized

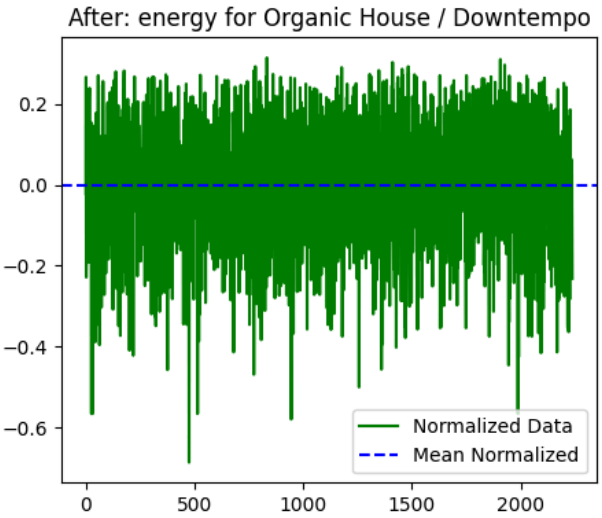

Before: valence for Organic House / Downtempo
Original Data
Mean

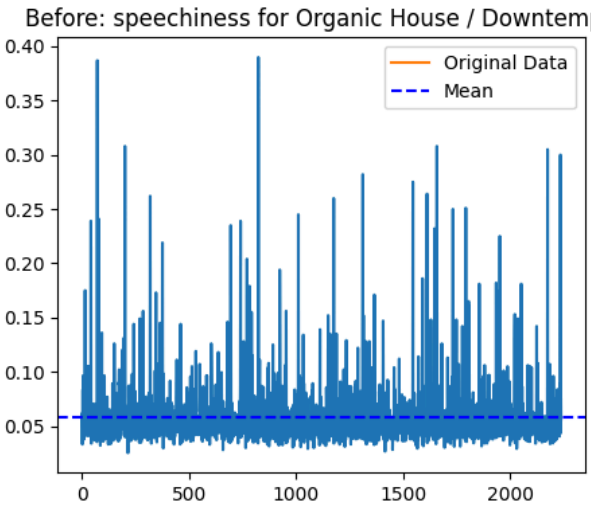

After: valence for Organic House / Downtempo
Normalized Data
Mean Normalized

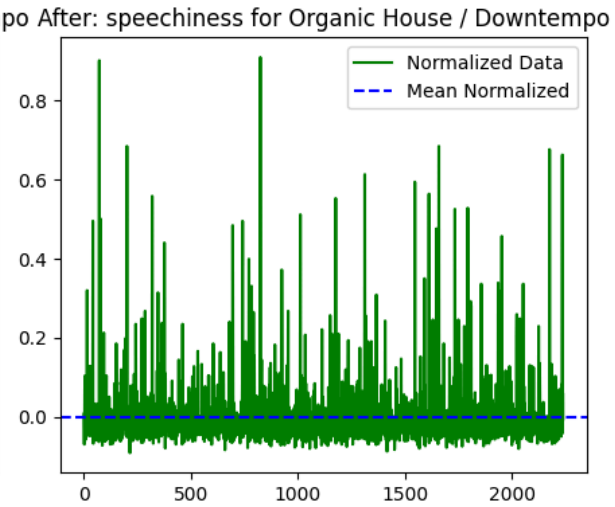

Before: danceability for Organic House / Downtempo
Original Data
Mean

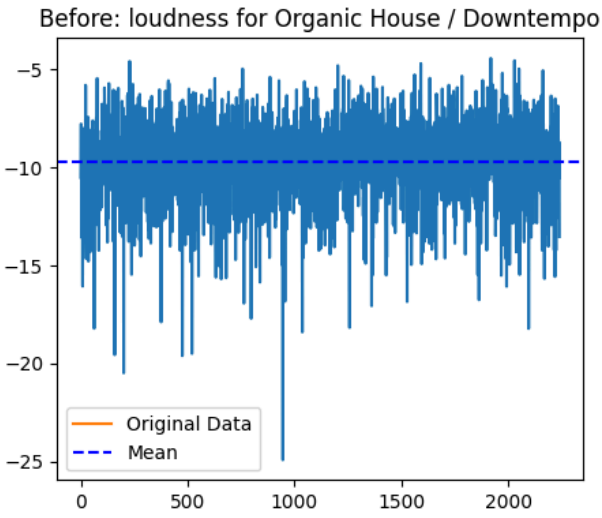

After: danceability for Organic House / Downtempo
Normalized Data
Mean Normalized

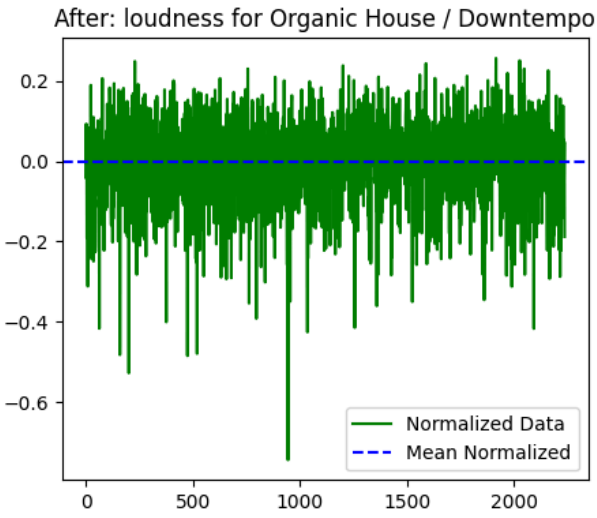

Before: energy for Organic House / Downtempo
Original Data
Mean

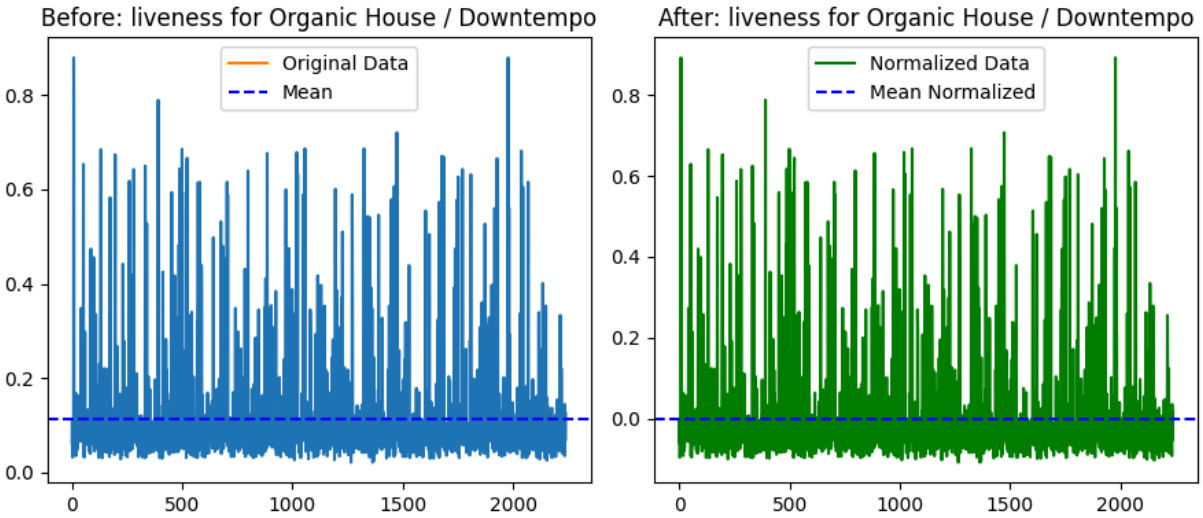

After: energy for Organic House / Downtempo
Normalized Data
Mean Normalized

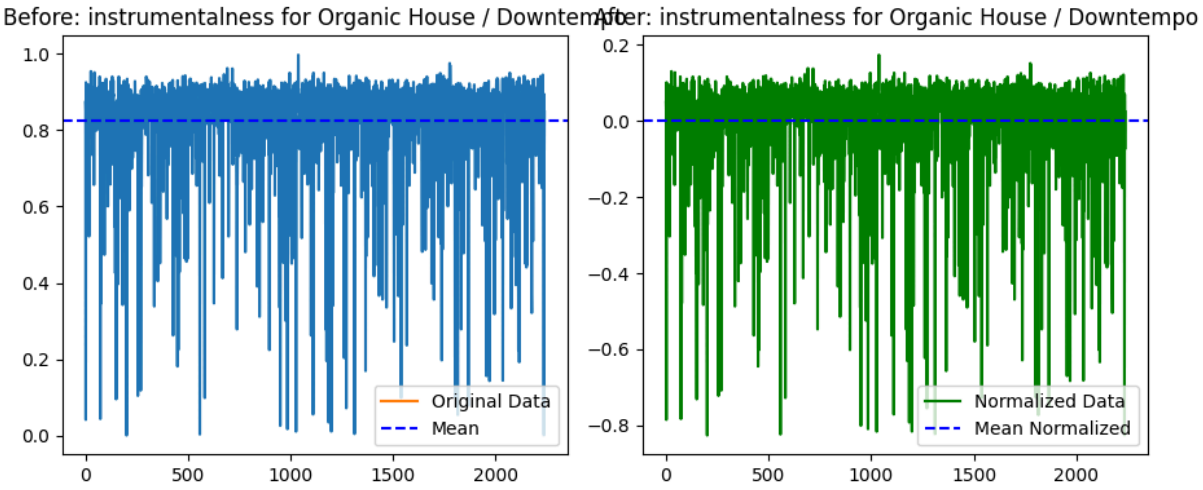

Before: speechiness for Organic House / Downtempo
Original Data
Mean

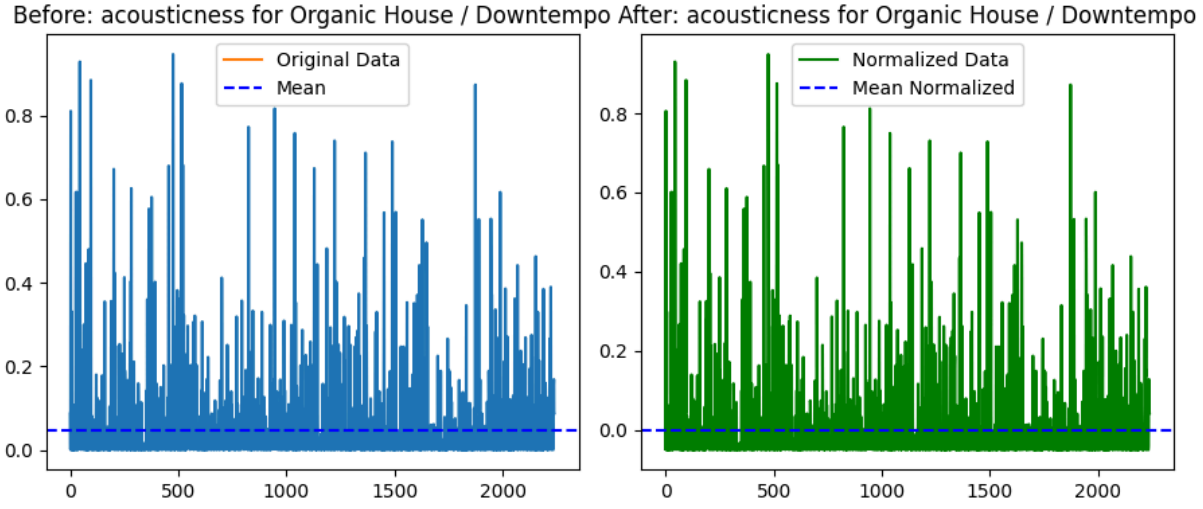

After: speechiness for Organic House / Downtempo
Normalized Data
Mean Normalized


Before: loudness for Organic House / Downtempo
Original Data
Mean


After: loudness for Organic House / Downtempo
Normalized Data
Mean Normalized



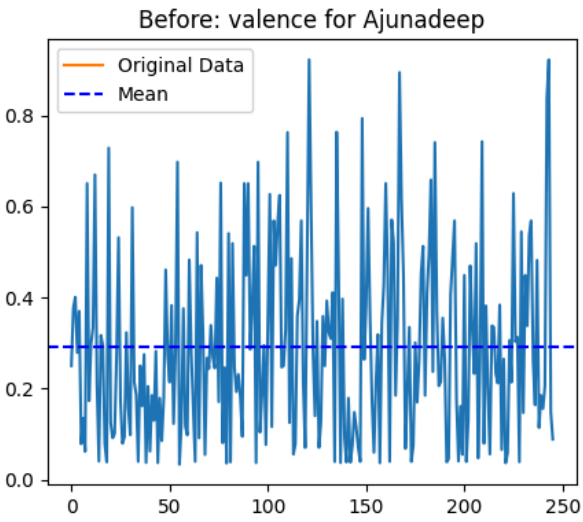



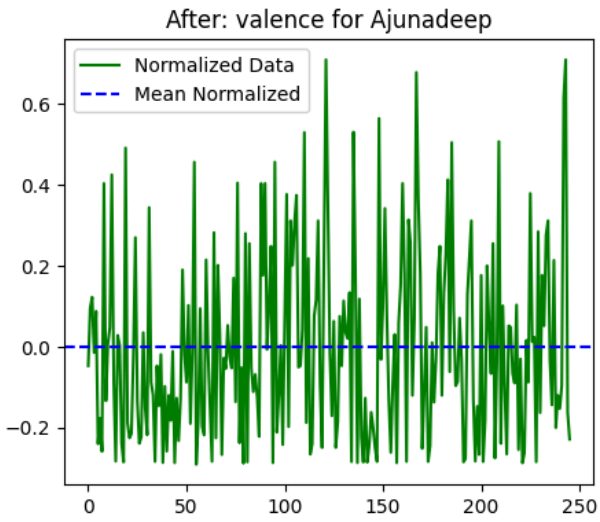



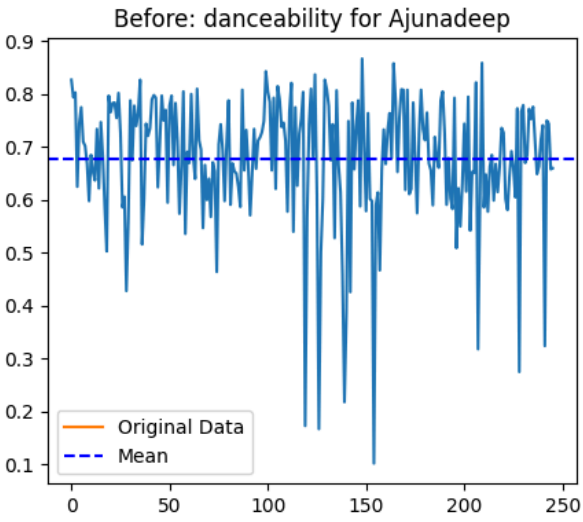

## By Label

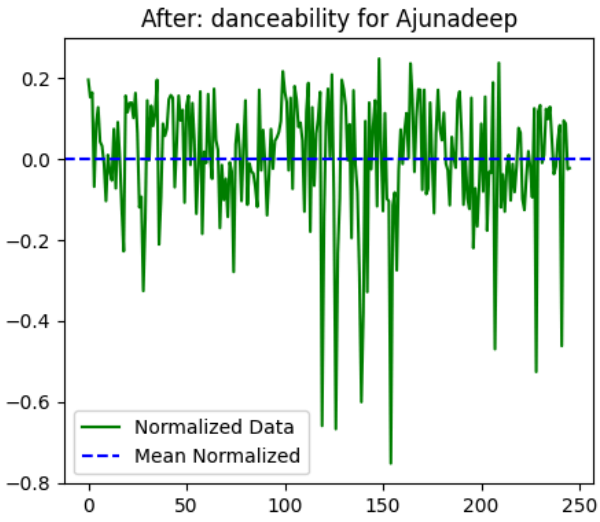
In [34]:
```
#by label
for label_id in subset_labels:
    label_name = label_ids_dict[label_id]
    label_matrix = mean_train[mean_train[:, toptracks_bpmeta_matrix_df_dict[
    x_features = label_matrix[:, core_x_features]
    for i, feature_index in enumerate(core_x_features):
```

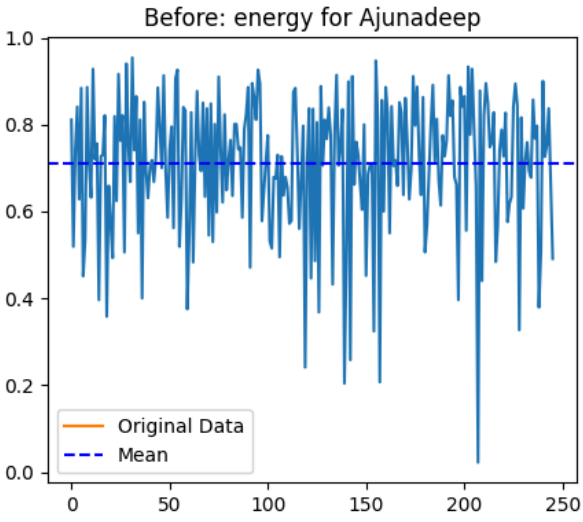
```
        feature = label_matrix[:, feature_index]
        feature_name = list(toptracks_bpmeta_matrix_df_dict.keys())[list(top
        title = f"{feature_name} for {label_name}"
        mean_plot(feature, title)
```

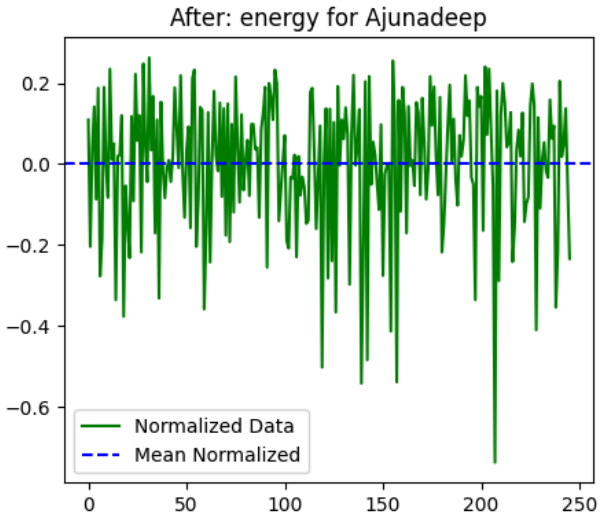


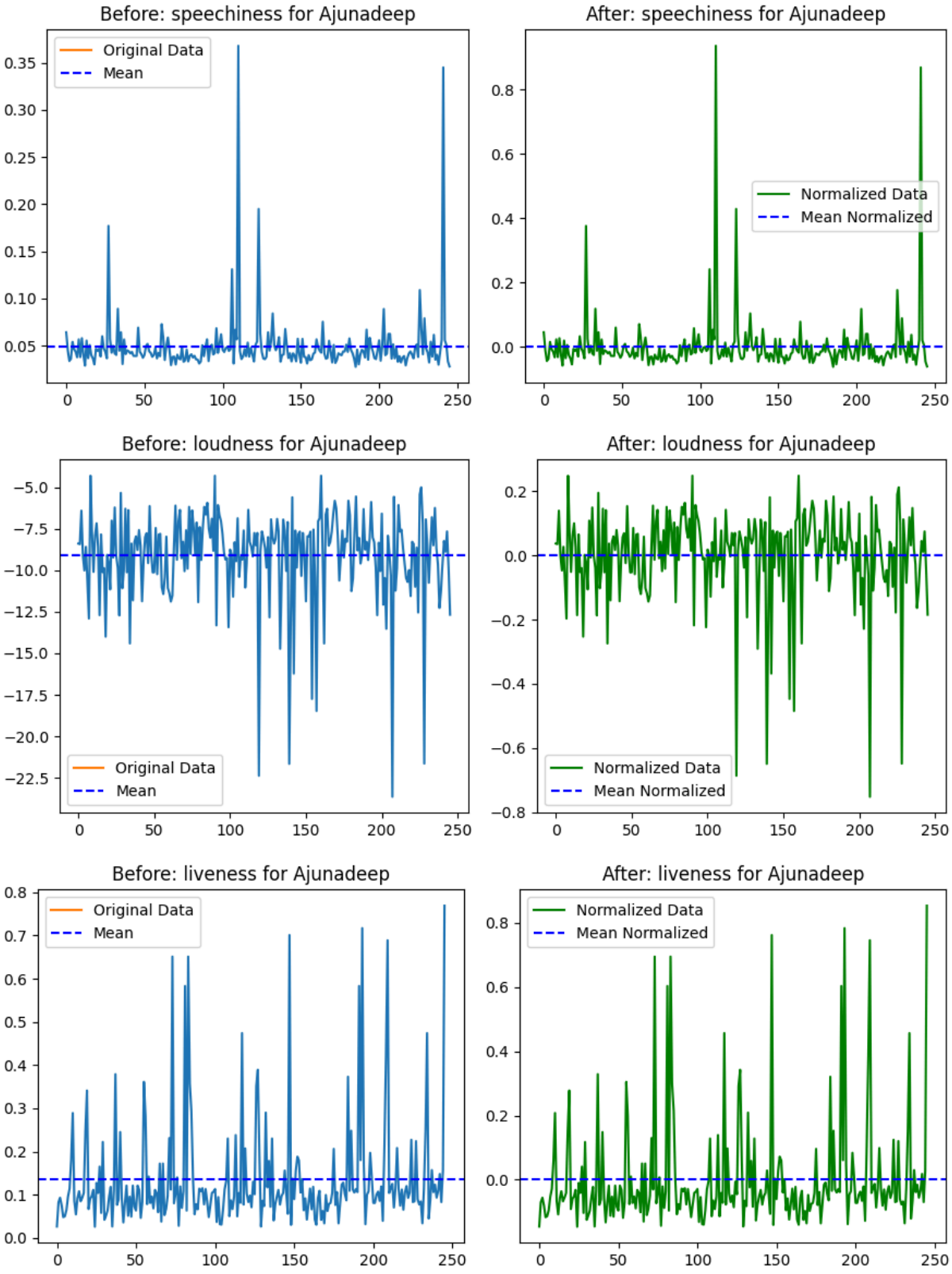



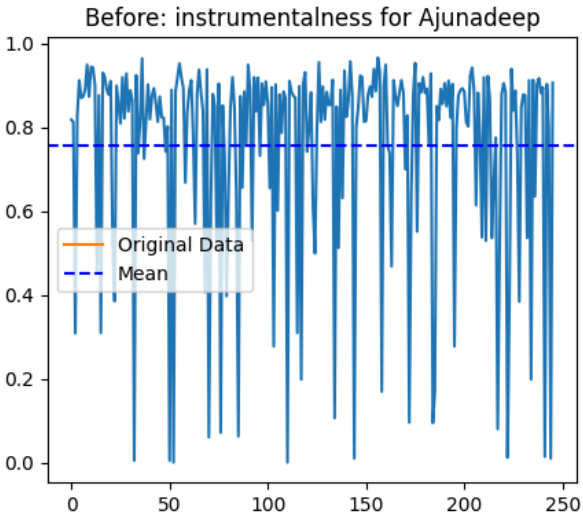



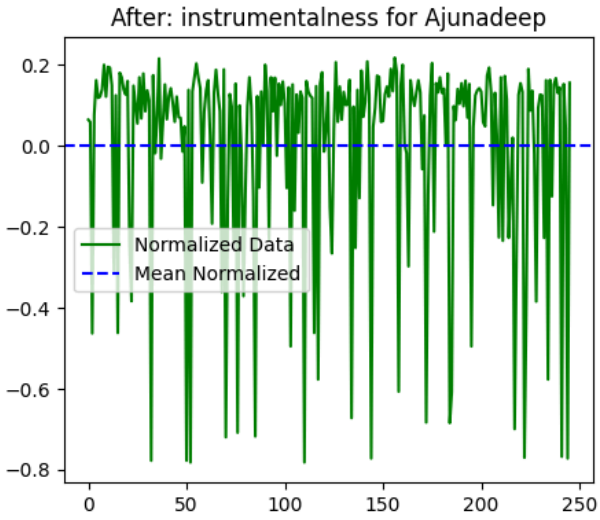



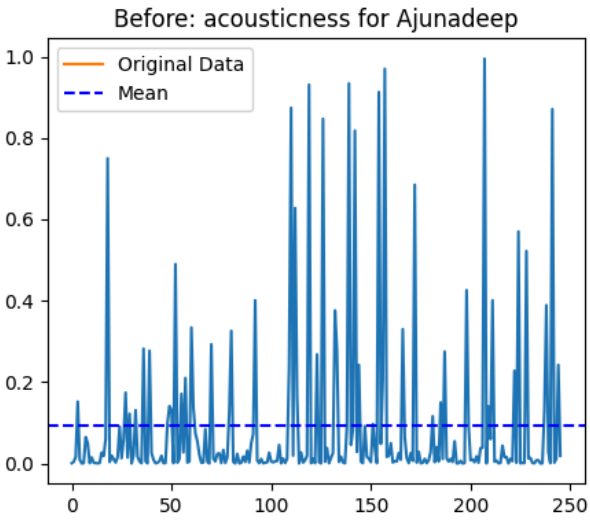



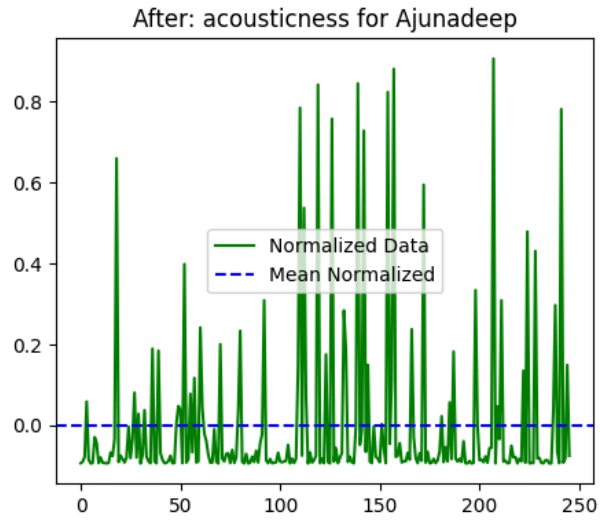



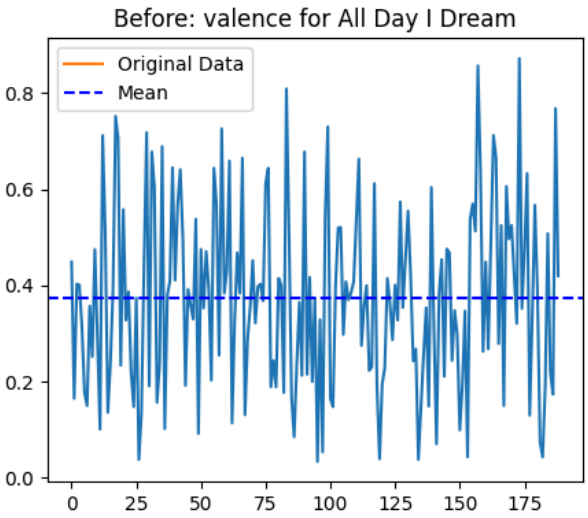

Before: speechiness for Ajunadeep
Original Data
Mean
After: speechiness for Ajunadeep
Normalized Data
Mean Normalized
Before: loudness for Ajunadeep
Original Data
Mean
After: loudness for Ajunadeep
Normalized Data
Mean Normalized
Before: liveness for Ajunadeep
Original Data
Mean
After: liveness for Ajunadeep
Normalized Data
Mean Normalized

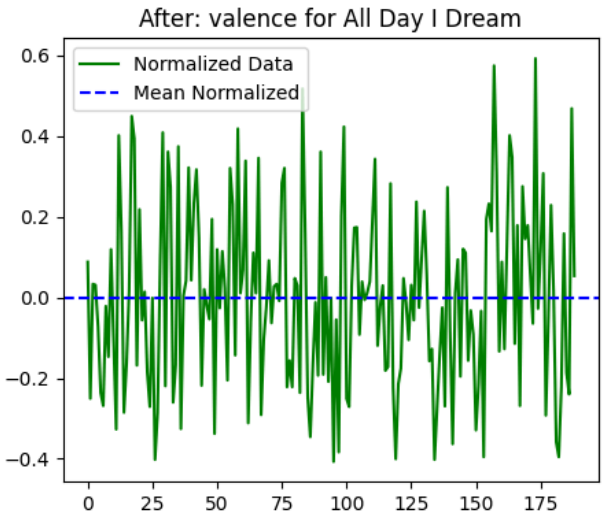

Before: instrumentalness for Ajunadeep
Original Data
Mean

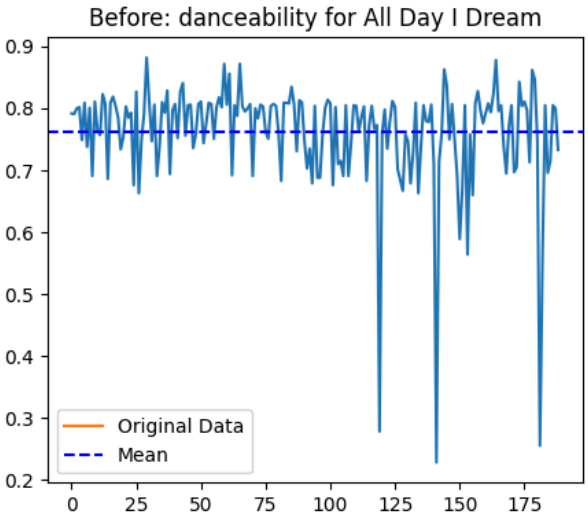

After: instrumentalness for Ajunadeep
Normalized Data
Mean Normalized

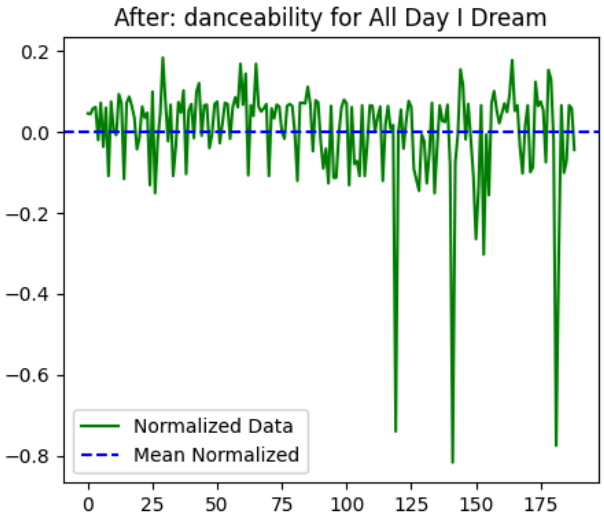

Before: acousticness for Ajunadeep
Original Data
Mean

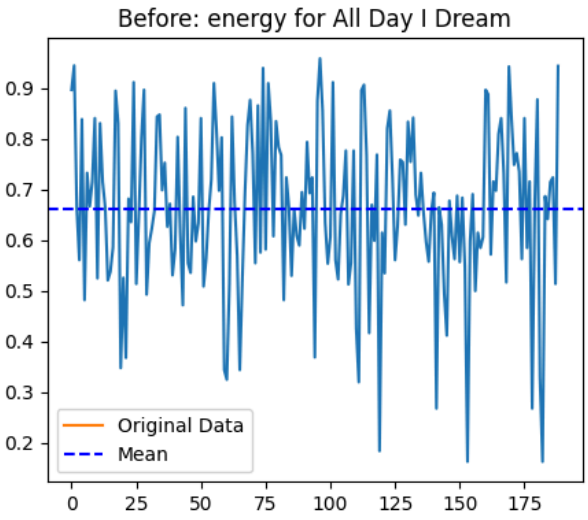

After: acousticness for Ajunadeep
Normalized Data
Mean Normalized

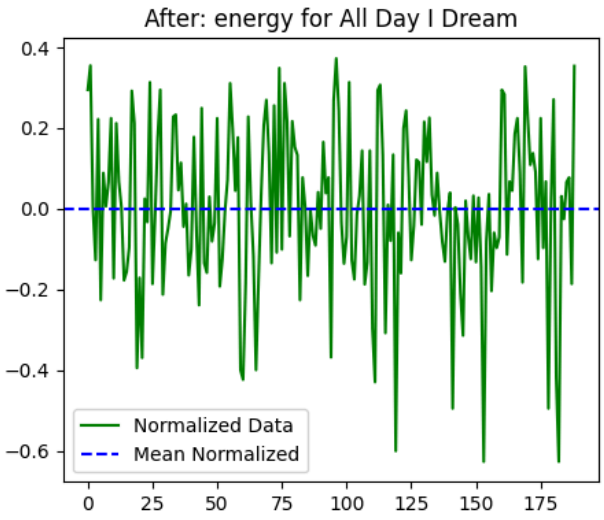

Before: valence for All Day I Dream
Original Data
Mean

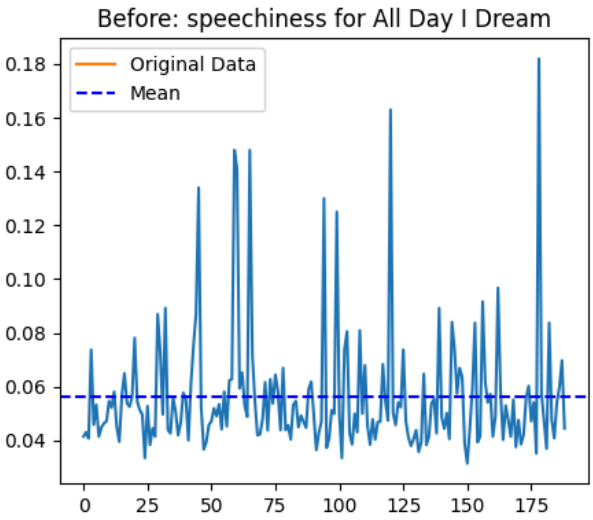

After: valence for All Day I Dream
Normalized Data
Mean Normalized

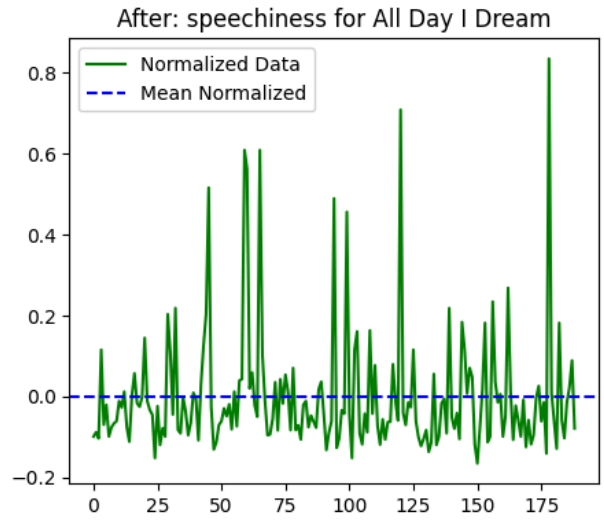

Before: danceability for All Day I Dream
Original Data
Mean

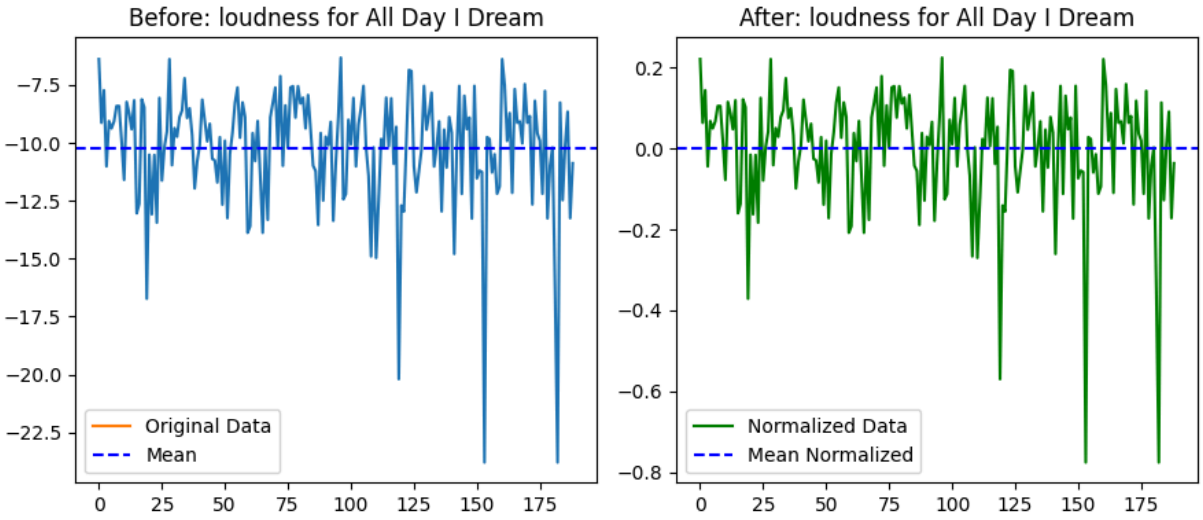

After: danceability for All Day I Dream
Normalized Data
Mean Normalized

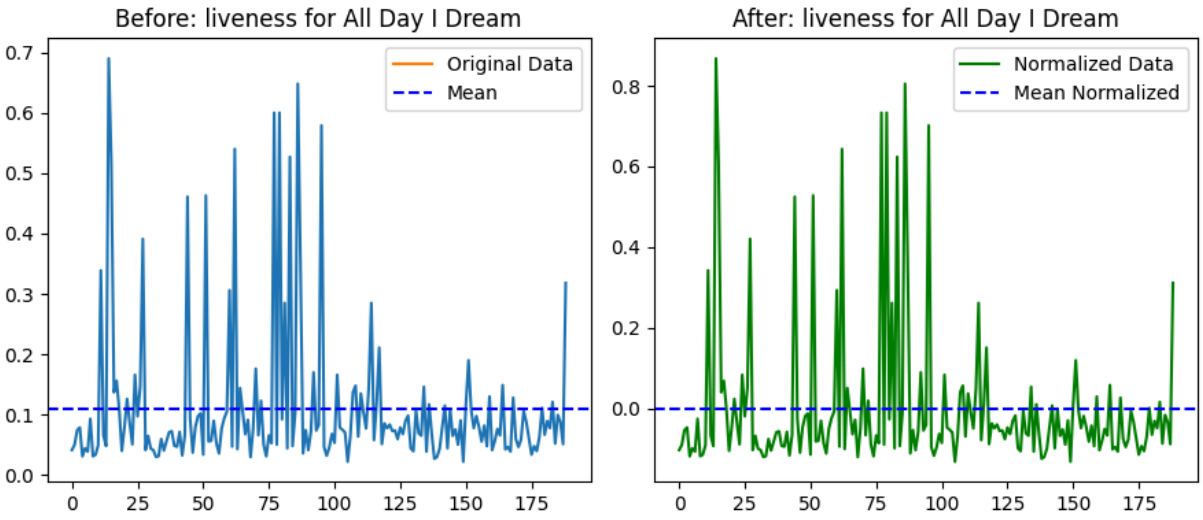

Before: energy for All Day I Dream
Original Data
Mean

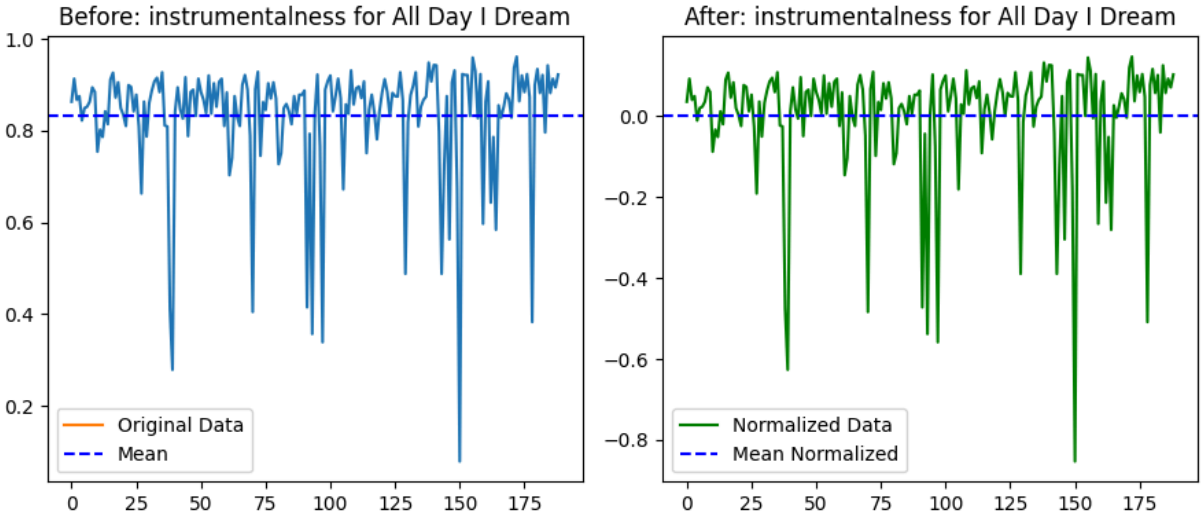

After: energy for All Day I Dream
Normalized Data
Mean Normalized

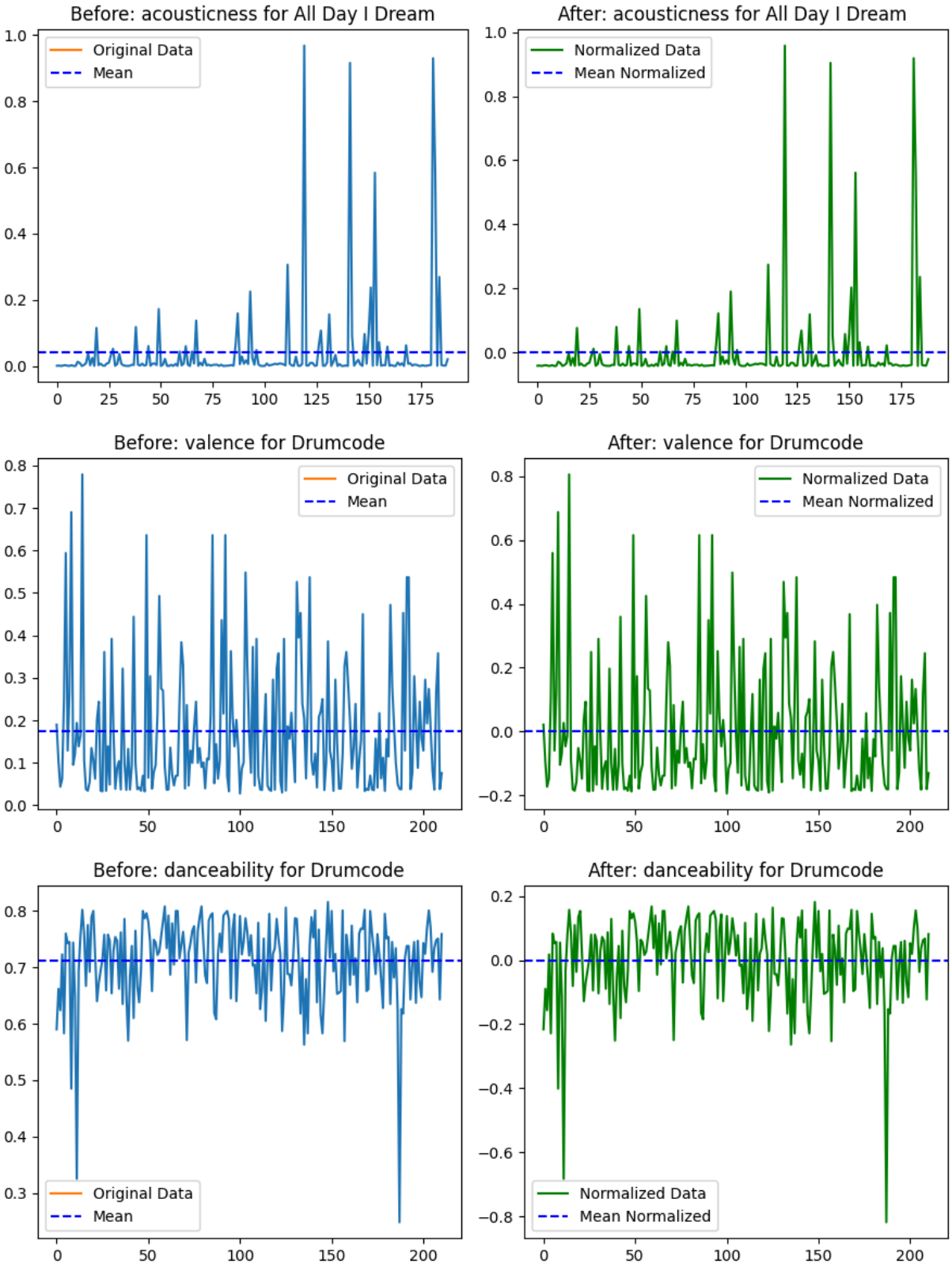

Before: speechiness for All Day I Dream
Original Data
Mean

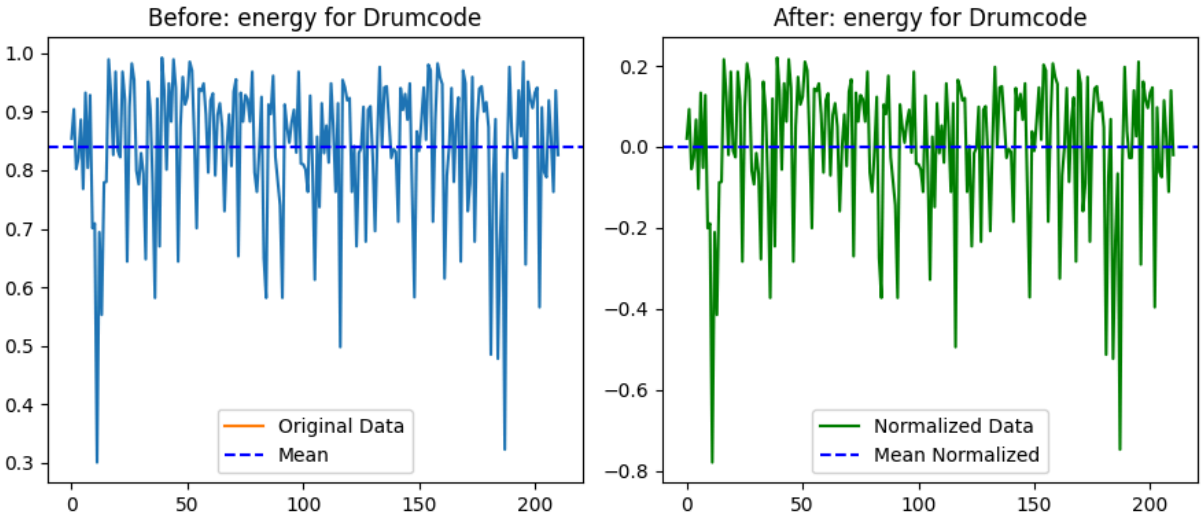

After: speechiness for All Day I Dream
Normalized Data
Mean Normalized


Before: loudness for All Day I Dream
Original Data
Mean
After: loudness for All Day I Dream
Normalized Data
Mean Normalized

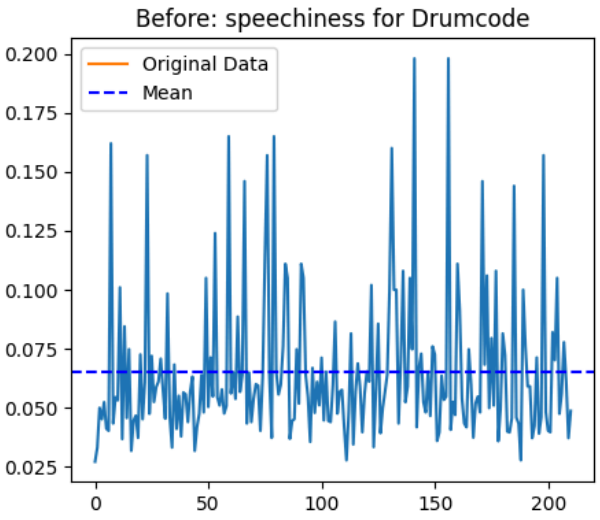

Before: liveness for All Day I Dream
Original Data
Mean
After: liveness for All Day I Dream
Normalized Data
Mean Normalized

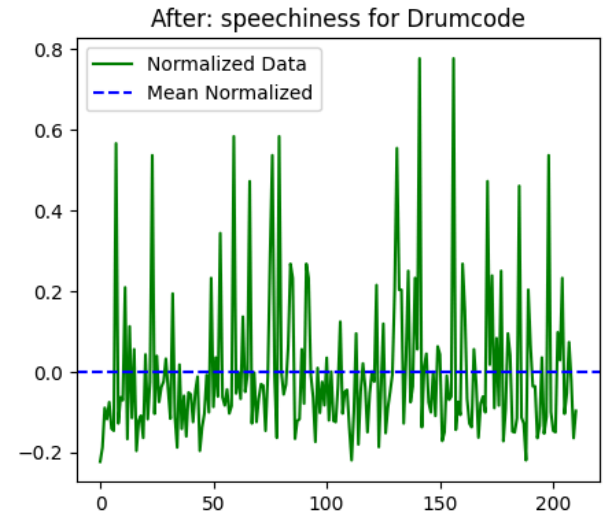

Before: instrumentalness for All Day I Dream
Original Data
Mean
After: instrumentalness for All Day I Dream
Normalized Data
Mean Normalized

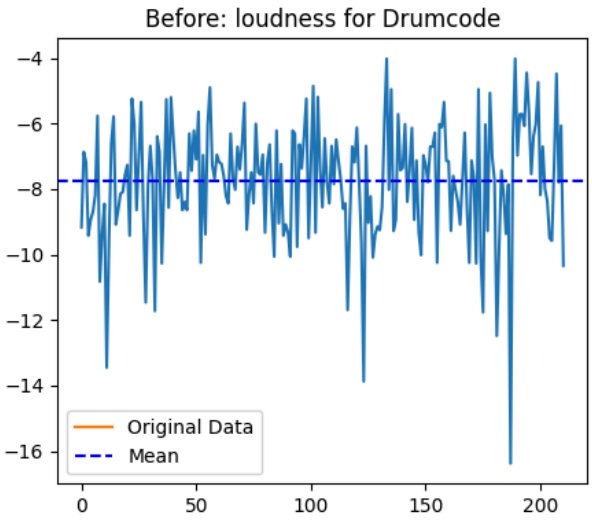

Before: acousticness for All Day I Dream
Original Data
Mean
After: acousticness for All Day I Dream
Normalized Data
Mean Normalized
Before: valence for Drumcode
Original Data
Mean
After: valence for Drumcode
Normalized Data
Mean Normalized
Before: danceability for Drumcode
Original Data
Mean
After: danceability for Drumcode
Normalized Data
Mean Normalized

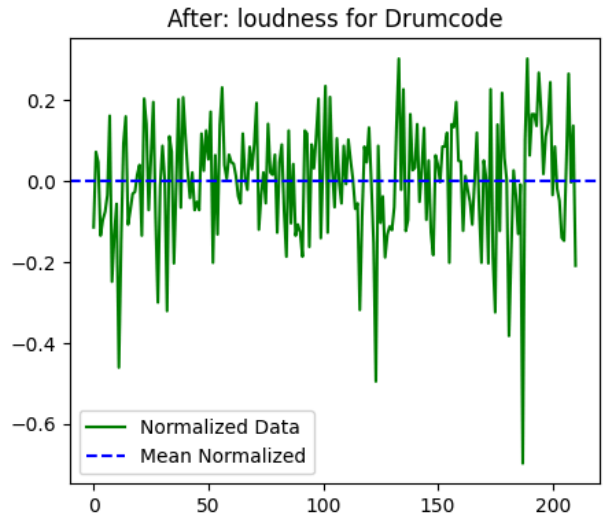

Before: energy for Drumcode
Original Data
Mean

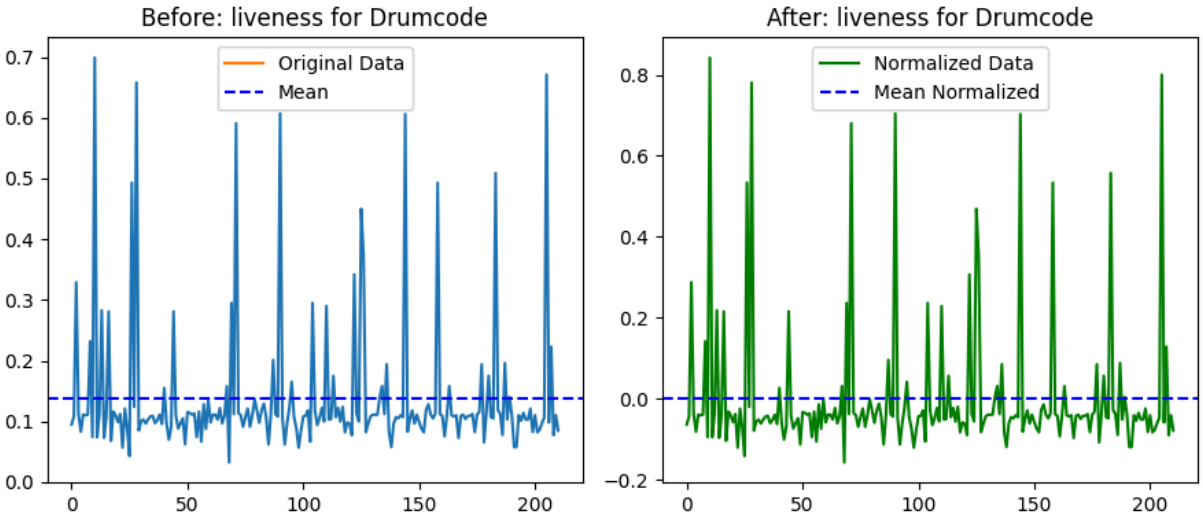

After: energy for Drumcode
Normalized Data
Mean Normalized


Before: speechiness for Drumcode
Original Data
Mean

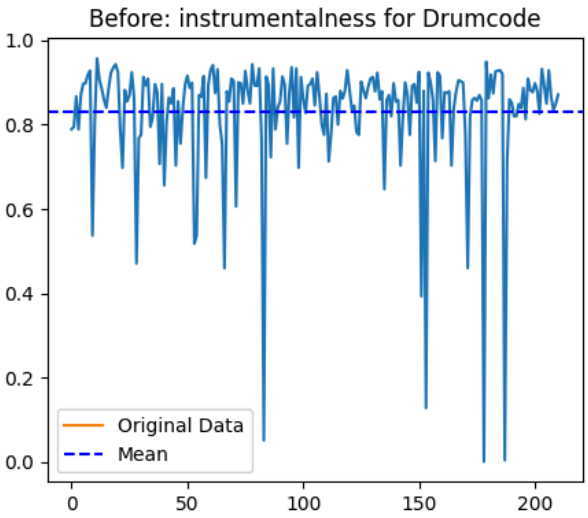

After: speechiness for Drumcode
Normalized Data
Mean Normalized

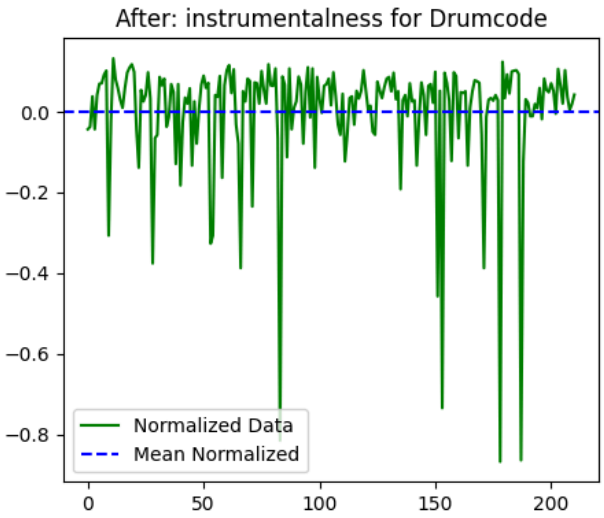

Before: loudness for Drumcode
Original Data
Mean

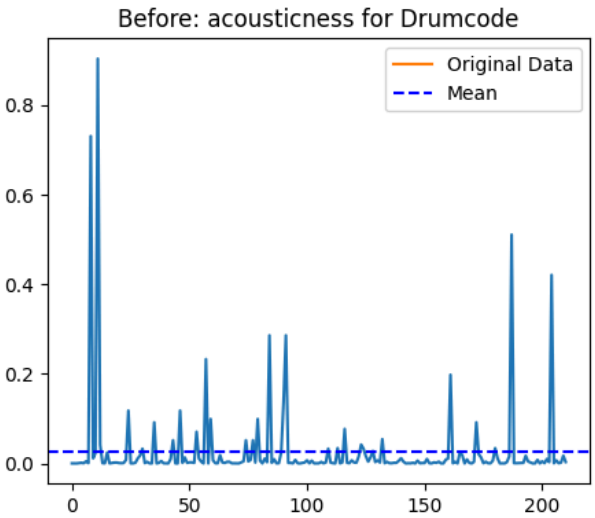

After: loudness for Drumcode
Normalized Data
Mean Normalized

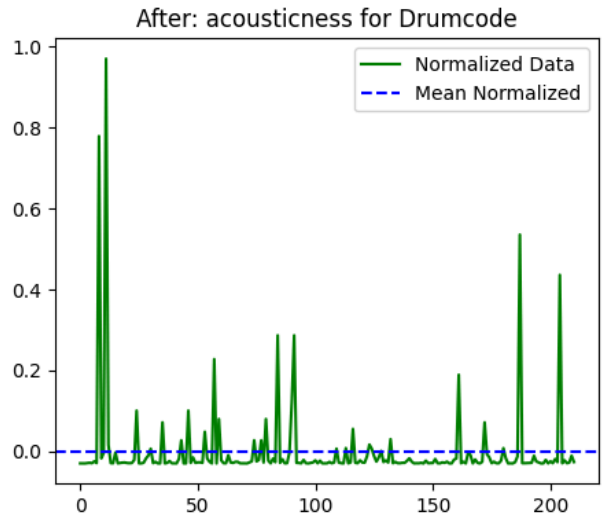

Before: liveness for Drumcode
Original Data
Mean

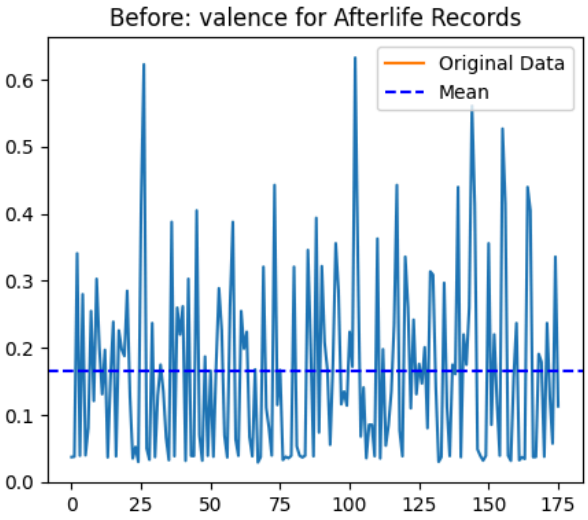

After: liveness for Drumcode
Normalized Data
Mean Normalized

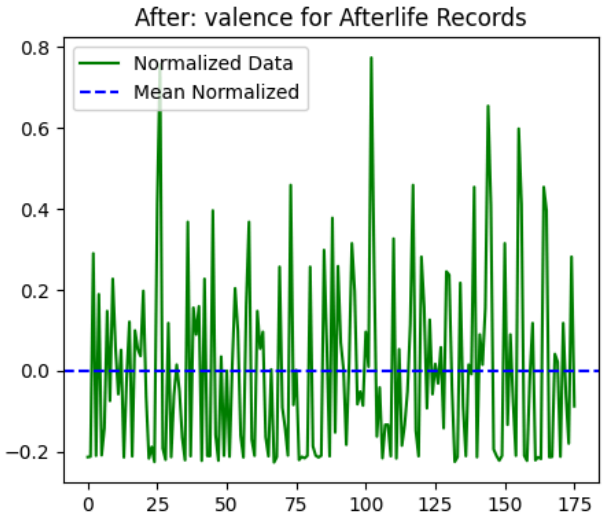

Before: instrumentalness for Drumcode
Original Data
Mean

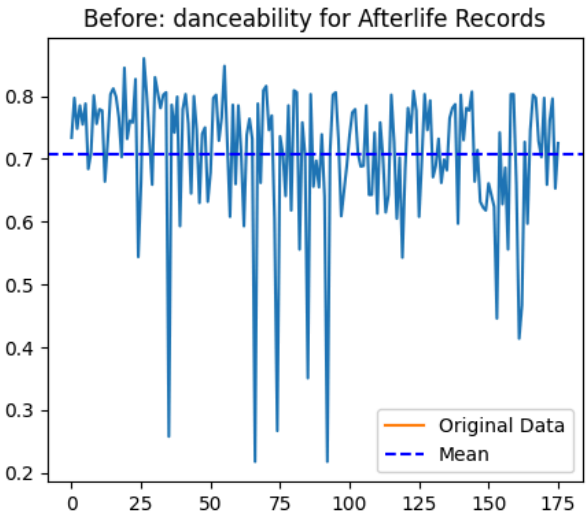

After: instrumentalness for Drumcode
Normalized Data
Mean Normalized

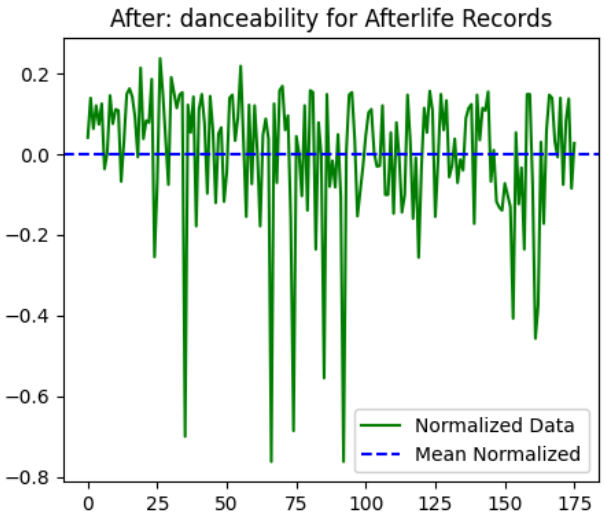

Before: acousticness for Drumcode
Original Data
Mean

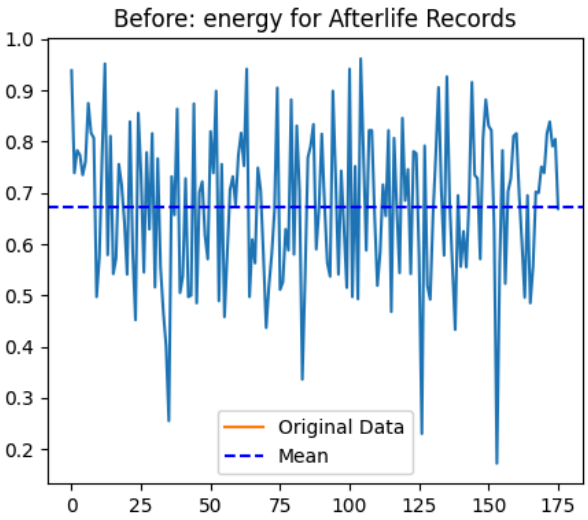

After: acousticness for Drumcode
Normalized Data
Mean Normalized

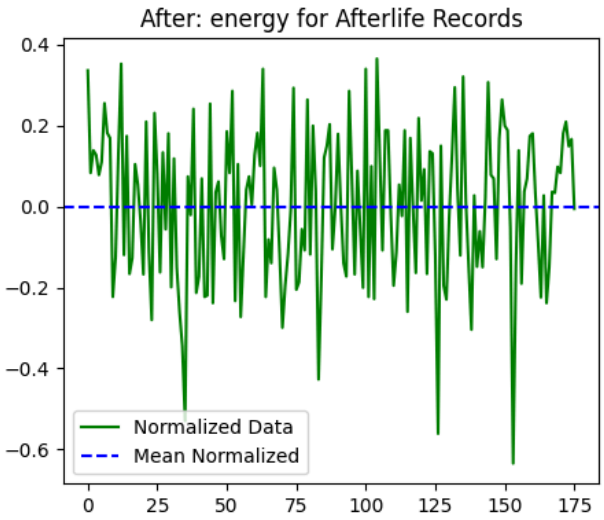

Before: valence for Afterlife Records
Original Data
Mean

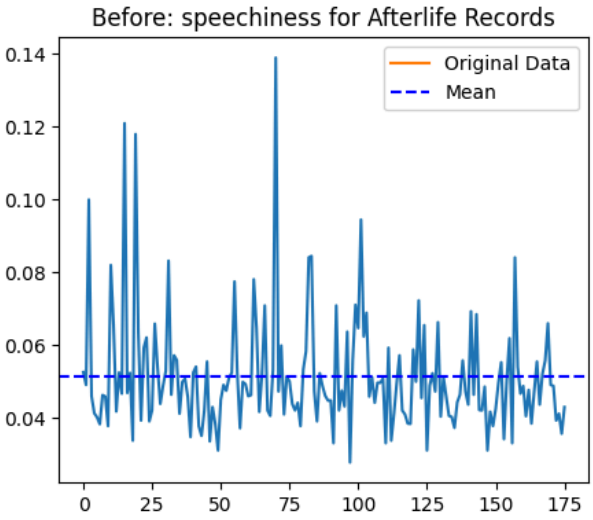

After: valence for Afterlife Records
Normalized Data
Mean Normalized

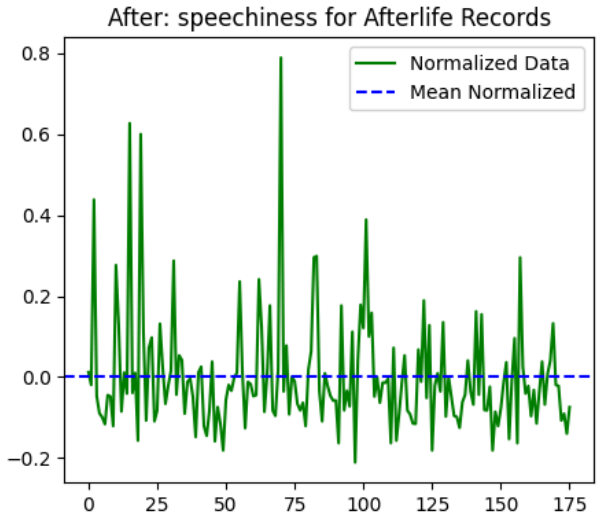

Before: danceability for Afterlife Records
Original Data
Mean

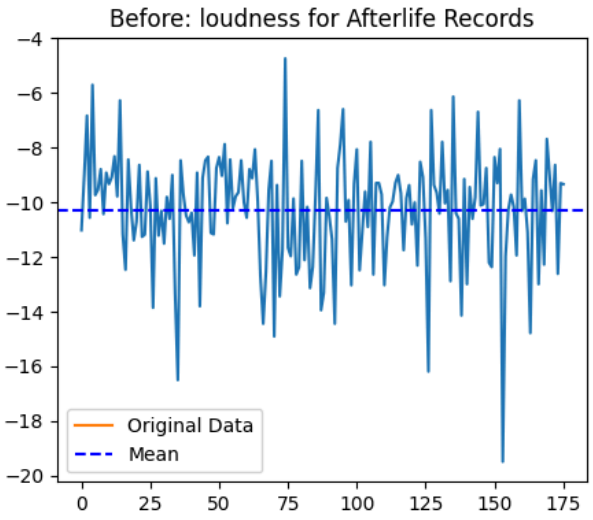

After: danceability for Afterlife Records
Normalized Data
Mean Normalized

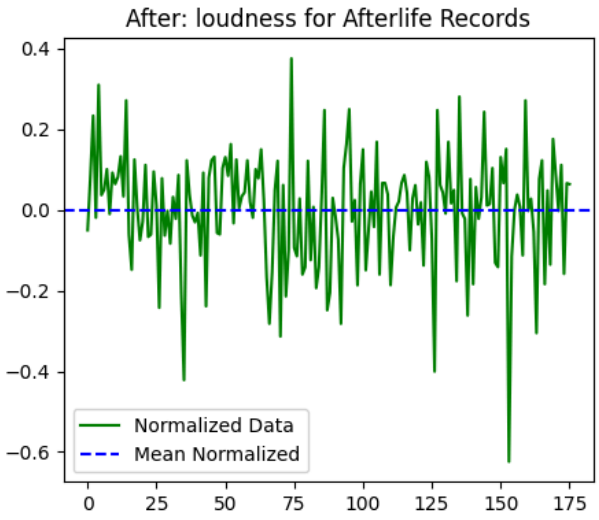

Before: energy for Afterlife Records
Original Data
Mean

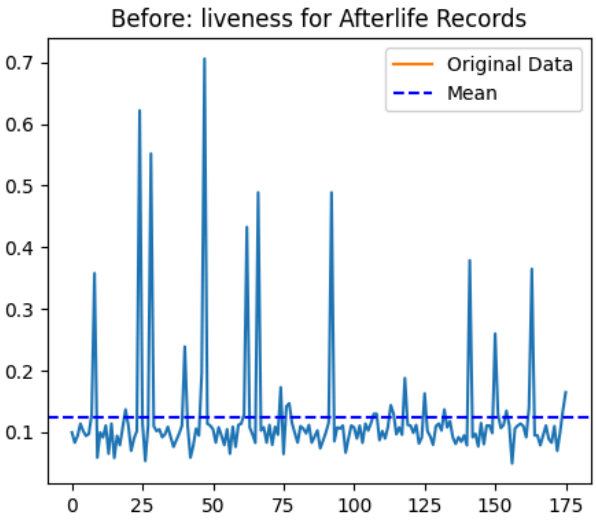

After: energy for Afterlife Records
Normalized Data
Mean Normalized

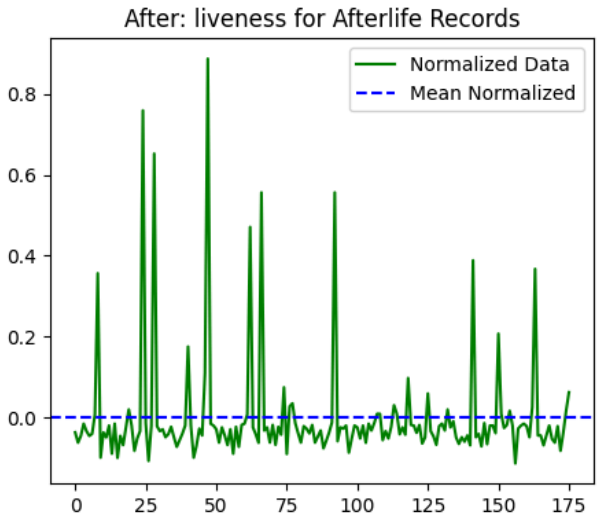

Before: speechiness for Afterlife Records
Original Data
Mean

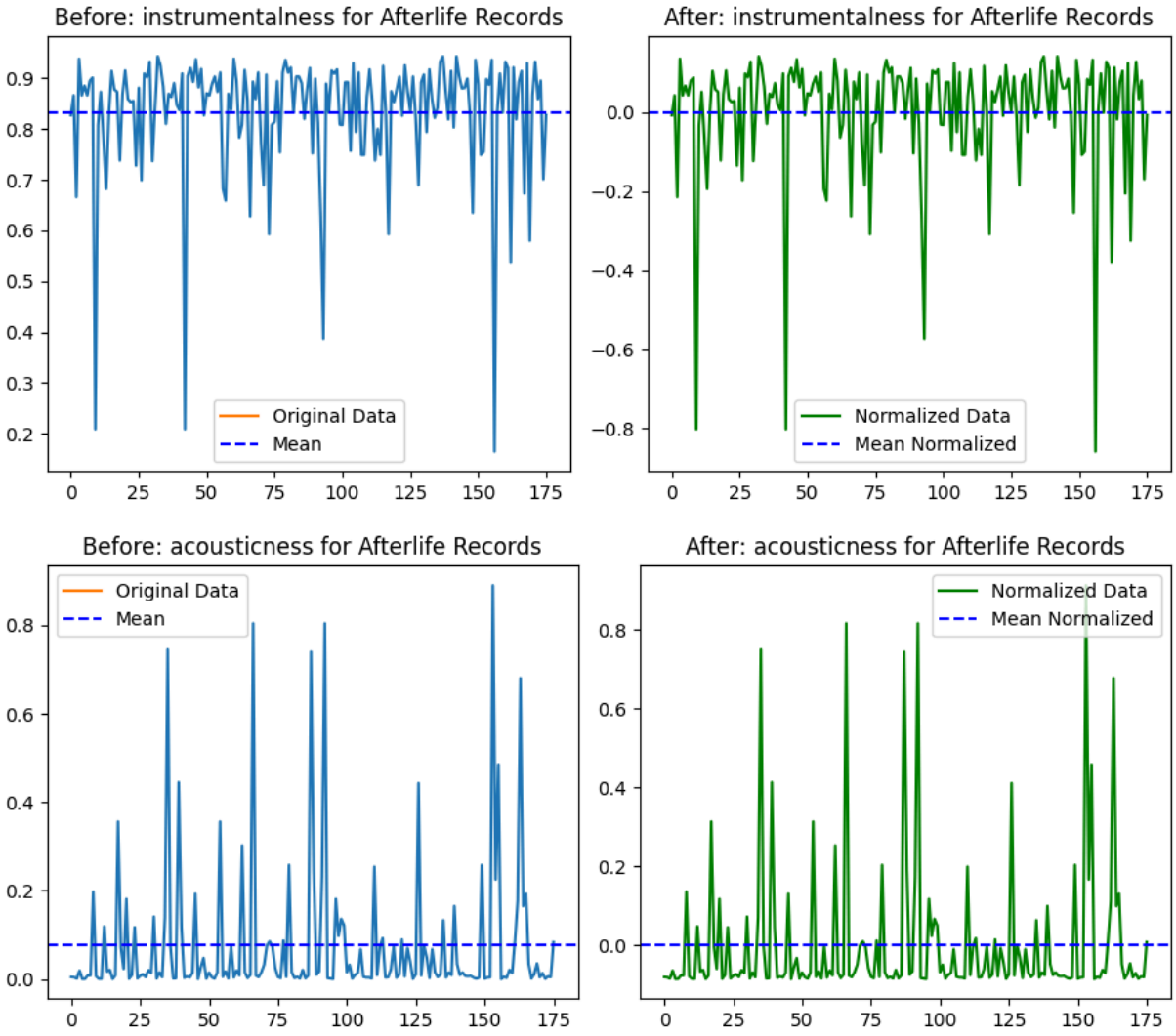

After: speechiness for Afterlife Records
Normalized Data
Mean Normalized


Before: loudness for Afterlife Records
Original Data
Mean


After: loudness for Afterlife Records
Normalized Data
Mean Normalized


Before: liveness for Afterlife Records
Original Data
Mean


After: liveness for Afterlife Records
Normalized Data
Mean Normalized

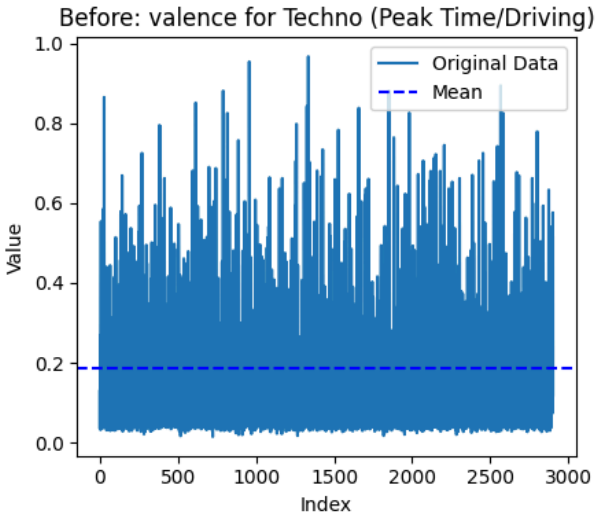


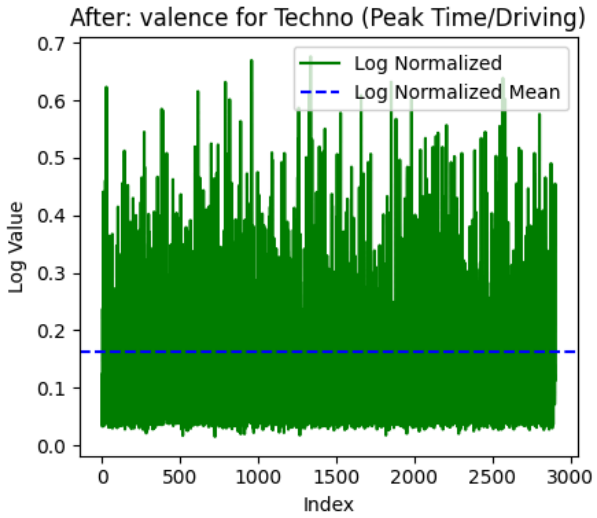

## Log Normalized

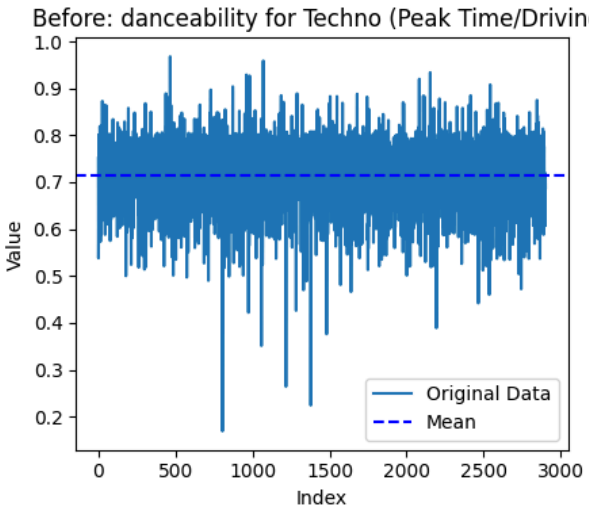
Loudness is a metric that, by default, has values ranging from positive to negative and so the charts don't populate. When I have some more knowledge and skills, I'll see if there's a way to handle values that range below -1 (the +1 in the definition above covers for -1 to zero). A bit out of my skillset range for now so I'm gonna let it rest.

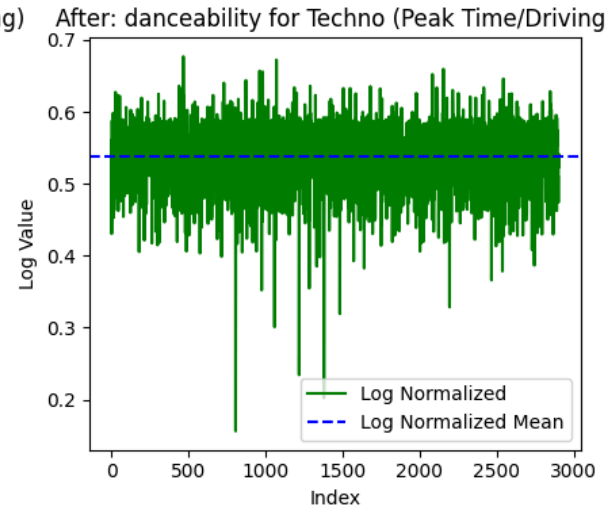
### By Genre

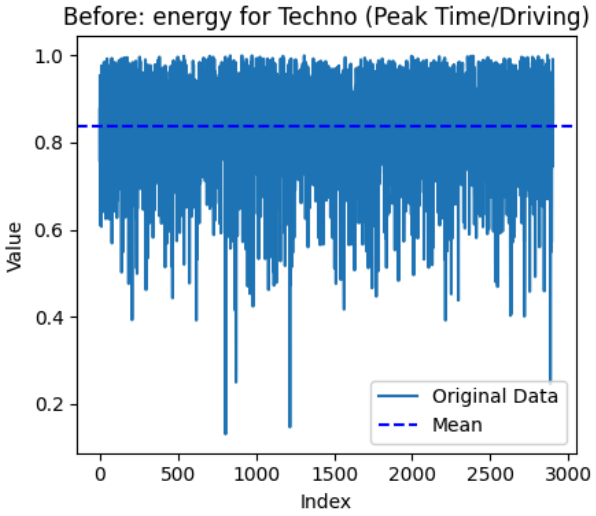
In [23]:
```
#by genre
for genre_id in subset_genres:
    genre_name = subset_genres_dict.get(genre_id, f"Genre {genre_id}")
    genre_matrix = log_train[log_train[:, toptracks_bpmeta_matrix_df_dict['g
    x_features = genre_matrix[:, core_x_features]
    for i, feature_index in enumerate(core_x_features):
        feature = genre_matrix[:, feature_index]
        feature_name = list(toptracks_bpmeta_matrix_df_dict.keys())[list(top
        title = f"{feature_name} for {genre_name}"
        log_plot(feature, title)
```

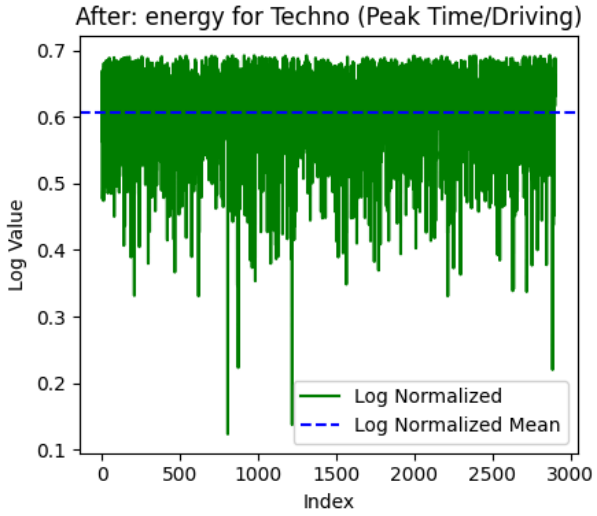

Before: valence for Techno (Peak Time/Driving)
Original Data
Mean
Value
Index

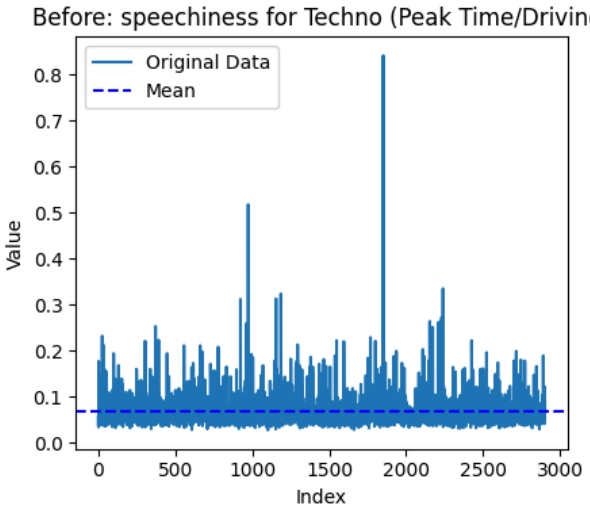

After: valence for Techno (Peak Time/Driving)
Log Normalized
Log Normalized Mean
Log Value
Index

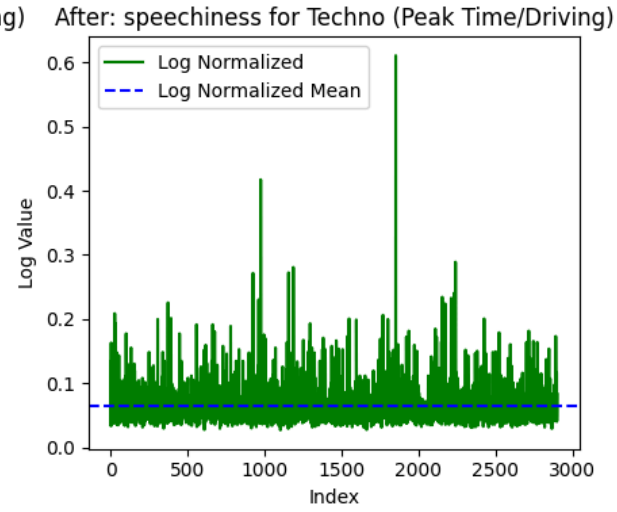

Before: danceability for Techno (Peak Time/Driving)
Original Data
Mean
Value
Index


After: danceability for Techno (Peak Time/Driving)
Log Normalized
Log Normalized Mean
Log Value
Index

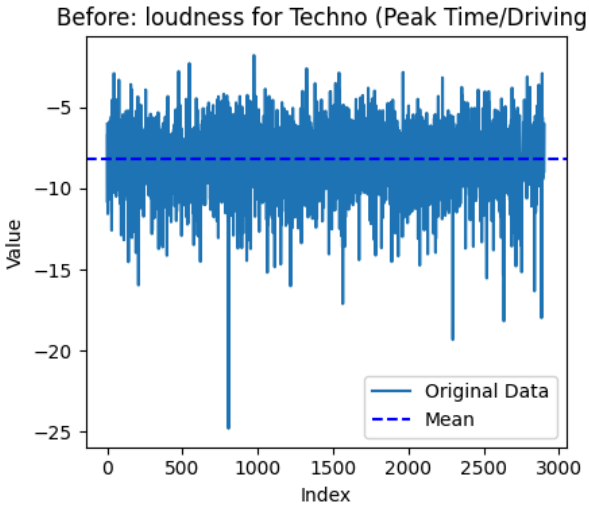

Before: energy for Techno (Peak Time/Driving)
Original Data
Mean
Value
Index

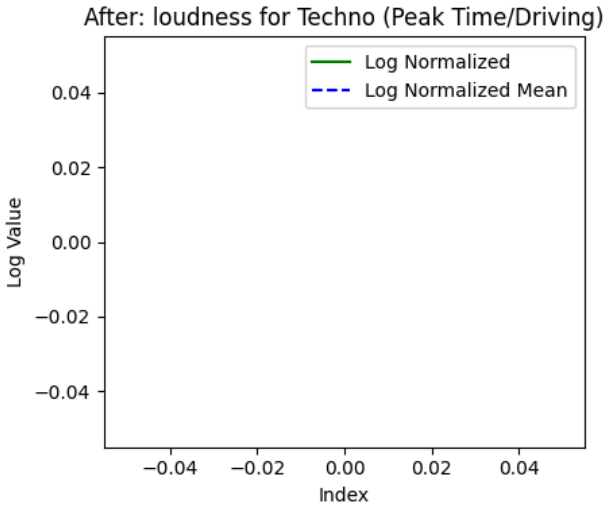

After: energy for Techno (Peak Time/Driving)
Log Normalized
Log Normalized Mean
Log Value
Index

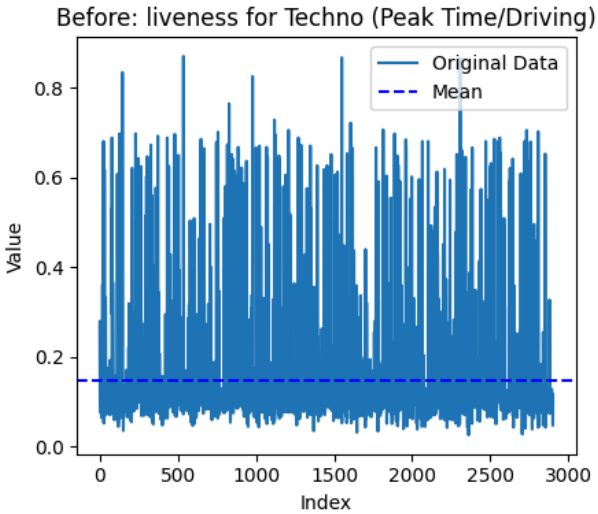


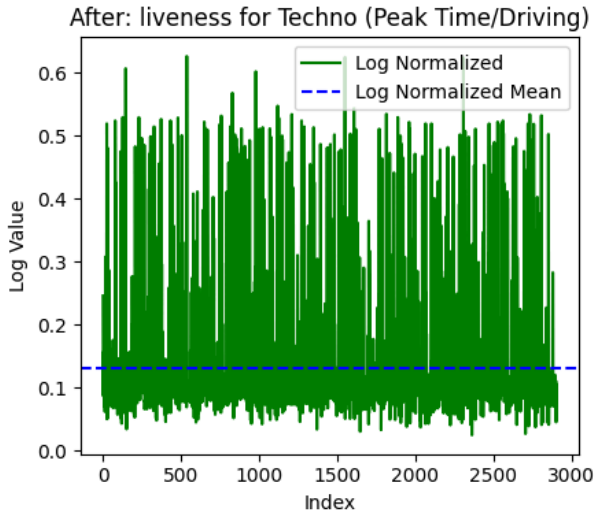



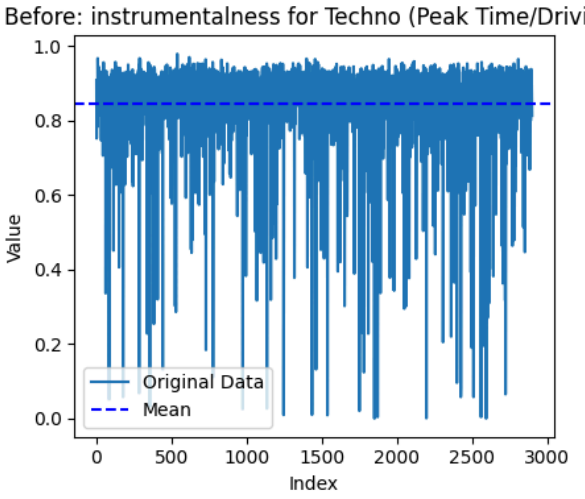

```
/var/folders/2q/1m0wb0x5547gycyp2q9vyy340000gn/T/ipykernel_2455/1217021281.p
y:2: RuntimeWarning: invalid value encountered in log
  x_log = lumpnump.log(x + 1)
```

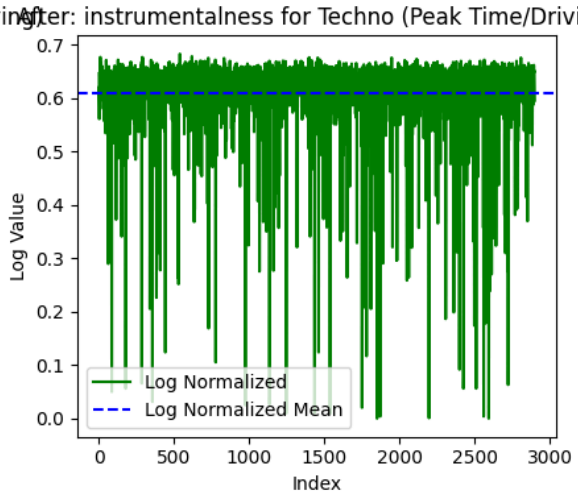


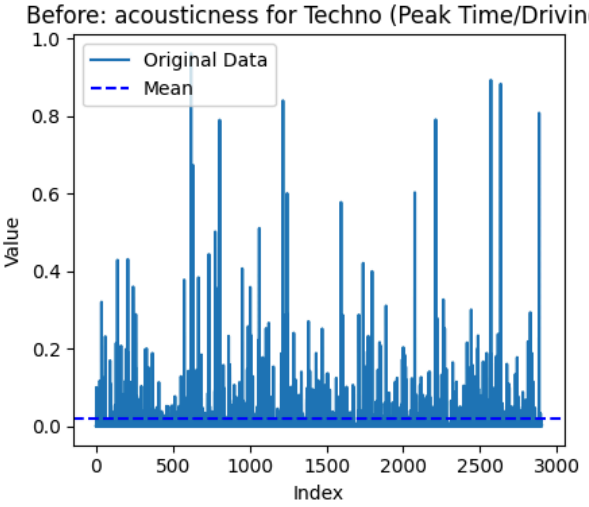



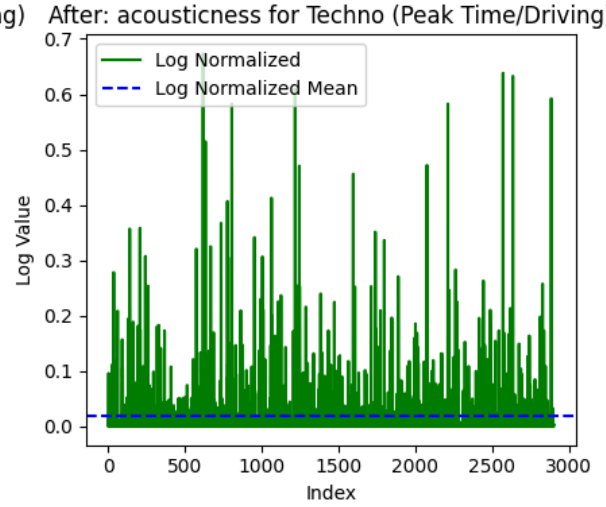



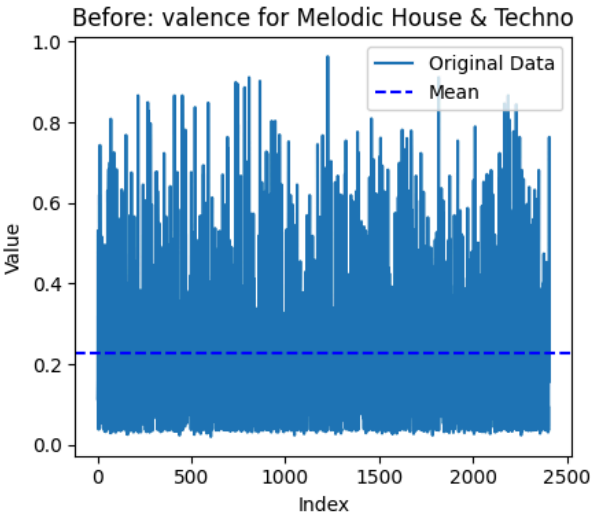



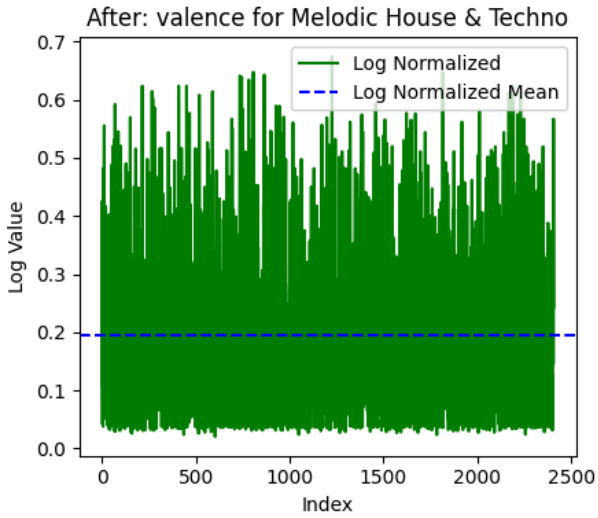

Before: instrumentalness for Techno (Peak Time/Driving)
Original Data
Mean
Value
Index

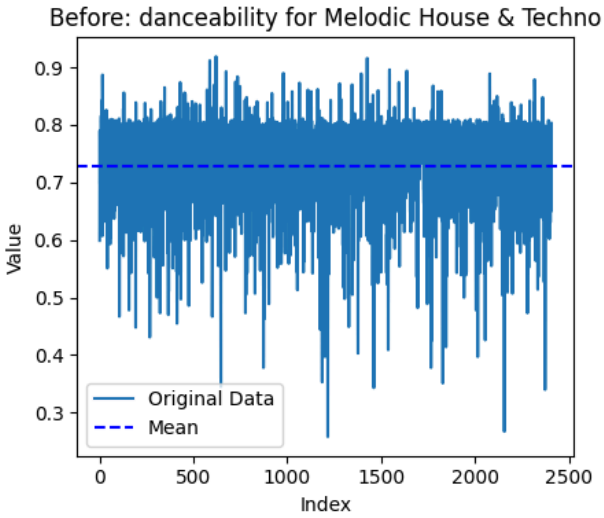

After: instrumentalness for Techno (Peak Time/Driving)
Log Normalized
Log Normalized Mean
Log Value
Index

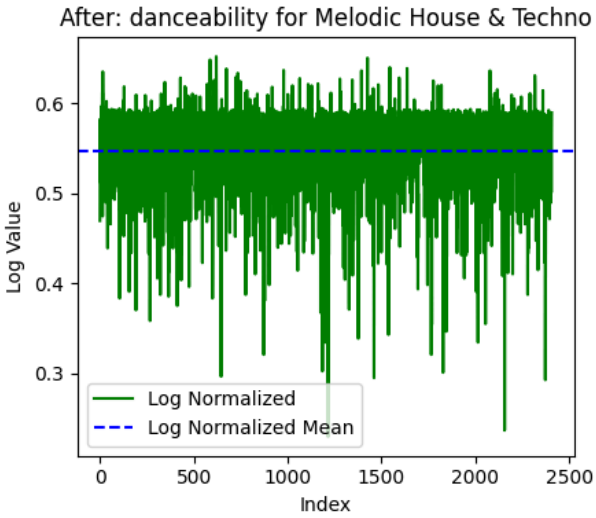

Before: acousticness for Techno (Peak Time/Driving)
Original Data
Mean
Value
Index

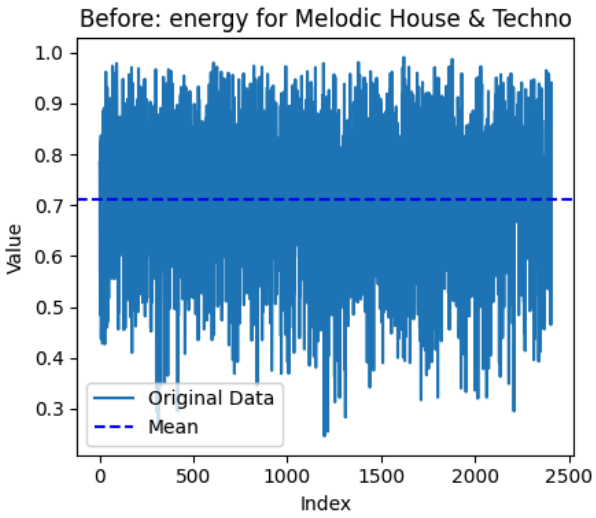

After: acousticness for Techno (Peak Time/Driving)
Log Normalized
Log Normalized Mean
Log Value
Index

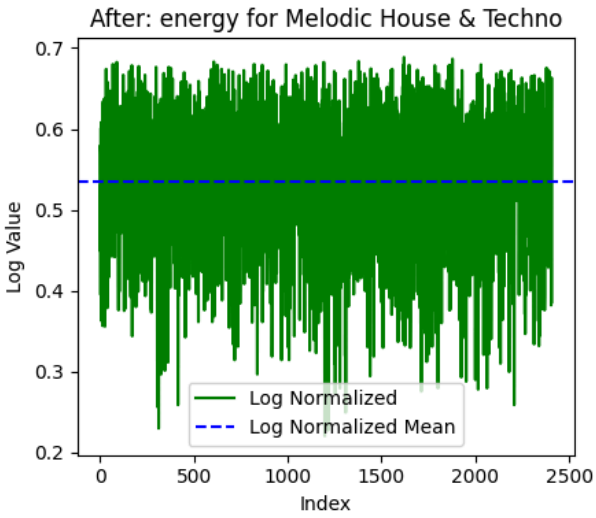

Before: valence for Melodic House & Techno
Original Data
Mean
Value
Index

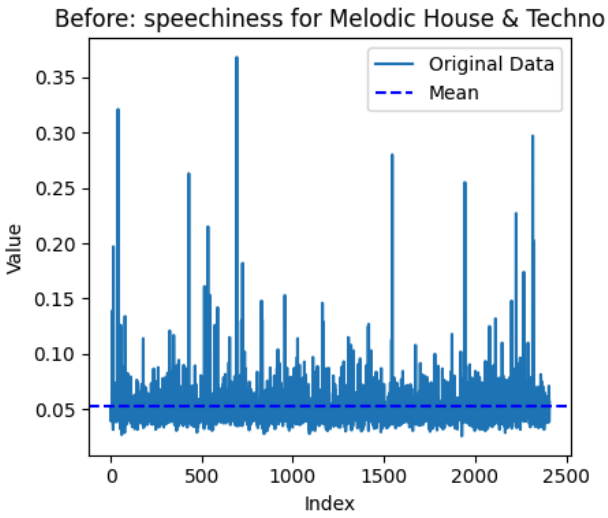

After: valence for Melodic House & Techno
Log Normalized
Log Normalized Mean
Log Value
Index

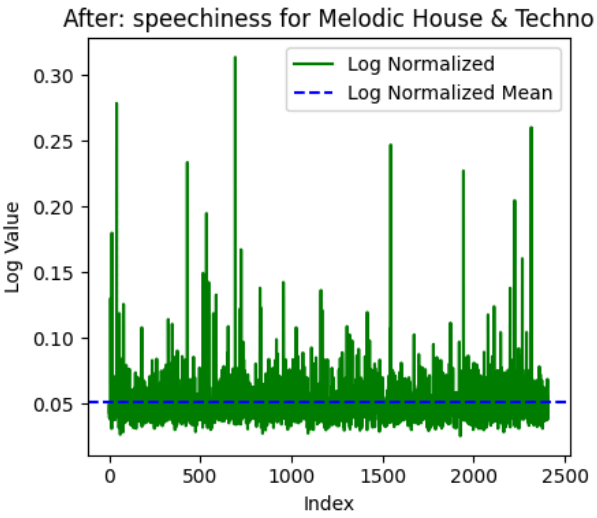

Before: danceability for Melodic House & Techno
Value
Index
Original Data
Mean

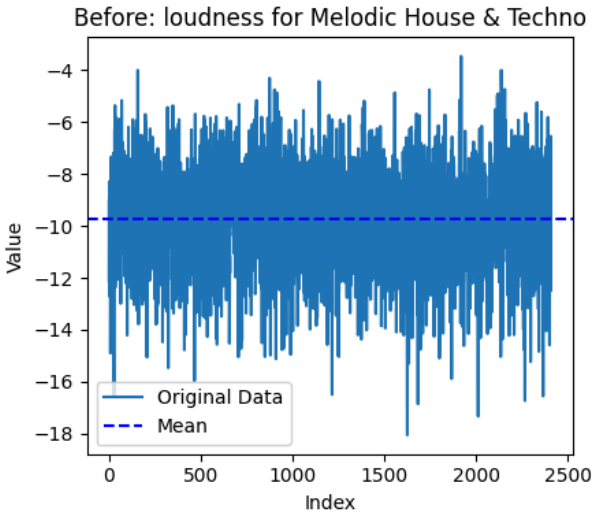

After: danceability for Melodic House & Techno
Log Value
Index
Log Normalized
Log Normalized Mean

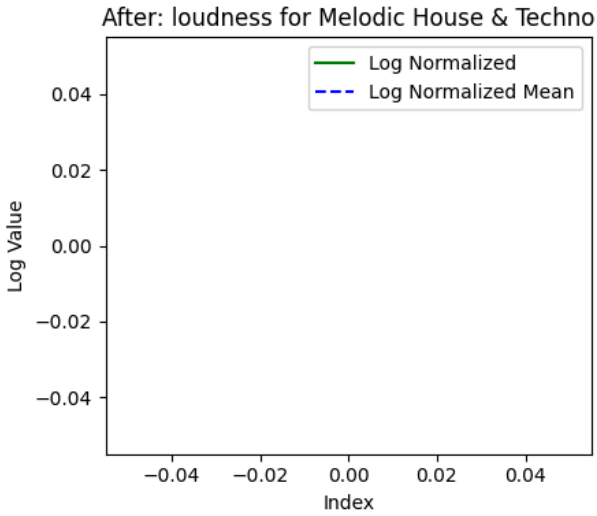

Before: energy for Melodic House & Techno
Value
Index
Original Data
Mean

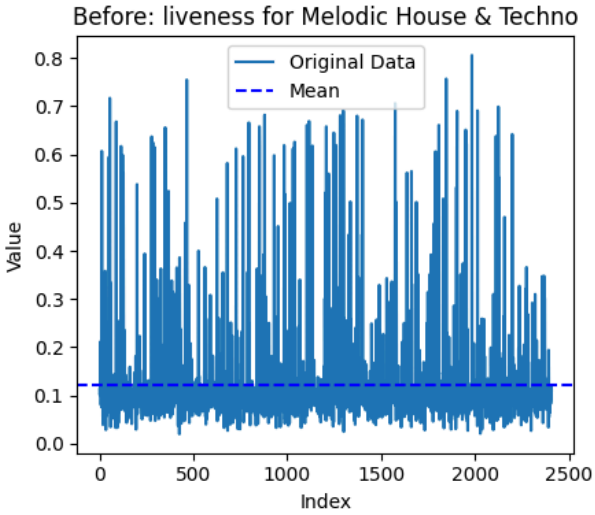

After: energy for Melodic House & Techno
Log Value
Index
Log Normalized
Log Normalized Mean

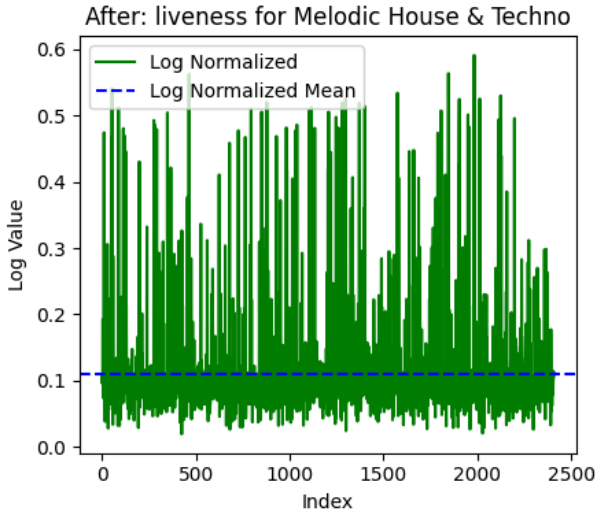

Before: speechiness for Melodic House & Techno
Value
Index
Original Data
Mean

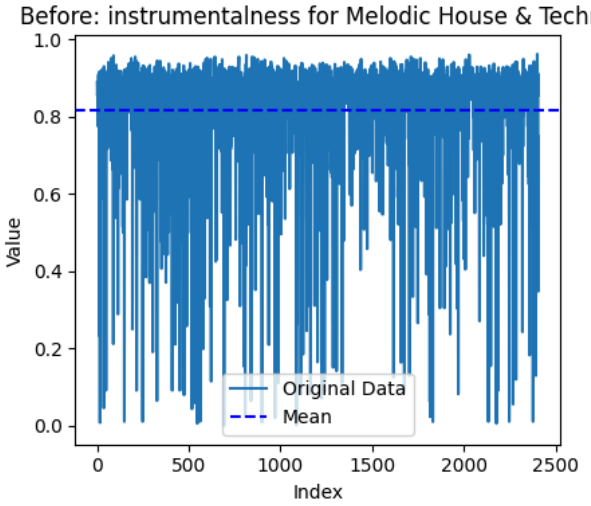

After: speechiness for Melodic House & Techno
Log Value
Index
Log Normalized
Log Normalized Mean

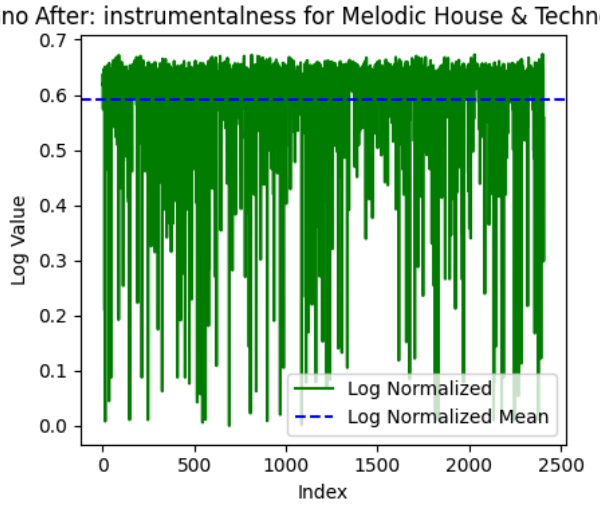

Before: loudness for Melodic House & Techno
Value
Index
Original Data
Mean

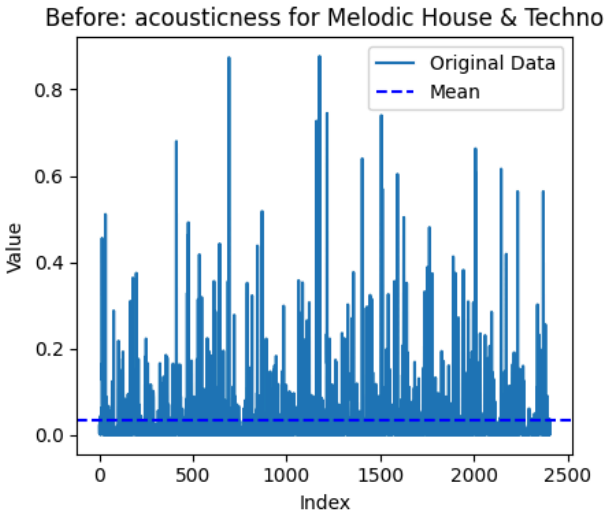

After: loudness for Melodic House & Techno
Log Value
Index
Log Normalized
Log Normalized Mean

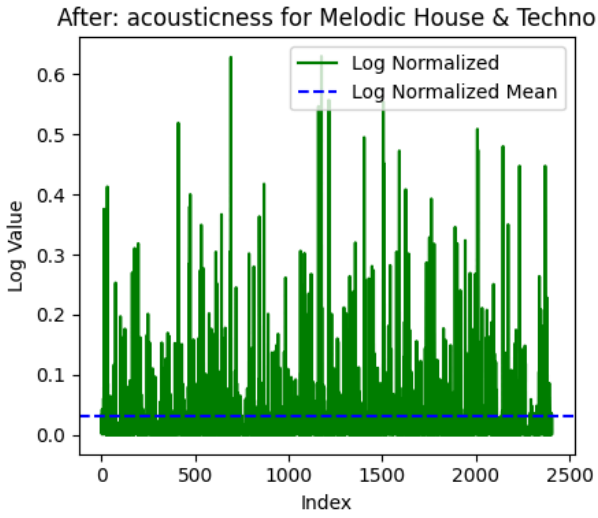

Before: liveness for Melodic House & Techno
Value
Index
Original Data
Mean

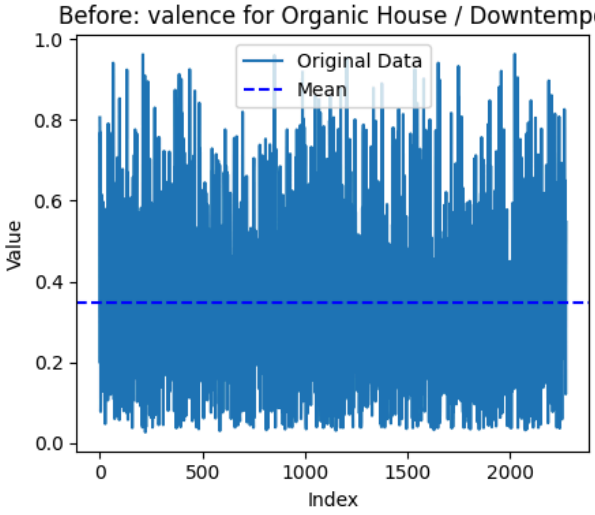

After: liveness for Melodic House & Techno
Log Value
Index
Log Normalized
Log Normalized Mean

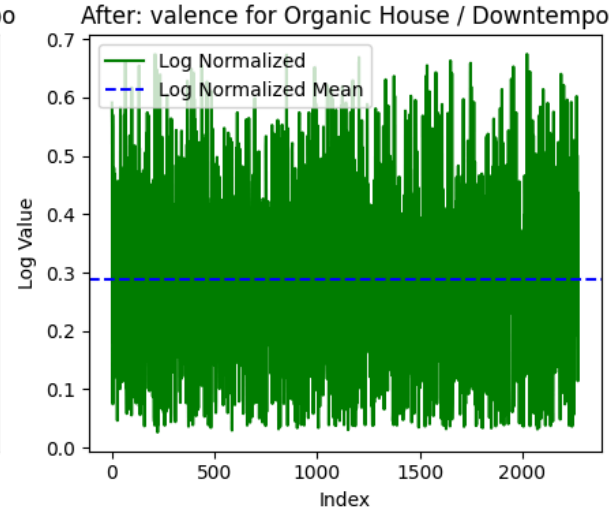

Before: instrumentalness for Melodic House & Techno
Value
Index
Original Data
Mean

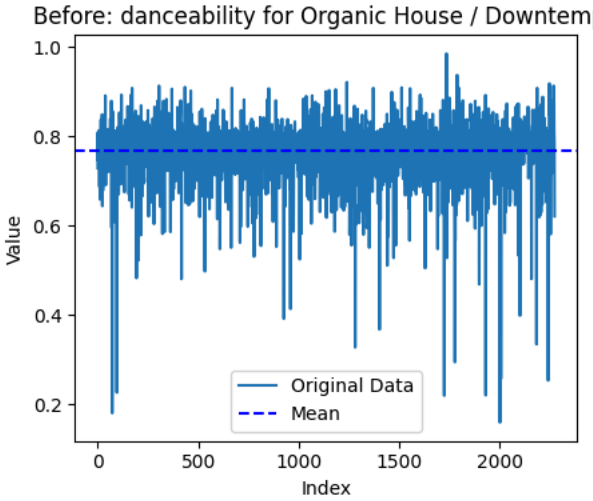

After: instrumentalness for Melodic House & Techno
Log Value
Index
Log Normalized
Log Normalized Mean

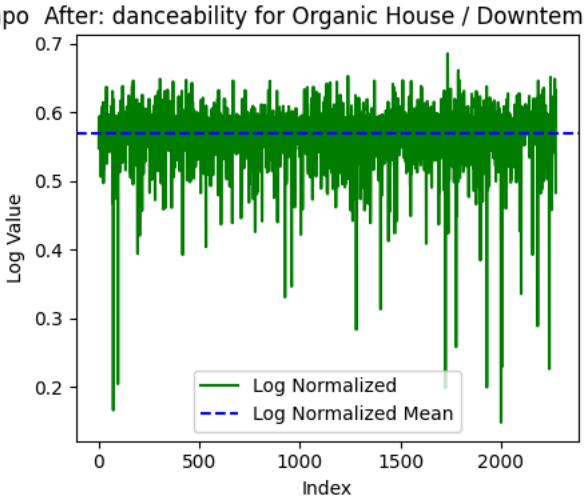

Before: acousticness for Melodic House & Techno
Original Data
Mean
Value
Index

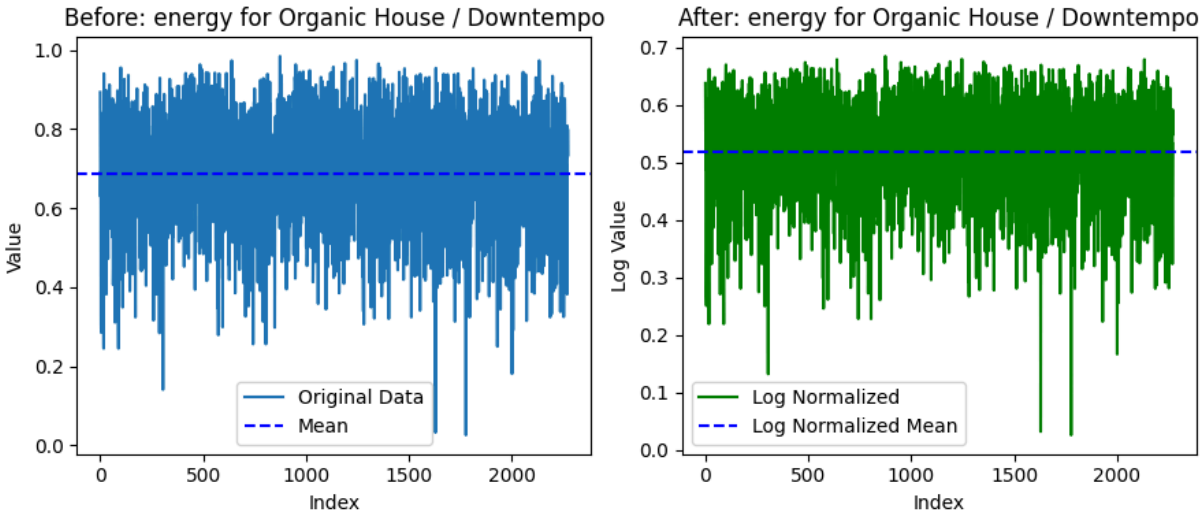

After: acousticness for Melodic House & Techno
Log Normalized
Log Normalized Mean
Log Value
Index

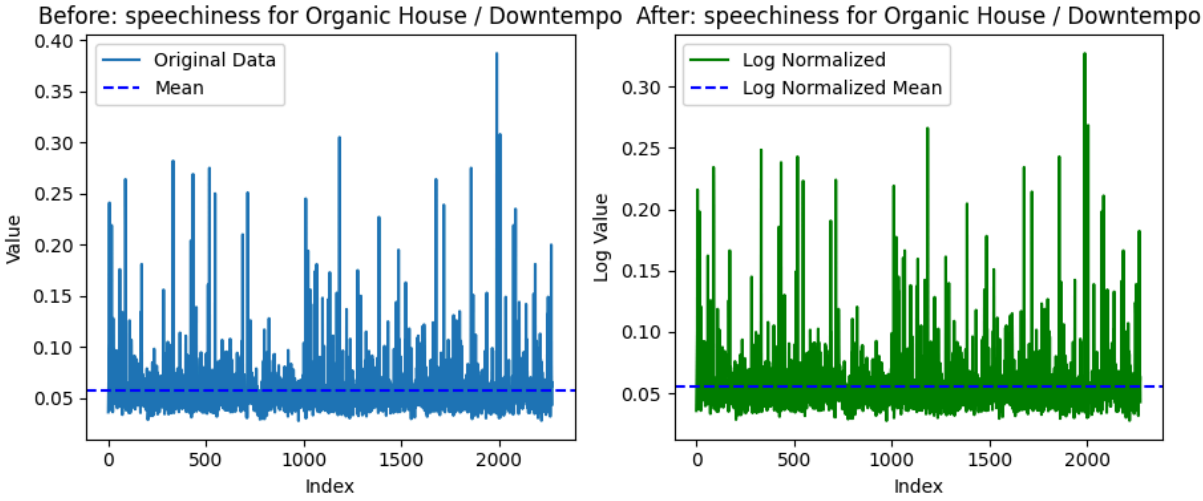

Before: valence for Organic House / Downtempo
Original Data
Mean
Value
Index

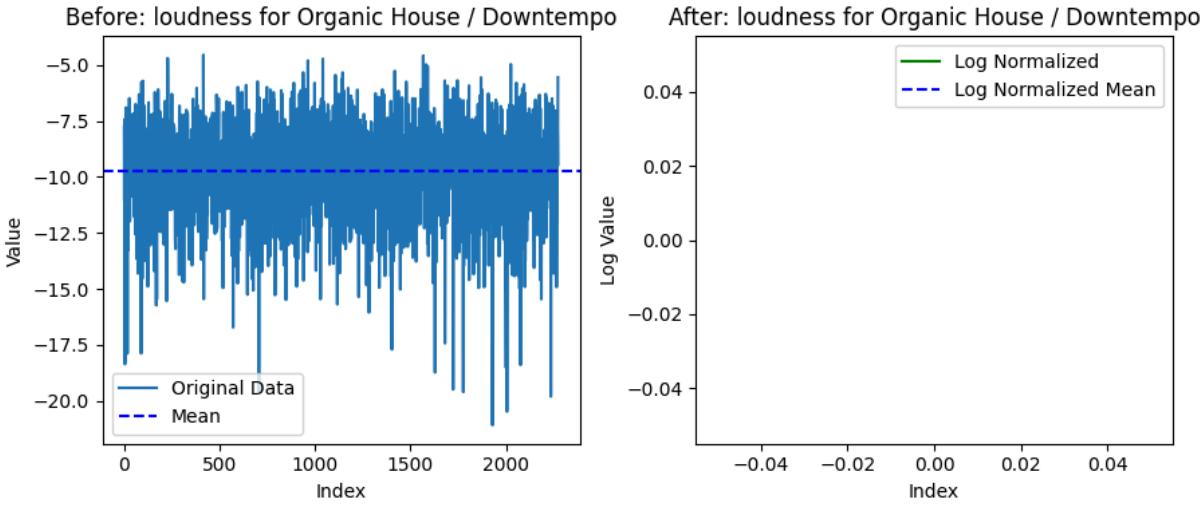

After: valence for Organic House / Downtempo
Log Normalized
Log Normalized Mean
Log Value
Index

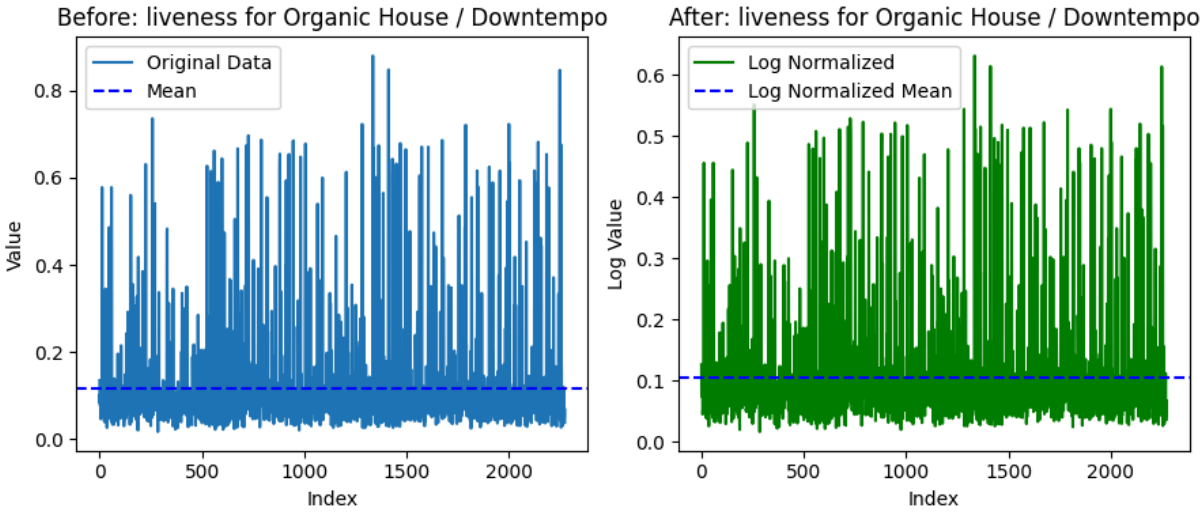

Before: danceability for Organic House / Downtempo
Original Data
Mean
Value
Index

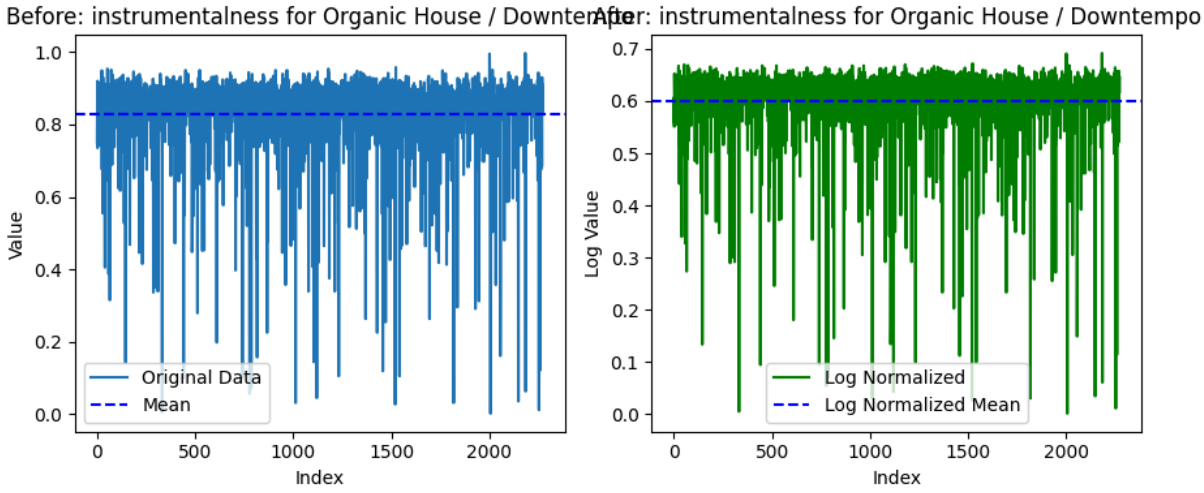

After: danceability for Organic House / Downtempo
Log Normalized
Log Normalized Mean
Log Value
Index

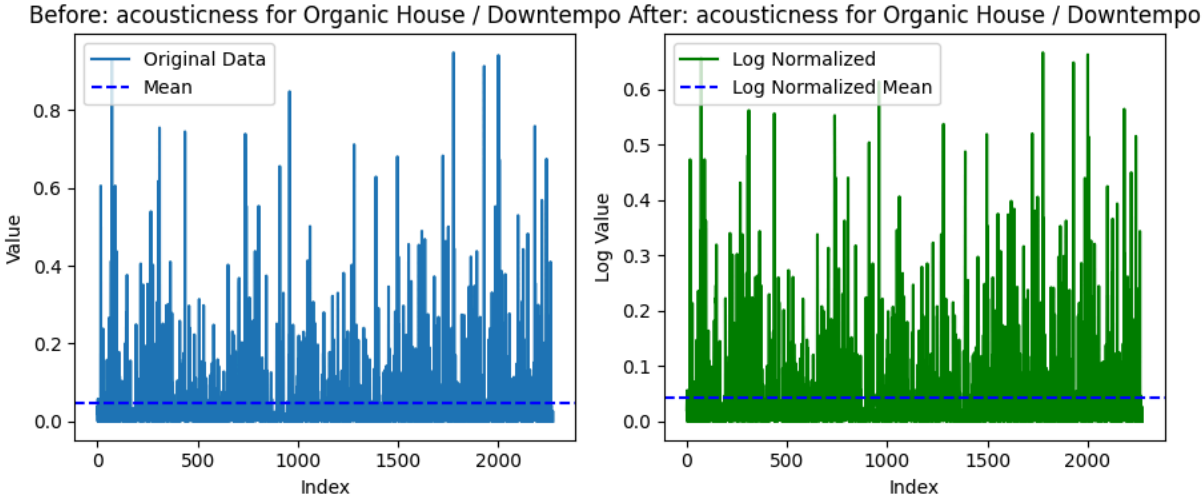

Before: energy for Organic House / Downtempo
Value
Index
Original Data
Mean
After: energy for Organic House / Downtempo
Log Value
Index
Log Normalized
Log Normalized Mean


Before: speechiness for Organic House / Downtempo
Value
Index
Original Data
Mean
After: speechiness for Organic House / Downtempo
Log Value
Index
Log Normalized
Log Normalized Mean


Before: loudness for Organic House / Downtempo
Value
Index
Original Data
Mean
After: loudness for Organic House / Downtempo
Log Value
Index
Log Normalized
Log Normalized Mean



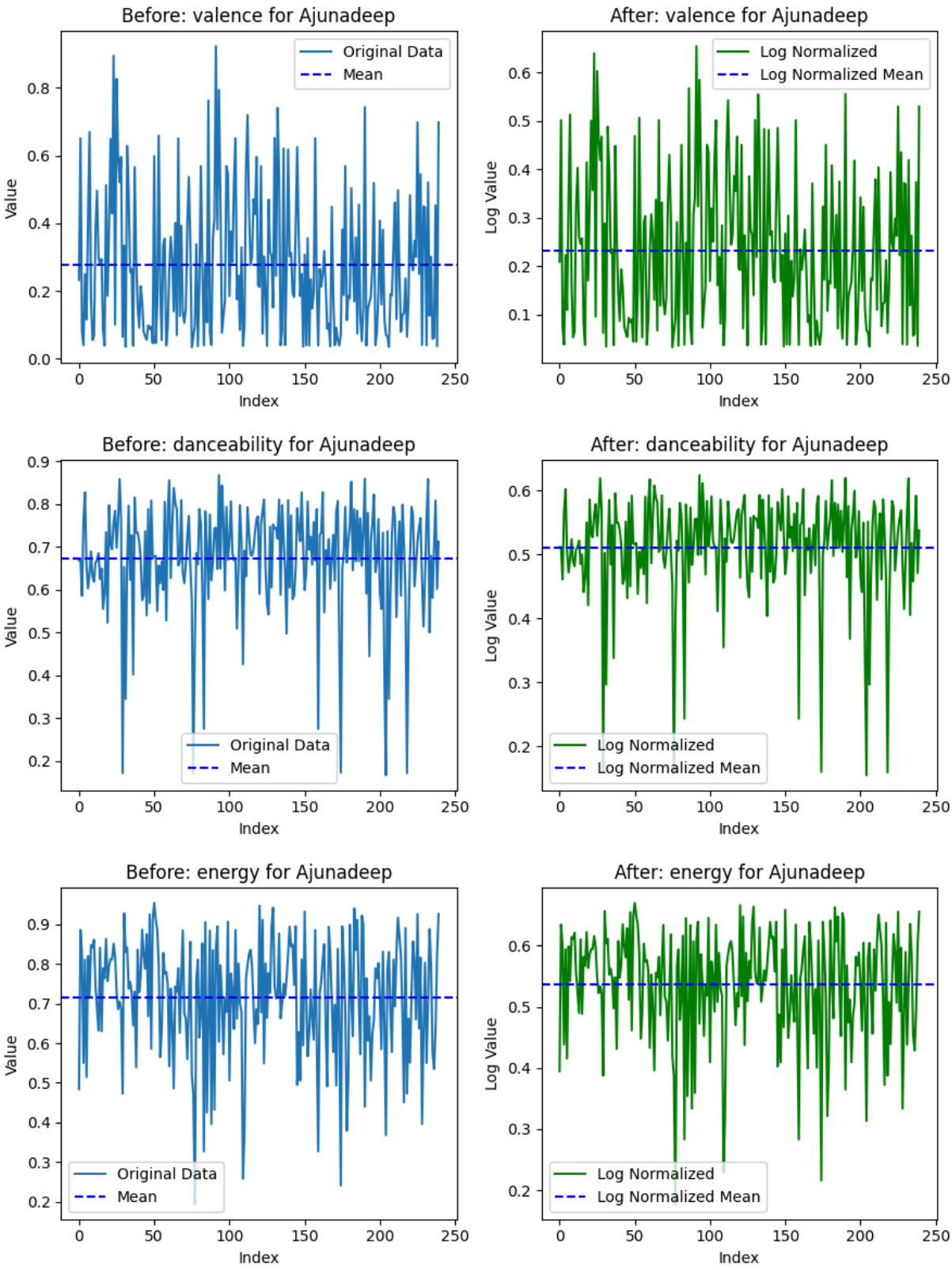



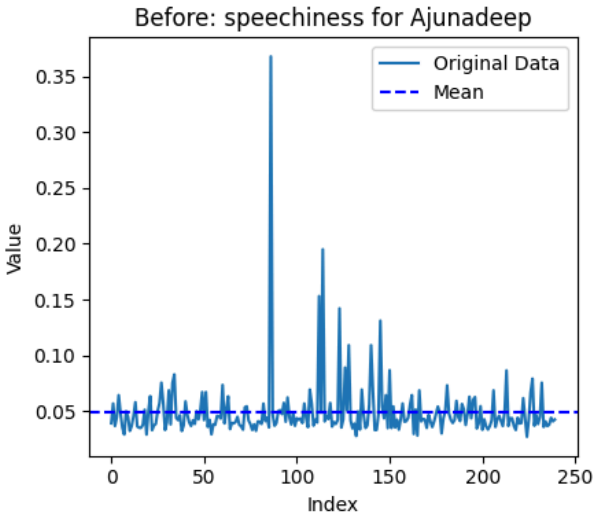



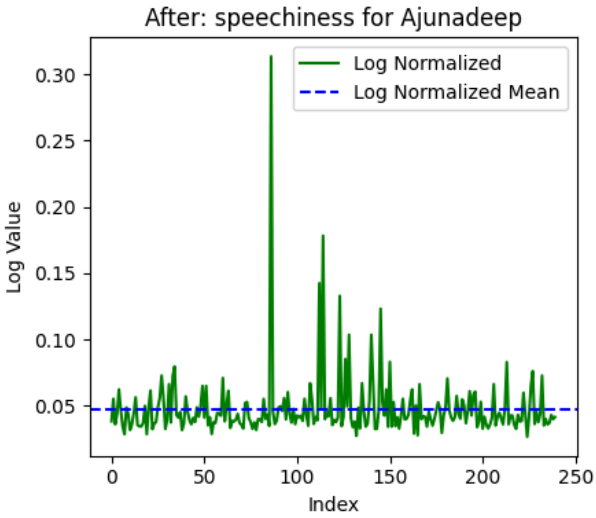

## By Label

In [24]:
```
#by label
for label_id in subset_labels:
    label_name = label_ids_dict.get(label_id, f"Label {label_id}")
    label_matrix = log_train[log_train[:, toptracks_bpmeta_matrix_df_dict['l
    x_features = label_matrix[:, core_x_features]
    for i, feature_index in enumerate(core_x_features):
```

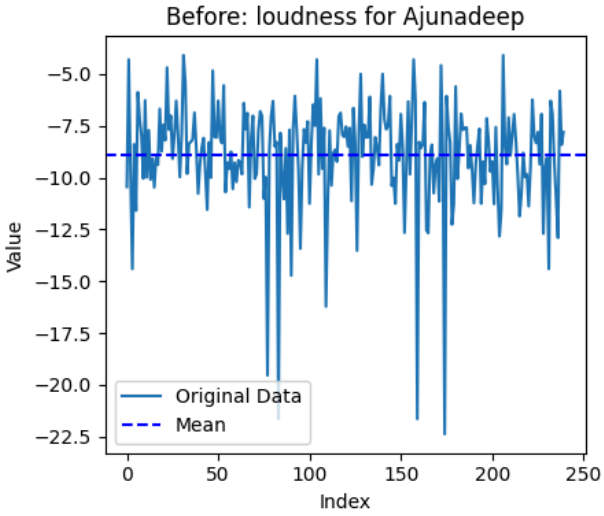
```
        feature = label_matrix[:, feature_index]
        feature_name = list(toptracks_bpmeta_matrix_df_dict.keys())[list(top
        title = f"{feature_name} for {label_name}"
        log_plot(feature, title)
```

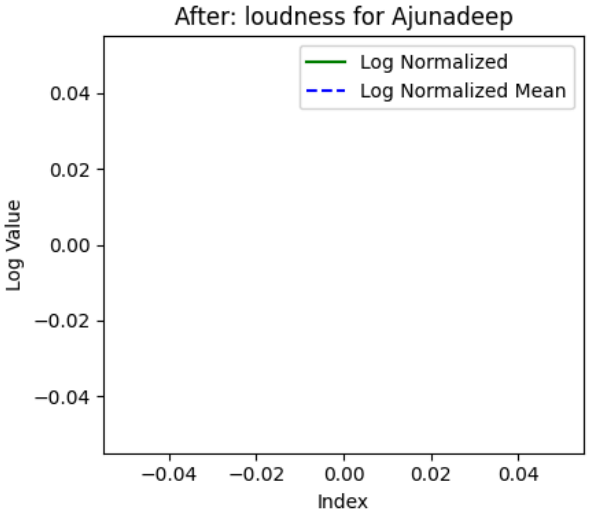


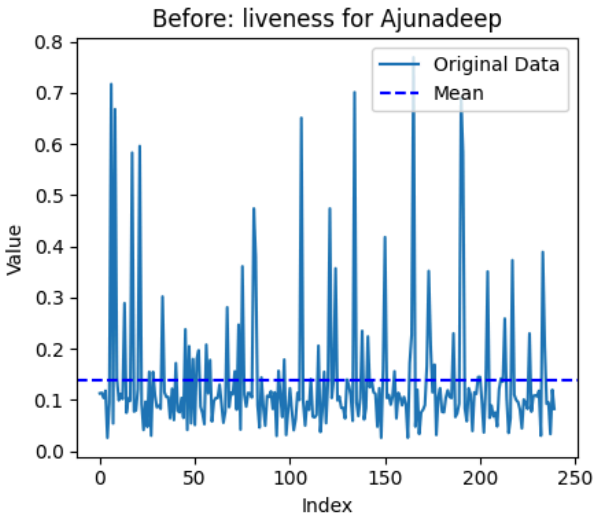


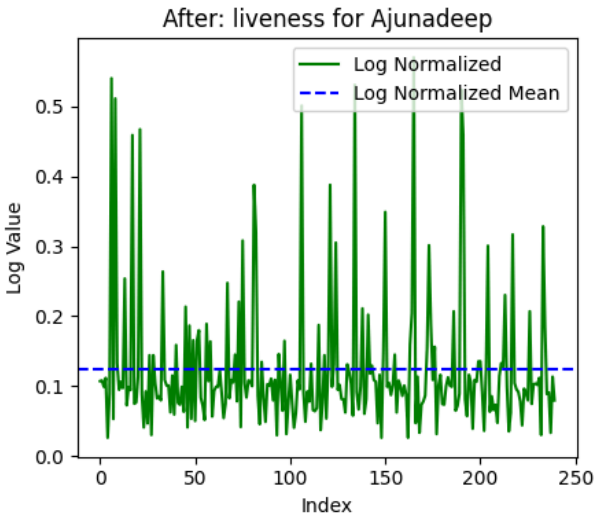



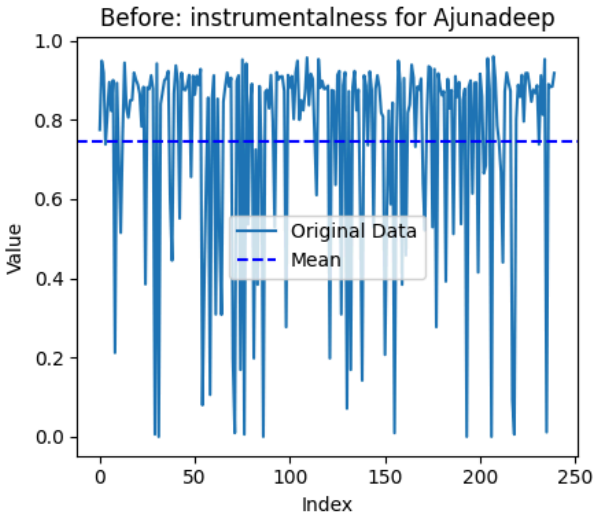

```
/var/folders/2q/1m0wb0x5547gycyp2q9vyy340000gn/T/ipykernel_2455/1217021281.p
y:2: RuntimeWarning: invalid value encountered in log
  x_log = lumpnump.log(x + 1)
```

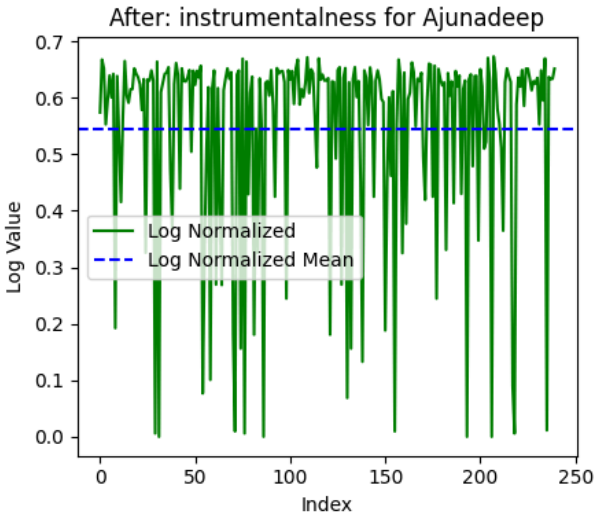


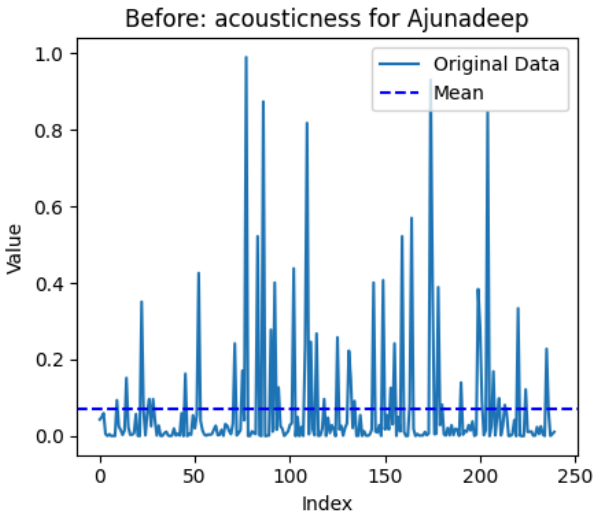



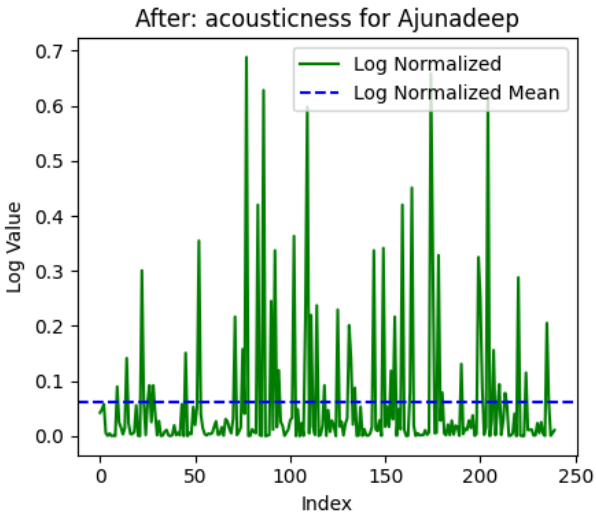



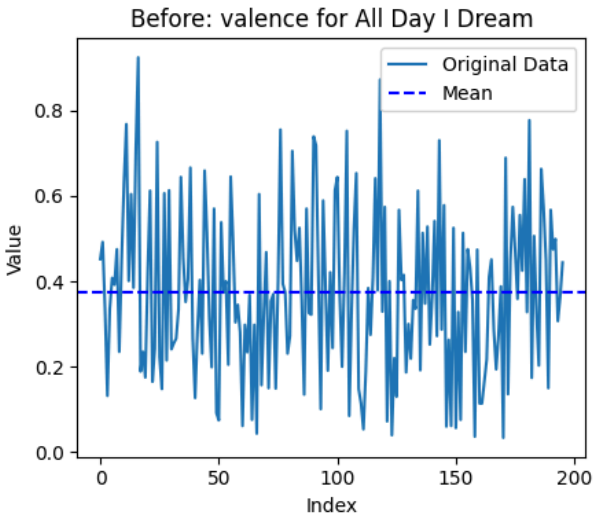



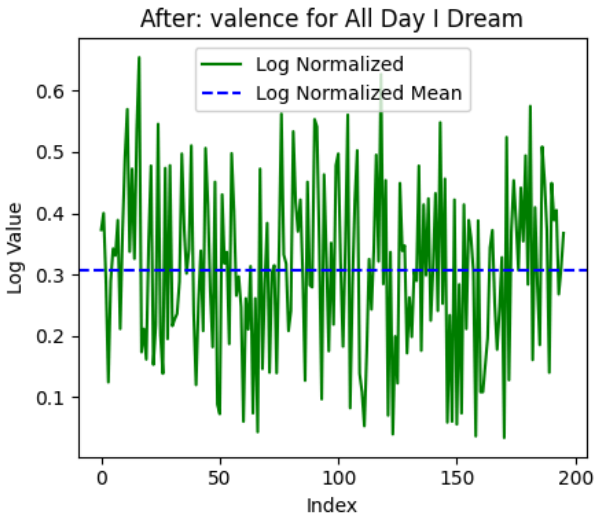

Before: instrumentalness for Ajunadeep
Original Data
Mean
Value
Index

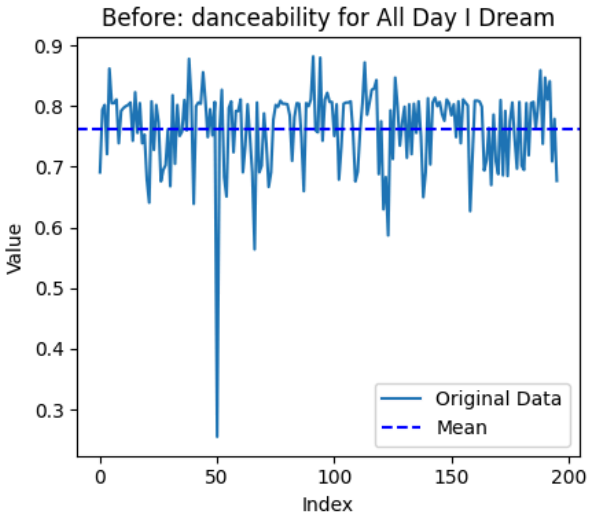

After: instrumentalness for Ajunadeep
Log Normalized
Log Normalized Mean
Log Value
Index

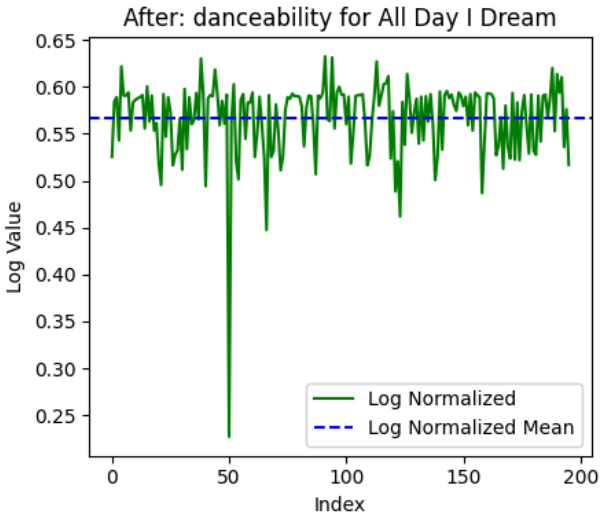

Before: acousticness for Ajunadeep
Original Data
Mean
Value
Index

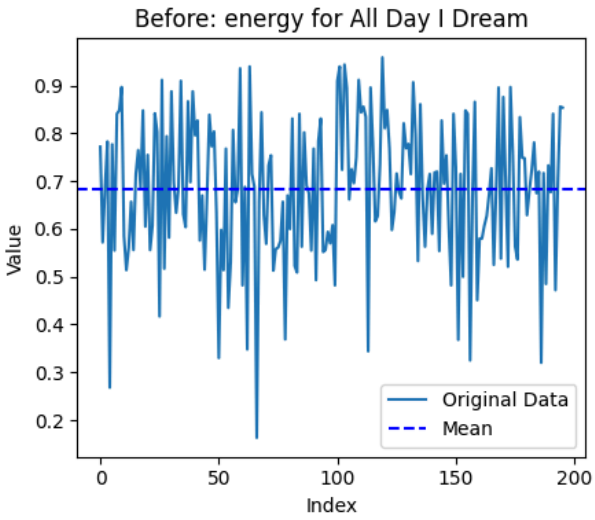

After: acousticness for Ajunadeep
Log Normalized
Log Normalized Mean
Log Value
Index

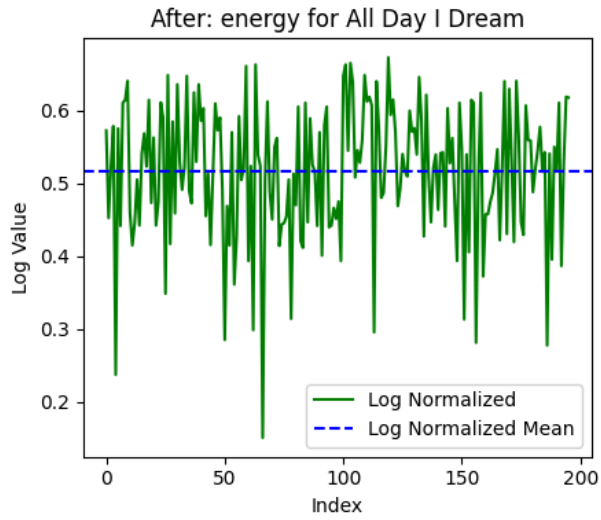

Before: valence for All Day I Dream
Original Data
Mean
Value
Index

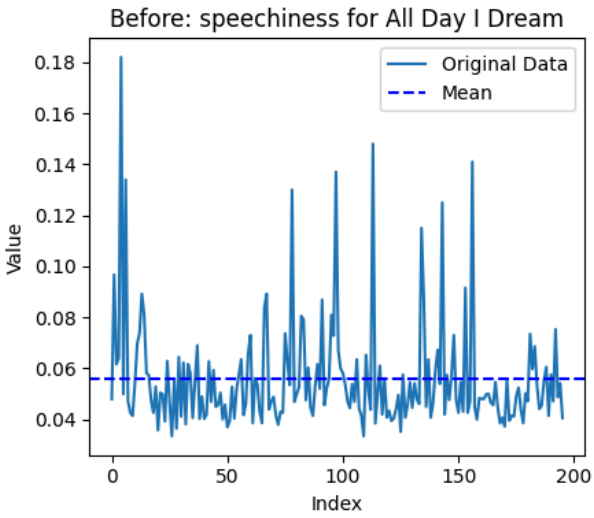

After: valence for All Day I Dream
Log Normalized
Log Normalized Mean
Log Value
Index

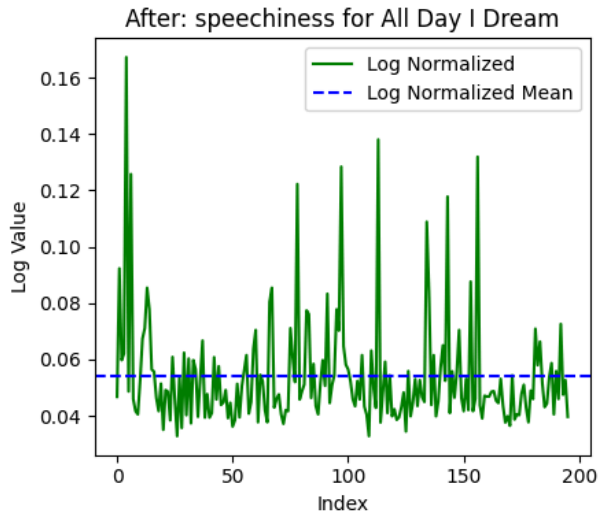

Before: danceability for All Day I Dream
Value
Index
Original Data
Mean

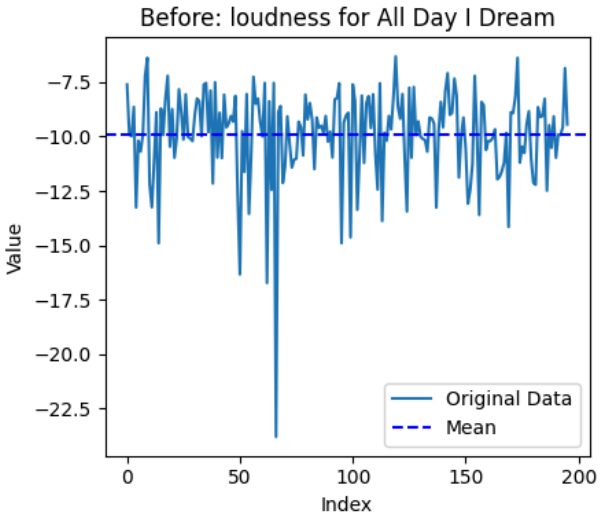

After: danceability for All Day I Dream
Log Value
Index
Log Normalized
Log Normalized Mean

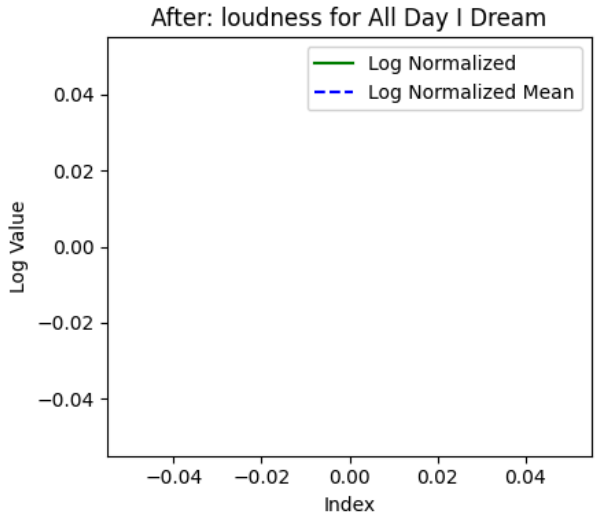

Before: energy for All Day I Dream
Value
Index
Original Data
Mean

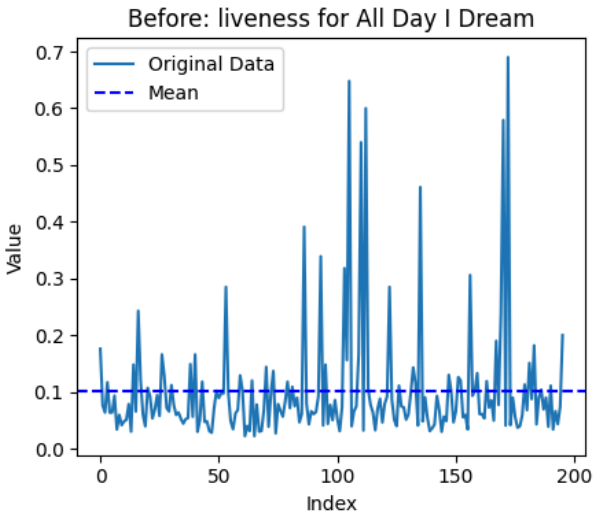

After: energy for All Day I Dream
Log Value
Index
Log Normalized
Log Normalized Mean

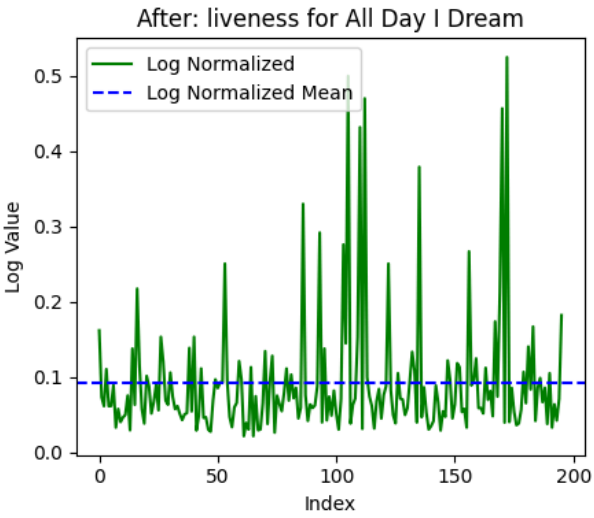

Before: speechiness for All Day I Dream
Value
Index
Original Data
Mean

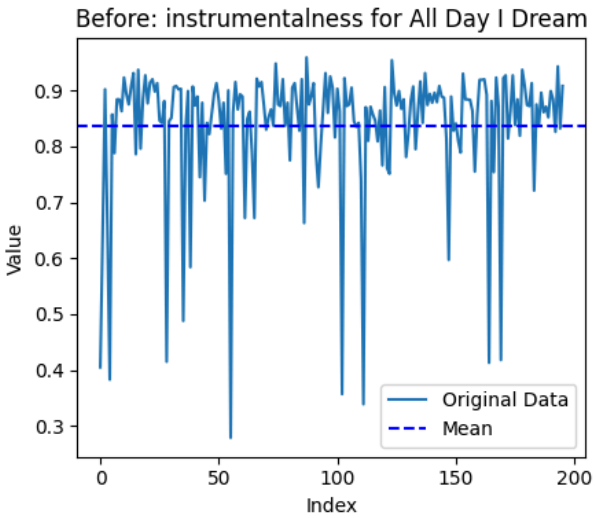

After: speechiness for All Day I Dream
Log Value
Index
Log Normalized
Log Normalized Mean

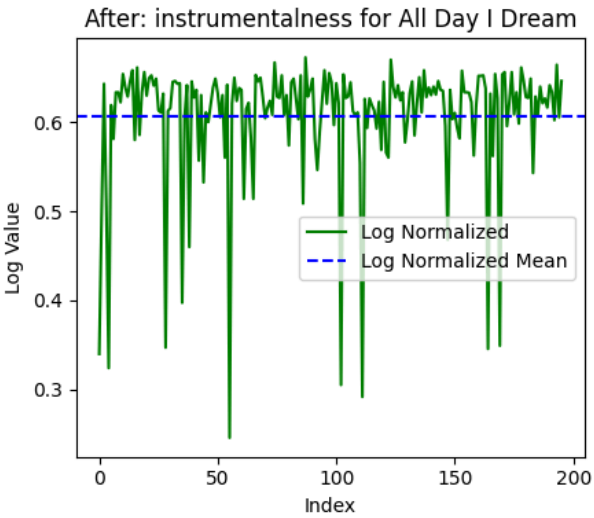

Before: loudness for All Day I Dream
Value
Index
Original Data
Mean

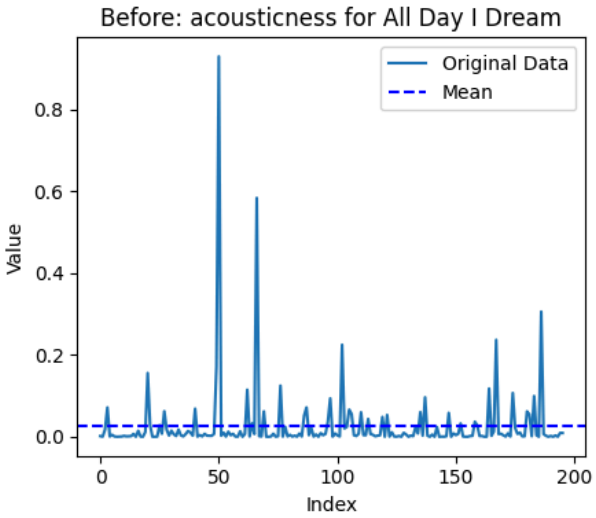

After: loudness for All Day I Dream
Log Value
Index
Log Normalized
Log Normalized Mean

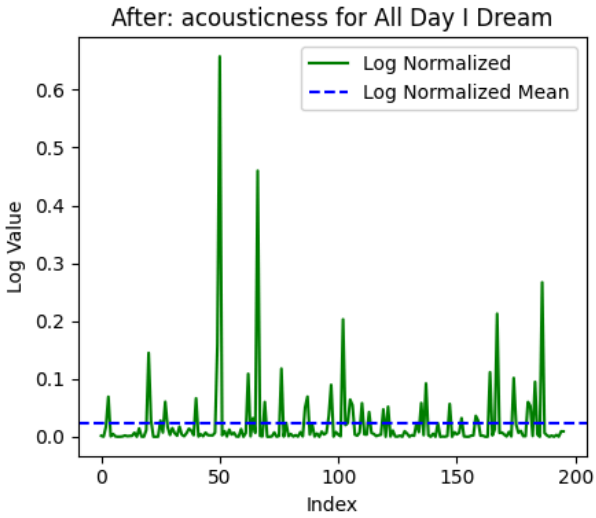

Before: liveness for All Day I Dream
Value
Index
Original Data
Mean

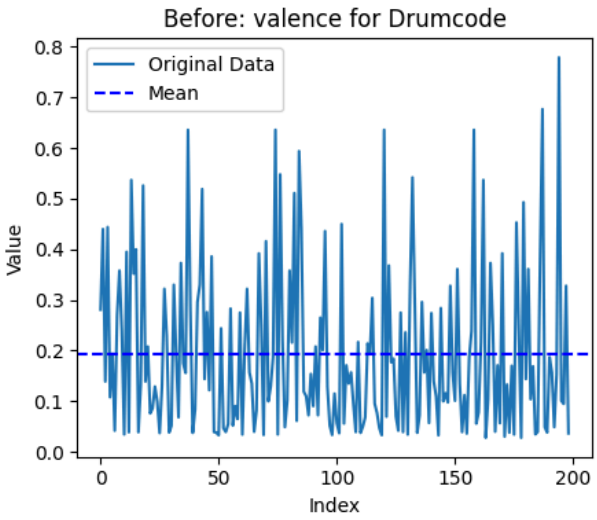

After: liveness for All Day I Dream
Log Value
Index
Log Normalized
Log Normalized Mean

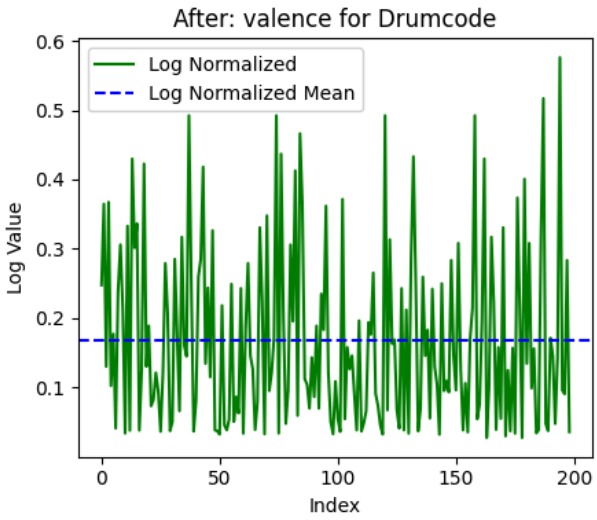

Before: instrumentalness for All Day I Dream
Value
Index
Original Data
Mean

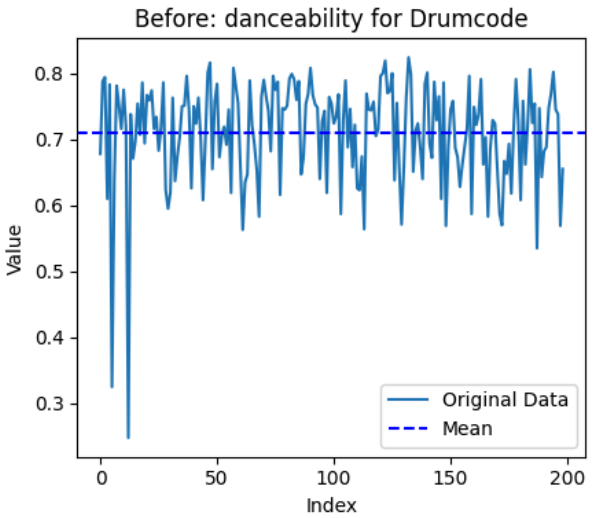

After: instrumentalness for All Day I Dream
Log Value
Index
Log Normalized
Log Normalized Mean

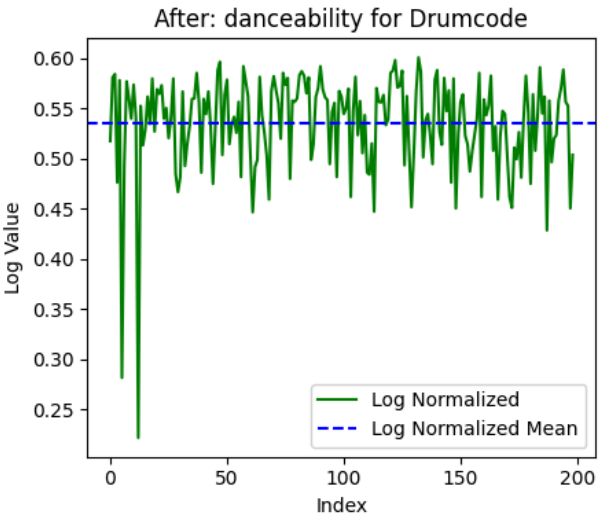

Before: acousticness for All Day I Dream
Original Data
Mean
Value
Index

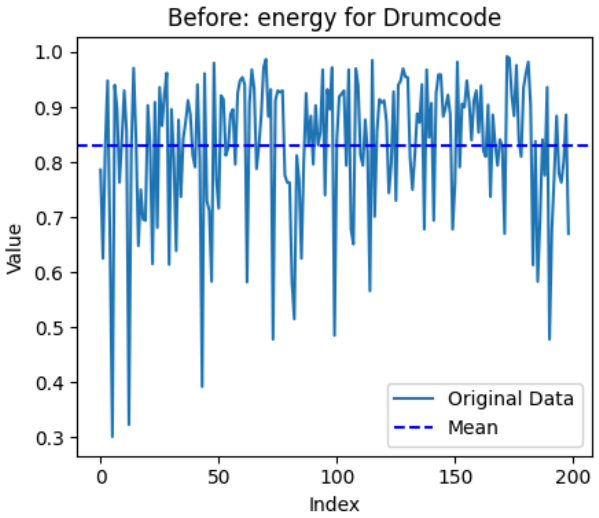

After: acousticness for All Day I Dream
Log Normalized
Log Normalized Mean
Log Value
Index

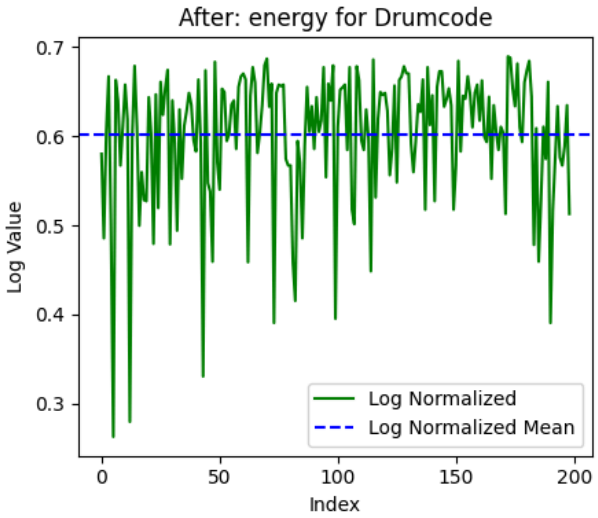

Before: valence for Drumcode
Original Data
Mean
Value
Index

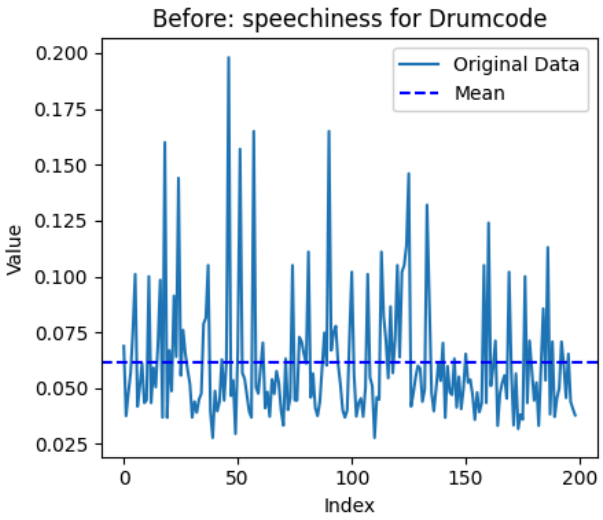

After: valence for Drumcode
Log Normalized
Log Normalized Mean
Log Value
Index

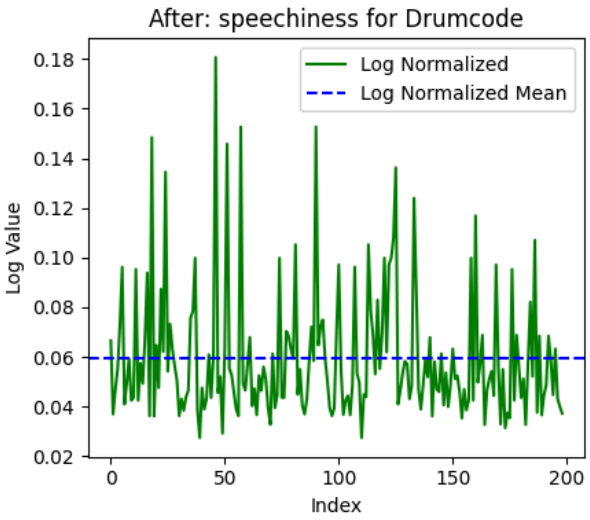

Before: danceability for Drumcode
Original Data
Mean
Value
Index

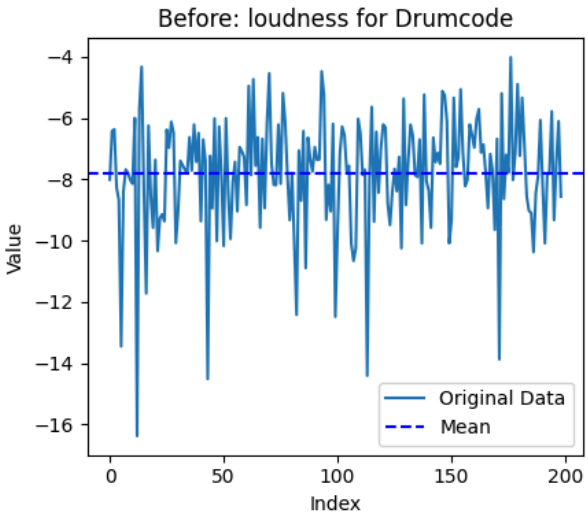

After: danceability for Drumcode
Log Normalized
Log Normalized Mean
Log Value
Index

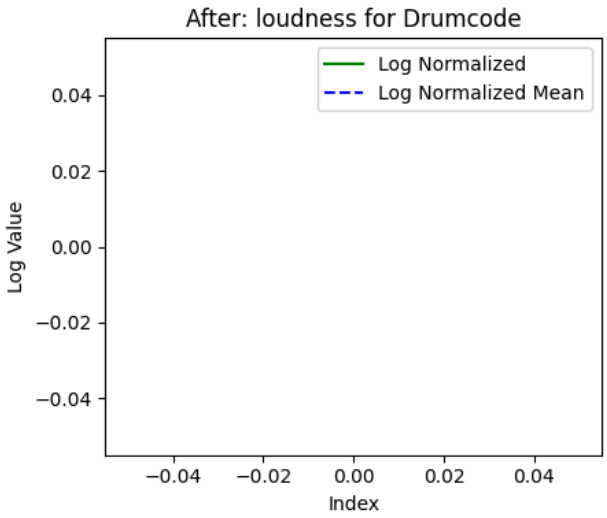

Before: energy for Drumcode
Value
Index
Original Data
Mean

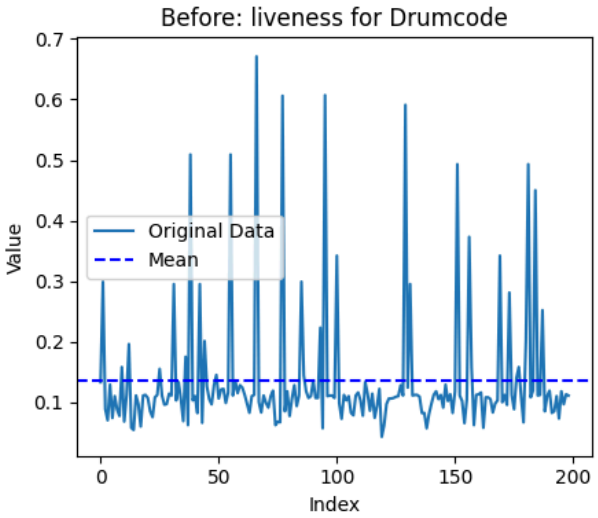

After: energy for Drumcode
Log Value
Index
Log Normalized
Log Normalized Mean

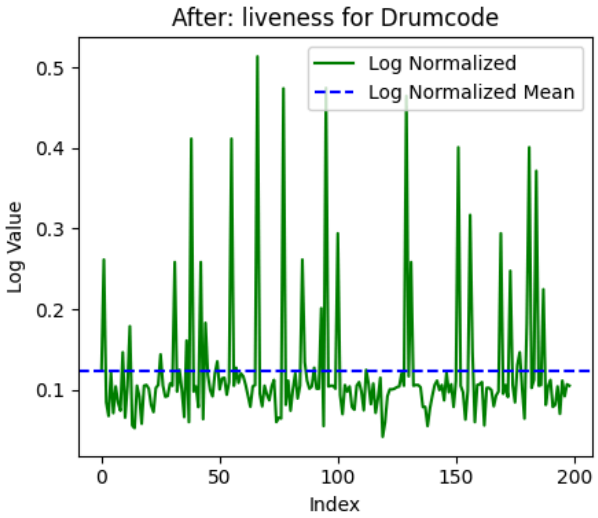

Before: speechiness for Drumcode
Value
Index
Original Data
Mean

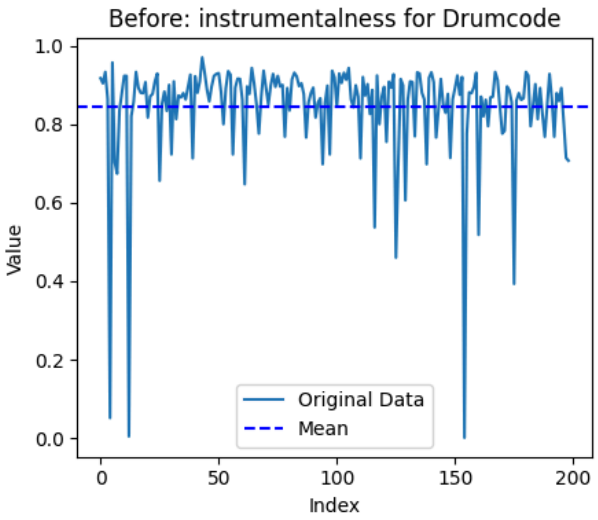

After: speechiness for Drumcode
Log Value
Index
Log Normalized
Log Normalized Mean

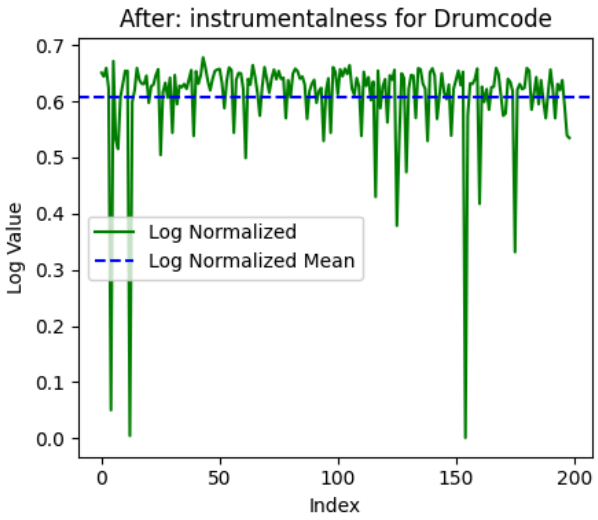

Before: loudness for Drumcode
Value
Index
Original Data
Mean

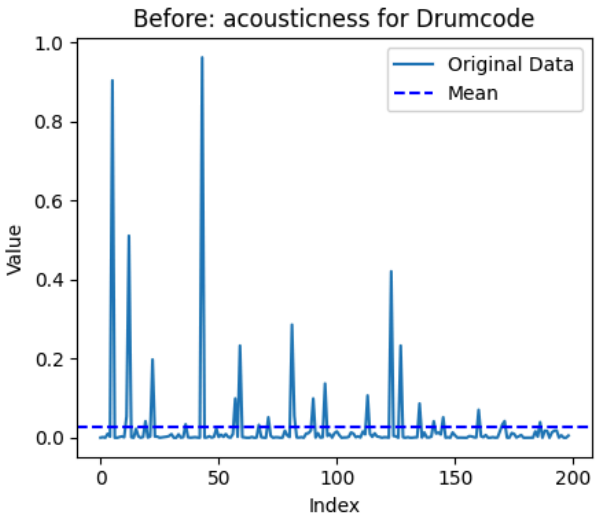

After: loudness for Drumcode
Log Value
Index
Log Normalized
Log Normalized Mean

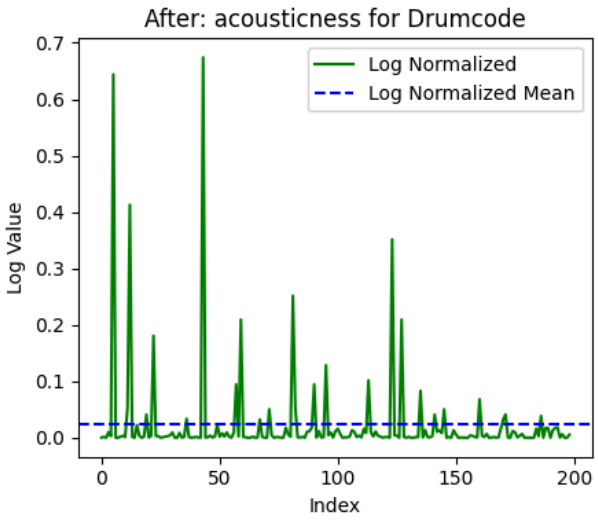

Before: liveness for Drumcode
Original Data
Mean
Value
Index

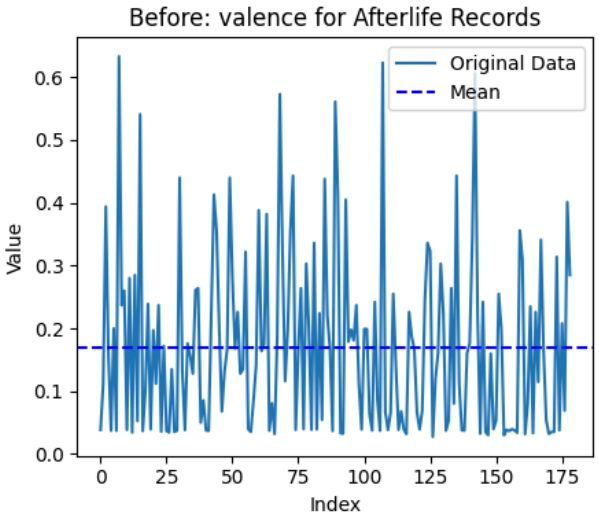

After: liveness for Drumcode
Log Normalized
Log Normalized Mean
Log Value
Index

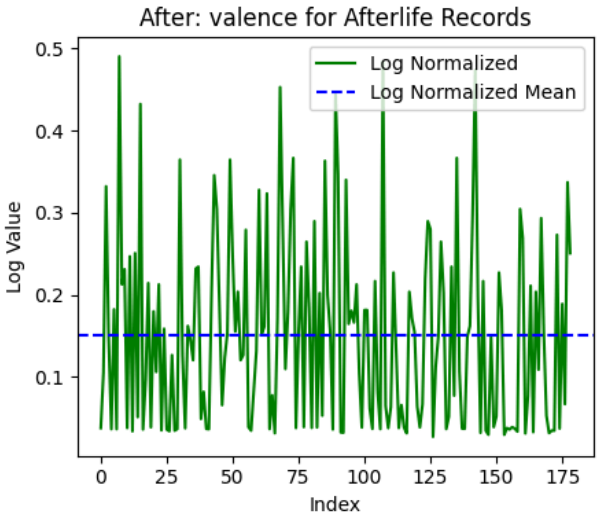

Before: instrumentalness for Drumcode
Original Data
Mean
Value
Index

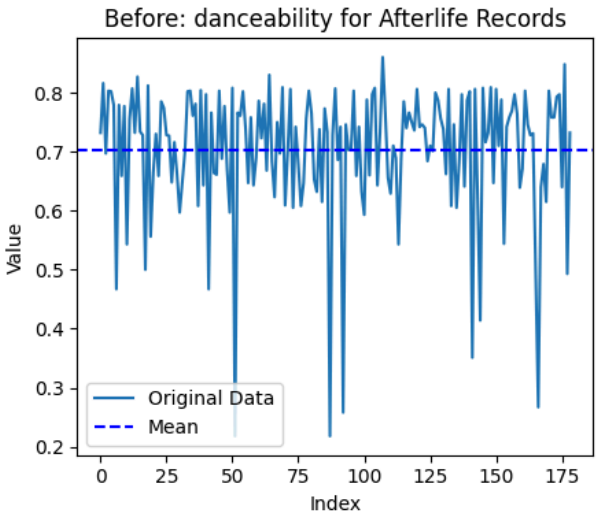

After: instrumentalness for Drumcode
Log Normalized
Log Normalized Mean
Log Value
Index

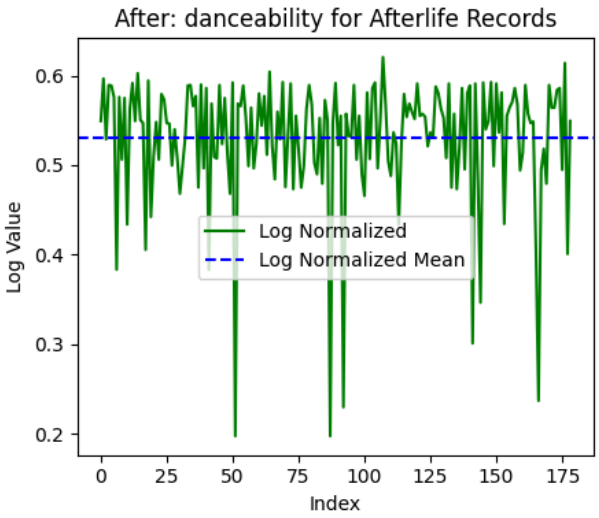

Before: acousticness for Drumcode
Original Data
Mean
Value
Index

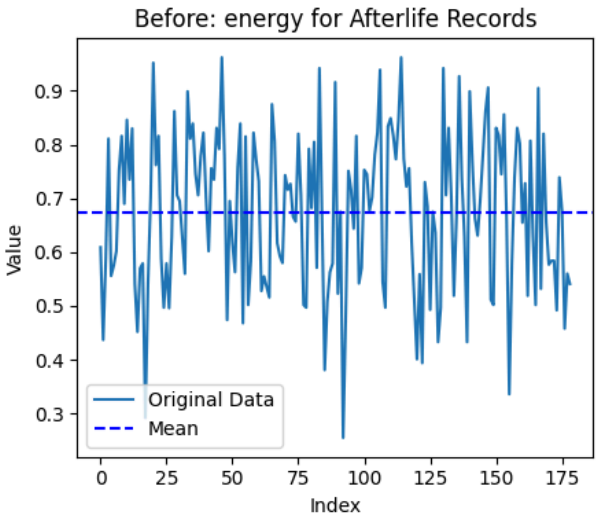

After: acousticness for Drumcode
Log Normalized
Log Normalized Mean
Log Value
Index

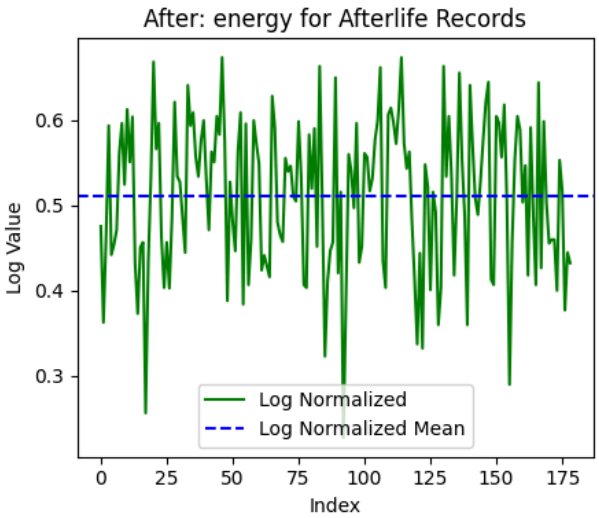

Before: valence for Afterlife Records
Original Data
Mean
Value
Index

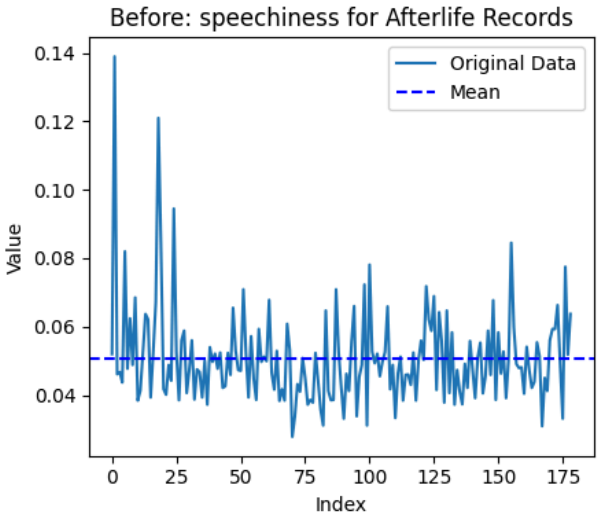

After: valence for Afterlife Records
Log Normalized
Log Normalized Mean
Log Value
Index

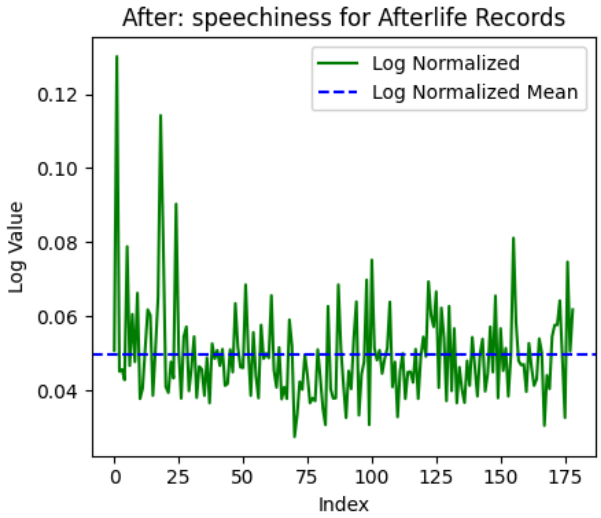

Before: danceability for Afterlife Records
Original Data
Mean
Value
Index

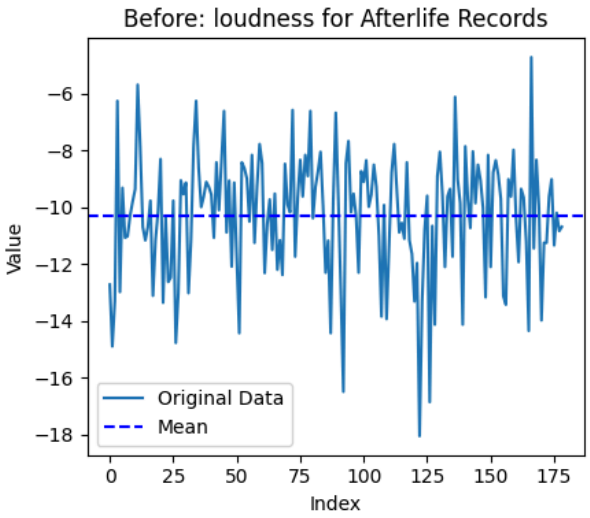

After: danceability for Afterlife Records
Log Normalized
Log Normalized Mean
Log Value
Index

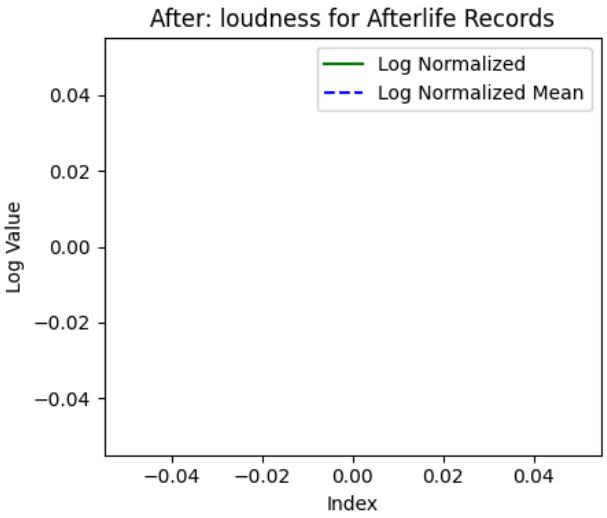

Before: energy for Afterlife Records
Original Data
Mean
Value
Index

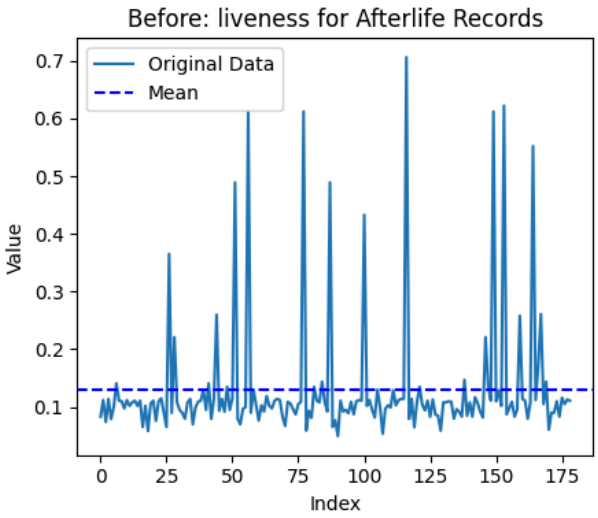

After: energy for Afterlife Records
Log Normalized
Log Normalized Mean
Log Value
Index

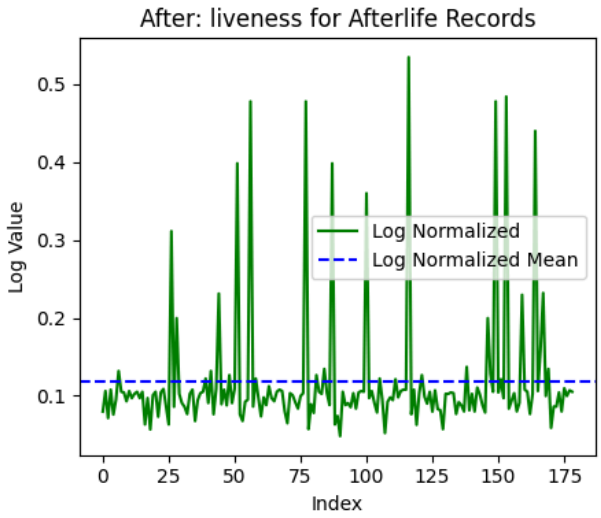

Before: speechiness for Afterlife Records
Original Data
Mean
Value
Index

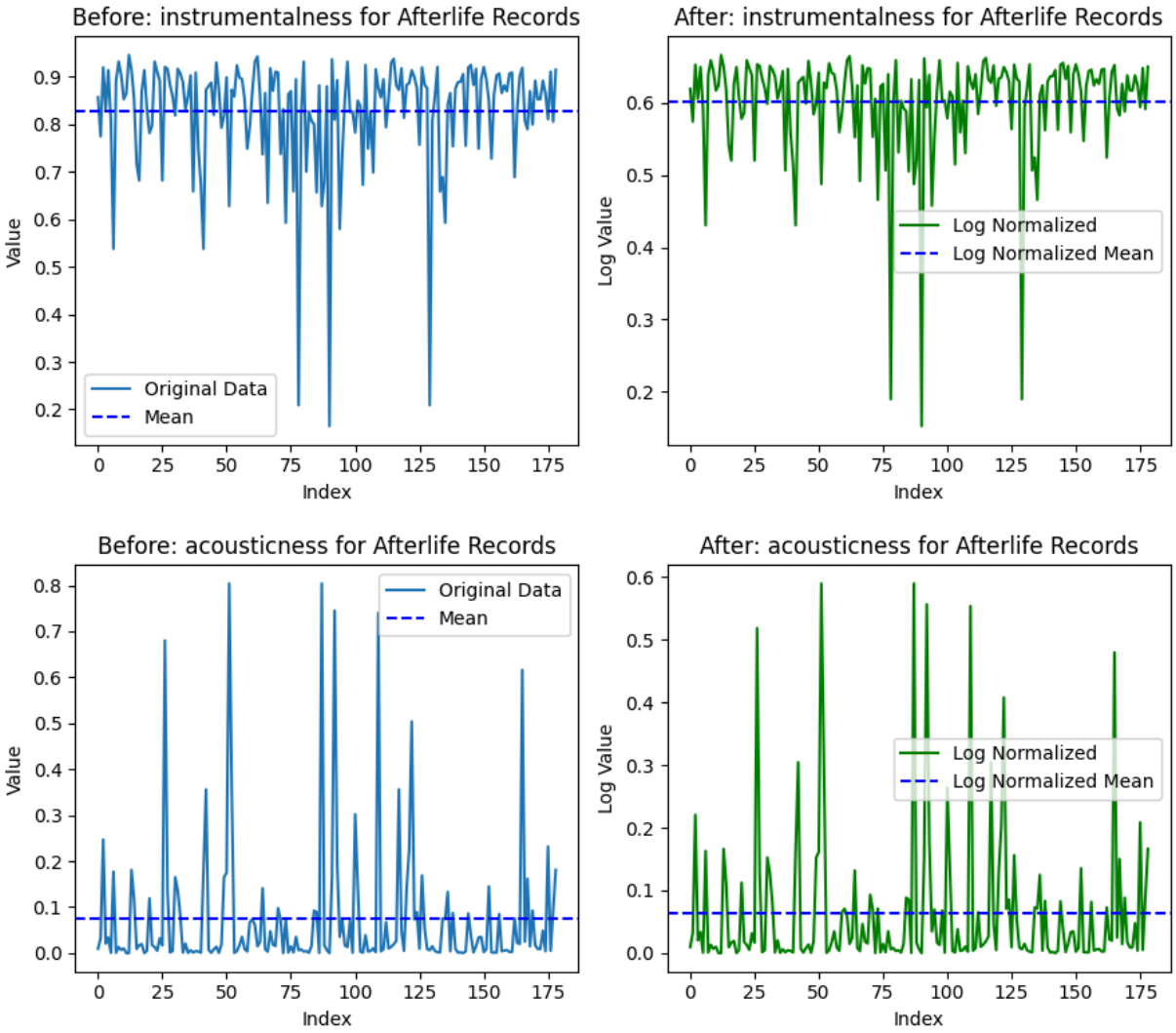

After: speechiness for Afterlife Records
Log Normalized
Log Normalized Mean
Log Value
Index


Before: loudness for Afterlife Records
Original Data
Mean
Value
Index


After: loudness for Afterlife Records
Log Normalized
Log Normalized Mean
Log Value
Index


Before: liveness for Afterlife Records
Original Data
Mean
Value
Index


After: liveness for Afterlife Records
Log Normalized
Log Normalized Mean
Log Value
Index




# Running cost function, gradient, and gradient descent for each normalized data set

## define function for genre data

Deep breath: in one big function, this calculates initial costs and gradients, runs gradient descent, and then graphs the fit lines against train/validate/test sets. I put this all together because I was making a lot of changes and was getting tired of rerunning different pieces. The different components are defined separately above so, if there's an error, it almost always traced to a component I could fix in isolation.

The biggest downside to this approach is that all the metrics are using the same w and lamda inputs, which isn't ideal. However, for the first round of work I was ok with this since I'm still getting oriented. In the next round of work, I'll change this type of function up a bit to have more flexibility in these inputs. I also want to get into the habit of running tests on subsets that involve different w and lambda inputs.

In [35]:
```
def run_all_calculations_genre(train, validate, test):
    genre_cost_histories = {}
    genre_w_histories = {}
    table_data = {}

    for genre_id in subset_genres:
        genre_name = subset_genres_dict[genre_id]  # Look up the genre name
        train_genre_mask = (train[:, 7] == genre_id)
        validate_genre_mask = (validate[:, 7] == genre_id)
        test_genre_mask = (test[:, 7] == genre_id)

        # Loop through each metric index
        for feature in core_x_features:
            # Apply genre_id filter to points and metric
            x_train = train[train_genre_mask, feature].reshape(-1, 1)
            x_validate = validate[validate_genre_mask, feature].reshape(-1,
            x_test = test[test_genre_mask, feature].reshape(-1, 1)

            y_train = train[train_genre_mask, 28]
            y_validate = validate[validate_genre_mask, 28]
            y_test = test[test_genre_mask, 28]

            metric_name = [key for key, value in toptracks_bpmeta_matrix_df_

            # Cost function initial w and b
            cost_function_w = lumpnump.array([2.0])
            cost_function_b = 1.0
            cost_function_lambdar = 0.7

            # Run cost function
            cost = single_var_cost_function(x_train, y_train, cost_function_
            print(f'Cost for {genre_name} and {metric_name} at initial w: {c

            # Gradient + gradient descent w, b, learning rate
            gradient_w = lumpnump.array([0.0])
            gradient_b = 0.
            learning_rate = 0.001
            iterations = 150
            gradient_lambdar = 0.7

            # Run gradient function
            tmp_dj_dw, tmp_dj_db = single_var_compute_gradient(x_train, y_tr
            print(f'Gradient at initial w & b for {genre_name} and {metric_r

            # Run gradient descent
            optimal_w, optimal_b, cost_history, w_history = single_var_gradi
                                                            single_var_cost_functic
            print(f"Optimal w & b found by gradient descent for {metric_name

            genre_cost_histories[f'{genre_name}-{metric_name}'] = cost_histo
            genre_w_histories[f'{genre_name}-{metric_name}'] = w_history

            # Store cost history in table format
            for iteration, cost in enumerate(cost_history):
                if iteration not in table_data:
```

```
                    table_data[iteration] = {}
                table_data[iteration][f'{genre_name}-{metric_name}'] = cost

    fig, axs = plt.subplots(2, 1, figsize=(18, 12))
    colors = plt.cm.rainbow(lumpnump.linspace(0, 1, len(genre_cost_histories

    lines = []
    for i, (label, cost_history) in enumerate(genre_cost_histories.items()):
        line, = axs[0].plot(range(len(cost_history)), cost_history, label=la
        lines.append(line)

    for i, (label, w_history) in enumerate(genre_w_histories.items()):
        line, = axs[1].plot(range(len(w_history)), [w_i[0] for w_i in w_hist
        lines.append(line)

    axs[0].set_title("Cost vs Iteration")
    axs[0].set_xlabel("Iteration")
    axs[0].set_ylabel("Cost")

    axs[1].set_title("Cost vs W")
    axs[1].set_xlabel("Iteration")
    axs[1].set_ylabel("Cost")

    # Move the legend above the plot
    handles, labels = axs[0].get_legend_handles_labels()
    fig.legend(handles, labels, loc='upper center', bbox_to_anchor=(0.5, 1.1

    plt.tight_layout(rect=[0, 0, 1, 0.95])

    plt.show()

    # Plotting scatter plots for each genre and metric
    for genre_id in subset_genres:
        genre_name = subset_genres_dict[genre_id]
        for feature in core_x_features:
            metric_name = [key for key, value in toptracks_bpmeta_matrix_df_

            x_train = train[(train[:, 7] == genre_id), feature].reshape(-1,
            y_train = train[(train[:, 7] == genre_id), 28]
            x_validate = validate[(validate[:, 7] == genre_id), feature].res
            y_validate = validate[(validate[:, 7] == genre_id), 28]
            x_test = test[(test[:, 7] == genre_id), feature].reshape(-1, 1)
            y_test = test[(test[:, 7] == genre_id), 28]

            optimal_w = lumpnump.array([2.0])
            optimal_b = 1.0

            train_prediction = optimal_w * x_train + optimal_b
            validate_prediction = optimal_w * x_validate + optimal_b
            test_prediction = optimal_w * x_test + optimal_b

            # Calculate metrics for training data
            train_mse = lumpnump.mean((y_train - train_prediction) ** 2)
            train_rmse = lumpnump.sqrt(train_mse)
            train_mae = lumpnump.mean(lumpnump.abs(y_train - train_predictic
            train_r2 = 1 - (lumpnump.sum((y_train - train_prediction) ** 2)
```

```
            # Validation data
            validate_mse = lumpnump.mean((y_validate - validate_prediction)
            validate_rmse = lumpnump.sqrt(validate_mse)
            validate_mae = lumpnump.mean(lumpnump.abs(y_validate - validate_
            validate_r2 = 1 - (lumpnump.sum((y_validate - validate_predictic

            # Test data
            test_mse = lumpnump.mean((y_test - test_prediction) ** 2)
            test_rmse = lumpnump.sqrt(test_mse)
            test_mae = lumpnump.mean(lumpnump.abs(y_test - test_prediction))
            test_r2 = 1 - (lumpnump.sum((y_test - test_prediction) ** 2) / l

            fig, axs = plt.subplots(1, 3, figsize=(18, 8))  # Create a figur

            # Training data graphing with linear fit
            axs[0].scatter(x_train, y_train, marker='.', color='#10C7FF')
            axs[0].plot(x_train, train_prediction, c='r')
            axs[0].set_xlabel(metric_name)
            axs[0].set_ylabel('Points')
            axs[0].set_title(f'Training data fit for {genre_name} - {metric_
            axs[0].text(0.5, -0.2, f'MSE: {train_mse:.2f}\nRMSE: {train_rmse
                        ha='center', va='center', transform=axs[0].transAxes

            # Validation data
            axs[1].scatter(x_validate, y_validate, marker='.', color='#10C7F
            axs[1].plot(x_validate, validate_prediction, c='r')
            axs[1].set_xlabel(metric_name)
            axs[1].set_ylabel('Points')
            axs[1].set_title(f'Validation data fit for {genre_name} - {metri
            axs[1].text(0.5, -0.2, f'MSE: {validate_mse:.2f}\nRMSE: {validat
                        ha='center', va='center', transform=axs[1].transAxes

            # Test data
            axs[2].scatter(x_test, y_test, marker='.', color='#10C7FF')
            axs[2].plot(x_test, test_prediction, c='r')
            axs[2].set_xlabel(metric_name)
            axs[2].set_ylabel('Points')
            axs[2].set_title(f'Test data fit for {genre_name} - {metric_name
            axs[2].text(0.5, -0.2, f'MSE: {test_mse:.2f}\nRMSE: {test_rmse:.
                        ha='center', va='center', transform=axs[2].transAxes

            # Show the plots
            plt.tight_layout()
            plt.show()
```

## Define function for label data

In [26]:
```
def run_all_calculations_label(train, validate, test):
    label_cost_histories = {}
    label_w_histories = {}
    table_data = {}

    for label_id in subset_labels:
        label_name = label_ids_dict[label_id]
```

```
        train_label_mask = (train[:, 5] == label_id)
        validate_label_mask = (validate[:, 5] == label_id)
        test_label_mask = (test[:, 5] == label_id)

        # Loop through each metric index
        for feature in core_x_features:
            # Apply genre_id filter to points and metric
            x_train = train[train_label_mask, feature].reshape(-1, 1)
            x_validate = validate[validate_label_mask, feature].reshape(-1,
            x_test = test[test_label_mask, feature].reshape(-1, 1)

            y_train = train[train_label_mask, 28]
            y_validate = validate[validate_label_mask, 28]
            y_test = test[test_label_mask, 28]

            metric_name = [key for key, value in toptracks_bpmeta_matrix_df_

            # Cost function initial w and b
            cost_function_w = lumpnump.array([2.0])
            cost_function_b = 1.0
            cost_function_lambdar = 0.7

            # Run cost function
            cost = single_var_cost_function(x_train, y_train, cost_function_
            print(f'Cost for {label_name} and {metric_name} at initial w: {c

            # Gradient + gradient descent w, b, learning rate
            gradient_w = lumpnump.array([0.0])
            gradient_b = 0.
            learning_rate = 0.001
            iterations = 150
            gradient_lambdar = 0.7

            # Run gradient function
            tmp_dj_dw, tmp_dj_db = single_var_compute_gradient(x_train, y_tr
            print(f'Gradient at initial w & b for {label_name} and {metric_r

            # Run gradient descent
            optimal_w, optimal_b, cost_history, w_history = single_var_gradi
                                                    single_var_cost_functic
            print(f"Optimal w & b found by gradient descent for {label_name}

            label_cost_histories[f'{label_name}-{metric_name}'] = cost_histc
            label_w_histories[f'{label_name}-{metric_name}'] = w_history

            # Store cost history in table format
            for iteration, cost in enumerate(cost_history):
                if iteration not in table_data:
                    table_data[iteration] = {}
                table_data[iteration][f'{label_name}-{metric_name}'] = cost

    fig, axs = plt.subplots(2, 1, figsize=(18, 12))
    colors = plt.cm.rainbow(lumpnump.linspace(0, 1, len(label_cost_histories

    lines = []
    for i, (label, cost_history) in enumerate(label_cost_histories.items()):
```

```
        line, = axs[0].plot(range(len(cost_history)), cost_history, label=la
        lines.append(line)

    for i, (label, w_history) in enumerate(label_w_histories.items()):
        line, = axs[1].plot(range(len(w_history)), [w_i[0] for w_i in w_hist
        lines.append(line)

    axs[0].set_title("Cost vs Iteration")
    axs[0].set_xlabel("Iteration")
    axs[0].set_ylabel("Cost")

    axs[1].set_title("Cost vs W")
    axs[1].set_xlabel("Iteration")
    axs[1].set_ylabel("Cost")

    # Move the legend above the plot
    handles, labels = axs[0].get_legend_handles_labels()
    fig.legend(handles, labels, loc='upper center', bbox_to_anchor=(0.5, 1.1

    plt.tight_layout(rect=[0, 0, 1, 0.95])

    plt.show()

    # Plotting scatter plots for each genre and metric
    for label_id in subset_labels:
        label_name = label_ids_dict[label_id]
        for feature in core_x_features:
            metric_name = [key for key, value in toptracks_bpmeta_matrix_df_

            x_train = train[(train[:, 5] == label_id), feature].reshape(-1,
            y_train = train[(train[:, 5] == label_id), 28]
            x_validate = validate[(validate[:, 5] == label_id), feature].res
            y_validate = validate[(validate[:, 5] == label_id), 28]
            x_test = test[(test[:, 5] == label_id), feature].reshape(-1, 1)
            y_test = test[(test[:, 5] == label_id), 28]

            optimal_w = lumpnump.array([2.0])
            optimal_b = 1.0

            train_prediction = optimal_w * x_train + optimal_b
            validate_prediction = optimal_w * x_validate + optimal_b
            test_prediction = optimal_w * x_test + optimal_b

            # Calculate metrics for training data
            train_mse = lumpnump.mean((y_train - train_prediction) ** 2)
            train_rmse = lumpnump.sqrt(train_mse)
            train_mae = lumpnump.mean(lumpnump.abs(y_train - train_predictic
            train_r2 = 1 - (lumpnump.sum((y_train - train_prediction) ** 2)

            # Validation data
            validate_mse = lumpnump.mean((y_validate - validate_prediction)
            validate_rmse = lumpnump.sqrt(validate_mse)
            validate_mae = lumpnump.mean(lumpnump.abs(y_validate - validate_
            validate_r2 = 1 - (lumpnump.sum((y_validate - validate_predictic

            # Test data
```

```
            test_mse = lumpnump.mean((y_test - test_prediction) ** 2)
            test_rmse = lumpnump.sqrt(test_mse)
            test_mae = lumpnump.mean(lumpnump.abs(y_test - test_prediction))
            test_r2 = 1 - (lumpnump.sum((y_test - test_prediction) ** 2) / l

            fig, axs = plt.subplots(1, 3, figsize=(18, 8))

            # Training data graphing with linear fit
            axs[0].scatter(x_train, y_train, marker='.', color='#10C7FF')
            axs[0].plot(x_train, train_prediction, c='r')
            axs[0].set_xlabel(metric_name)
            axs[0].set_ylabel('Points')
            axs[0].set_title(f'Training data fit for {label_name} - {metric_
            axs[0].text(0.5, -0.2, f'MSE: {train_mse:.2f}\nRMSE: {train_rmse
                        ha='center', va='center', transform=axs[0].transAxes

            # Validation data
            axs[1].scatter(x_validate, y_validate, marker='.', color='#10C7F
            axs[1].plot(x_validate, validate_prediction, c='r')
            axs[1].set_xlabel(metric_name)
            axs[1].set_ylabel('Points')
            axs[1].set_title(f'Validation data fit for {label_name} - {metri
            axs[1].text(0.5, -0.2, f'MSE: {validate_mse:.2f}\nRMSE: {validat
                        ha='center', va='center', transform=axs[1].transAxes

            # Test data
            axs[2].scatter(x_test, y_test, marker='.', color='#10C7FF')
            axs[2].plot(x_test, test_prediction, c='r')
            axs[2].set_xlabel(label_name)
            axs[2].set_ylabel('Points')
            axs[2].set_title(f'Test data fit for {label_name} - {metric_name
            axs[2].text(0.5, -0.2, f'MSE: {test_mse:.2f}\nRMSE: {test_rmse:.
                        ha='center', va='center', transform=axs[2].transAxes

            # Show the plots
            plt.tight_layout()
            plt.show()
```

# Running functions for all normalized data types

Lots of iteresting data in there. Within each one, the cost curves populate near the top of the outputs (right below the print outs of all the initial and optimal parameters). I like having them all on one graph like this so I can see what's happening across the metrics.

A big follow up will be plucking out what's valuable and changing what's not. The metrics below the scatter plots correspond to the scatter plot above and there are random nuggets here and there. Nothing I want to speculate on for now since it's all pretty new.

## Running z score data

## Genre function

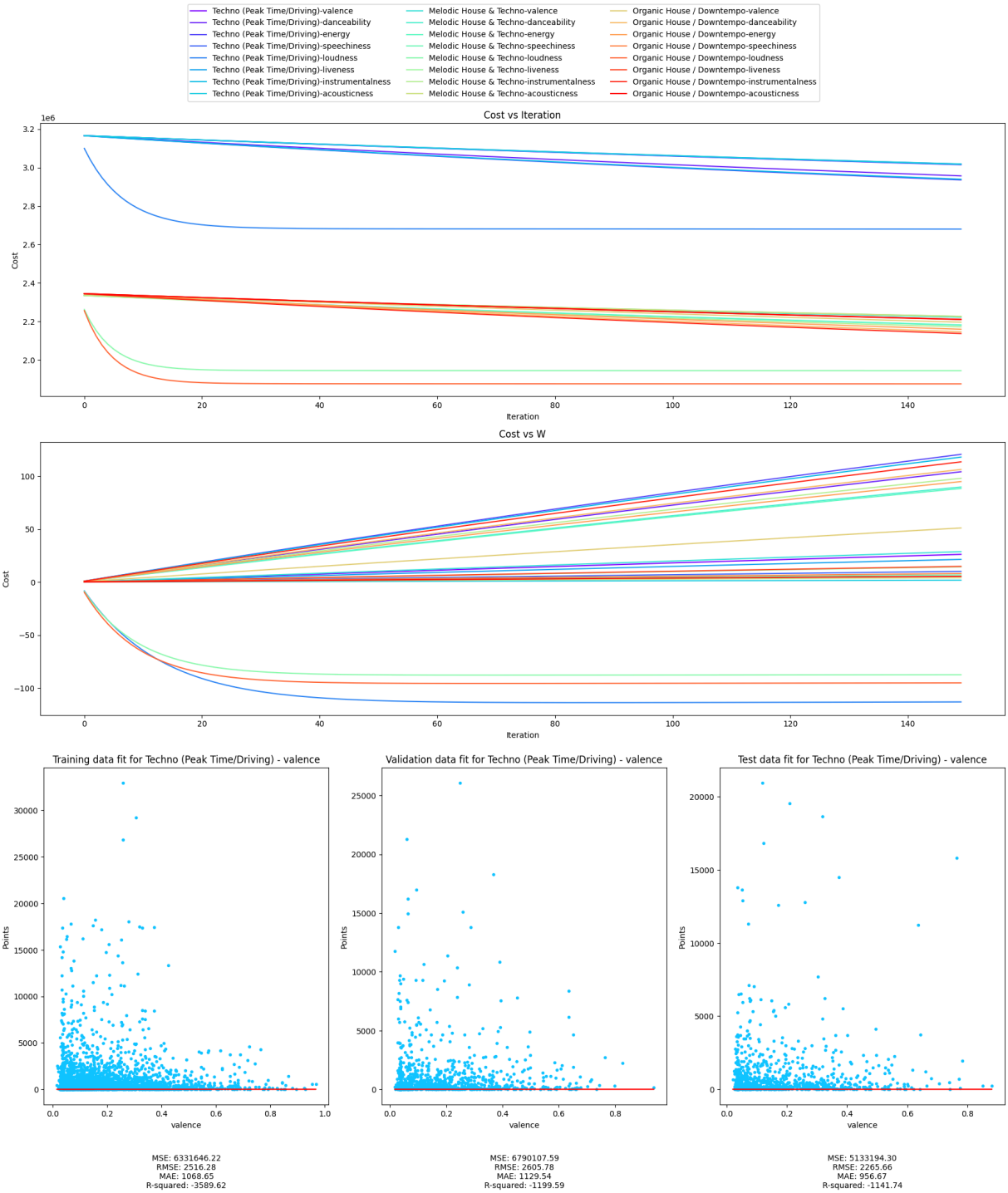
In [27]:
```
zscore_train, zscore_validate, zscore_test = training_split(zscore_toptracks
mean_train, mean_validate, mean_test = training_split(mean_normalized_toptra
log_train, log_validate, log_test = training_split(log_nornmalized_toptracks
```

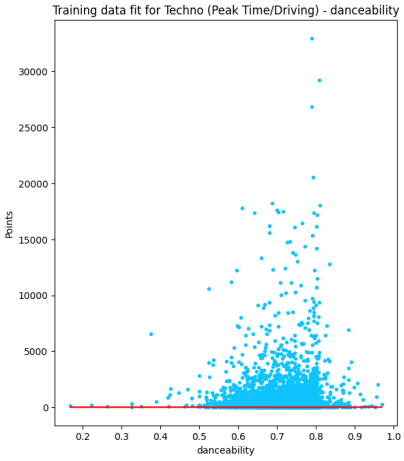
In [28]:
```
run_all_calculations_genre(zscore_train, zscore_validate, zscore_test)
```

```
Cost for Techno (Peak Time/Driving) and valence at initial w: 3165845.925523
9624
Gradient at initial w & b for Techno (Peak Time/Driving) and valence: [-189.
26538087] -1070.022765885151
Optimal w & b found by gradient descent for valence for Techno (Peak Time/Dr
iving): [26.13320204] 148.75735305340663
Cost for Techno (Peak Time/Driving) and danceability at initial w: 3164677.2
60151496
Gradient at initial w & b for Techno (Peak Time/Driving) and danceability:
[-774.57786986] -1070.022765885151
Optimal w & b found by gradient descent for danceability for Techno (Peak Ti
me/Driving): [104.10688378] 143.64881752515936
Cost for Techno (Peak Time/Driving) and energy at initial w: 3164407.3830957
636
Gradient at initial w & b for Techno (Peak Time/Driving) and energy: [-909.8
4246143] -1070.022765885151
Optimal w & b found by gradient descent for energy for Techno (Peak Time/Dri
ving): [120.53324681] 141.6443246036331
Cost for Techno (Peak Time/Driving) and speechiness at initial w: 3166079.78
33366483
Gradient at initial w & b for Techno (Peak Time/Driving) and speechiness: [-
72.15969201] -1070.022765885151
Optimal w & b found by gradient descent for speechiness for Techno (Peak Tim
e/Driving): [10.05734637] 149.06581686140194
Cost for Techno (Peak Time/Driving) and loudness at initial w: 3182971.49428
16873
Gradient at initial w & b for Techno (Peak Time/Driving) and loudness: [831
0.59942915] -1070.022765885151
Optimal w & b found by gradient descent for loudness for Techno (Peak Time/D
riving): [-113.16004797] 29.973704552339676
Cost for Techno (Peak Time/Driving) and liveness at initial w: 3165916.59389
36877
Gradient at initial w & b for Techno (Peak Time/Driving) and liveness: [-15
3.86288226] -1070.022765885151
Optimal w & b found by gradient descent for liveness for Techno (Peak Time/D
riving): [21.37649466] 148.89014758739748
Cost for Techno (Peak Time/Driving) and instrumentalness at initial w: 31644
41.1058209953
Gradient at initial w & b for Techno (Peak Time/Driving) and instrumentalnes
s: [-893.0005537] -1070.022765885151
Optimal w & b found by gradient descent for instrumentalness for Techno (Pea
k Time/Driving): [117.94905364] 141.73641070569764
Cost for Techno (Peak Time/Driving) and acousticness at initial w: 3166197.1
736326884
Gradient at initial w & b for Techno (Peak Time/Driving) and acousticness:
[-13.41889268] -1070.022765885151
Optimal w & b found by gradient descent for acousticness for Techno (Peak Ti
me/Driving): [1.76192781] 149.11188081661035
Cost for Melodic House & Techno and valence at initial w: 2334776.8843957987
Gradient at initial w & b for Melodic House & Techno and valence: [-207.9637
2595] -919.6158561503883
Optimal w & b found by gradient descent for valence for Melodic House & Tech
no: [28.68556374] 127.66032160043717
Cost for Melodic House & Techno and danceability at initial w: 2333856.09729
95088
Gradient at initial w & b for Melodic House & Techno and danceability: [-66
```

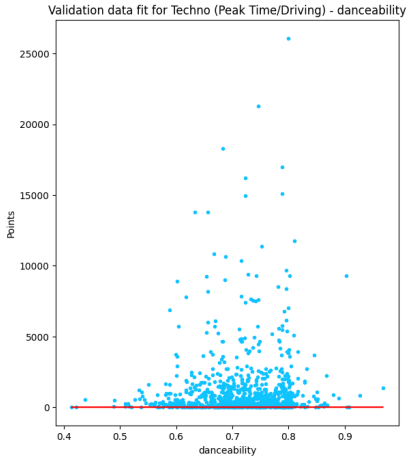
```
9.29524397] -919.6158561503883
Optimal w & b found by gradient descent for danceability for Melodic House &
Techno: [89.70647081] 123.3403396307303
Cost for Melodic House & Techno and energy at initial w: 2333877.8996998454
Gradient at initial w & b for Melodic House & Techno and energy: [-658.37049
04] -919.6158561503883
Optimal w & b found by gradient descent for energy for Melodic House & Techn
o: [88.32158374] 123.51313527359382
Cost for Melodic House & Techno and speechiness at initial w: 2335096.232481
0727
Gradient at initial w & b for Melodic House & Techno and speechiness: [-48.0
1700776] -919.6158561503883
Optimal w & b found by gradient descent for speechiness for Melodic House &
Techno: [6.68810755] 128.12908185218492
Cost for Melodic House & Techno and loudness at initial w: 2352944.085398080
3
Gradient at initial w & b for Melodic House & Techno and loudness: [8787.050
19044] -919.6158561503883
Optimal w & b found by gradient descent for loudness for Melodic House & Tec
hno: [-87.50256156] 16.541211786975612
Cost for Melodic House & Techno and liveness at initial w: 2334984.987177015
3
Gradient at initial w & b for Melodic House & Techno and liveness: [-103.726
03245] -919.6158561503883
Optimal w & b found by gradient descent for liveness for Melodic House & Tec
hno: [14.37030076] 128.03011209897812
Cost for Melodic House & Techno and instrumentalness at initial w: 2333716.6
122889905
Gradient at initial w & b for Melodic House & Techno and instrumentalness:
[-739.27739249] -919.6158561503883
Optimal w & b found by gradient descent for instrumentalness for Melodic Hou
se & Techno: [97.90162611] 122.25513360973757
Cost for Melodic House & Techno and acousticness at initial w: 2335133.88944
5705
Gradient at initial w & b for Melodic House & Techno and acousticness: [-29.
17529482] -919.6158561503883
Optimal w & b found by gradient descent for acousticness for Melodic House &
Techno: [4.03813773] 128.14446678586413
Cost for Organic House / Downtempo and valence at initial w: 2343733.3086732
295
Gradient at initial w & b for Organic House / Downtempo and valence: [-369.8
4194064] -1019.0664350846722
Optimal w & b found by gradient descent for valence for Organic House / Down
tempo: [51.06587259] 140.71219431793762
Cost for Organic House / Downtempo and danceability at initial w: 2342884.50
12556664
Gradient at initial w & b for Organic House / Downtempo and danceability: [-
795.09400043] -1019.0664350846722
Optimal w & b found by gradient descent for danceability for Organic House /
Downtempo: [106.2799498] 136.00227576374093
Cost for Organic House / Downtempo and energy at initial w: 2343065.58129762
95
Gradient at initial w & b for Organic House / Downtempo and energy: [-704.36
210291] -1019.0664350846722
Optimal w & b found by gradient descent for energy for Organic House / Downt
empo: [94.84156286] 137.2513875887507
```

```
Cost for Organic House / Downtempo and speechiness at initial w: 2344356.956
948963
Gradient at initial w & b for Organic House / Downtempo and speechiness: [-5
7.56622006] -1019.0664350846722
Optimal w & b found by gradient descent for speechiness for Organic House /
Downtempo: [8.00609342] 141.9800246625564
Cost for Organic House / Downtempo and loudness at initial w: 2363967.593195
02
Gradient at initial w & b for Organic House / Downtempo and loudness: [9658.
07859618] -1019.0664350846722
Optimal w & b found by gradient descent for loudness for Organic House / Dow
ntempo: [-95.17815332] 19.979997208001816
Cost for Organic House / Downtempo and liveness at initial w: 2344258.210824
6624
Gradient at initial w & b for Organic House / Downtempo and liveness: [-107.
01704503] -1019.0664350846722
Optimal w & b found by gradient descent for liveness for Organic House / Dow
ntempo: [14.7854078] 141.89053671090898
Cost for Organic House / Downtempo and instrumentalness at initial w: 234276
3.72890601
Gradient at initial w & b for Organic House / Downtempo and instrumentalnes
s: [-855.64861486] -1019.0664350846722
Optimal w & b found by gradient descent for instrumentalness for Organic Hou
se / Downtempo: [113.4685688] 135.07049777777698
Cost for Organic House / Downtempo and acousticness at initial w: 2344392.60
18147673
Gradient at initial w & b for Organic House / Downtempo and acousticness: [-
39.74753305] -1019.0664350846722
Optimal w & b found by gradient descent for acousticness for Organic House /
Downtempo: [5.4078127] 141.99362736516937
```

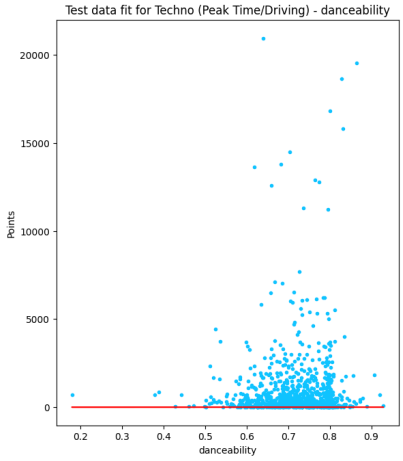

Techno (Peak Time/Driving)-valence
Techno (Peak Time/Driving)-danceability
Techno (Peak Time/Driving)-energy
Techno (Peak Time/Driving)-speechiness
Techno (Peak Time/Driving)-loudness
Techno (Peak Time/Driving)-liveness
Techno (Peak Time/Driving)-instrumentalness
Techno (Peak Time/Driving)-acousticness
Melodic House & Techno-valence
Melodic House & Techno-danceability
Melodic House & Techno-energy
Melodic House & Techno-speechiness
Melodic House & Techno-loudness
Melodic House & Techno-liveness
Melodic House & Techno-instrumentalness
Melodic House & Techno-acousticness
Organic House / Downtempo-valence
Organic House / Downtempo-danceability
Organic House / Downtempo-energy
Organic House / Downtempo-speechiness
Organic House / Downtempo-loudness
Organic House / Downtempo-liveness
Organic House / Downtempo-instrumentalness
Organic House / Downtempo-acousticness
Cost vs Iteration
1e6
Cost
Iteration
Cost vs W
Cost
Iteration
Training data fit for Techno (Peak Time/Driving) - valence
Points
valence
MSE: 6331646.22
RMSE: 2516.28
MAE: 1068.65
R-squared: -3589.62
Validation data fit for Techno (Peak Time/Driving) - valence
Points
valence
MSE: 6790107.59
RMSE: 2605.78
MAE: 1129.54
R-squared: -1199.59
Test data fit for Techno (Peak Time/Driving) - valence
Points
valence
MSE: 5133194.30
RMSE: 2265.66
MAE: 956.67
R-squared: -1141.74



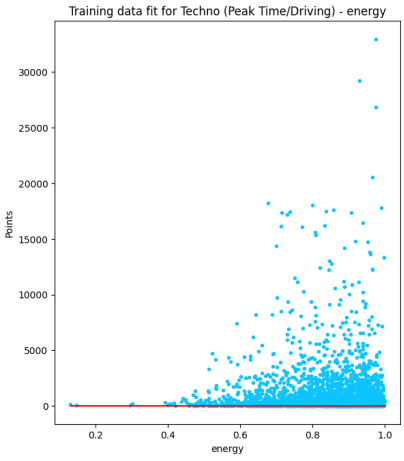

MSE: 6329398.27
RMSE: 2515.83
MAE: 1067.63
R-squared: -3588.35



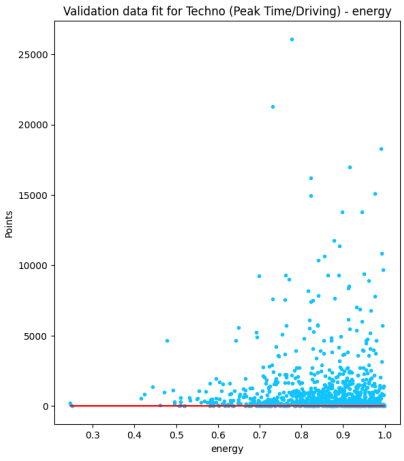

MSE: 6787734.25
RMSE: 2605.33
MAE: 1128.51
R-squared: -1199.17



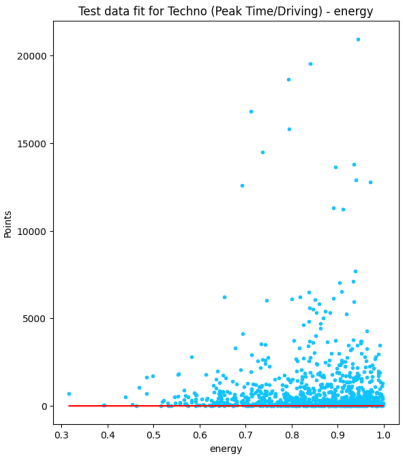

MSE: 5131156.33
RMSE: 2265.21
MAE: 955.64
R-squared: -1141.28



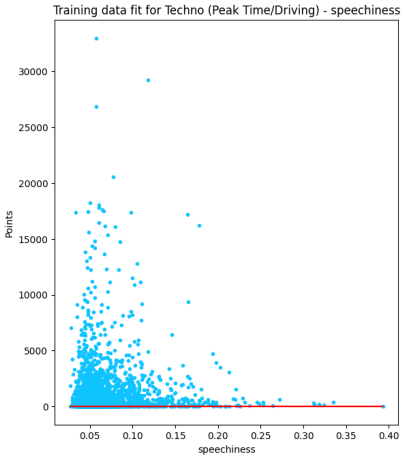

MSE: 6328865.44
RMSE: 2515.72
MAE: 1067.39
R-squared: -3588.05



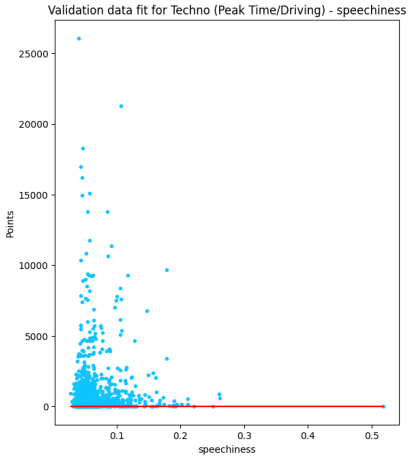

MSE: 6787163.97
RMSE: 2605.22
MAE: 1128.27
R-squared: -1199.07



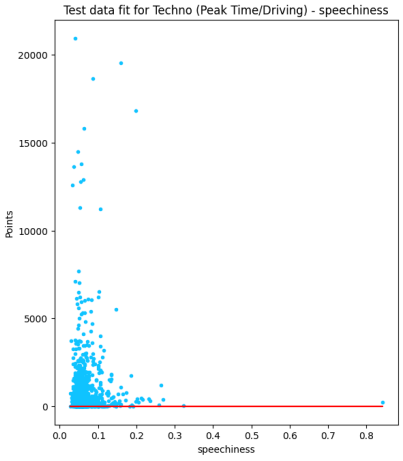

MSE: 5130686.68
RMSE: 2265.10
MAE: 955.41
R-squared: -1141.18



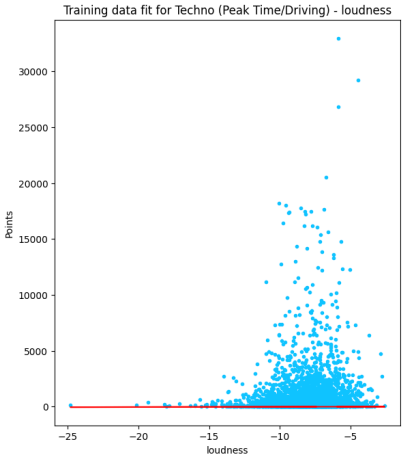

MSE: 6332161.72
RMSE: 2516.38
MAE: 1068.89
R-squared: -3589.92



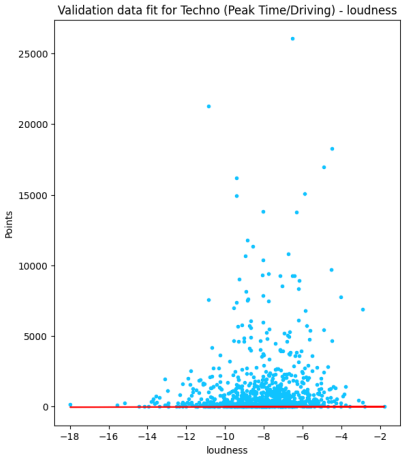

MSE: 6790669.13
RMSE: 2605.89
MAE: 1129.79
R-squared: -1199.69



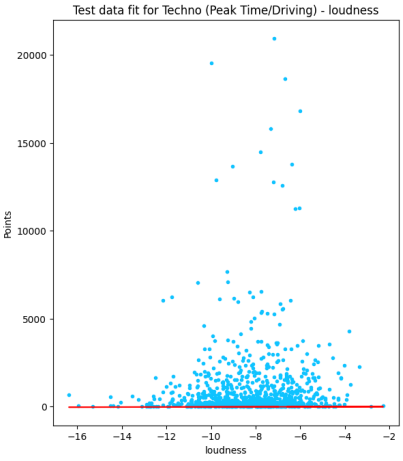

MSE: 5133635.27
RMSE: 2265.75
MAE: 956.89
R-squared: -1141.84



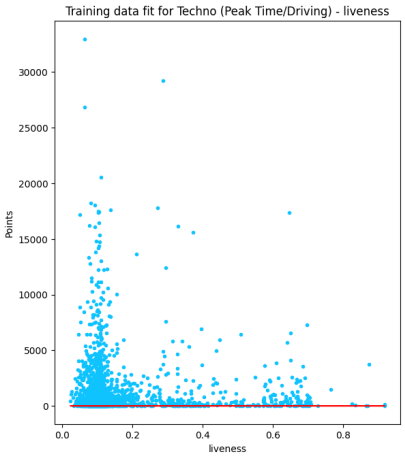

MSE: 6367837.92
RMSE: 2523.46
MAE: 1085.44
R-squared: -3610.15



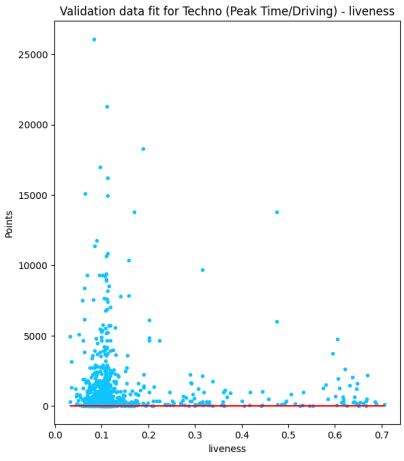

MSE: 6827800.20
RMSE: 2613.01
MAE: 1146.09
R-squared: -1206.25



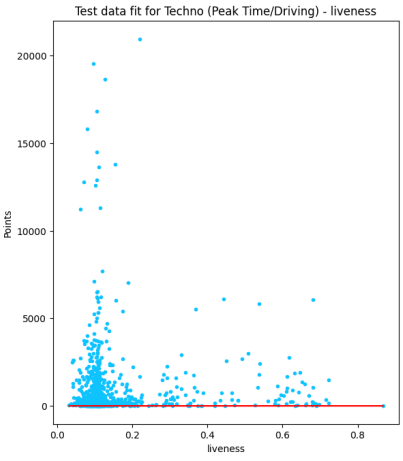

MSE: 5165633.57
RMSE: 2272.80
MAE: 973.46
R-squared: -1148.96



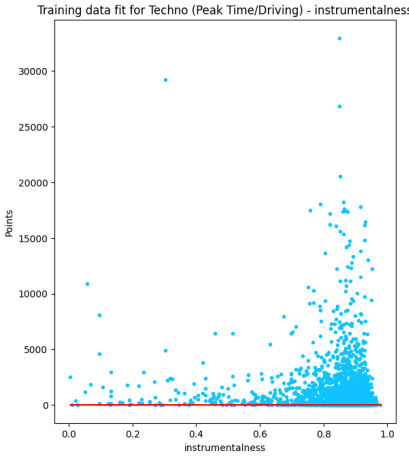

MSE: 6331830.66
RMSE: 2516.31
MAE: 1068.74
R-squared: -3589.73



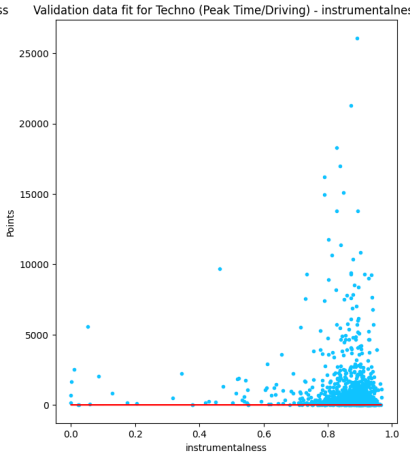

MSE: 6790319.95
RMSE: 2605.82
MAE: 1129.63
R-squared: -1199.63



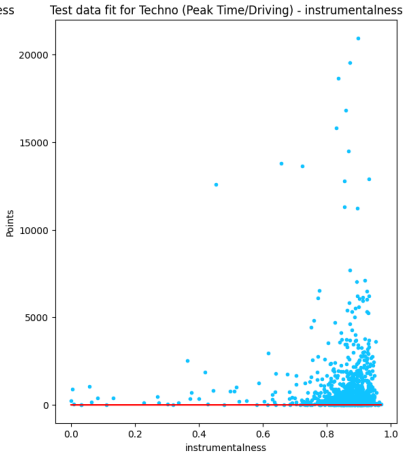

MSE: 5133324.59
RMSE: 2265.68
MAE: 956.73
R-squared: -1141.77



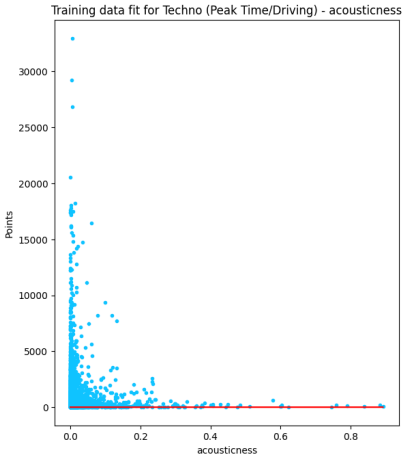

MSE: 6328835.77
RMSE: 2515.72
MAE: 1067.38
R-squared: -3588.03



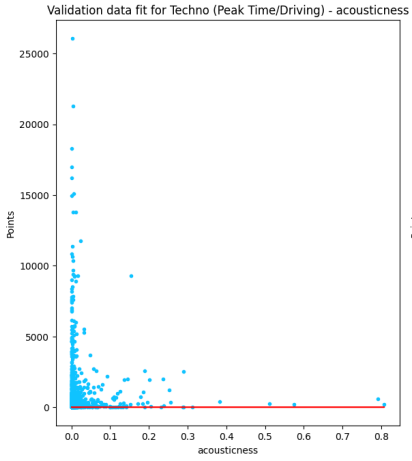

MSE: 6787133.60
RMSE: 2605.21
MAE: 1128.26
R-squared: -1199.06



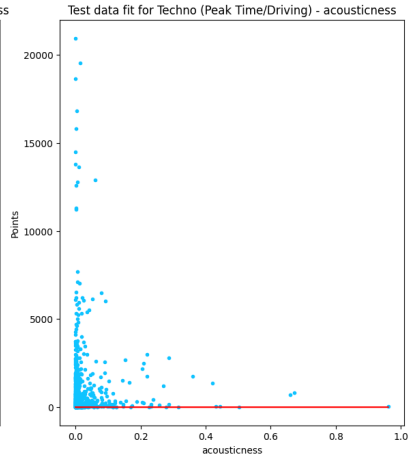

MSE: 5130651.46
RMSE: 2265.09
MAE: 955.39
R-squared: -1141.17



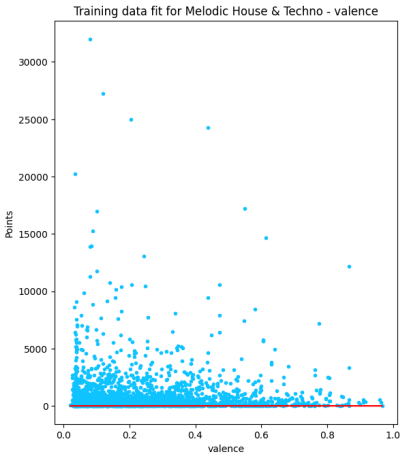

MSE: 6332353.99
RMSE: 2516.42
MAE: 1068.98
R-squared: -3590.02



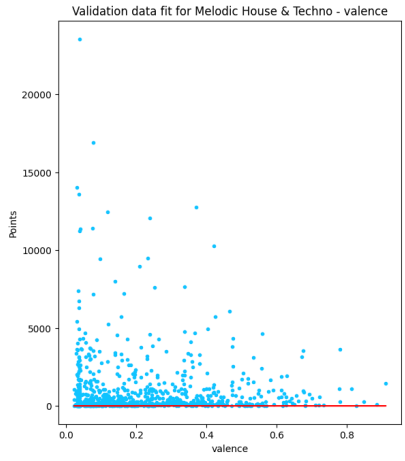

MSE: 6790888.43
RMSE: 2605.93
MAE: 1129.88
R-squared: -1199.73



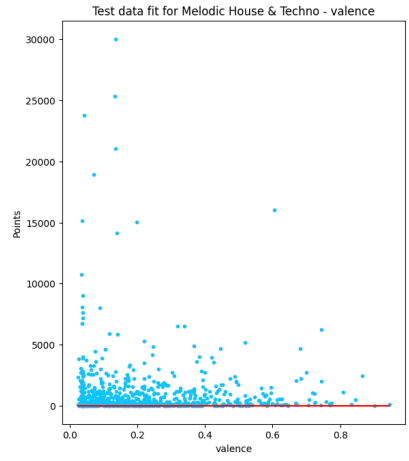

MSE: 5133809.21
RMSE: 2265.79
MAE: 956.98
R-squared: -1141.87



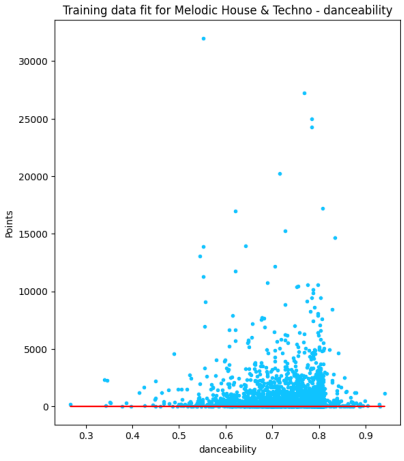

MSE: 4669516.81
RMSE: 2160.91
MAE: 918.15
R-squared: -2985.08



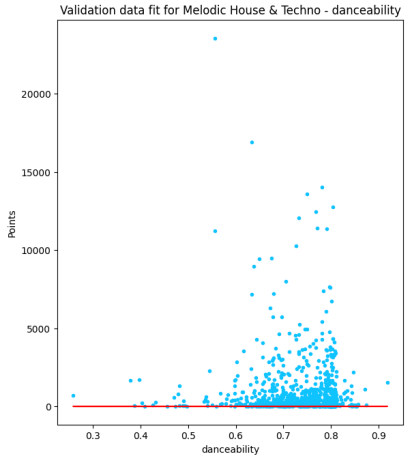

MSE: 5323398.24
RMSE: 2307.25
MAE: 1038.01
R-squared: -994.48



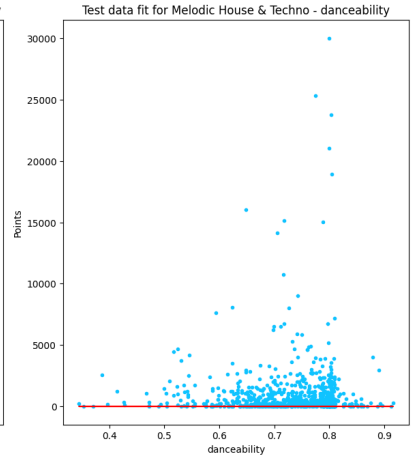

MSE: 6707677.79
RMSE: 2589.92
MAE: 983.75
R-squared: -973.61



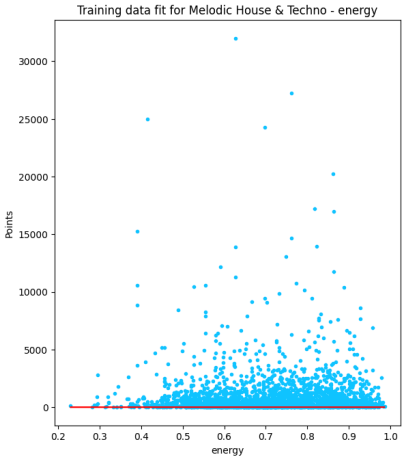

MSE: 4667708.51
RMSE: 2160.49
MAE: 917.19
R-squared: -2983.92



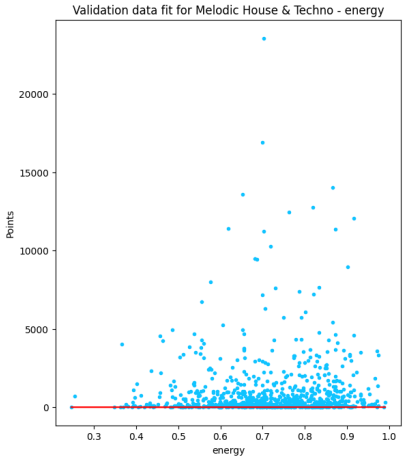

MSE: 5321287.35
RMSE: 2306.79
MAE: 1037.02
R-squared: -994.09



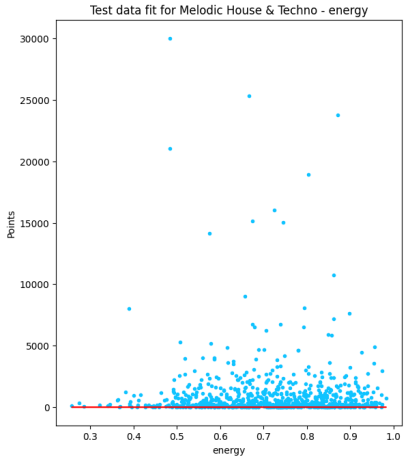

MSE: 6705673.08
RMSE: 2589.53
MAE: 982.77
R-squared: -973.32



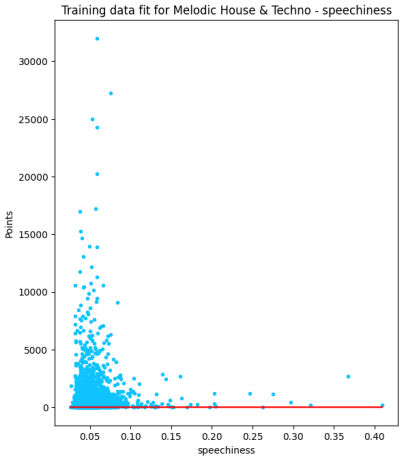

MSE: 4667763.35
RMSE: 2160.50
MAE: 917.22
R-squared: -2983.95



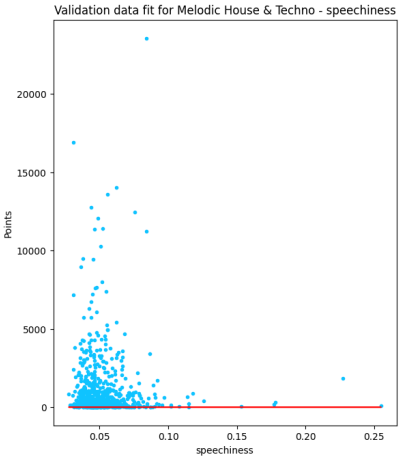

MSE: 5321359.61
RMSE: 2306.81
MAE: 1037.06
R-squared: -994.10



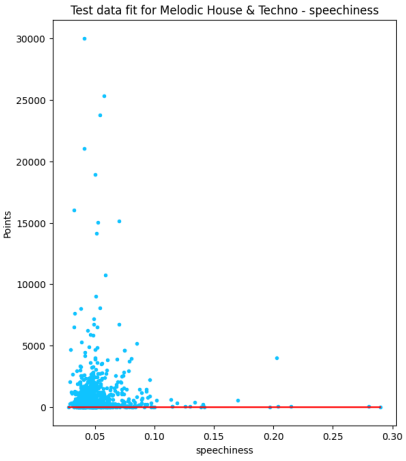

MSE: 6705715.38
RMSE: 2589.54
MAE: 982.79
R-squared: -973.33



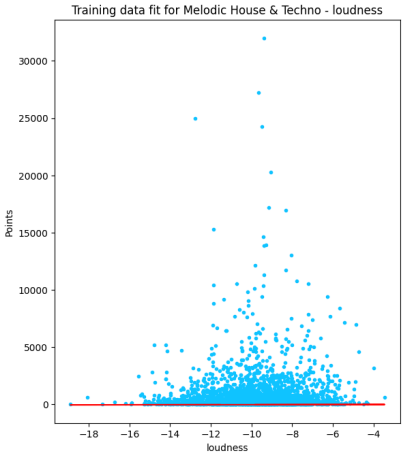

MSE: 4670191.85
RMSE: 2161.06
MAE: 918.51
R-squared: -2985.51



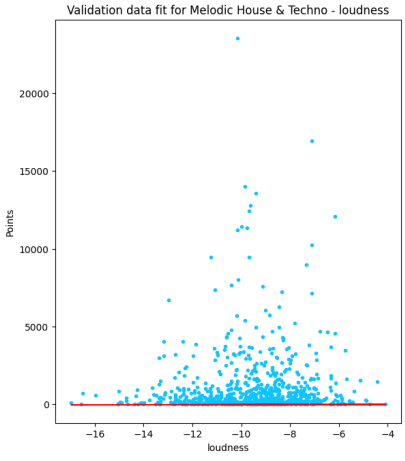

MSE: 5324115.75
RMSE: 2307.40
MAE: 1038.34
R-squared: -994.61



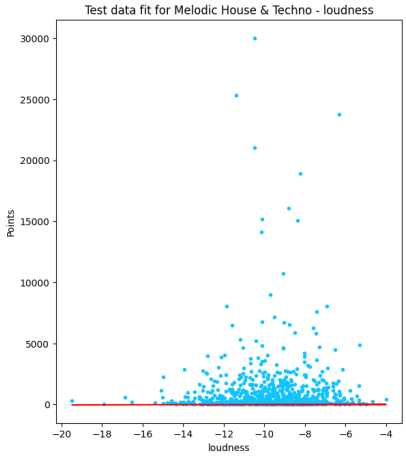

MSE: 6708327.60
RMSE: 2590.04
MAE: 984.08
R-squared: -973.71



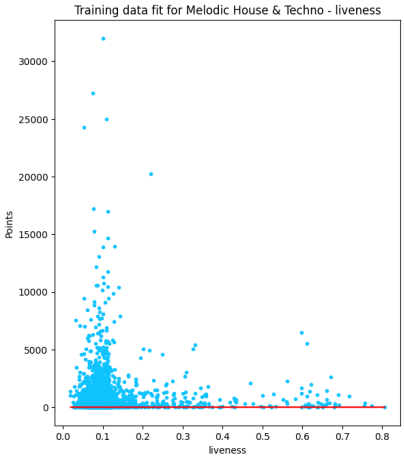

MSE: 4706514.93
RMSE: 2169.45
MAE: 938.07
R-squared: -3008.74



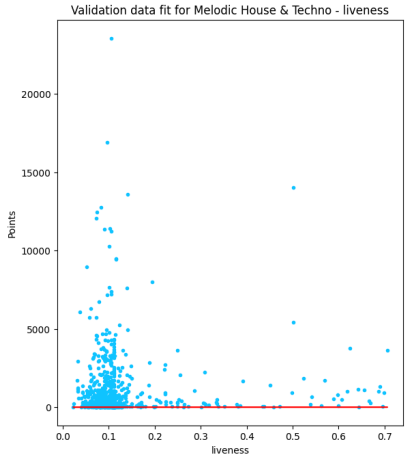

MSE: 5364719.20
RMSE: 2316.19
MAE: 1057.70
R-squared: -1002.21



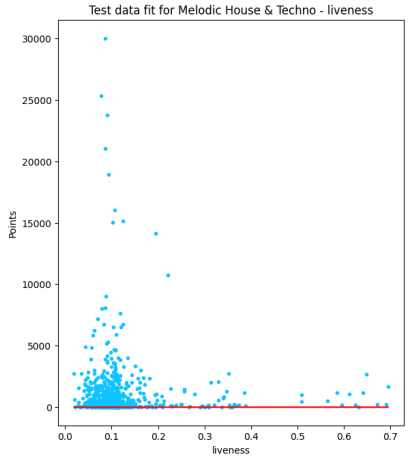

MSE: 6747438.72
RMSE: 2597.58
MAE: 1003.74
R-squared: -979.39



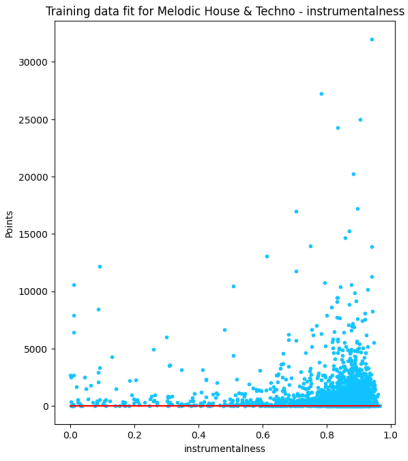

MSE: 4669947.56
RMSE: 2161.01
MAE: 918.38
R-squared: -2985.35



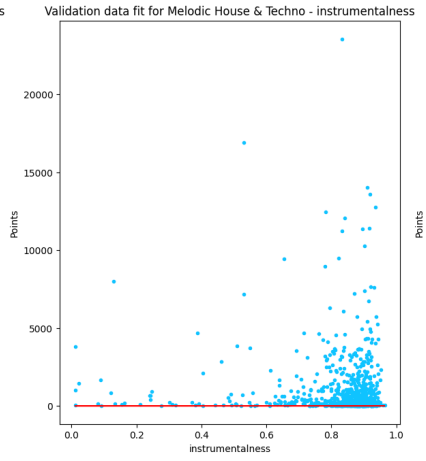

MSE: 5323821.78
RMSE: 2307.34
MAE: 1038.20
R-squared: -994.56



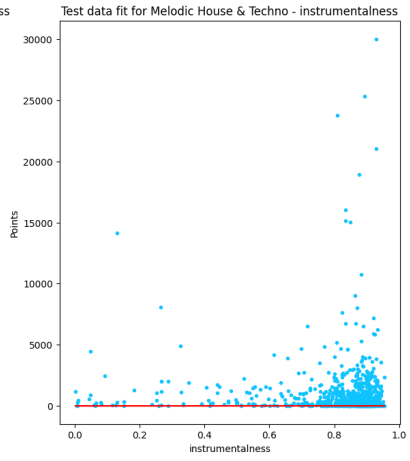

MSE: 6708085.23
RMSE: 2590.00
MAE: 983.96
R-squared: -973.67



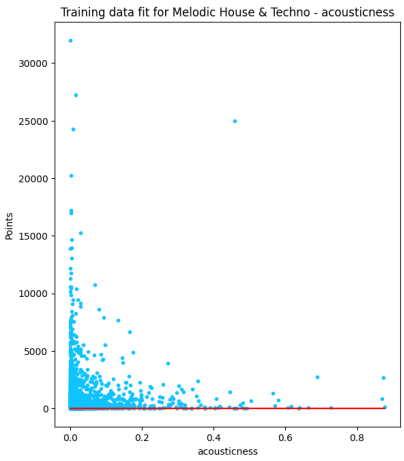

MSE: 4667392.12
RMSE: 2160.41
MAE: 917.02
R-squared: -2983.72



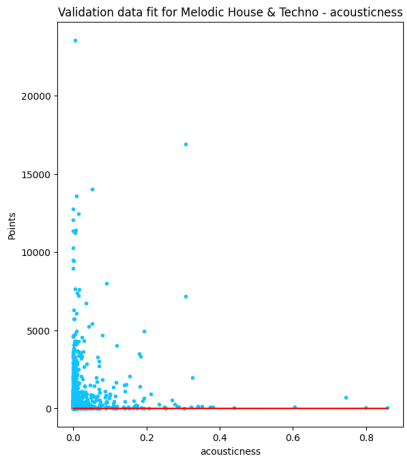

MSE: 5320903.72
RMSE: 2306.71
MAE: 1036.85
R-squared: -994.01



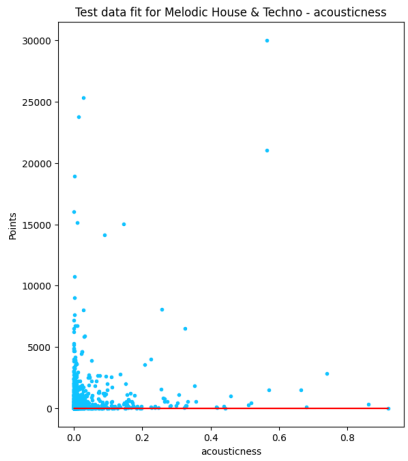

MSE: 6705352.10
RMSE: 2589.47
MAE: 982.61
R-squared: -973.27



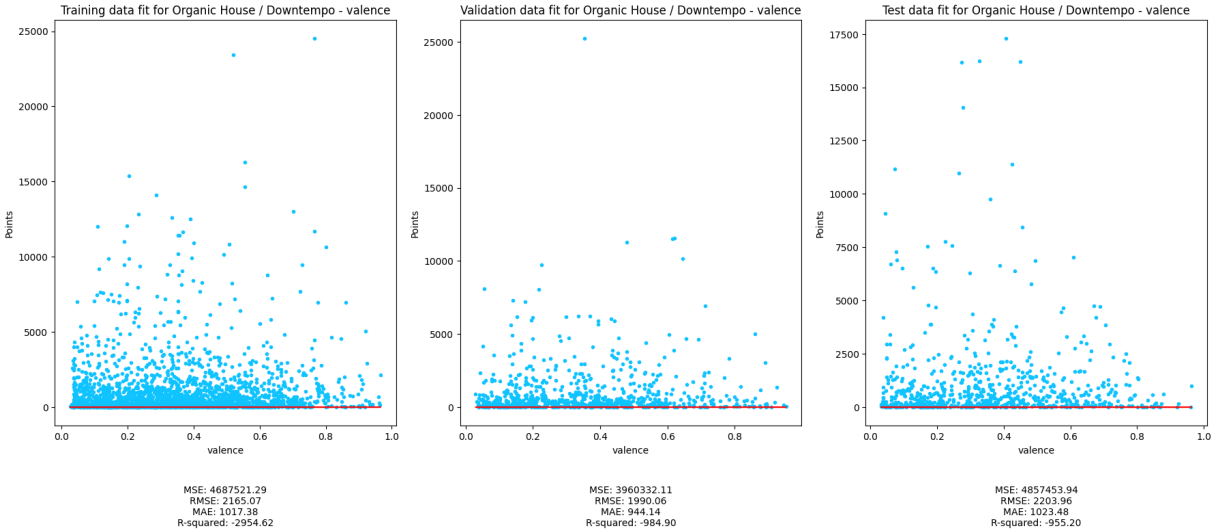

MSE: 4670258.36
RMSE: 2161.08
MAE: 918.55
R-squared: -2985.55

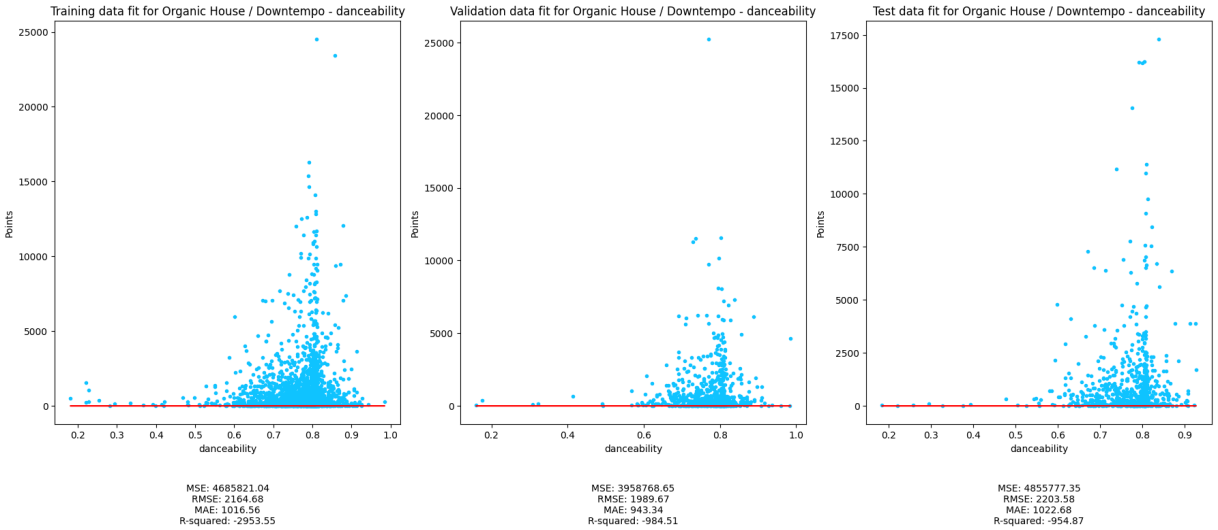


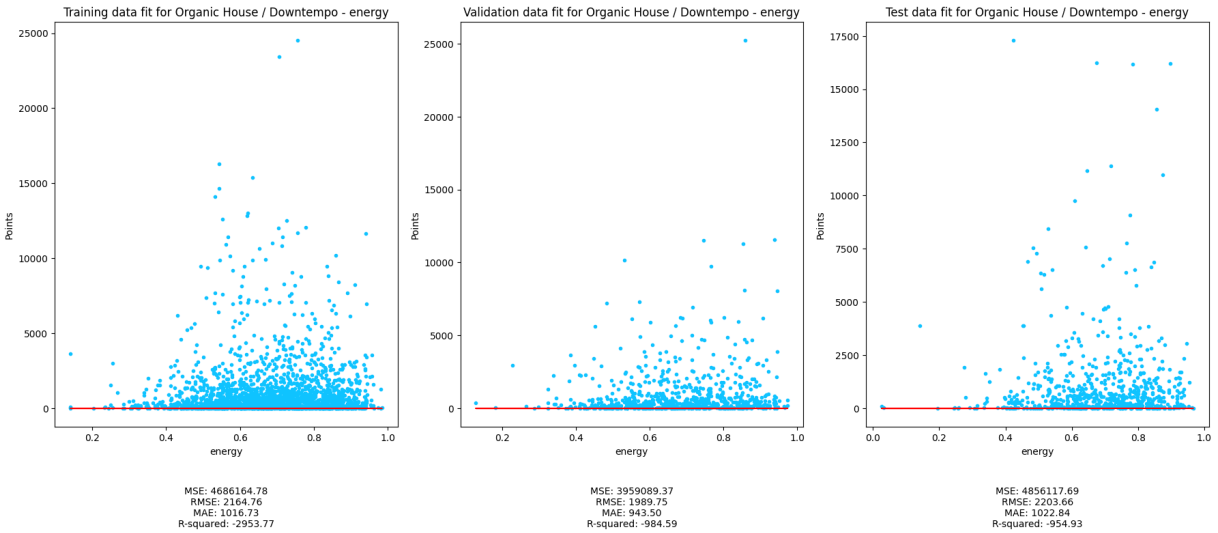

MSE: 5324209.31
RMSE: 2307.42
MAE: 1038.39
R-squared: -994.63

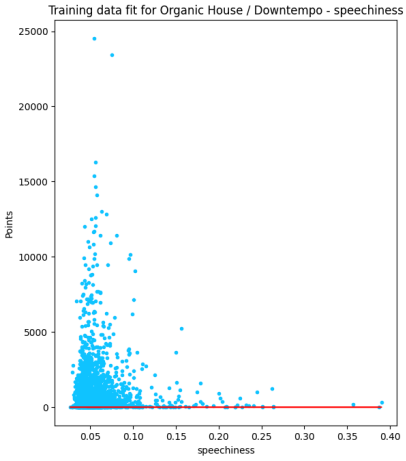



MSE: 6708386.69
RMSE: 2590.06
MAE: 984.11
R-squared: -973.72

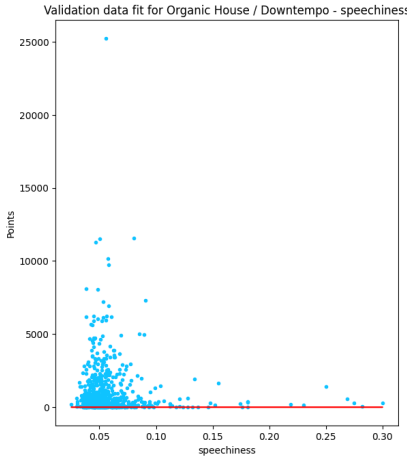

Training data fit for Organic House / Downtempo - valence
Points
valence
MSE: 4687521.29
RMSE: 2165.07
MAE: 1017.38
R-squared: -2954.62
Validation data fit for Organic House / Downtempo - valence
Points
valence
MSE: 3960332.11
RMSE: 1990.06
MAE: 944.14
R-squared: -984.90
Test data fit for Organic House / Downtempo - valence
Points
valence
MSE: 4857453.94
RMSE: 2203.96
MAE: 1023.48
R-squared: -955.20


Training data fit for Organic House / Downtempo - danceability
Points
danceability
MSE: 4685821.04
RMSE: 2164.68
MAE: 1016.56
R-squared: -2953.55
Validation data fit for Organic House / Downtempo - danceability
Points
danceability
MSE: 3958768.65
RMSE: 1989.67
MAE: 943.34
R-squared: -984.51
Test data fit for Organic House / Downtempo - danceability
Points
danceability
MSE: 4855777.35
RMSE: 2203.58
MAE: 1022.68
R-squared: -954.87

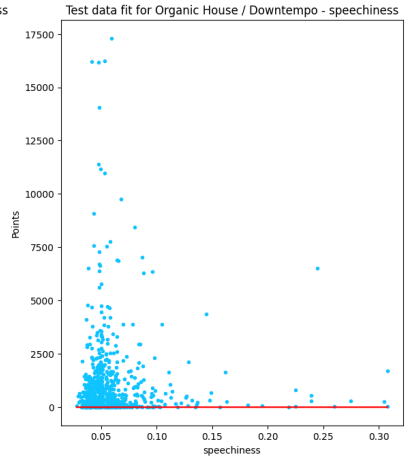

Training data fit for Organic House / Downtempo - energy
Points
energy
MSE: 4686164.78
RMSE: 2164.76
MAE: 1016.73
R-squared: -2953.77
Validation data fit for Organic House / Downtempo - energy
Points
energy
MSE: 3959089.37
RMSE: 1989.75
MAE: 943.50
R-squared: -984.59
Test data fit for Organic House / Downtempo - energy
Points
energy
MSE: 4856117.69
RMSE: 2203.66
MAE: 1022.84
R-squared: -954.93



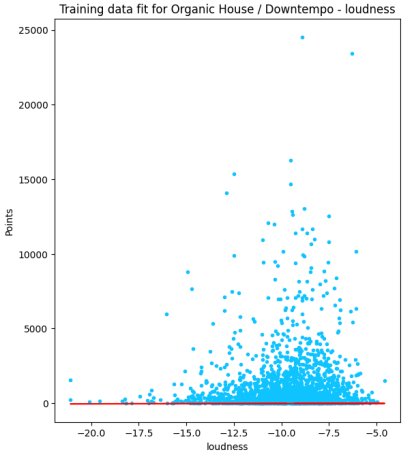

MSE: 4688708.83
RMSE: 2165.34
MAE: 1017.95
R-squared: -2955.37



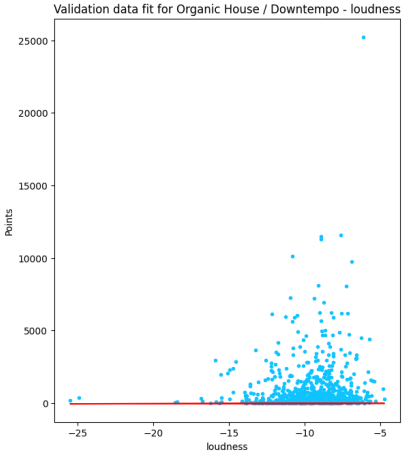

MSE: 3961460.63
RMSE: 1990.34
MAE: 944.72
R-squared: -985.18



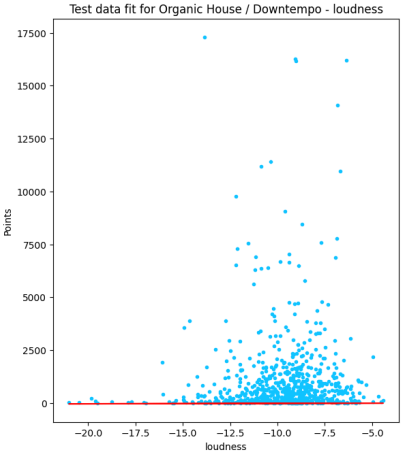

MSE: 4858662.71
RMSE: 2204.24
MAE: 1024.05
R-squared: -955.43



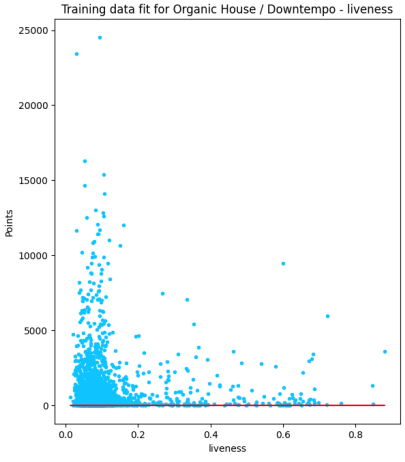

MSE: 4729092.51
RMSE: 2174.65
MAE: 1037.59
R-squared: -2980.83



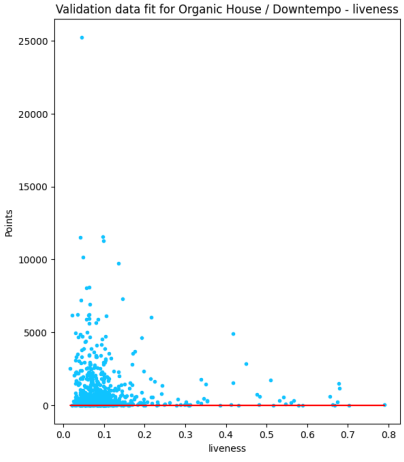

MSE: 3998428.19
RMSE: 1999.61
MAE: 964.07
R-squared: -994.38



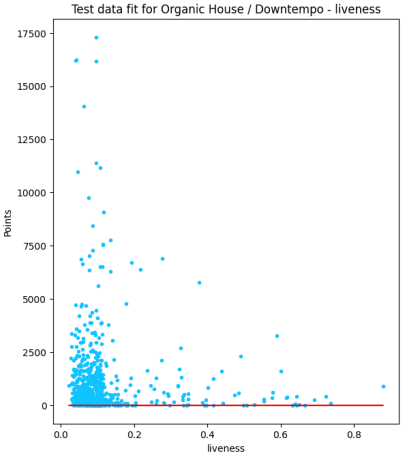

MSE: 4899195.99
RMSE: 2213.41
MAE: 1043.64
R-squared: -963.41



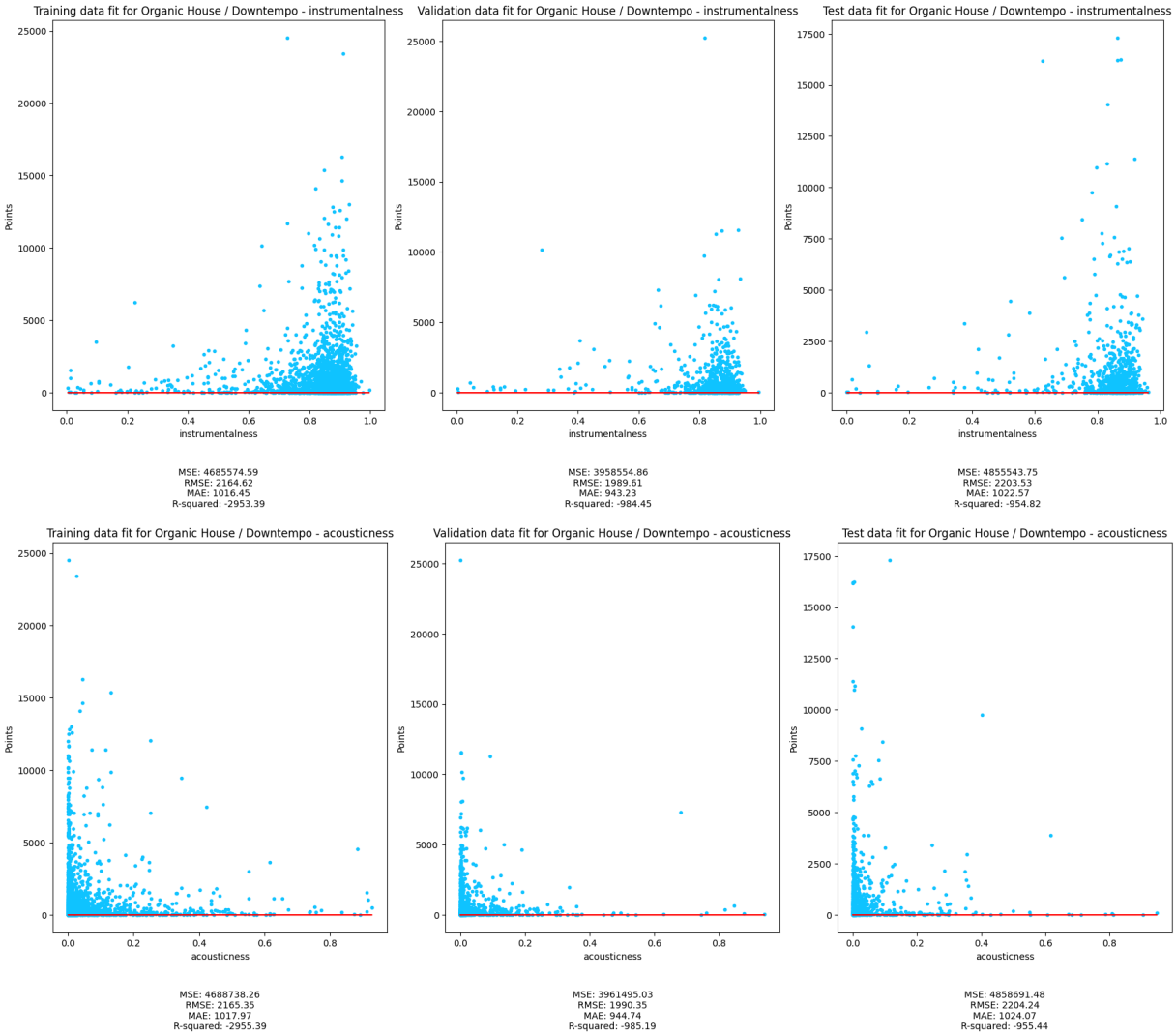

MSE: 4688478.95
RMSE: 2165.29
MAE: 1017.84
R-squared: -2955.23




MSE: 3961267.02
RMSE: 1990.29
MAE: 944.62
R-squared: -985.13




MSE: 4858440.01
RMSE: 2204.19
MAE: 1023.95
R-squared: -955.39



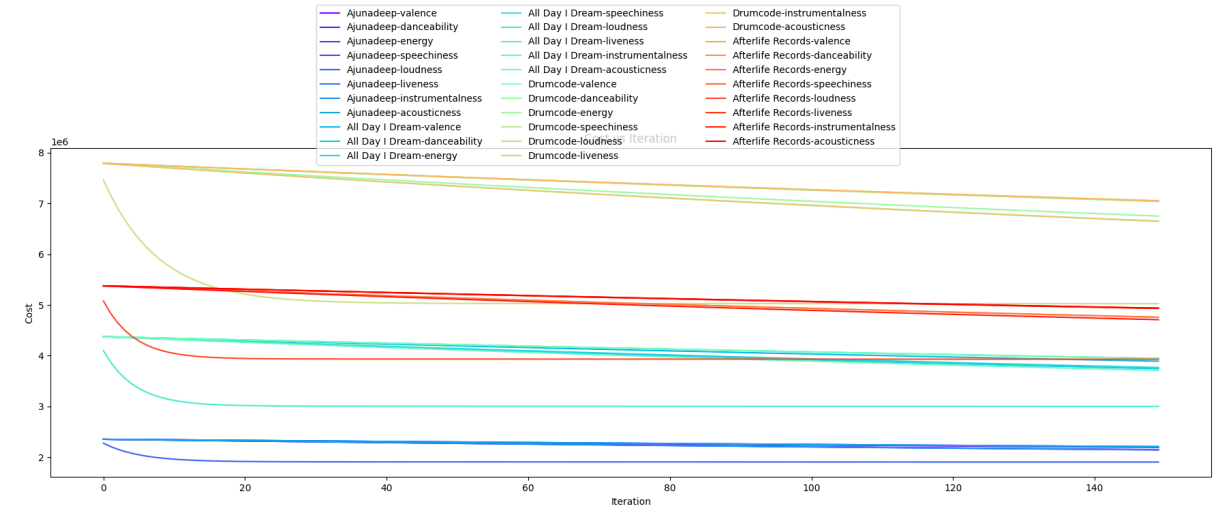

## Label function

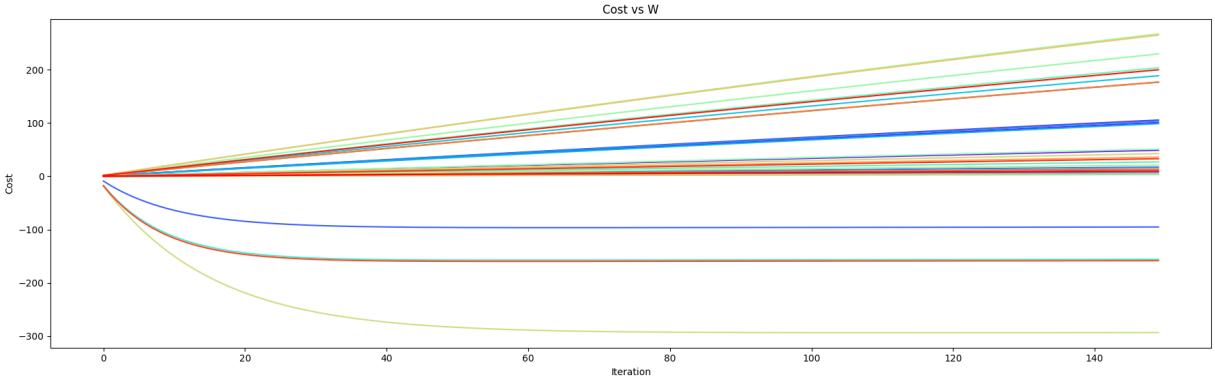
In [29]:
```
run_all_calculations_label(zscore_train, zscore_validate, zscore_test)
```

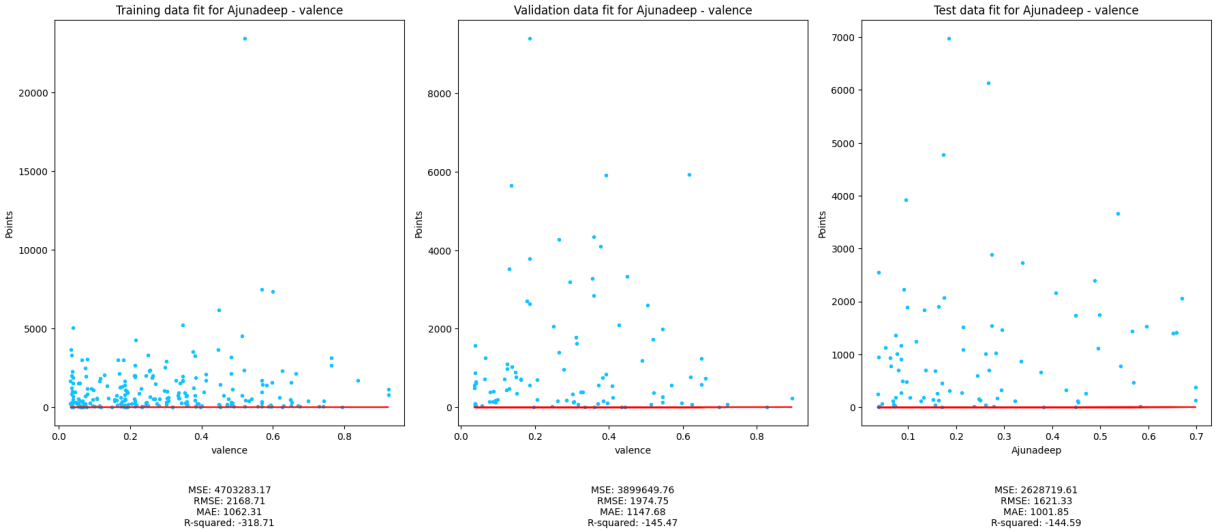
```
Cost for Ajunadeep and valence at initial w: 2351551.590604445
Gradient at initial w & b for Ajunadeep and valence: [-351.69248189] -1063.8
641975308642
Optimal w & b found by gradient descent for Ajunadeep and valence [49.019663
11] 147.22930665065965
Cost for Ajunadeep and danceability at initial w: 2350766.023915347
Gradient at initial w & b for Ajunadeep and danceability: [-745.17690823] -1
063.8641975308642
Optimal w & b found by gradient descent for Ajunadeep and danceability [100.
89369847] 143.3626235532783
Cost for Ajunadeep and energy at initial w: 2350686.1810668726
Gradient at initial w & b for Ajunadeep and energy: [-785.21847737] -1063.86
41975308642
Optimal w & b found by gradient descent for Ajunadeep and energy [105.626712
75] 142.75031625069792
Cost for Ajunadeep and speechiness at initial w: 2352145.507291593
Gradient at initial w & b for Ajunadeep and speechiness: [-54.36825062] -106
3.8641975308642
Optimal w & b found by gradient descent for Ajunadeep and speechiness [7.627
78279] 148.2309008845928
Cost for Ajunadeep and loudness at initial w: 2370472.291522215
Gradient at initial w & b for Ajunadeep and loudness: [9026.56969959] -1063.
8641975308642
Optimal w & b found by gradient descent for Ajunadeep and loudness [-95.3002
9739] 33.81858543512092
Cost for Ajunadeep and liveness at initial w: 2352003.3265876705
Gradient at initial w & b for Ajunadeep and liveness: [-125.58756255] -1063.
8641975308642
Optimal w & b found by gradient descent for Ajunadeep and liveness [17.19421
101] 148.07972329093036
Cost for Ajunadeep and instrumentalness at initial w: 2350701.8027943186
Gradient at initial w & b for Ajunadeep and instrumentalness: [-777.4977625
7] -1063.8641975308642
Optimal w & b found by gradient descent for Ajunadeep and instrumentalness
[103.7604518] 142.70579108443954
Cost for Ajunadeep and acousticness at initial w: 2352117.391967008
Gradient at initial w & b for Ajunadeep and acousticness: [-68.52663707] -10
63.8641975308642
Optimal w & b found by gradient descent for Ajunadeep and acousticness [9.12
313644] 148.18999845001312
Cost for All Day I Dream and valence at initial w: 4374706.631515093
Gradient at initial w & b for All Day I Dream and valence: [-718.42869249] -
1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and valence [98.
80149711] 251.37612053067534
Cost for All Day I Dream and danceability at initial w: 4373318.4026616085
Gradient at initial w & b for All Day I Dream and danceability: [-1413.31968
075] -1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and danceability
[188.83500227] 243.5109501154327
Cost for All Day I Dream and energy at initial w: 4373529.406873181
Gradient at initial w & b for All Day I Dream and energy: [-1307.61892958] -
1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and energy [176.
67953591] 245.38599541744273
Cost for All Day I Dream and speechiness at initial w: 4375943.12054534
```

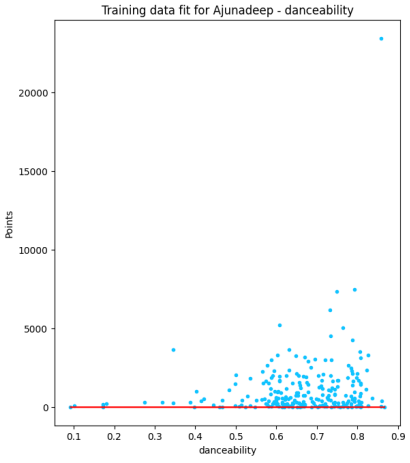
```
Gradient at initial w & b for All Day I Dream and speechiness: [-99.6625704
2] -1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and speechiness
[13.84877095] 254.11465653365138
Cost for All Day I Dream and loudness at initial w: 4410534.751823575
Gradient at initial w & b for All Day I Dream and loudness: [17099.48701878]
-1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and loudness [-1
55.93307648] 46.10080570373604
Cost for All Day I Dream and liveness at initial w: 4375753.756968231
Gradient at initial w & b for All Day I Dream and liveness: [-194.41427559]
-1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and liveness [2
7.0469155] 253.96259289258728
Cost for All Day I Dream and instrumentalness at initial w: 4373070.27178680
9
Gradient at initial w & b for All Day I Dream and instrumentalness: [-1537.5
7026244] -1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and instrumental
ness [203.70808193] 241.64284502616215
Cost for All Day I Dream and acousticness at initial w: 4376092.281508058
Gradient at initial w & b for All Day I Dream and acousticness: [-25.0638698
9] -1823.8920187793428
Optimal w & b found by gradient descent for All Day I Dream and acousticness
[3.18208255] 254.16436111494025
Cost for Drumcode and valence at initial w: 7795014.984720605
Gradient at initial w & b for Drumcode and valence: [-370.39920707] -2396.70
70707070707
Optimal w & b found by gradient descent for Drumcode and valence [51.0035628
7] 333.3622981595192
Cost for Drumcode and danceability at initial w: 7792336.2016281355
Gradient at initial w & b for Drumcode and danceability: [-1710.78772222] -2
396.7070707070707
Optimal w & b found by gradient descent for Drumcode and danceability [229.8
4719819] 321.9994651496664
Cost for Drumcode and energy at initial w: 7791717.043620086
Gradient at initial w & b for Drumcode and energy: [-2020.70379798] -2396.70
70707070707
Optimal w & b found by gradient descent for Drumcode and energy [267.4502131
2] 317.40737649714833
Cost for Drumcode and speechiness at initial w: 7795455.409453174
Gradient at initial w & b for Drumcode and speechiness: [-150.03332172] -239
6.7070707070707
Optimal w & b found by gradient descent for Drumcode and speechiness [20.886
70363] 333.90141457514136
Cost for Drumcode and loudness at initial w: 7832790.381346354
Gradient at initial w & b for Drumcode and loudness: [18463.60373232] -2396.
7070707070707
Optimal w & b found by gradient descent for Drumcode and loudness [-293.6191
324] 51.01955471657453
Cost for Drumcode and liveness at initial w: 7795139.740135742
Gradient at initial w & b for Drumcode and liveness: [-307.97264242] -2396.7
070707070707
Optimal w & b found by gradient descent for Drumcode and liveness [42.512489
54] 333.5632322514754
Cost for Drumcode and instrumentalness at initial w: 7791748.123532021
```

```
Gradient at initial w & b for Drumcode and instrumentalness: [-2005.1641218
2] -2396.7070707070707
Optimal w & b found by gradient descent for Drumcode and instrumentalness [2
65.23213131] 317.5620650998996
Cost for Drumcode and acousticness at initial w: 7795680.87670997
Gradient at initial w & b for Drumcode and acousticness: [-37.25694851] -239
6.7070707070707
Optimal w & b found by gradient descent for Drumcode and acousticness [5.083
16731] 333.9896671517275
Cost for Afterlife Records and valence at initial w: 5377436.893933163
Gradient at initial w & b for Afterlife Records and valence: [-264.27614912]
-1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and valence [3
6.11990852] 256.9822427740441
Cost for Afterlife Records and danceability at initial w: 5375335.149494011
Gradient at initial w & b for Afterlife Records and danceability: [-1316.132
04094] -1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and danceabili
ty [177.02031998] 248.30510894872745
Cost for Afterlife Records and energy at initial w: 5375343.786917847
Gradient at initial w & b for Afterlife Records and energy: [-1311.78135088]
-1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and energy [17
6.9031103] 248.54501652893362
Cost for Afterlife Records and speechiness at initial w: 5377774.7565425895
Gradient at initial w & b for Afterlife Records and speechiness: [-95.178107
6] -1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and speechines
s [13.26855769] 257.38737866941847
Cost for Afterlife Records and loudness at initial w: 5413716.446326303
Gradient at initial w & b for Afterlife Records and loudness: [17776.4264912
3] -1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and loudness
[-158.47064913] 42.35717422849542
Cost for Afterlife Records and liveness at initial w: 5377490.97411017
Gradient at initial w & b for Afterlife Records and liveness: [-237.1703766
1] -1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and liveness
[32.97353187] 257.1281010246346
Cost for Afterlife Records and instrumentalness at initial w: 5374949.056414
514
Gradient at initial w & b for Afterlife Records and instrumentalness: [-150
9.48525146] -1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and instrument
alness [200.01236296] 245.32869040826162
Cost for Afterlife Records and acousticness at initial w: 5377812.102801613
Gradient at initial w & b for Afterlife Records and acousticness: [-76.54089
643] -1847.3216374269007
Optimal w & b found by gradient descent for Afterlife Records and acousticne
ss [10.13322064] 257.3858780688038
```

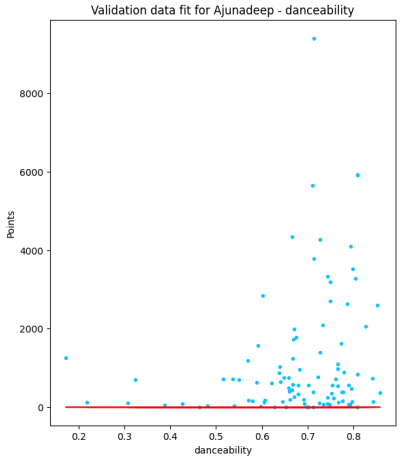

Ajunadeep-valence
Ajunadeep-danceability
Ajunadeep-energy
Ajunadeep-speechiness
Ajunadeep-loudness
Ajunadeep-liveness
Ajunadeep-instrumentalness
Ajunadeep-acousticness
All Day I Dream-valence
All Day I Dream-danceability
All Day I Dream-energy
All Day I Dream-speechiness
All Day I Dream-loudness
All Day I Dream-liveness
All Day I Dream-instrumentalness
All Day I Dream-acousticness
Drumcode-valence
Drumcode-danceability
Drumcode-energy
Drumcode-speechiness
Drumcode-loudness
Drumcode-liveness
Drumcode-instrumentalness
Drumcode-acousticness
Afterlife Records-valence
Afterlife Records-danceability
Afterlife Records-energy
Afterlife Records-speechiness
Afterlife Records-loudness
Afterlife Records-liveness
Afterlife Records-instrumentalness
Afterlife Records-acousticness
1e6
Cost
Iteration


Cost vs W
Cost
Iteration

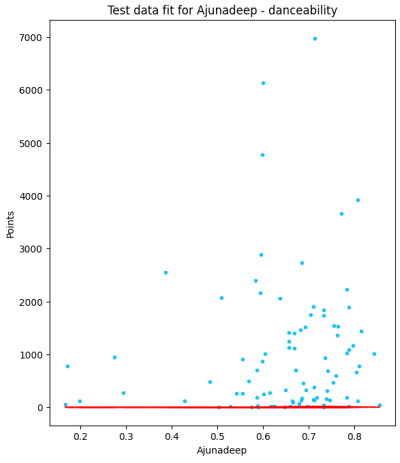

Training data fit for Ajunadeep - valence
Points
valence
MSE: 4703283.17
RMSE: 2168.71
MAE: 1062.31
R-squared: -318.71
Validation data fit for Ajunadeep - valence
Points
valence
MSE: 3899649.76
RMSE: 1974.75
MAE: 1147.68
R-squared: -145.47
Test data fit for Ajunadeep - valence
Points
Ajunadeep
MSE: 2628719.61
RMSE: 1621.33
MAE: 1001.85
R-squared: -144.59



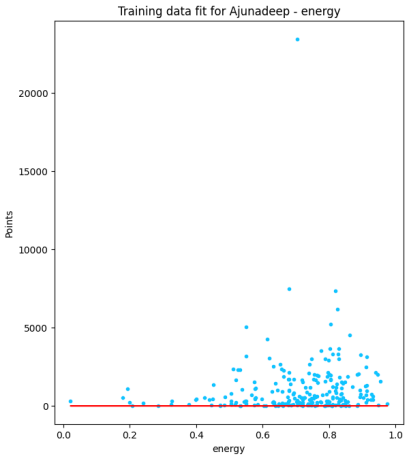

MSE: 4701700.13
RMSE: 2168.34
MAE: 1061.59
R-squared: -318.60



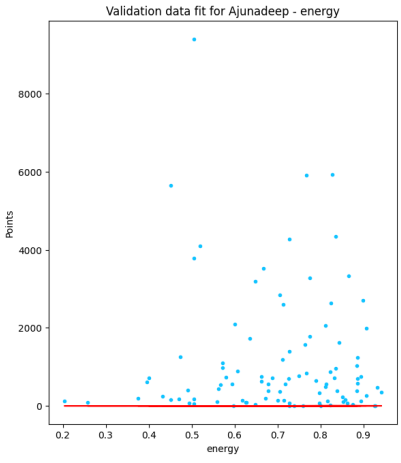

MSE: 3897902.39
RMSE: 1974.31
MAE: 1146.93
R-squared: -145.41



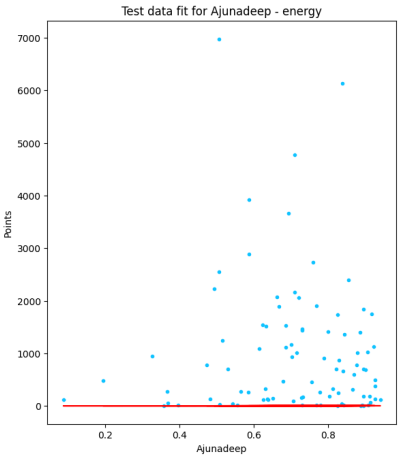

MSE: 2627112.28
RMSE: 1620.84
MAE: 1001.06
R-squared: -144.50



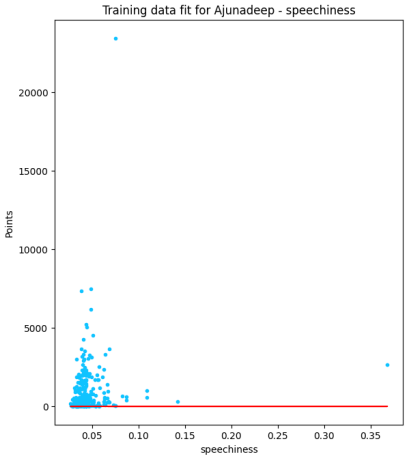

MSE: 4701497.02
RMSE: 2168.29
MAE: 1061.50
R-squared: -318.59



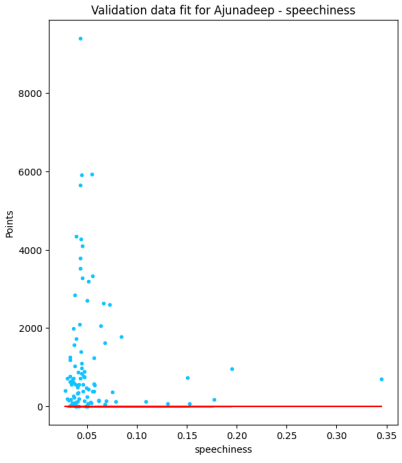

MSE: 3897790.10
RMSE: 1974.28
MAE: 1146.88
R-squared: -145.40



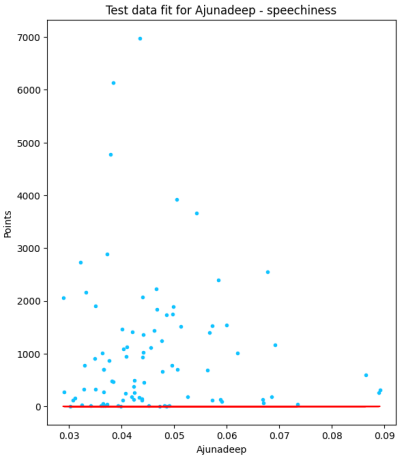

MSE: 2626865.35
RMSE: 1620.76
MAE: 1000.94
R-squared: -144.49



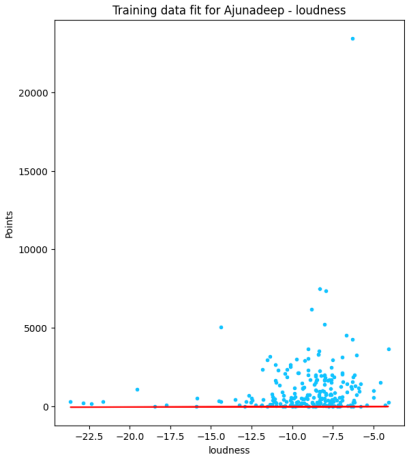

MSE: 4704311.46
RMSE: 2168.94
MAE: 1062.77
R-squared: -318.78



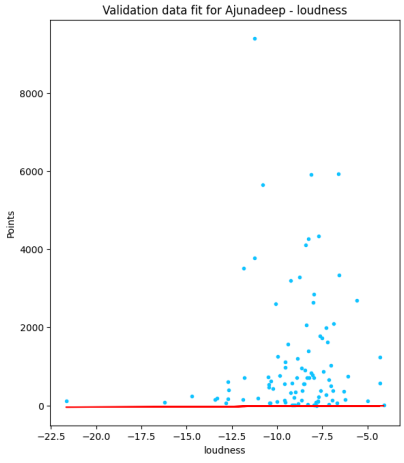

MSE: 3900761.83
RMSE: 1975.03
MAE: 1148.17
R-squared: -145.52



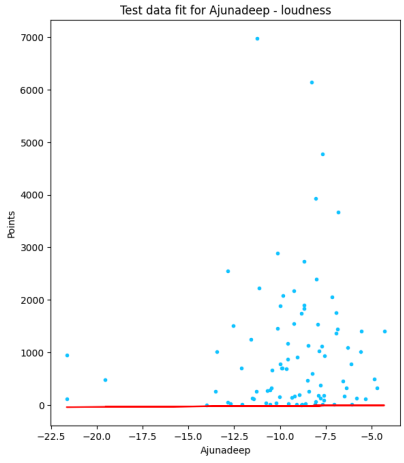

MSE: 2629532.72
RMSE: 1621.58
MAE: 1002.25
R-squared: -144.63



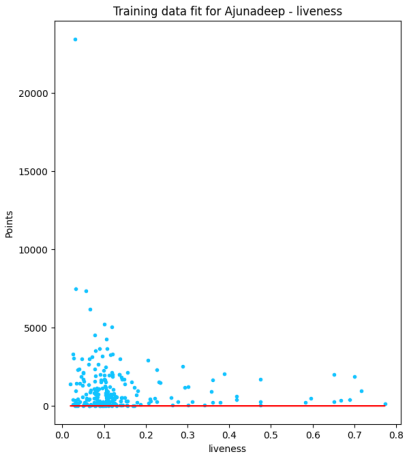

MSE: 4743715.64
RMSE: 2178.01
MAE: 1081.14
R-squared: -321.46



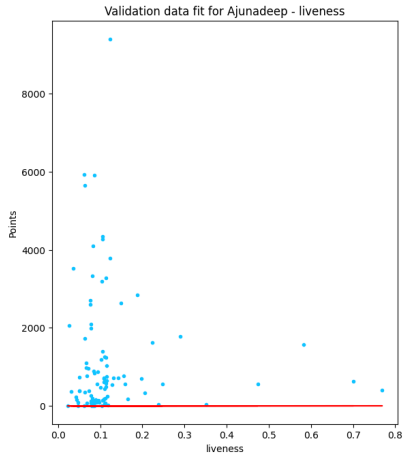

MSE: 3942616.89
RMSE: 1985.60
MAE: 1166.24
R-squared: -147.09



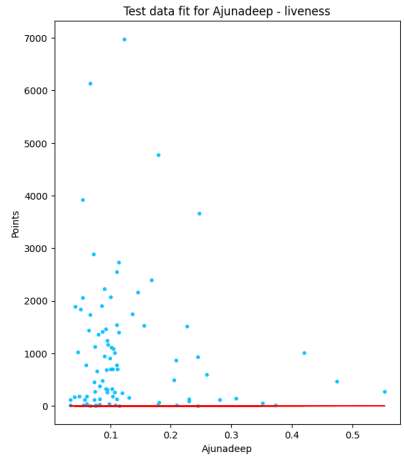

MSE: 2667473.67
RMSE: 1633.24
MAE: 1020.99
R-squared: -146.73



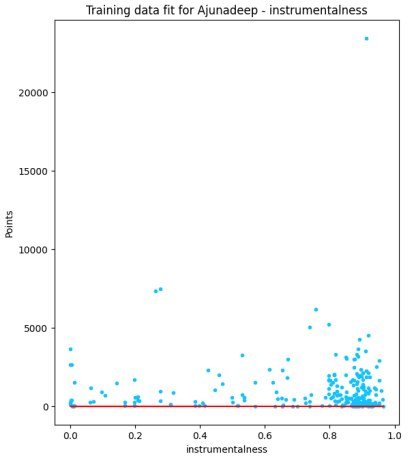

MSE: 4703910.71
RMSE: 2168.85
MAE: 1062.59
R-squared: -318.75



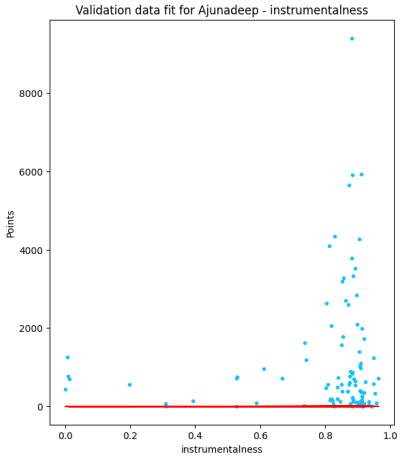

MSE: 3900441.34
RMSE: 1974.95
MAE: 1148.03
R-squared: -145.50



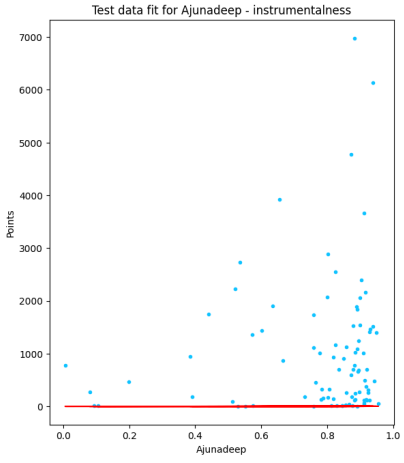

MSE: 2629193.19
RMSE: 1621.48
MAE: 1002.08
R-squared: -144.61



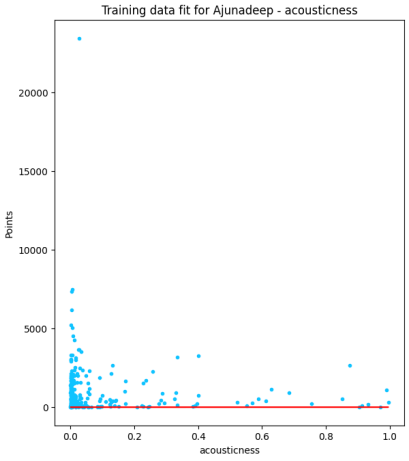

MSE: 4701426.57
RMSE: 2168.28
MAE: 1061.47
R-squared: -318.58



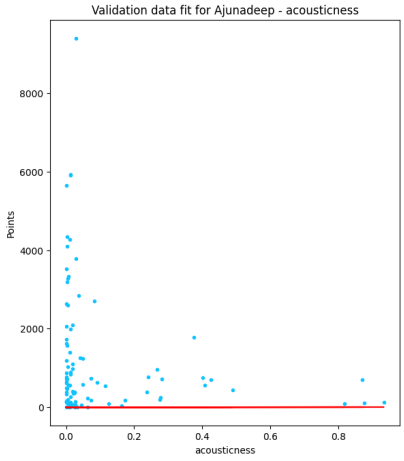

MSE: 3897350.47
RMSE: 1974.17
MAE: 1146.69
R-squared: -145.39



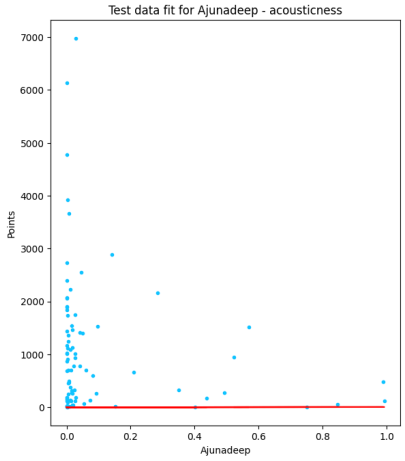

MSE: 2626614.78
RMSE: 1620.68
MAE: 1000.82
R-squared: -144.47



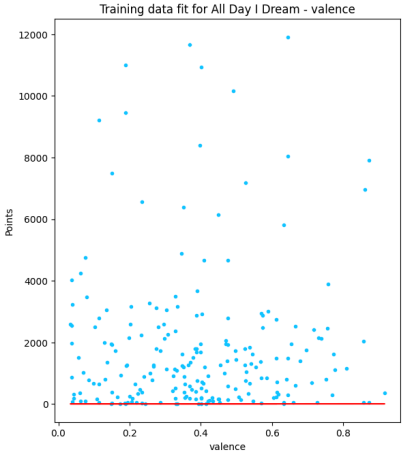

MSE: 4704088.71
RMSE: 2168.89
MAE: 1062.67
R-squared: -318.76



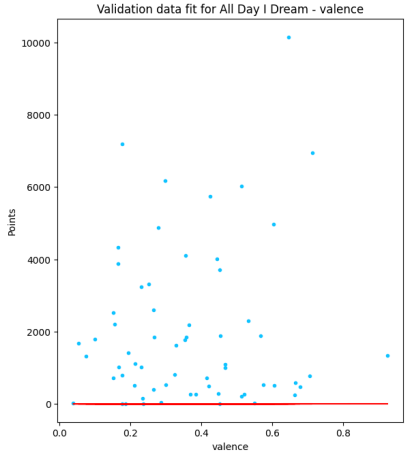

MSE: 3900588.64
RMSE: 1974.99
MAE: 1148.09
R-squared: -145.51



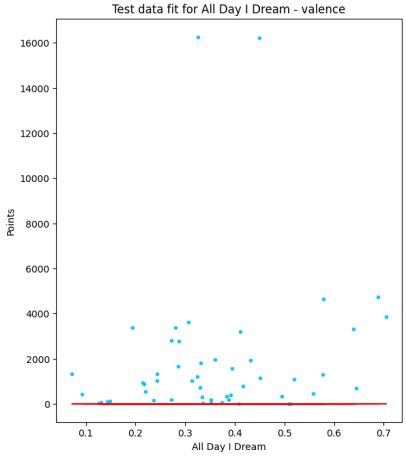

MSE: 2629347.75
RMSE: 1621.53
MAE: 1002.16
R-squared: -144.62



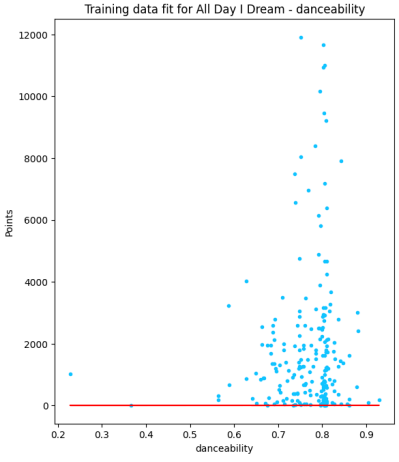

MSE: 8749460.11
RMSE: 2957.95
MAE: 1822.12
R-squared: -342.25



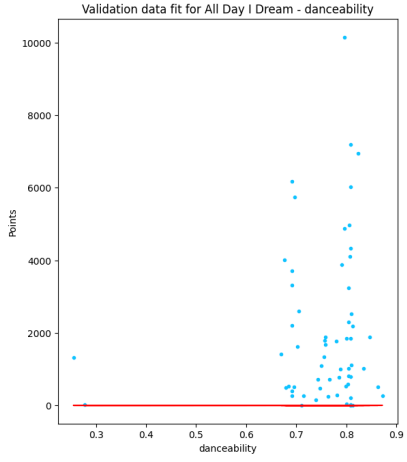

MSE: 8430526.94
RMSE: 2903.54
MAE: 1963.48
R-squared: -115.09



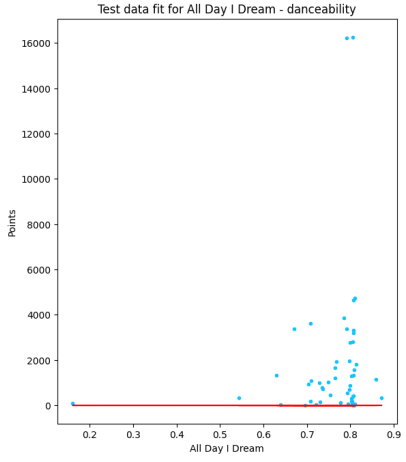

MSE: 13530181.96
RMSE: 3678.34
MAE: 1853.24
R-squared: -67.35



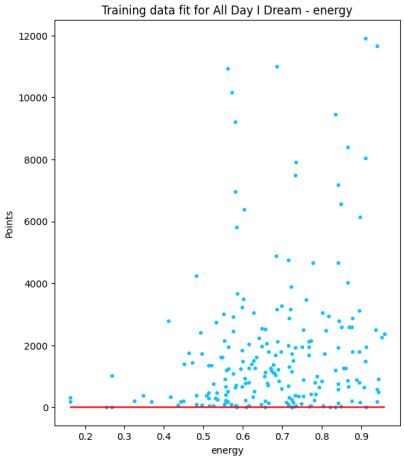

MSE: 8746702.59
RMSE: 2957.48
MAE: 1821.37
R-squared: -342.14



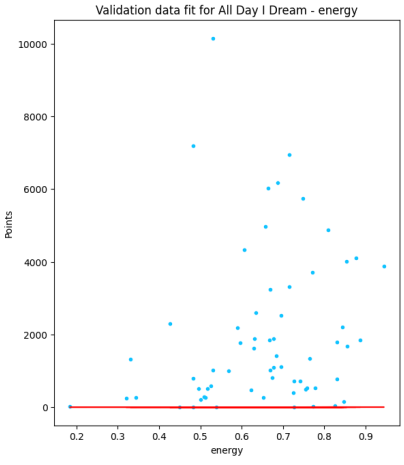

MSE: 8427473.50
RMSE: 2903.01
MAE: 1962.70
R-squared: -115.04



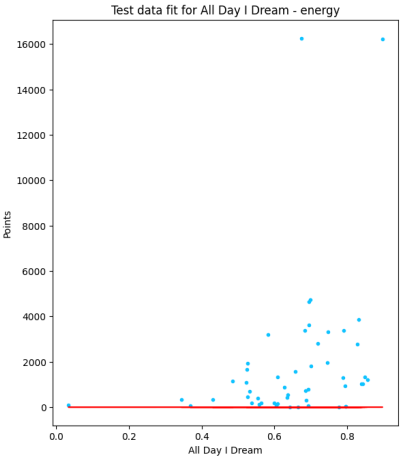

MSE: 13527228.57
RMSE: 3677.94
MAE: 1852.45
R-squared: -67.34



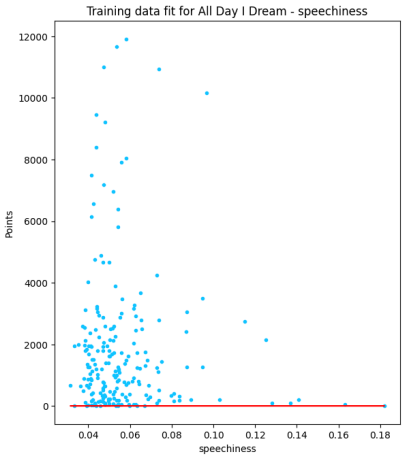

MSE: 8747349.46
RMSE: 2957.59
MAE: 1821.55
R-squared: -342.17



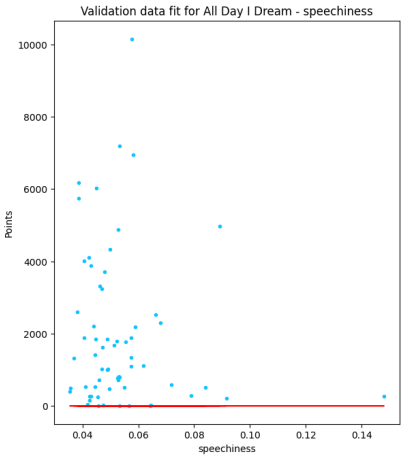

MSE: 8428288.71
RMSE: 2903.15
MAE: 1962.91
R-squared: -115.05



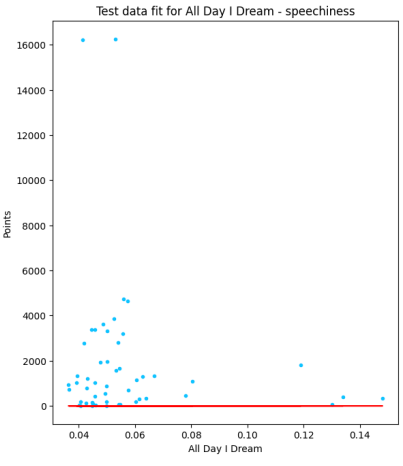

MSE: 13528017.46
RMSE: 3678.05
MAE: 1852.66
R-squared: -67.34



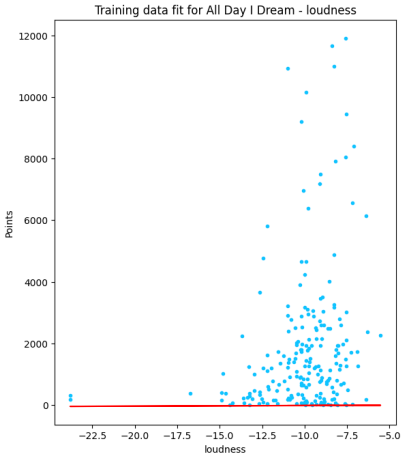

MSE: 8751873.93
RMSE: 2958.36
MAE: 1822.78
R-squared: -342.35



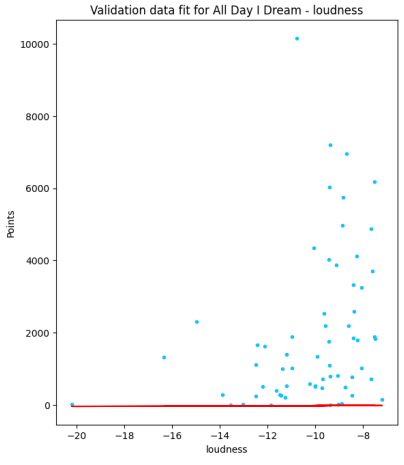

MSE: 8432971.17
RMSE: 2903.96
MAE: 1964.10
R-squared: -115.12



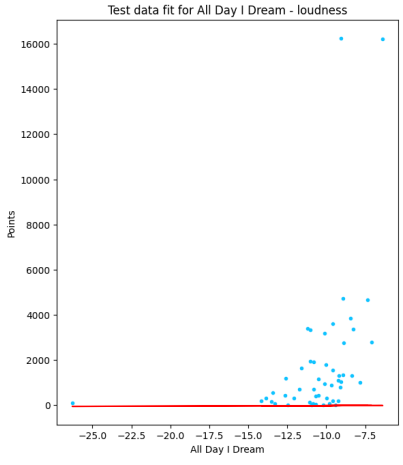

MSE: 13532415.42
RMSE: 3678.64
MAE: 1853.85
R-squared: -67.36



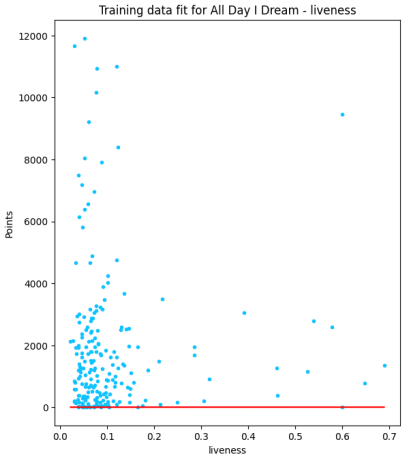

MSE: 8826181.72
RMSE: 2970.89
MAE: 1843.04
R-squared: -345.26



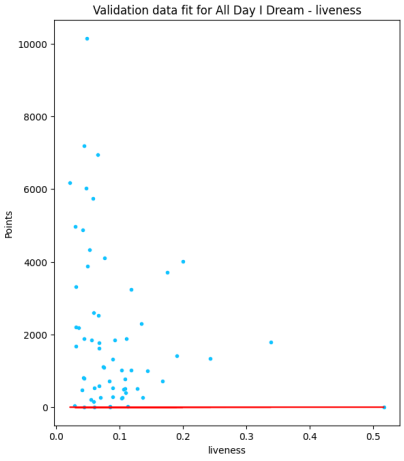

MSE: 8513635.85
RMSE: 2917.81
MAE: 1984.52
R-squared: -116.23



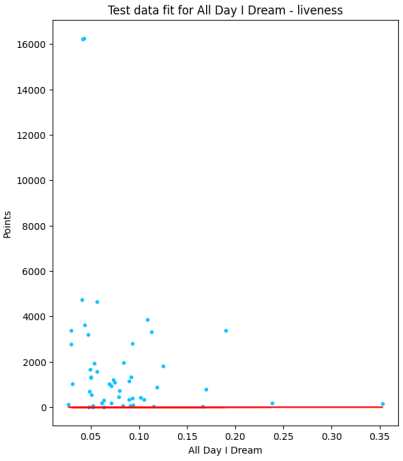

MSE: 13612004.34
RMSE: 3689.44
MAE: 1875.18
R-squared: -67.76



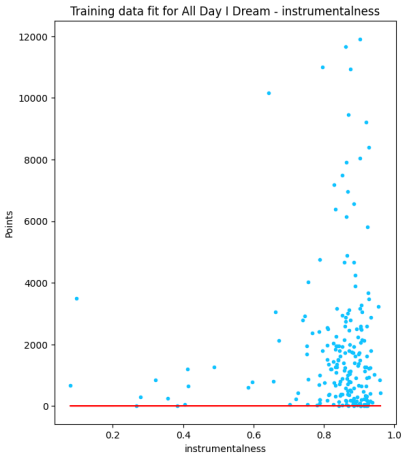

MSE: 8751510.76
RMSE: 2958.30
MAE: 1822.68
R-squared: -342.33



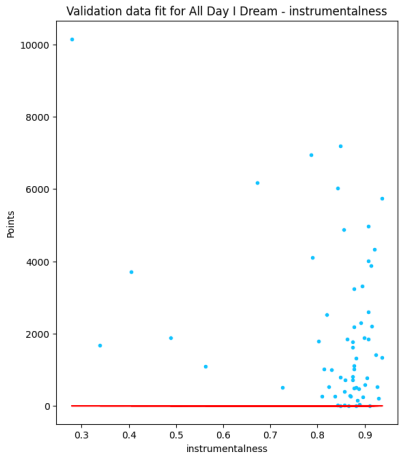

MSE: 8432647.69
RMSE: 2903.90
MAE: 1964.02
R-squared: -115.11



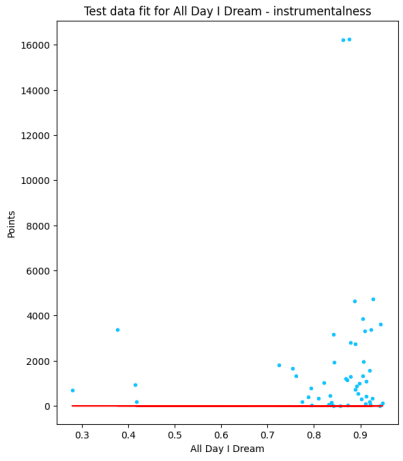

MSE: 13532210.07
RMSE: 3678.62
MAE: 1853.79
R-squared: -67.36



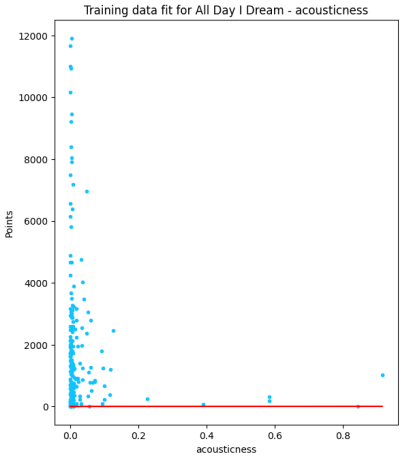

MSE: 8746220.57
RMSE: 2957.40
MAE: 1821.24
R-squared: -342.12



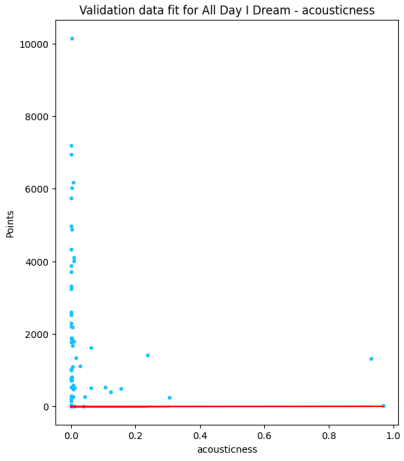

MSE: 8426855.09
RMSE: 2902.90
MAE: 1962.54
R-squared: -115.03



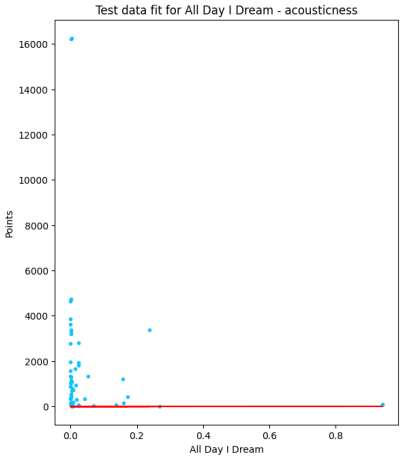

MSE: 13526687.28
RMSE: 3677.86
MAE: 1852.30
R-squared: -67.33



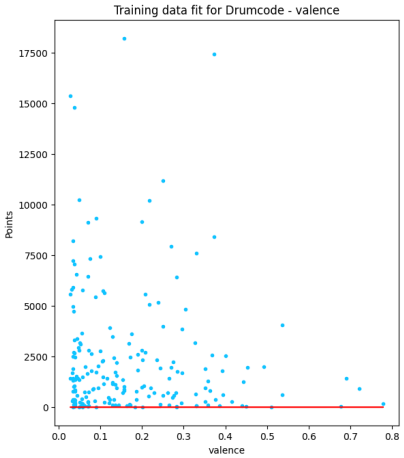

MSE: 8752069.22
RMSE: 2958.39
MAE: 1822.83
R-squared: -342.35



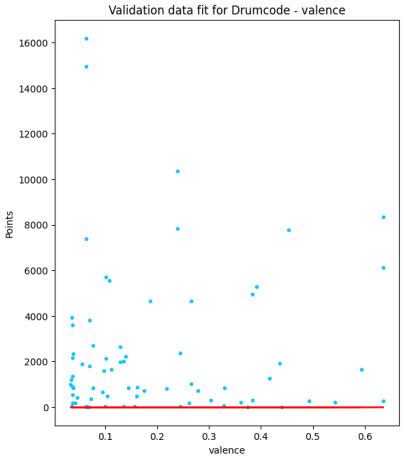

MSE: 8432988.72
RMSE: 2903.96
MAE: 1964.10
R-squared: -115.12



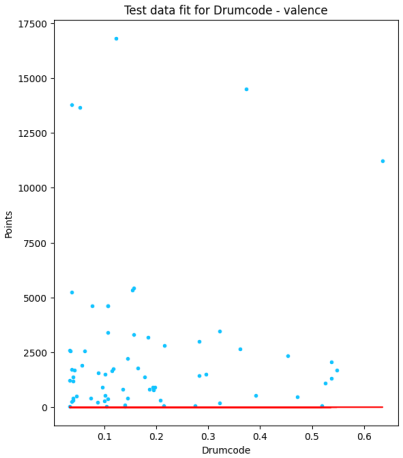

MSE: 13532478.96
RMSE: 3678.65
MAE: 1853.86
R-squared: -67.36



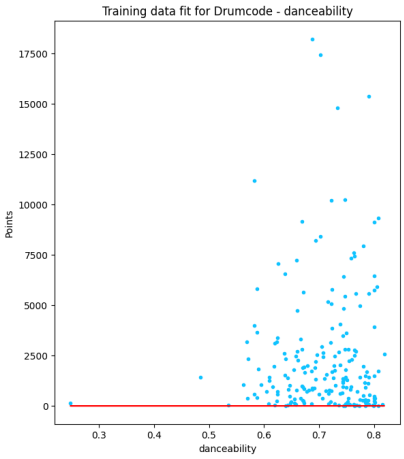

MSE: 15589877.12
RMSE: 3948.40
MAE: 2395.37
R-squared: -312.31



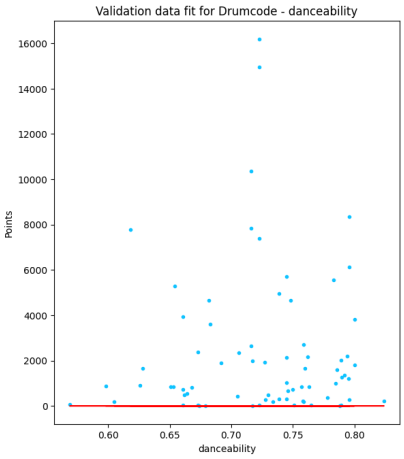

MSE: 16114987.18
RMSE: 4014.35
MAE: 2374.79
R-squared: -109.76



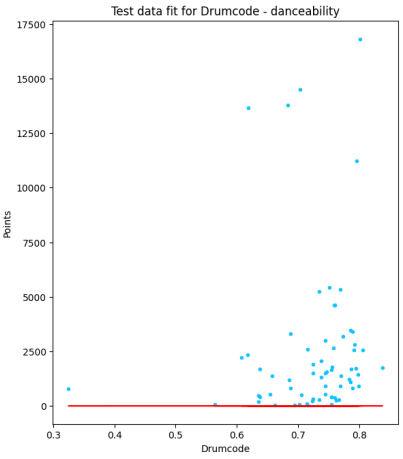

MSE: 19041521.73
RMSE: 4363.66
MAE: 2536.78
R-squared: -101.71



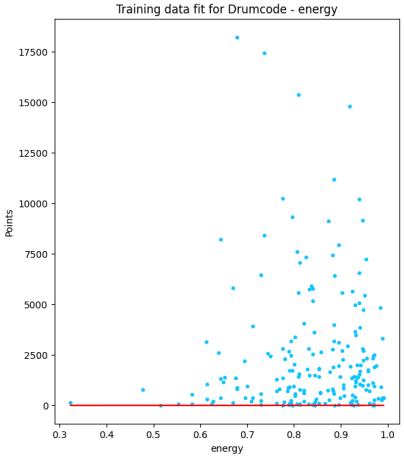

MSE: 15584714.46
RMSE: 3947.75
MAE: 2394.29
R-squared: -312.21



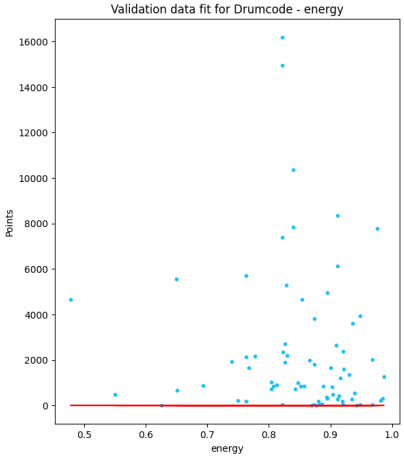

MSE: 16109979.96
RMSE: 4013.72
MAE: 2373.74
R-squared: -109.73



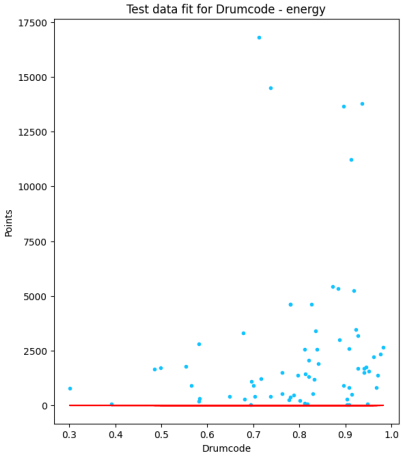

MSE: 19036025.35
RMSE: 4363.03
MAE: 2535.69
R-squared: -101.68



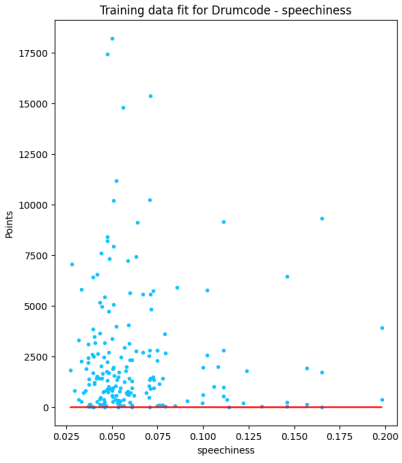

MSE: 15583474.65
RMSE: 3947.59
MAE: 2394.03
R-squared: -312.18



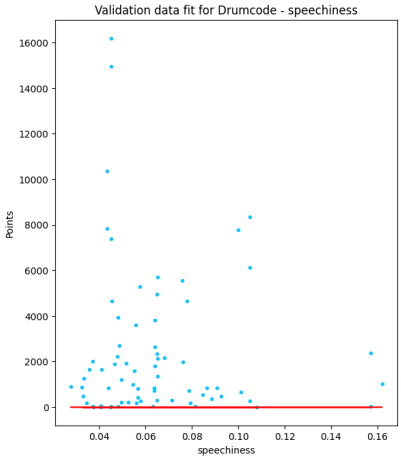

MSE: 16108771.43
RMSE: 4013.57
MAE: 2373.50
R-squared: -109.72



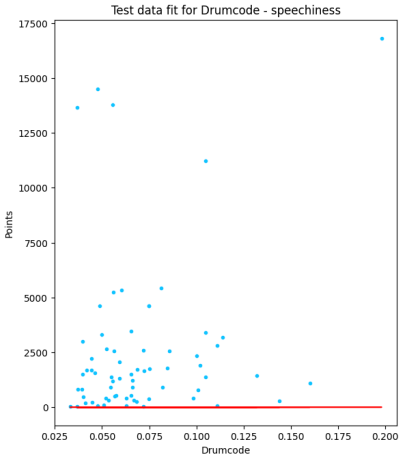

MSE: 19035309.62
RMSE: 4362.95
MAE: 2535.55
R-squared: -101.68



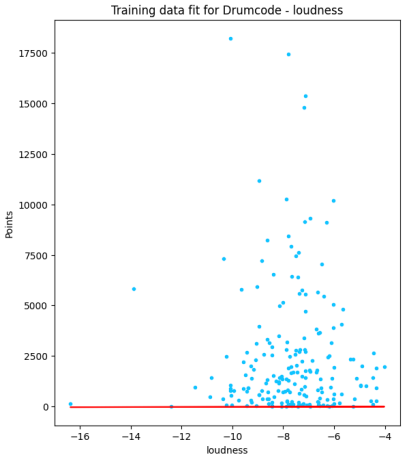

MSE: 15590907.71
RMSE: 3948.53
MAE: 2395.58
R-squared: -312.33



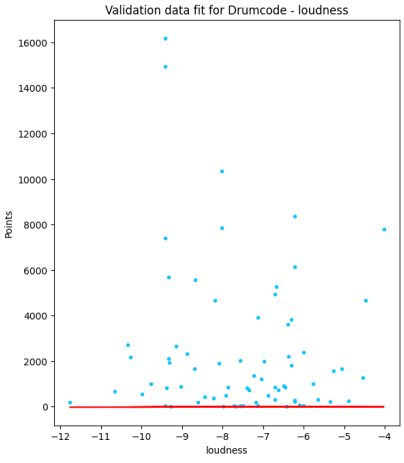

MSE: 16116246.13
RMSE: 4014.50
MAE: 2375.05
R-squared: -109.77



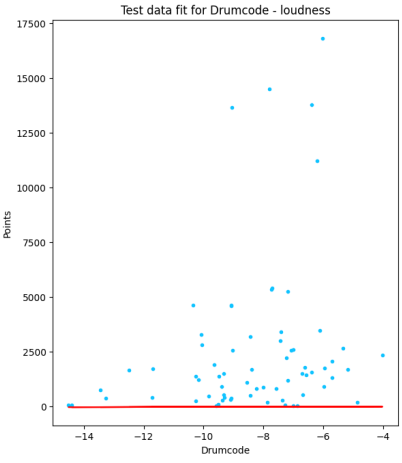

MSE: 19042678.43
RMSE: 4363.79
MAE: 2537.01
R-squared: -101.72



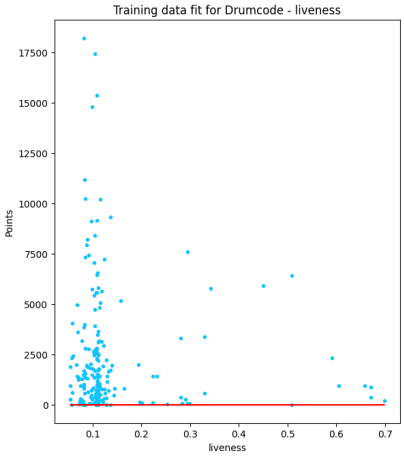

MSE: 15665224.62
RMSE: 3957.93
MAE: 2411.04
R-squared: -313.83



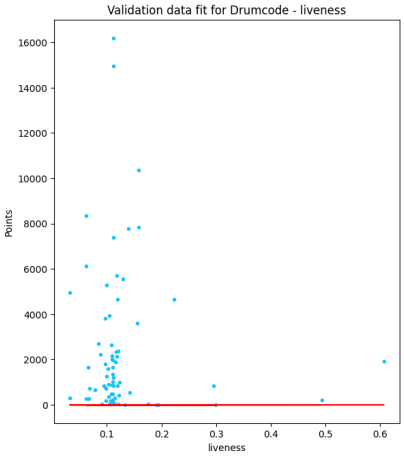

MSE: 16188531.16
RMSE: 4023.50
MAE: 2390.22
R-squared: -110.27



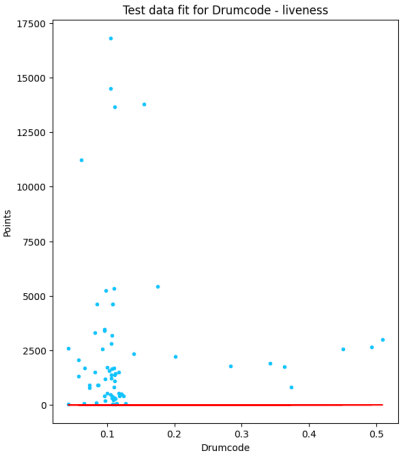

MSE: 19128771.59
RMSE: 4373.65
MAE: 2553.91
R-squared: -102.18



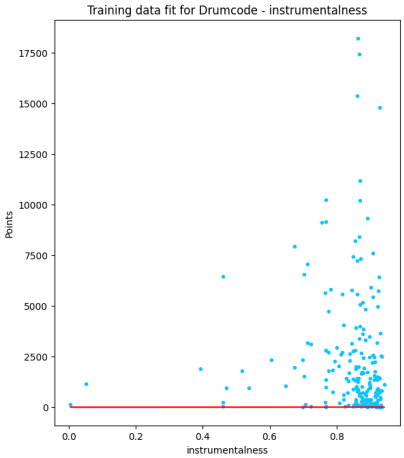

MSE: 15590170.15
RMSE: 3948.44
MAE: 2395.43
R-squared: -312.32



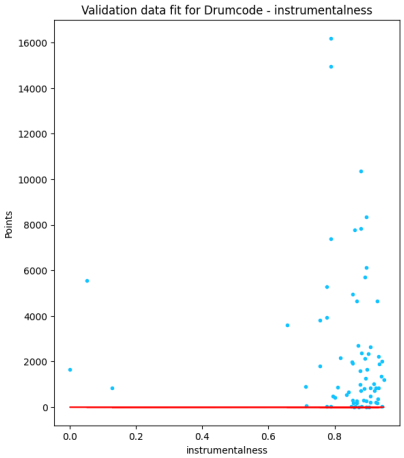

MSE: 16115644.59
RMSE: 4014.43
MAE: 2374.92
R-squared: -109.77



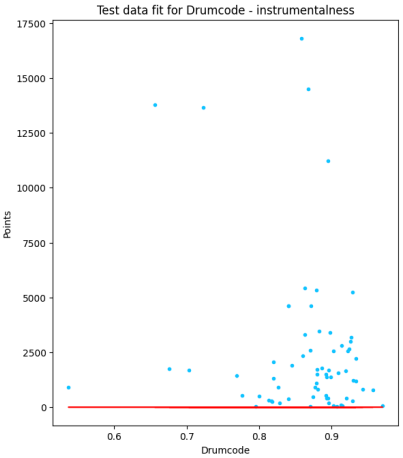

MSE: 19042039.23
RMSE: 4363.72
MAE: 2536.88
R-squared: -101.72



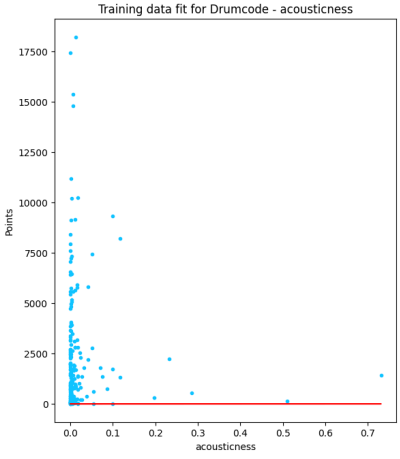

MSE: 15583484.70
RMSE: 3947.59
MAE: 2394.03
R-squared: -312.19



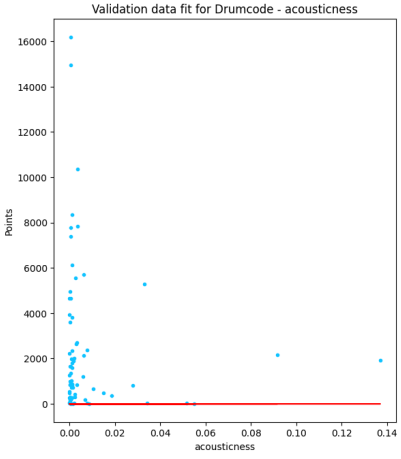

MSE: 16108974.42
RMSE: 4013.60
MAE: 2373.54
R-squared: -109.72



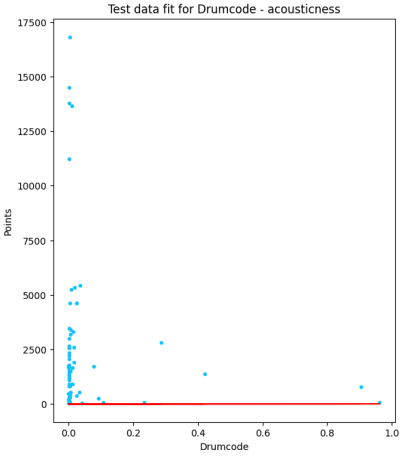

MSE: 19034626.91
RMSE: 4362.87
MAE: 2535.42
R-squared: -101.68



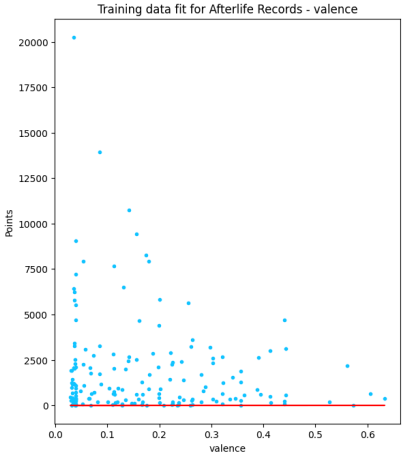

MSE: 15591322.15
RMSE: 3948.58
MAE: 2395.67
R-squared: -312.34



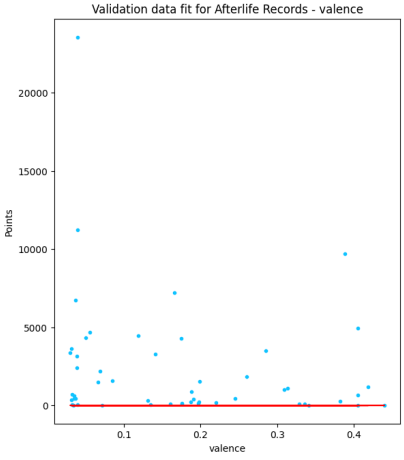

MSE: 16116783.61
RMSE: 4014.57
MAE: 2375.16
R-squared: -109.77



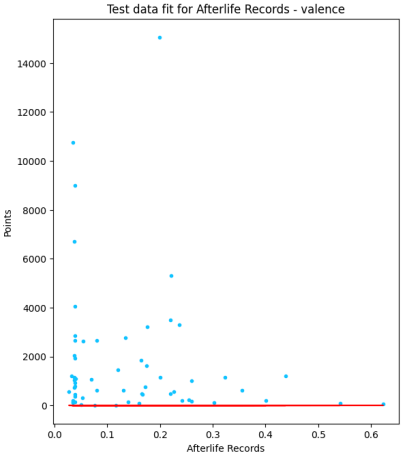

MSE: 19042877.79
RMSE: 4363.81
MAE: 2537.05
R-squared: -101.72



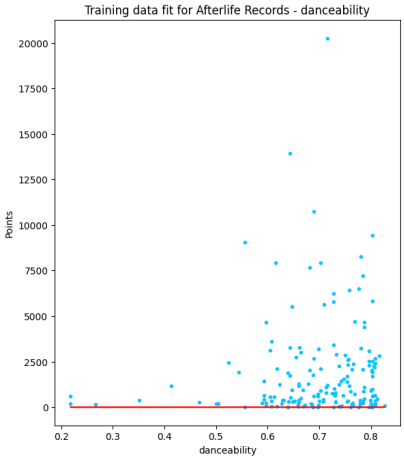

MSE: 10754660.96
RMSE: 3279.43
MAE: 1845.98
R-squared: -249.31



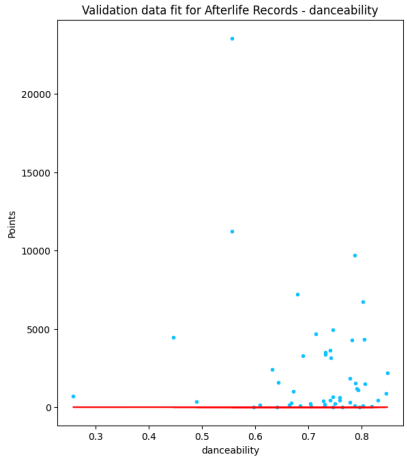

MSE: 20388109.93
RMSE: 4515.32
MAE: 2291.64
R-squared: -69.04



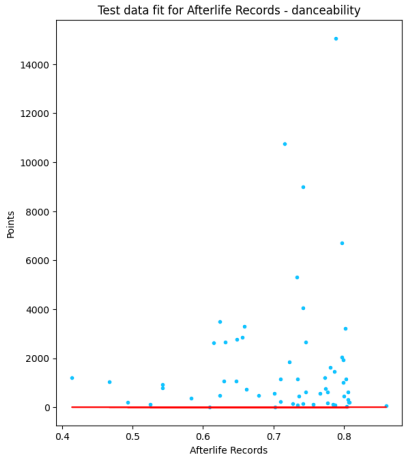

MSE: 10443317.92
RMSE: 3231.61
MAE: 1761.59
R-squared: -82.94



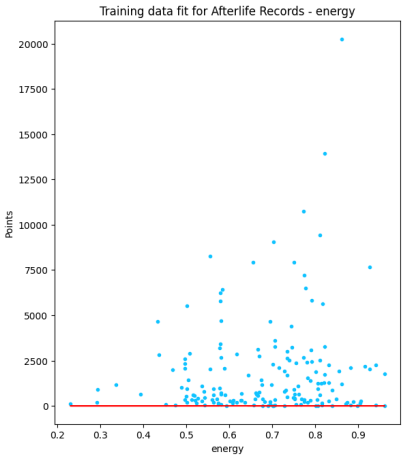

MSE: 10750752.46
RMSE: 3278.83
MAE: 1844.92
R-squared: -249.22



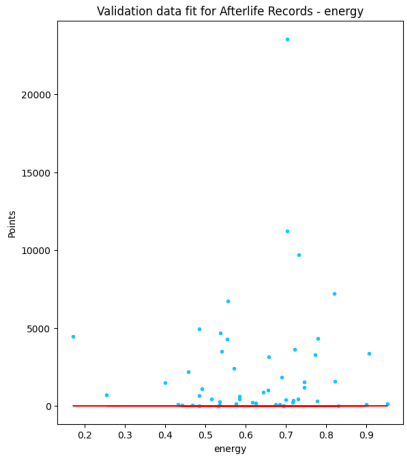

MSE: 20383136.33
RMSE: 4514.77
MAE: 2290.60
R-squared: -69.02



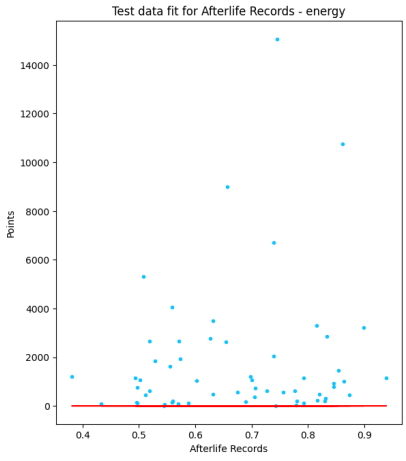

MSE: 10439341.30
RMSE: 3231.00
MAE: 1760.47
R-squared: -82.91



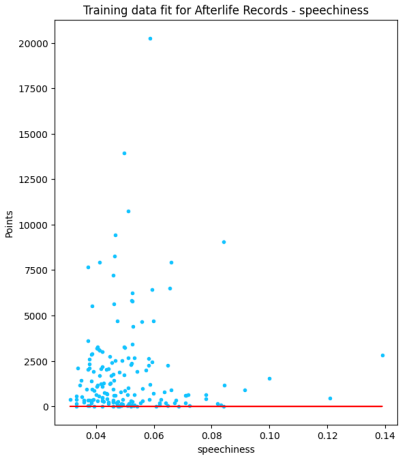

MSE: 10750880.67
RMSE: 3278.85
MAE: 1844.96
R-squared: -249.22



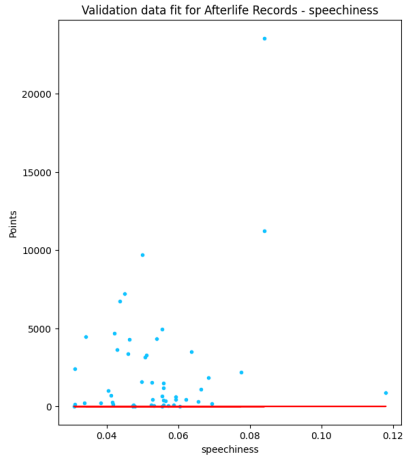

MSE: 20383941.27
RMSE: 4514.86
MAE: 2290.76
R-squared: -69.03



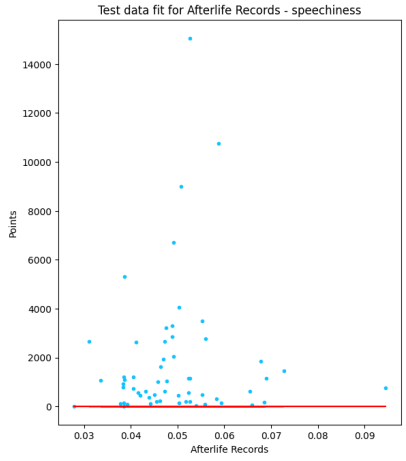

MSE: 10439597.59
RMSE: 3231.04
MAE: 1760.54
R-squared: -82.91



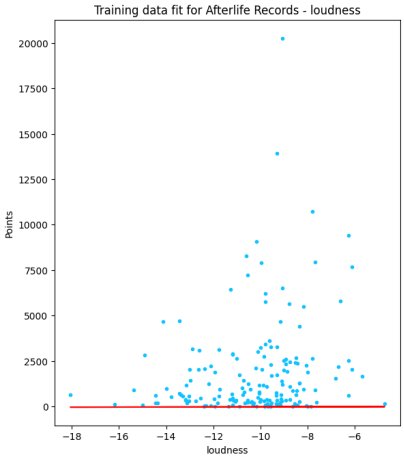

MSE: 10755553.91
RMSE: 3279.57
MAE: 1846.22
R-squared: -249.33



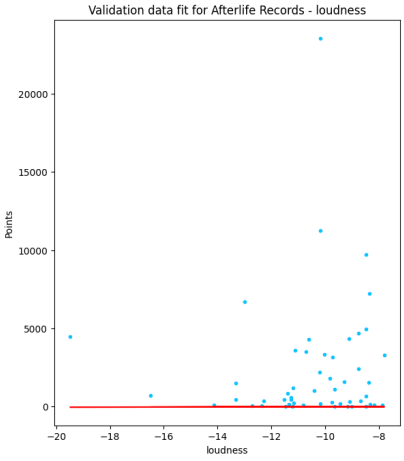

MSE: 20389197.05
RMSE: 4515.44
MAE: 2291.87
R-squared: -69.05



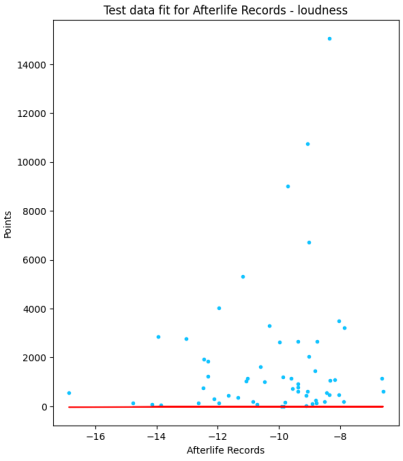

MSE: 10444000.73
RMSE: 3231.72
MAE: 1761.78
R-squared: -82.95



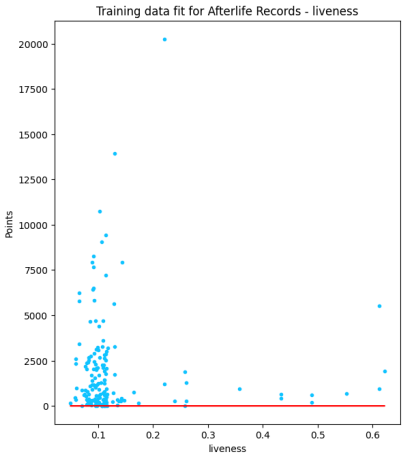

MSE: 10832033.18
RMSE: 3291.21
MAE: 1866.81
R-squared: -251.11



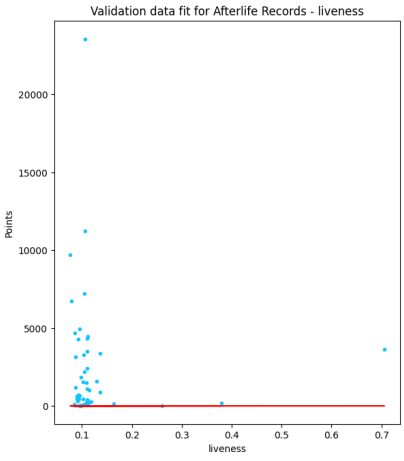

MSE: 20486348.79
RMSE: 4526.18
MAE: 2312.97
R-squared: -69.38



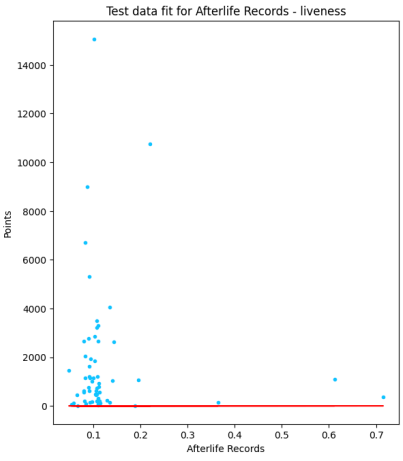

MSE: 10516861.96
RMSE: 3242.97
MAE: 1782.34
R-squared: -83.53



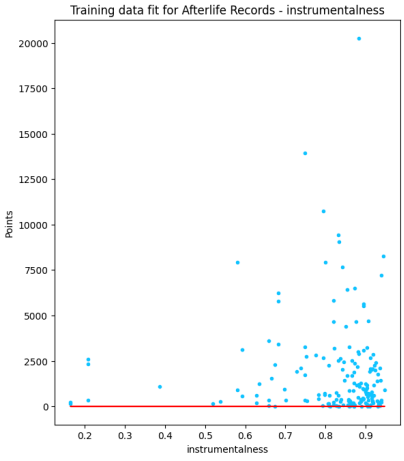

MSE: 10754980.71
RMSE: 3279.48
MAE: 1846.06
R-squared: -249.32



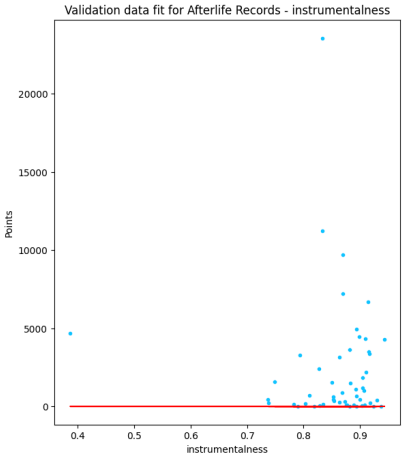

MSE: 20388553.62
RMSE: 4515.37
MAE: 2291.73
R-squared: -69.04



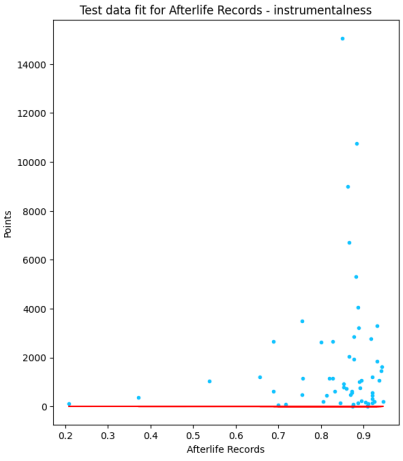

MSE: 10443448.95
RMSE: 3231.63
MAE: 1761.63
R-squared: -82.94



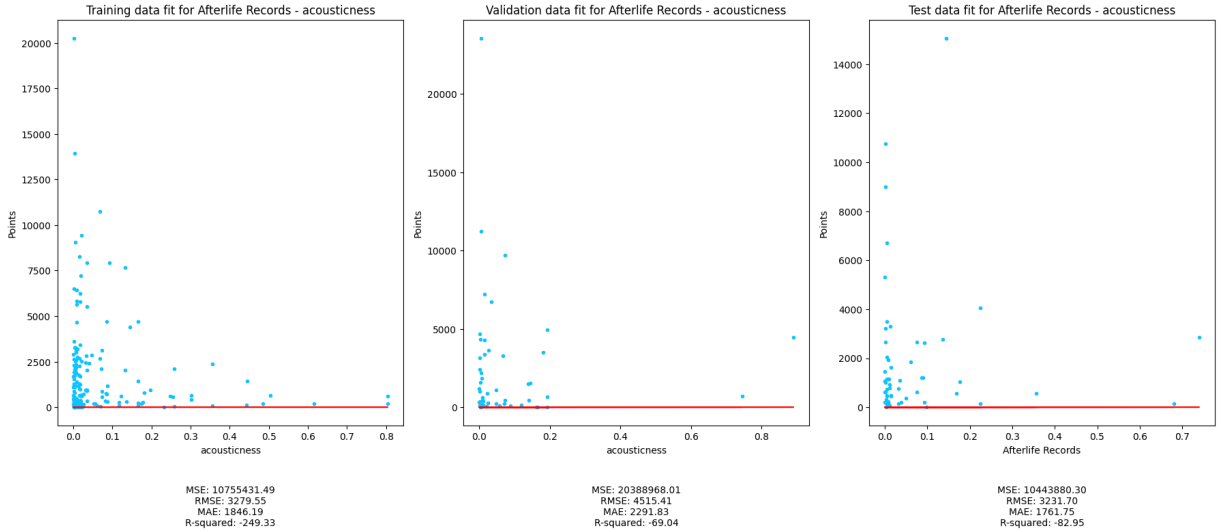

MSE: 10749884.38
RMSE: 3278.70
MAE: 1844.69
R-squared: -249.20




MSE: 20381831.13
RMSE: 4514.62
MAE: 2290.35
R-squared: -69.02




MSE: 10438468.44
RMSE: 3230.86
MAE: 1760.24
R-squared: -82.90




# Running mean normalized data

## genre function

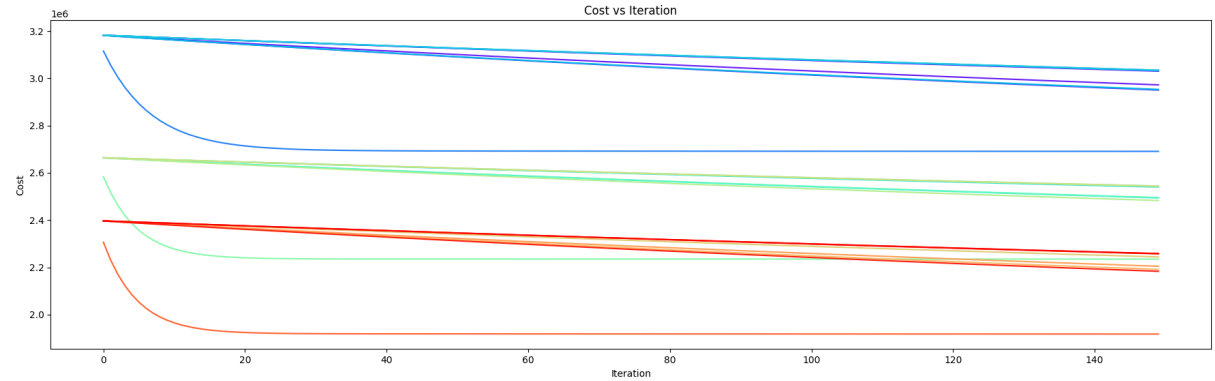
In [30]:
```
run_all_calculations_genre(mean_train, mean_validate, mean_test)
```

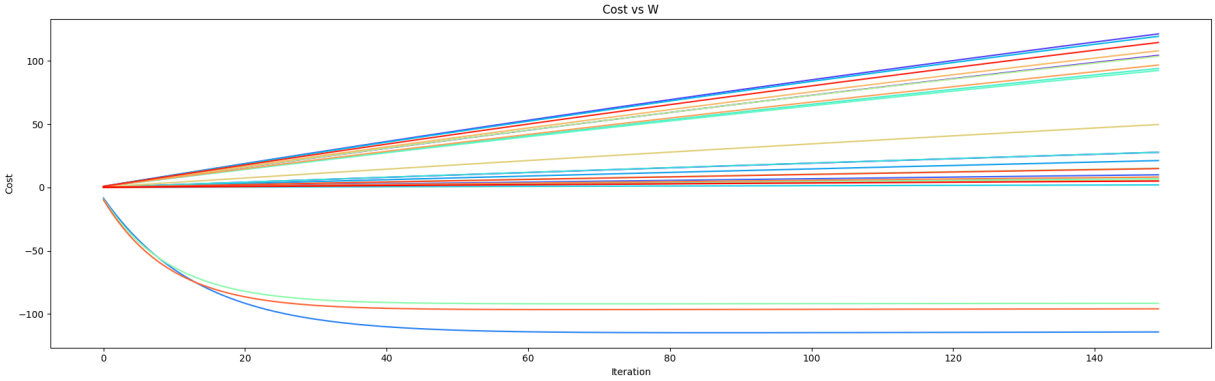
```
Cost for Techno (Peak Time/Driving) and valence at initial w: 3181641.603628
1306
Gradient at initial w & b for Techno (Peak Time/Driving) and valence: [-200.
31391919] -1070.8836893862326
Optimal w & b found by gradient descent for valence for Techno (Peak Time/Dr
iving): [27.77834419] 148.85482410820424
Cost for Techno (Peak Time/Driving) and danceability at initial w: 3180490.6
55909219
Gradient at initial w & b for Techno (Peak Time/Driving) and danceability:
[-776.76999525] -1070.8836893862326
Optimal w & b found by gradient descent for danceability for Techno (Peak Ti
me/Driving): [104.39389859] 143.74504739731418
Cost for Techno (Peak Time/Driving) and energy at initial w: 3180212.4059041
226
Gradient at initial w & b for Techno (Peak Time/Driving) and energy: [-916.2
2602102] -1070.8836893862326
Optimal w & b found by gradient descent for energy for Techno (Peak Time/Dri
ving): [121.36379882] 141.6865913823149
Cost for Techno (Peak Time/Driving) and speechiness at initial w: 3181898.04
33146656
Gradient at initial w & b for Techno (Peak Time/Driving) and speechiness: [-
71.91720295] -1070.8836893862326
Optimal w & b found by gradient descent for speechiness for Techno (Peak Tim
e/Driving): [10.02051653] 149.1859825544398
Cost for Techno (Peak Time/Driving) and loudness at initial w: 3198827.31114
7933
Gradient at initial w & b for Techno (Peak Time/Driving) and loudness: [833
0.04817396] -1070.8836893862326
Optimal w & b found by gradient descent for loudness for Techno (Peak Time/D
riving): [-114.21775728] 29.413739722825113
Cost for Techno (Peak Time/Driving) and liveness at initial w: 3181735.98147
99856
Gradient at initial w & b for Techno (Peak Time/Driving) and liveness: [-15
3.05809769] -1070.8836893862326
Optimal w & b found by gradient descent for liveness for Techno (Peak Time/D
riving): [21.24171494] 149.00980216998857
Cost for Techno (Peak Time/Driving) and instrumentalness at initial w: 31802
37.5827043313
Gradient at initial w & b for Techno (Peak Time/Driving) and instrumentalnes
s: [-903.6507334] -1070.8836893862326
Optimal w & b found by gradient descent for instrumentalness for Techno (Pea
k Time/Driving): [119.44800922] 141.76455995661246
Cost for Techno (Peak Time/Driving) and acousticness at initial w: 3182011.5
26286266
Gradient at initial w & b for Techno (Peak Time/Driving) and acousticness:
[-15.13024177] -1070.8836893862326
Optimal w & b found by gradient descent for acousticness for Techno (Peak Ti
me/Driving): [2.01659314] 149.23143579294168
Cost for Melodic House & Techno and valence at initial w: 2663241.911532439
Gradient at initial w & b for Melodic House & Techno and valence: [-202.3799
8242] -963.6845310596833
Optimal w & b found by gradient descent for valence for Melodic House & Tech
no: [27.81801219] 133.82899011742444
Cost for Melodic House & Techno and danceability at initial w: 2662244.68119
24293
Gradient at initial w & b for Melodic House & Techno and danceability: [-70
```

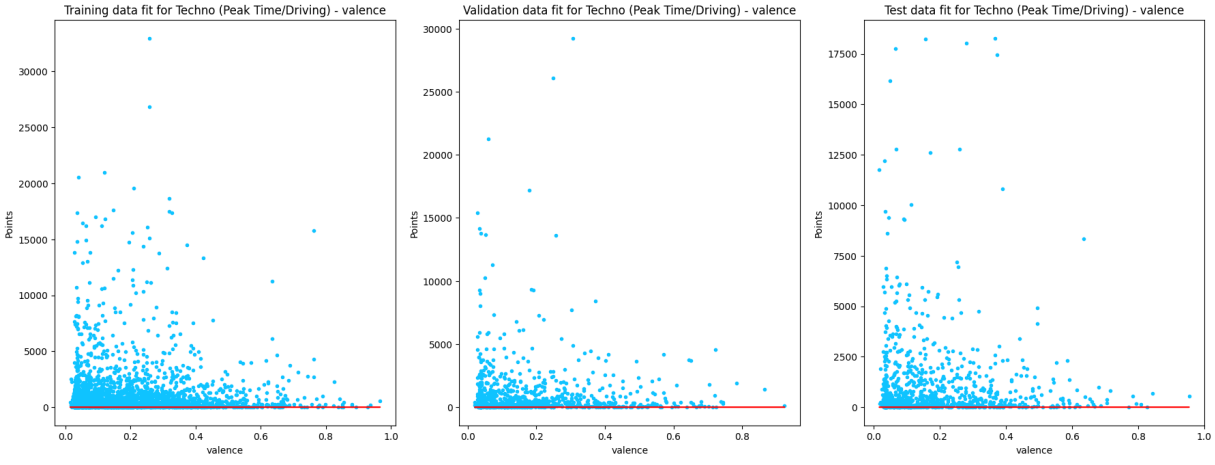
```
1.94695615] -963.6845310596833
Optimal w & b found by gradient descent for danceability for Melodic House &
Techno: [94.06524636] 129.23929774545502
Cost for Melodic House & Techno and energy at initial w: 2662269.9156939127
Gradient at initial w & b for Melodic House & Techno and energy: [-689.30577
467] -963.6845310596833
Optimal w & b found by gradient descent for energy for Melodic House & Techn
o: [92.4450701] 129.43086739999433
Cost for Melodic House & Techno and speechiness at initial w: 2663546.585142
4974
Gradient at initial w & b for Melodic House & Techno and speechiness: [-49.7
8174385] -963.6845310596833
Optimal w & b found by gradient descent for speechiness for Melodic House &
Techno: [6.92531333] 134.26928928499478
Cost for Melodic House & Techno and loudness at initial w: 2682218.324127331
4
Gradient at initial w & b for Melodic House & Techno and loudness: [9197.239
72838] -963.6845310596833
Optimal w & b found by gradient descent for loudness for Melodic House & Tec
hno: [-91.58307919] 17.479927844494558
Cost for Melodic House & Techno and liveness at initial w: 2663426.786272200
3
Gradient at initial w & b for Melodic House & Techno and liveness: [-109.767
49082] -963.6845310596833
Optimal w & b found by gradient descent for liveness for Melodic House & Tec
hno: [15.21886894] 134.16404735740045
Cost for Melodic House & Techno and instrumentalness at initial w: 2662083.2
642848073
Gradient at initial w & b for Melodic House & Techno and instrumentalness:
[-782.89444404] -963.6845310596833
Optimal w & b found by gradient descent for instrumentalness for Melodic Hou
se & Techno: [103.74380093] 128.03863608206612
Cost for Melodic House & Techno and acousticness at initial w: 2663565.06564
1017
Gradient at initial w & b for Melodic House & Techno and acousticness: [-40.
52964785] -963.6845310596833
Optimal w & b found by gradient descent for acousticness for Melodic House &
Techno: [5.71736599] 134.28136952681047
Cost for Organic House / Downtempo and valence at initial w: 2395988.4166122
35
Gradient at initial w & b for Organic House / Downtempo and valence: [-361.3
0452563] -1036.7685688405797
Optimal w & b found by gradient descent for valence for Organic House / Down
tempo: [49.74742603] 143.2150433096094
Cost for Organic House / Downtempo and danceability at initial w: 2395096.24
34287304
Gradient at initial w & b for Organic House / Downtempo and danceability: [-
808.24123958] -1036.7685688405797
Optimal w & b found by gradient descent for danceability for Organic House /
Downtempo: [108.02831646] 138.37026149729354
Cost for Organic House / Downtempo and energy at initial w: 2395276.56686236
1
Gradient at initial w & b for Organic House / Downtempo and energy: [-717.87
925498] -1036.7685688405797
Optimal w & b found by gradient descent for energy for Organic House / Downt
empo: [96.74498873] 139.65023197453357
```

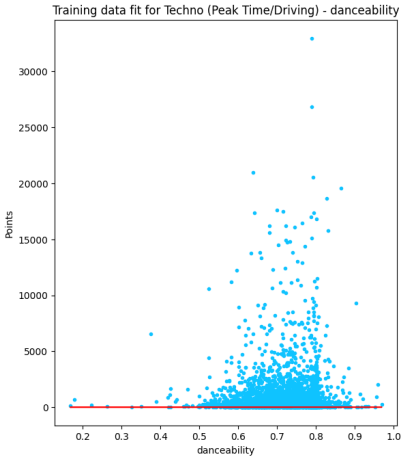
```
Cost for Organic House / Downtempo and speechiness at initial w: 2396591.917
902075
Gradient at initial w & b for Organic House / Downtempo and speechiness: [-5
9.10461182] -1036.7685688405797
Optimal w & b found by gradient descent for speechiness for Organic House /
Downtempo: [8.2184004] 144.4456385835178
Cost for Organic House / Downtempo and loudness at initial w: 2416409.898455
8936
Gradient at initial w & b for Organic House / Downtempo and loudness: [9760.
07068614] -1036.7685688405797
Optimal w & b found by gradient descent for loudness for Organic House / Dow
ntempo: [-95.93721183] 21.58277093769111
Cost for Organic House / Downtempo and liveness at initial w: 2396494.345923
426
Gradient at initial w & b for Organic House / Downtempo and liveness: [-107.
96875005] -1036.7685688405797
Optimal w & b found by gradient descent for liveness for Organic House / Dow
ntempo: [14.89663249] 144.3555386360928
Cost for Organic House / Downtempo and instrumentalness at initial w: 239498
5.109098314
Gradient at initial w & b for Organic House / Downtempo and instrumentalnes
s: [-863.96845496] -1036.7685688405797
Optimal w & b found by gradient descent for instrumentalness for Organic Hou
se / Downtempo: [114.58025207] 137.50072946107528
Cost for Organic House / Downtempo and acousticness at initial w: 2396634.92
3699211
Gradient at initial w & b for Organic House / Downtempo and acousticness: [-
37.60738775] -1036.7685688405797
Optimal w & b found by gradient descent for acousticness for Organic House /
Downtempo: [5.06157402] 144.46122391585396
```


Techno (Peak Time/Driving)-valence
Techno (Peak Time/Driving)-danceability
Techno (Peak Time/Driving)-energy
Techno (Peak Time/Driving)-speechiness
Techno (Peak Time/Driving)-loudness
Techno (Peak Time/Driving)-liveness
Techno (Peak Time/Driving)-instrumentalness
Techno (Peak Time/Driving)-acousticness
Melodic House & Techno-valence
Melodic House & Techno-danceability
Melodic House & Techno-energy
Melodic House & Techno-speechiness
Melodic House & Techno-loudness
Melodic House & Techno-liveness
Melodic House & Techno-instrumentalness
Melodic House & Techno-acousticness
Organic House / Downtempo-valence
Organic House / Downtempo-danceability
Organic House / Downtempo-energy
Organic House / Downtempo-speechiness
Organic House / Downtempo-loudness
Organic House / Downtempo-liveness
Organic House / Downtempo-instrumentalness
Organic House / Downtempo-acousticness

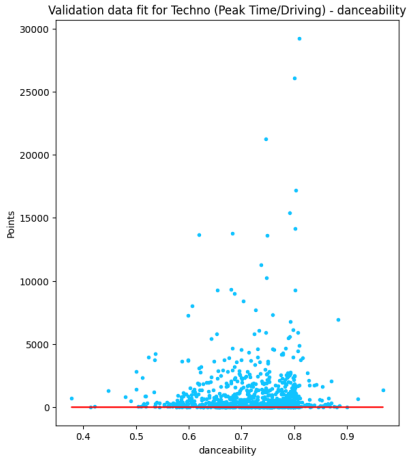

Cost vs Iteration
1e6
Cost
Iteration


Cost vs W
Cost
Iteration

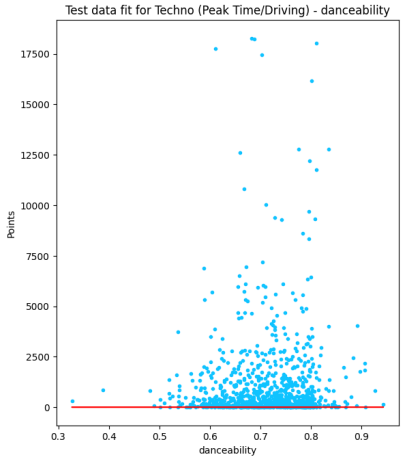

Training data fit for Techno (Peak Time/Driving) - valence
Points
valence
MSE: 6363279.94
RMSE: 2522.55
MAE: 1069.52
R-squared: -3594.28
Validation data fit for Techno (Peak Time/Driving) - valence
Points
valence
MSE: 6293393.87
RMSE: 2508.66
MAE: 1036.25
R-squared: -1109.47
Test data fit for Techno (Peak Time/Driving) - valence
Points
valence
MSE: 5585545.81
RMSE: 2363.38
MAE: 1049.93
R-squared: -1228.68



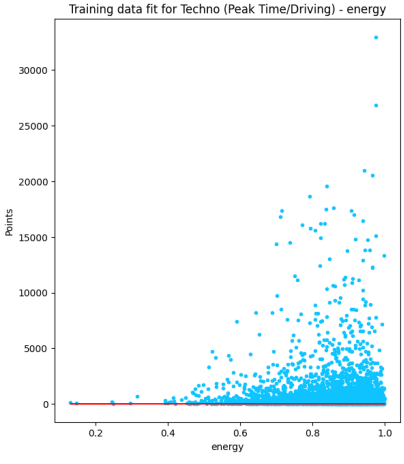

MSE: 6361026.60
RMSE: 2522.11
MAE: 1068.49
R-squared: -3593.01



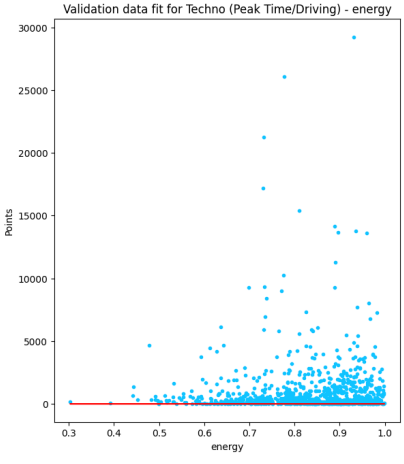

MSE: 6291198.05
RMSE: 2508.23
MAE: 1035.22
R-squared: -1109.08



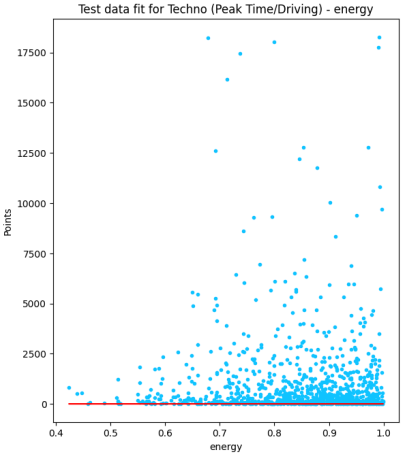

MSE: 5583338.63
RMSE: 2362.91
MAE: 1048.91
R-squared: -1228.20



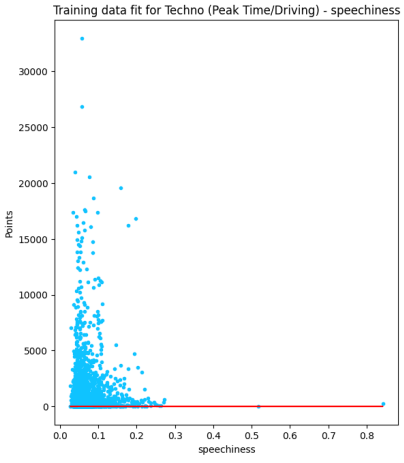

MSE: 6360485.74
RMSE: 2522.00
MAE: 1068.25
R-squared: -3592.70



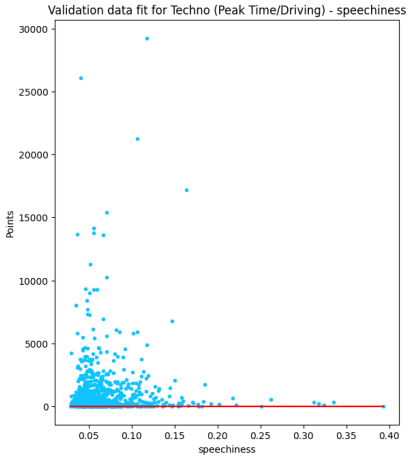

MSE: 6290723.71
RMSE: 2508.13
MAE: 1035.00
R-squared: -1109.00



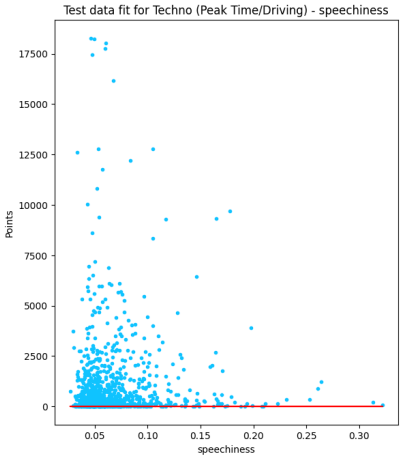

MSE: 5582798.56
RMSE: 2362.79
MAE: 1048.66
R-squared: -1228.08



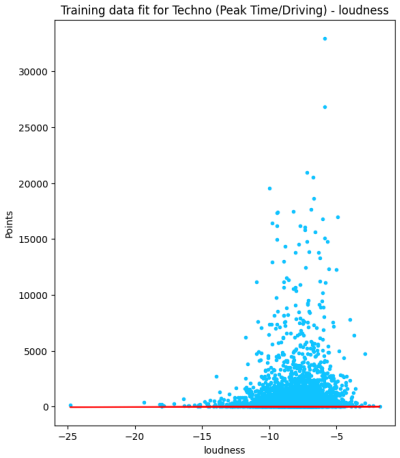

MSE: 6363797.13
RMSE: 2522.66
MAE: 1069.75
R-squared: -3594.57



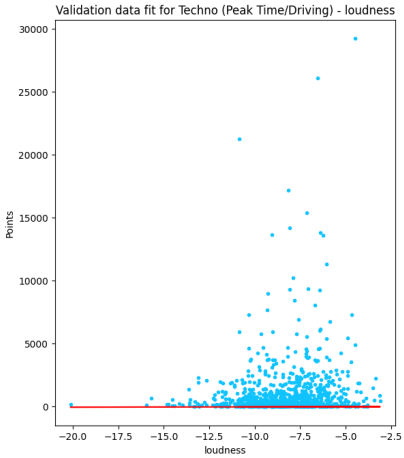

MSE: 6293886.81
RMSE: 2508.76
MAE: 1036.48
R-squared: -1109.56



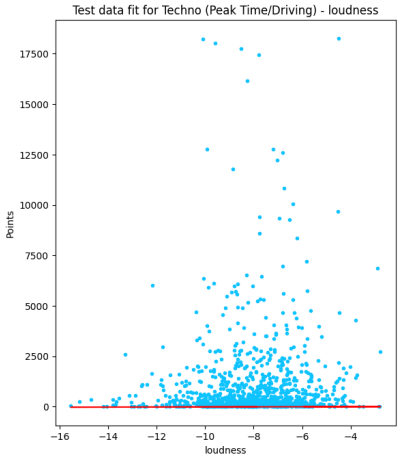

MSE: 5586048.96
RMSE: 2363.48
MAE: 1050.16
R-squared: -1228.79



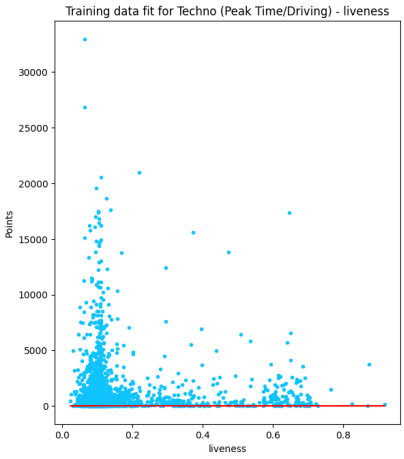

MSE: 6399407.09
RMSE: 2529.70
MAE: 1086.26
R-squared: -3614.69



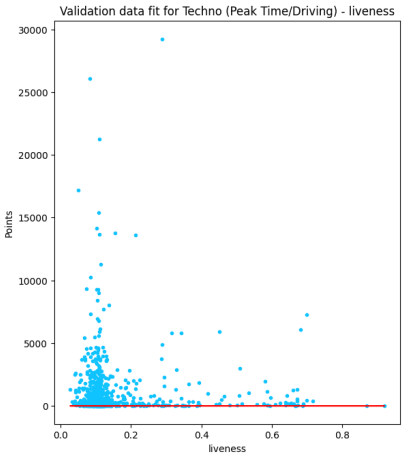

MSE: 6328719.15
RMSE: 2515.69
MAE: 1053.14
R-squared: -1115.71



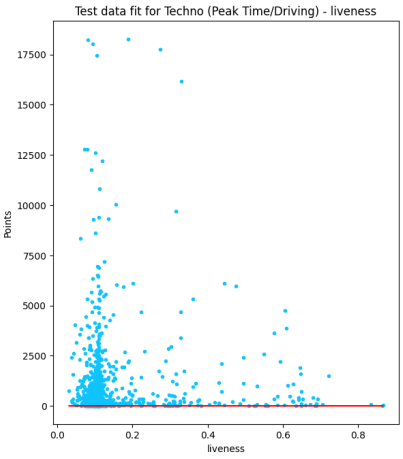

MSE: 5620682.11
RMSE: 2370.80
MAE: 1066.51
R-squared: -1236.42



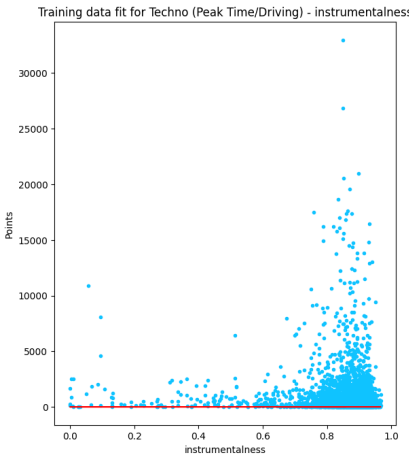

MSE: 6363461.13
RMSE: 2522.59
MAE: 1069.60
R-squared: -3594.38



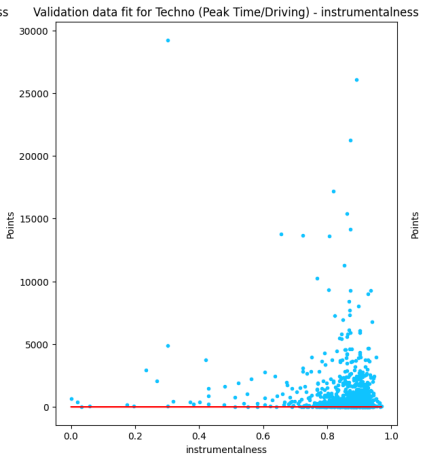

MSE: 6293579.99
RMSE: 2508.70
MAE: 1036.34
R-squared: -1109.50



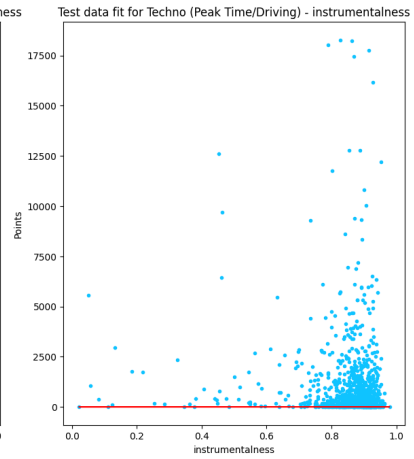

MSE: 5585709.74
RMSE: 2363.41
MAE: 1050.00
R-squared: -1228.72



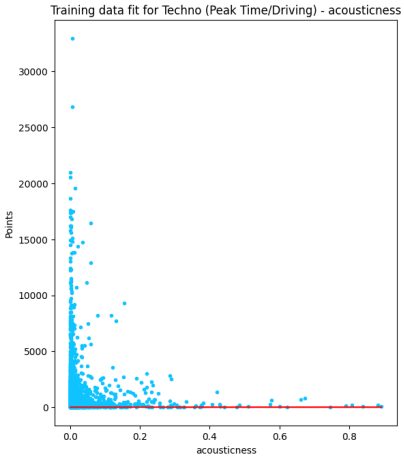

MSE: 6360468.39
RMSE: 2522.00
MAE: 1068.24
R-squared: -3592.69



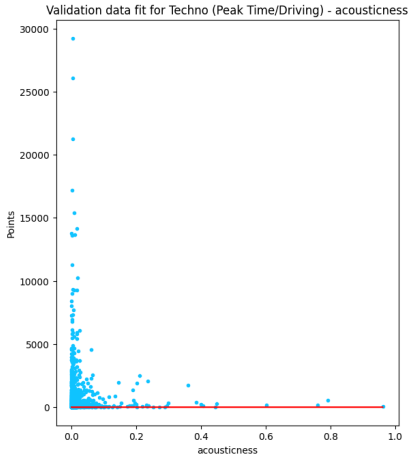

MSE: 6290649.07
RMSE: 2508.12
MAE: 1034.97
R-squared: -1108.99



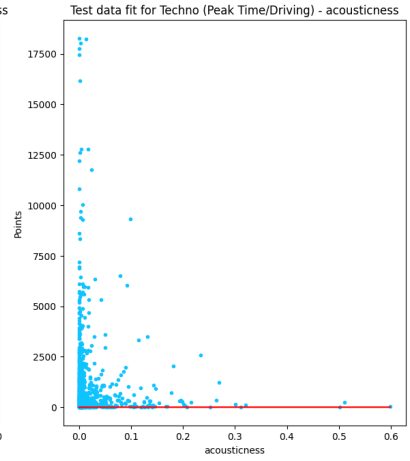

MSE: 5582768.44
RMSE: 2362.79
MAE: 1048.64
R-squared: -1228.07



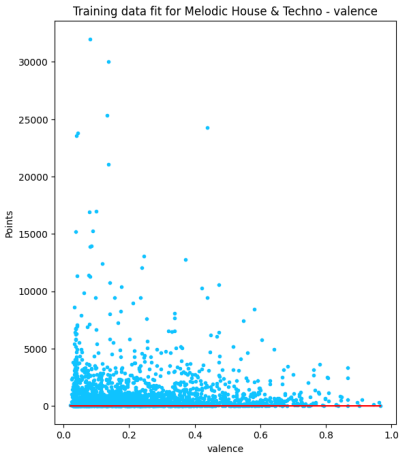

MSE: 6363988.82
RMSE: 2522.69
MAE: 1069.84
R-squared: -3594.68



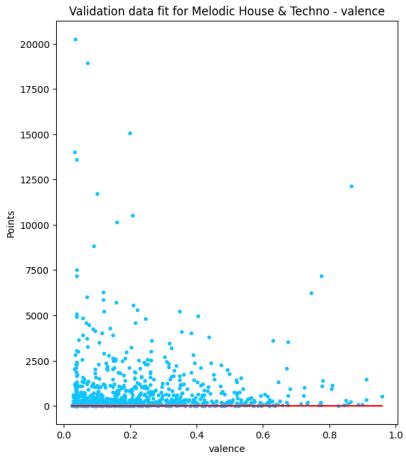

MSE: 6294076.39
RMSE: 2508.80
MAE: 1036.57
R-squared: -1109.59



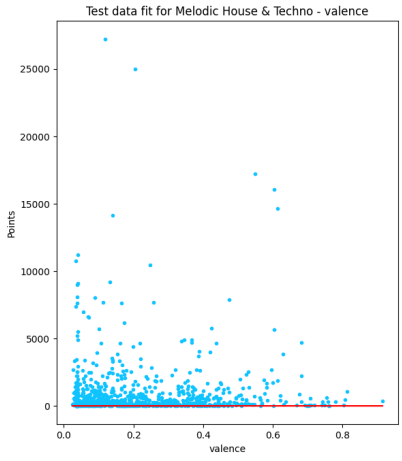

MSE: 5586253.73
RMSE: 2363.53
MAE: 1050.26
R-squared: -1228.84



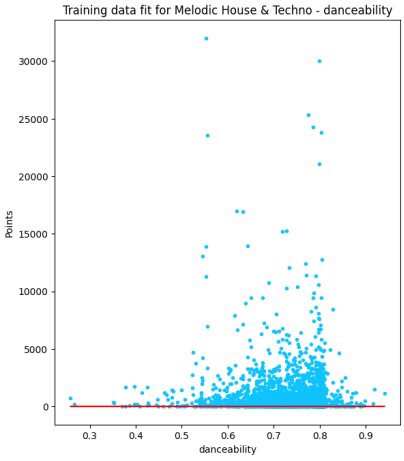

MSE: 5326407.62
RMSE: 2307.90
MAE: 962.24
R-squared: -2980.22



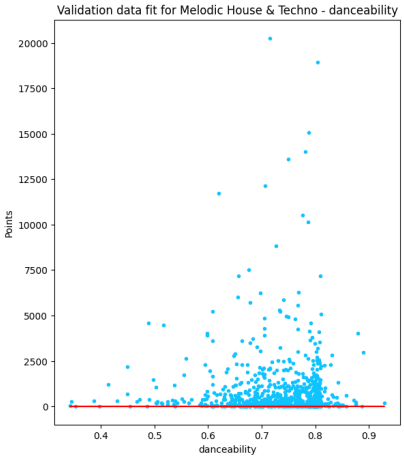

MSE: 4147632.90
RMSE: 2036.57
MAE: 883.06
R-squared: -1013.79



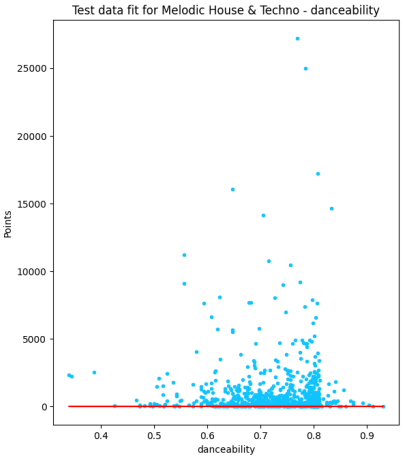

MSE: 5978040.23
RMSE: 2445.00
MAE: 1007.25
R-squared: -948.07



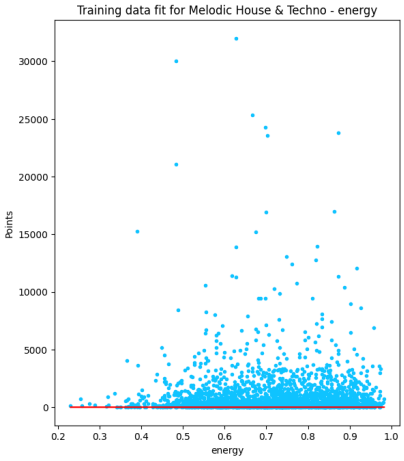

MSE: 5324483.32
RMSE: 2307.48
MAE: 961.26
R-squared: -2979.14



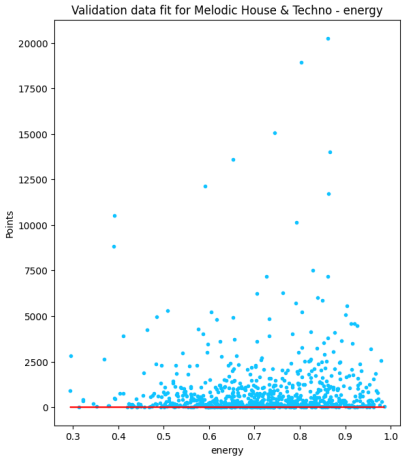

MSE: 4145890.24
RMSE: 2036.15
MAE: 882.10
R-squared: -1013.36



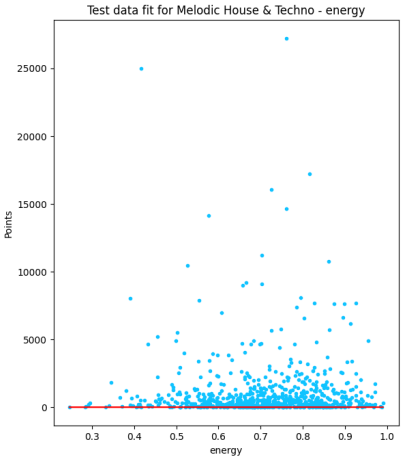

MSE: 5976018.13
RMSE: 2444.59
MAE: 1006.27
R-squared: -947.75



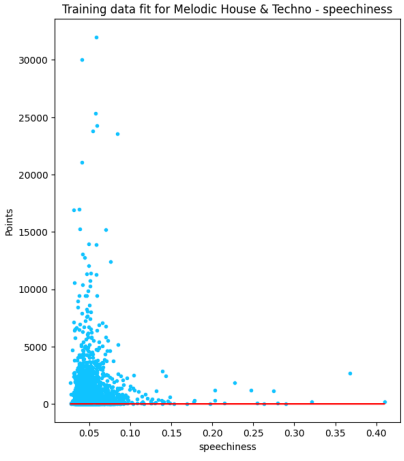

MSE: 5324541.68
RMSE: 2307.50
MAE: 961.29
R-squared: -2979.17



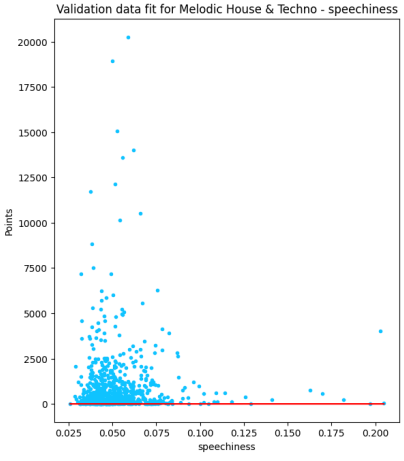

MSE: 4145934.99
RMSE: 2036.16
MAE: 882.13
R-squared: -1013.37



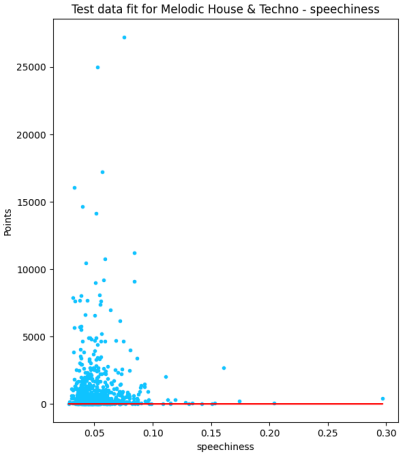

MSE: 5976077.13
RMSE: 2444.60
MAE: 1006.30
R-squared: -947.76



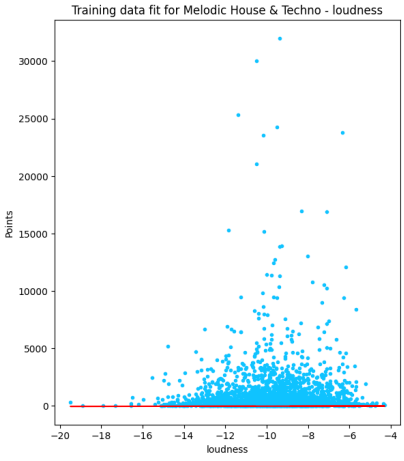

MSE: 5327089.31
RMSE: 2308.05
MAE: 962.58
R-squared: -2980.60



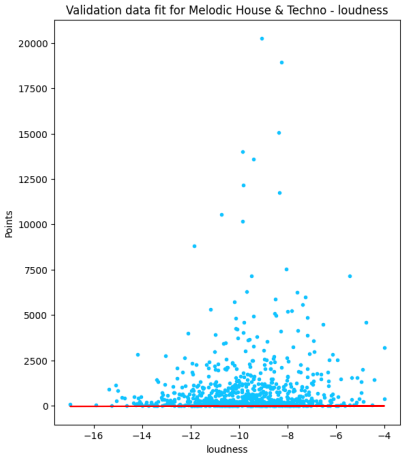

MSE: 4148287.04
RMSE: 2036.73
MAE: 883.43
R-squared: -1013.95



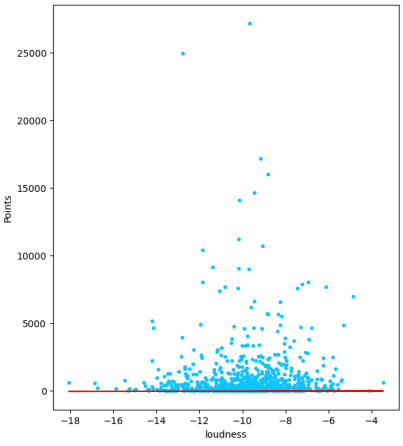

MSE: 5978734.95
RMSE: 2445.15
MAE: 1007.59
R-squared: -948.18



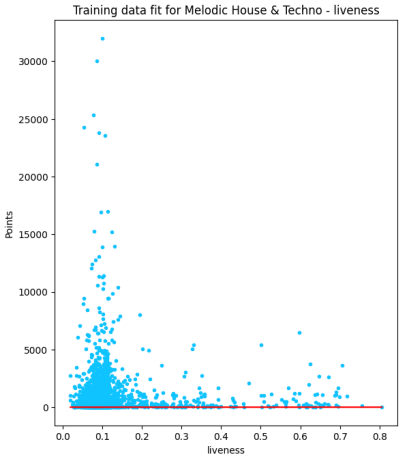

MSE: 5365133.52
RMSE: 2316.28
MAE: 982.13
R-squared: -3001.89



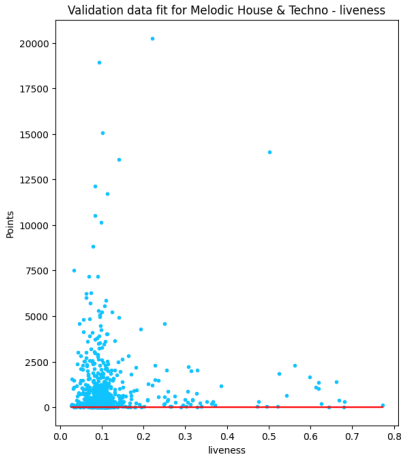

MSE: 4183082.23
RMSE: 2045.26
MAE: 902.90
R-squared: -1022.46



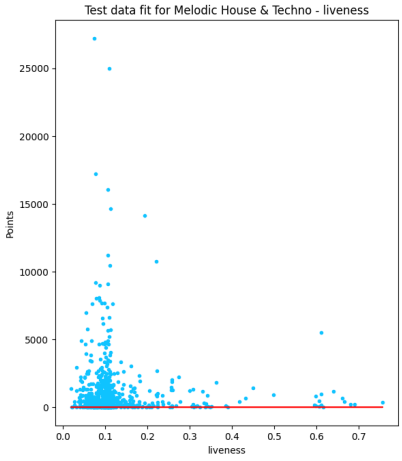

MSE: 6018580.00
RMSE: 2453.28
MAE: 1027.16
R-squared: -954.50



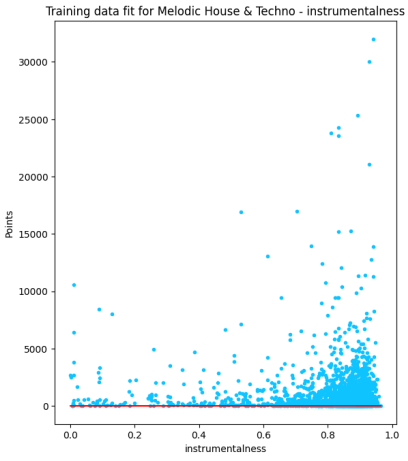

MSE: 5326834.33
RMSE: 2307.99
MAE: 962.45
R-squared: -2980.46



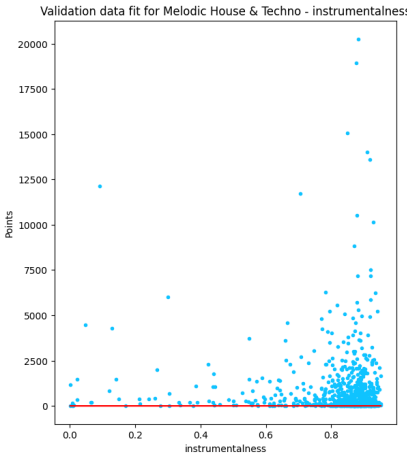

MSE: 4148059.28
RMSE: 2036.68
MAE: 883.30
R-squared: -1013.89



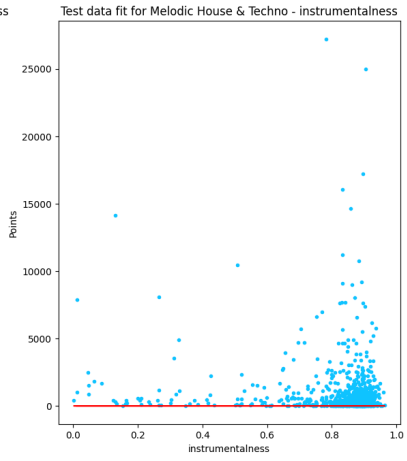

MSE: 5978458.60
RMSE: 2445.09
MAE: 1007.45
R-squared: -948.13



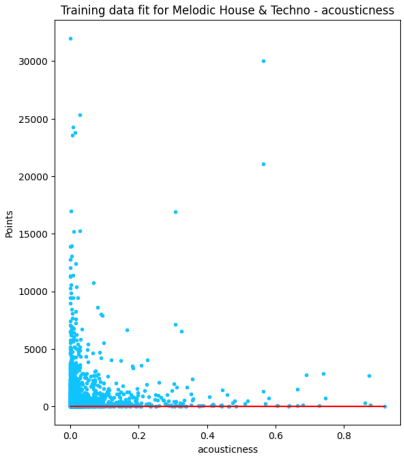

MSE: 5324152.65
RMSE: 2307.41
MAE: 961.10
R-squared: -2978.96



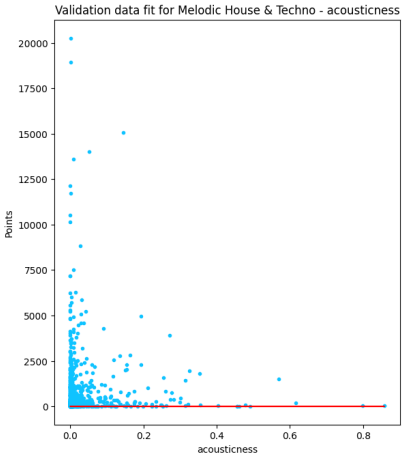

MSE: 4145584.15
RMSE: 2036.07
MAE: 881.93
R-squared: -1013.29



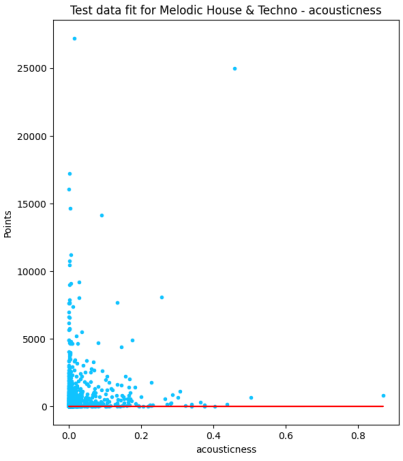

MSE: 5975664.36
RMSE: 2444.52
MAE: 1006.10
R-squared: -947.69



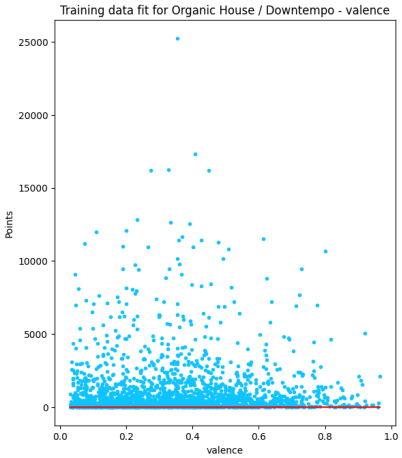

MSE: 5327157.71
RMSE: 2308.06
MAE: 962.62
R-squared: -2980.64



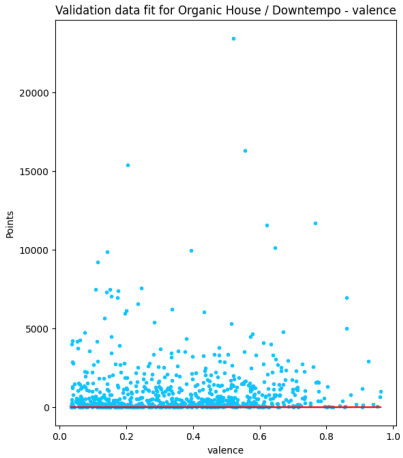

MSE: 4148353.65
RMSE: 2036.75
MAE: 883.47
R-squared: -1013.96



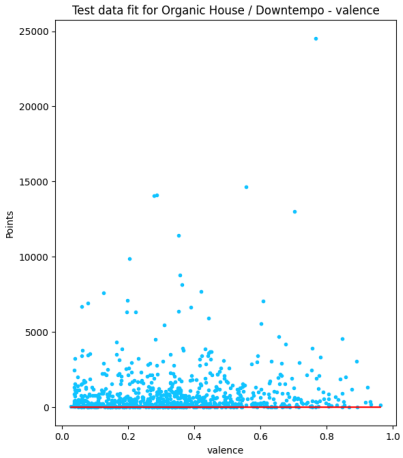

MSE: 5978813.79
RMSE: 2445.16
MAE: 1007.63
R-squared: -948.19



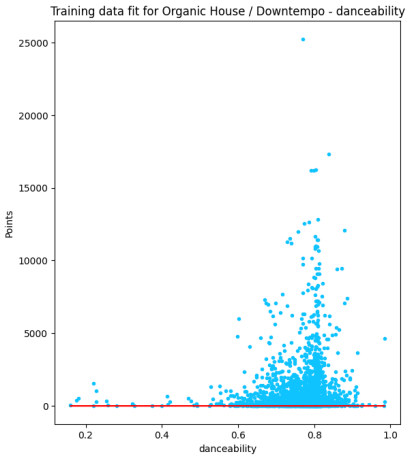

MSE: 4791976.66
RMSE: 2189.06
MAE: 1035.09
R-squared: -2842.81



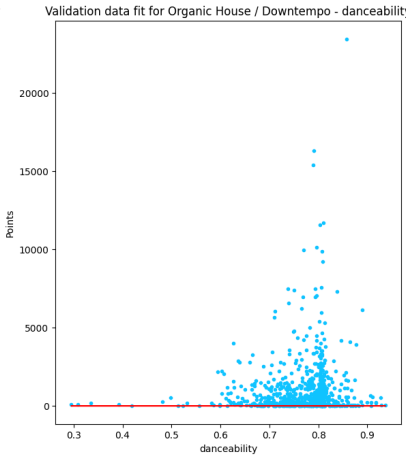

MSE: 4390096.88
RMSE: 2095.26
MAE: 978.38
R-squared: -1028.46



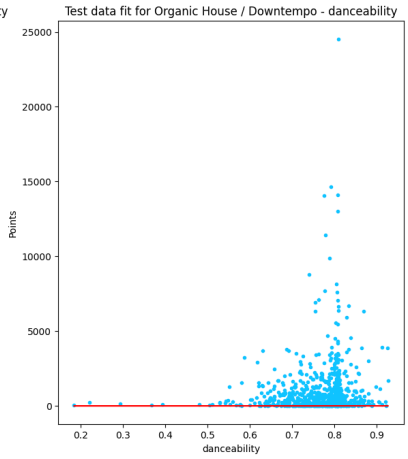

MSE: 4165962.76
RMSE: 2041.07
MAE: 943.88
R-squared: -1021.71



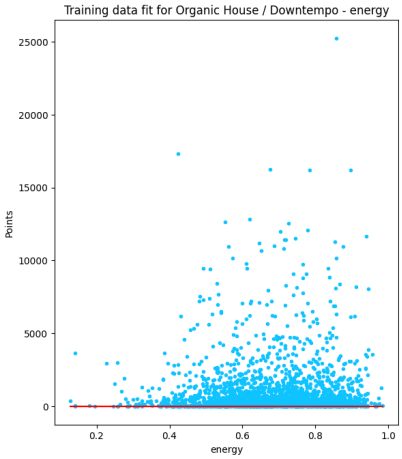

MSE: 4790242.84
RMSE: 2188.66
MAE: 1034.28
R-squared: -2841.78



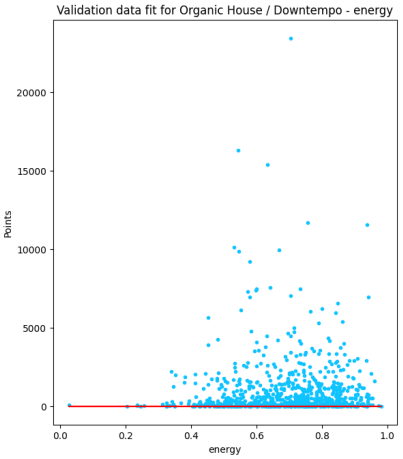

MSE: 4388501.78
RMSE: 2094.88
MAE: 977.59
R-squared: -1028.08



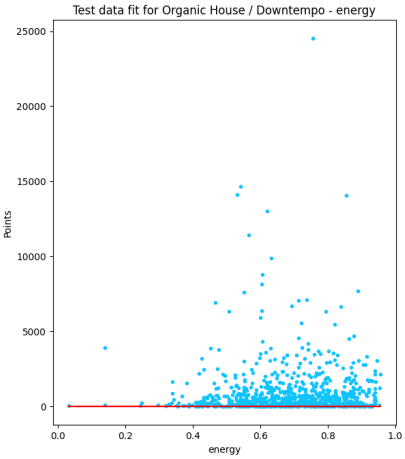

MSE: 4164399.44
RMSE: 2040.69
MAE: 943.07
R-squared: -1021.33



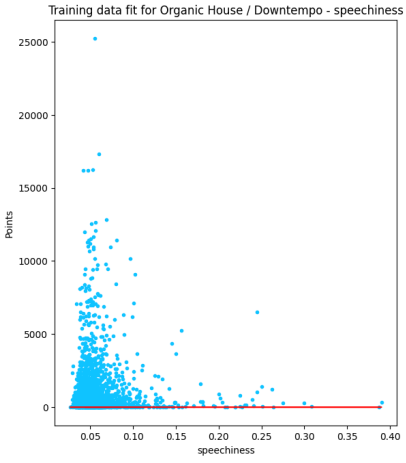

MSE: 4790607.69
RMSE: 2188.75
MAE: 1034.45
R-squared: -2842.00



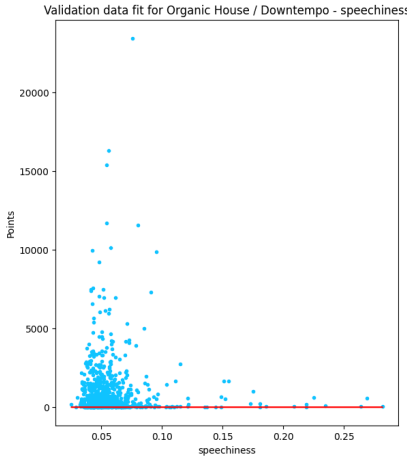

MSE: 4388810.87
RMSE: 2094.95
MAE: 977.74
R-squared: -1028.16



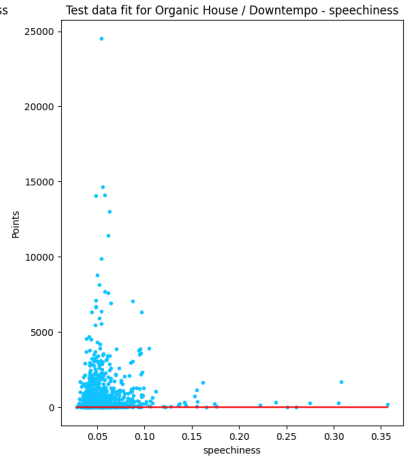

MSE: 4164698.17
RMSE: 2040.76
MAE: 943.22
R-squared: -1021.40



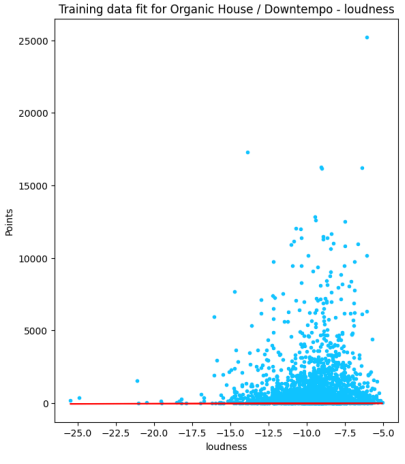

MSE: 4793178.03
RMSE: 2189.33
MAE: 1035.65
R-squared: -2843.52



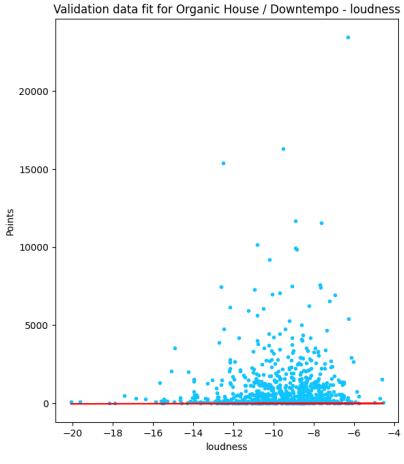

MSE: 4391285.88
RMSE: 2095.54
MAE: 978.97
R-squared: -1028.74



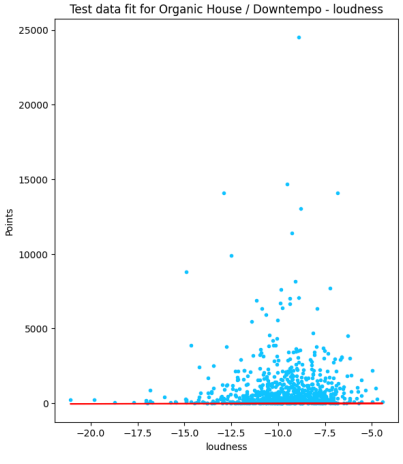

MSE: 4167073.53
RMSE: 2041.34
MAE: 944.47
R-squared: -1021.98



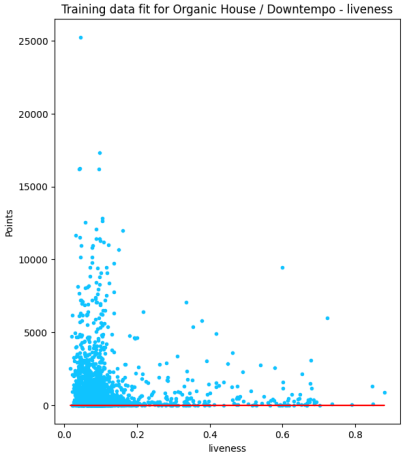

MSE: 4834194.43
RMSE: 2198.68
MAE: 1055.26
R-squared: -2867.86



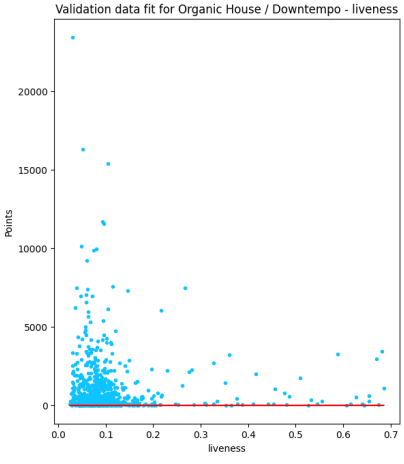

MSE: 4429902.59
RMSE: 2104.73
MAE: 998.49
R-squared: -1037.79



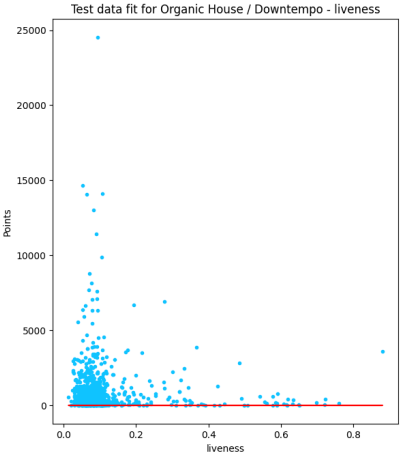

MSE: 4204365.53
RMSE: 2050.45
MAE: 964.00
R-squared: -1031.14



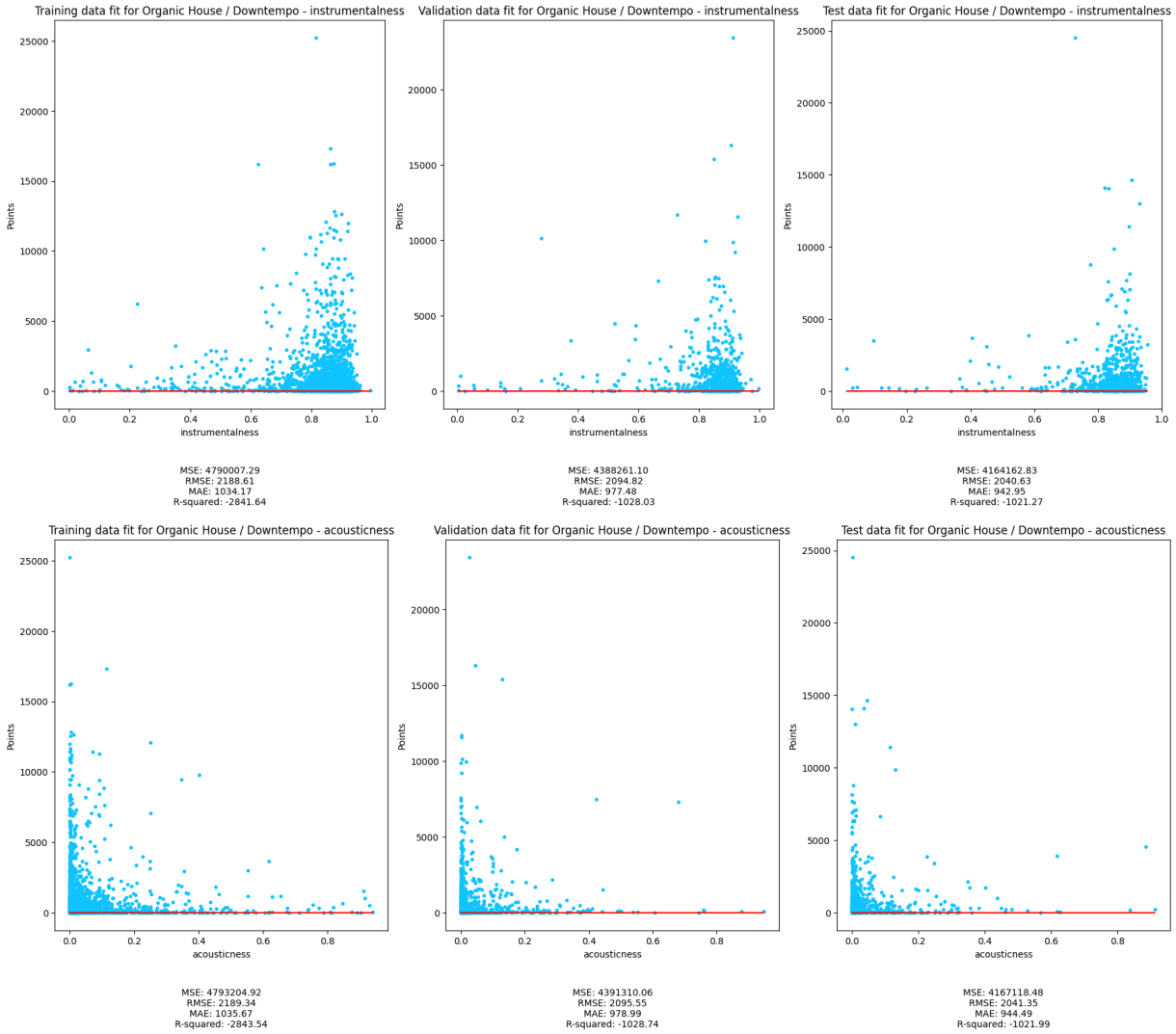

MSE: 4792943.41
RMSE: 2189.28
MAE: 1035.54
R-squared: -2843.38




MSE: 4391092.99
RMSE: 2095.49
MAE: 978.88
R-squared: -1028.69




MSE: 4166860.98
RMSE: 2041.29
MAE: 944.35
R-squared: -1021.93



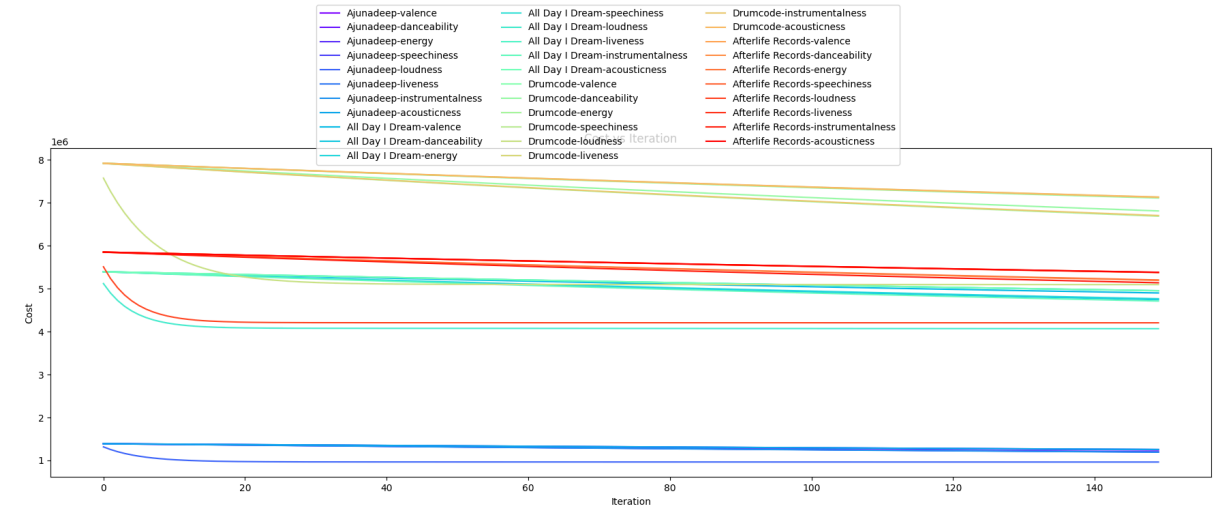

## Label function

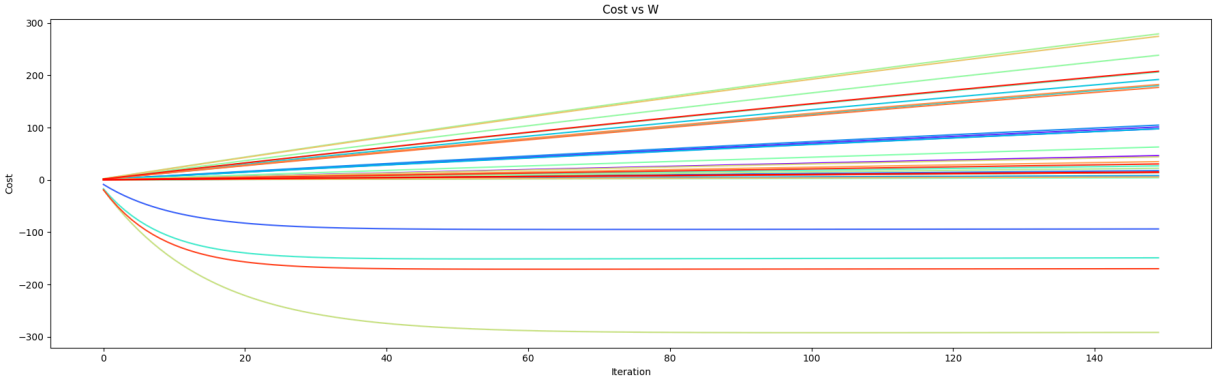
In [31]:
```
run_all_calculations_label(mean_train, mean_validate, mean_test)
```

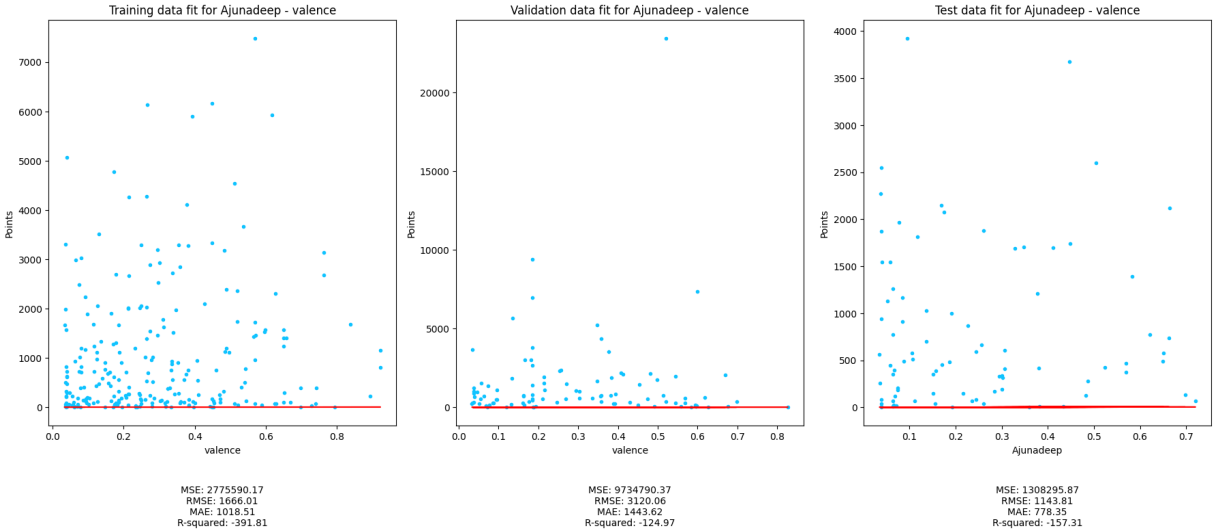
```
Cost for Ajunadeep and valence at initial w: 1387725.7764198936
Gradient at initial w & b for Ajunadeep and valence: [-332.24867439] -1020.0
853658536586
Optimal w & b found by gradient descent for Ajunadeep and valence [46.222680
2] 141.17484068347292
Cost for Ajunadeep and danceability at initial w: 1386946.6113316827
Gradient at initial w & b for Ajunadeep and danceability: [-722.56154065] -1
020.0853658536586
Optimal w & b found by gradient descent for Ajunadeep and danceability [97.6
1635423] 137.3013419002556
Cost for Ajunadeep and energy at initial w: 1386889.5282626583
Gradient at initial w & b for Ajunadeep and energy: [-751.1902561] -1020.085
3658536586
Optimal w & b found by gradient descent for Ajunadeep and energy [101.024897
03] 136.88770465501406
Cost for Ajunadeep and speechiness at initial w: 1388286.362969076
Gradient at initial w & b for Ajunadeep and speechiness: [-51.5882378] -102
0.0853658536586
Optimal w & b found by gradient descent for Ajunadeep and speechiness [7.203
32543] 142.13002816528893
Cost for Ajunadeep and loudness at initial w: 1406067.4265710819
Gradient at initial w & b for Ajunadeep and loudness: [8757.53063821] -1020.
0853658536586
Optimal w & b found by gradient descent for Ajunadeep and loudness [-93.8111
6775] 30.17440171729357
Cost for Ajunadeep and liveness at initial w: 1388133.8330429385
Gradient at initial w & b for Ajunadeep and liveness: [-127.96905813] -1020.
0853658536586
Optimal w & b found by gradient descent for Ajunadeep and liveness [17.68646
295] 141.98220896257774
Cost for Ajunadeep and instrumentalness at initial w: 1386822.6000688348
Gradient at initial w & b for Ajunadeep and instrumentalness: [-784.8027675
1] -1020.0853658536586
Optimal w & b found by gradient descent for Ajunadeep and instrumentalness
[104.6099545] 136.32727033513495
Cost for Ajunadeep and acousticness at initial w: 1388268.9276980667
Gradient at initial w & b for Ajunadeep and acousticness: [-60.3956852] -102
0.0853658536586
Optimal w & b found by gradient descent for Ajunadeep and acousticness [8.01
306699] 142.1002968573183
Cost for All Day I Dream and valence at initial w: 5397174.956949957
Gradient at initial w & b for All Day I Dream and valence: [-708.68799735] -
1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and valence [97.
61099059] 255.37964782688118
Cost for All Day I Dream and danceability at initial w: 5395724.648403957
Gradient at initial w & b for All Day I Dream and danceability: [-1434.64018
519] -1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and danceability
[191.81266666] 247.28273614253115
Cost for All Day I Dream and energy at initial w: 5395932.109515216
Gradient at initial w & b for All Day I Dream and energy: [-1330.68738095] -
1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and energy [180.
32825295] 249.28946433116934
Cost for All Day I Dream and speechiness at initial w: 5398391.986471953
```

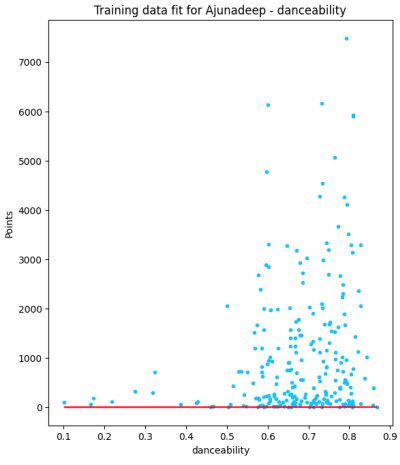
```
Gradient at initial w & b for All Day I Dream and speechiness: [-99.6815444
4] -1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and speechiness
[13.83559248] 257.99565413965917
Cost for All Day I Dream and loudness at initial w: 5433022.025444866
Gradient at initial w & b for All Day I Dream and loudness: [17114.38002116]
-1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and loudness [-1
48.99137693] 54.91136161385612
Cost for All Day I Dream and liveness at initial w: 5398205.6621033205
Gradient at initial w & b for All Day I Dream and liveness: [-192.92141746]
-1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and liveness [2
6.70190433] 257.8377066423841
Cost for All Day I Dream and instrumentalness at initial w: 5395482.83787715
1
Gradient at initial w & b for All Day I Dream and instrumentalness: [-1555.7
3547989] -1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and instrumental
ness [206.11239592] 245.3870206748837
Cost for All Day I Dream and acousticness at initial w: 5398526.586557087
Gradient at initial w & b for All Day I Dream and acousticness: [-32.3823913
4] -1851.7407407407406
Optimal w & b found by gradient descent for All Day I Dream and acousticness
[4.04757013] 258.03988550832037
Cost for Drumcode and valence at initial w: 7927059.9276170125
Gradient at initial w & b for Drumcode and valence: [-450.9659782] -2474.127
962085308
Optimal w & b found by gradient descent for Drumcode and valence [62.800881
2] 343.9922043146388
Cost for Drumcode and danceability at initial w: 7924417.605359445
Gradient at initial w & b for Drumcode and danceability: [-1773.12640758] -2
474.127962085308
Optimal w & b found by gradient descent for Drumcode and danceability [238.1
2671758] 332.29499412465805
Cost for Drumcode and energy at initial w: 7923751.35289899
Gradient at initial w & b for Drumcode and energy: [-2106.58554502] -2474.12
7962085308
Optimal w & b found by gradient descent for Drumcode and energy [278.941670
4] 327.4678573310433
Cost for Drumcode and speechiness at initial w: 7927643.054846534
Gradient at initial w & b for Drumcode and speechiness: [-159.24537725] -247
4.127962085308
Optimal w & b found by gradient descent for Drumcode and speechiness [22.155
76246] 344.6810304016941
Cost for Drumcode and loudness at initial w: 7965924.088331077
Gradient at initial w & b for Drumcode and loudness: [18925.70552607] -2474.
127962085308
Optimal w & b found by gradient descent for Drumcode and loudness [-291.5732
4587] 59.01347048108063
Cost for Drumcode and liveness at initial w: 7927324.848857084
Gradient at initial w & b for Drumcode and liveness: [-318.44651801] -2474.1
27962085308
Optimal w & b found by gradient descent for Drumcode and liveness [44.036837
24] 344.34527150849857
Cost for Drumcode and instrumentalness at initial w: 7923822.541353279
```

```
Gradient at initial w & b for Drumcode and instrumentalness: [-2070.9725176
3] -2474.127962085308
Optimal w & b found by gradient descent for Drumcode and instrumentalness [2
74.28115407] 327.9457895295698
Cost for Drumcode and acousticness at initial w: 7927878.062810335
Gradient at initial w & b for Drumcode and acousticness: [-41.70808402] -247
4.127962085308
Optimal w & b found by gradient descent for Drumcode and acousticness [5.543
735] 344.775112021452
Cost for Afterlife Records and valence at initial w: 5855075.587030719
Gradient at initial w & b for Afterlife Records and valence: [-255.67598182]
-1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and valence [3
4.85354854] 265.79494200659667
Cost for Afterlife Records and danceability at initial w: 5852874.98960226
Gradient at initial w & b for Afterlife Records and danceability: [-1356.983
59659] -1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and danceabili
ty [182.22322114] 256.7301018865911
Cost for Afterlife Records and energy at initial w: 5852967.197694568
Gradient at initial w & b for Afterlife Records and energy: [-1310.80816477]
-1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and energy [17
6.68708441] 257.47246077997096
Cost for Afterlife Records and speechiness at initial w: 5855387.029414902
Gradient at initial w & b for Afterlife Records and speechiness: [-99.797660
23] -1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and speechines
s [13.92041557] 266.1660053084514
Cost for Afterlife Records and loudness at initial w: 5893849.694075032
Gradient at initial w & b for Afterlife Records and loudness: [19031.8753522
7] -1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and loudness
[-169.8152901] 35.567995206736214
Cost for Afterlife Records and liveness at initial w: 5855147.459468773
Gradient at initial w & b for Afterlife Records and liveness: [-219.6762267]
-1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and liveness
[30.37004319] 265.9436590745777
Cost for Afterlife Records and instrumentalness at initial w: 5852450.732688
514
Gradient at initial w & b for Afterlife Records and instrumentalness: [-156
9.43376705] -1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and instrument
alness [207.52124799] 253.4317425309747
Cost for Afterlife Records and acousticness at initial w: 5855364.054683364
Gradient at initial w & b for Afterlife Records and acousticness: [-111.3394
6017] -1910.3352272727273
Optimal w & b found by gradient descent for Afterlife Records and acousticne
ss [15.09039348] 266.132354845158
```

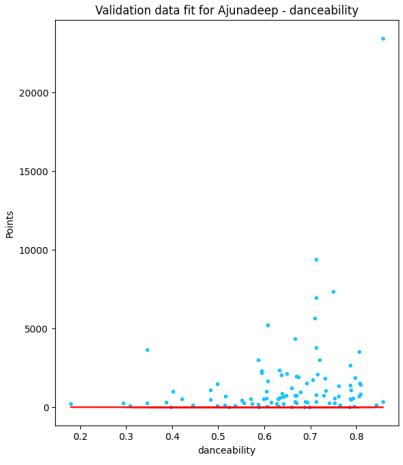

Ajunadeep-valence
Ajunadeep-danceability
Ajunadeep-energy
Ajunadeep-speechiness
Ajunadeep-loudness
Ajunadeep-liveness
Ajunadeep-instrumentalness
Ajunadeep-acousticness
All Day I Dream-valence
All Day I Dream-danceability
All Day I Dream-energy
All Day I Dream-speechiness
All Day I Dream-loudness
All Day I Dream-liveness
All Day I Dream-instrumentalness
All Day I Dream-acousticness
Drumcode-valence
Drumcode-danceability
Drumcode-energy
Drumcode-speechiness
Drumcode-loudness
Drumcode-liveness
Drumcode-instrumentalness
Drumcode-acousticness
Afterlife Records-valence
Afterlife Records-danceability
Afterlife Records-energy
Afterlife Records-speechiness
Afterlife Records-loudness
Afterlife Records-liveness
Afterlife Records-instrumentalness
Afterlife Records-acousticness
1e6
Cost
Iteration


Cost vs W
Cost
Iteration

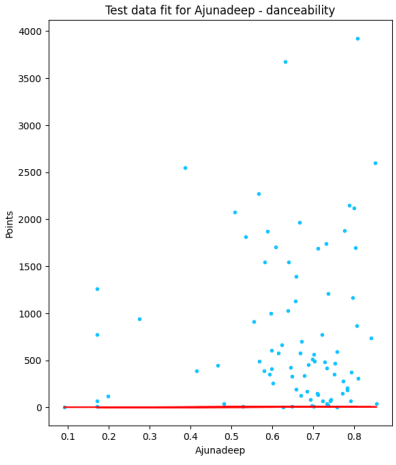

Training data fit for Ajunadeep - valence
Points
valence
MSE: 2775590.17
RMSE: 1666.01
MAE: 1018.51
R-squared: -391.81
Validation data fit for Ajunadeep - valence
Points
valence
MSE: 9734790.37
RMSE: 3120.06
MAE: 1443.62
R-squared: -124.97
Test data fit for Ajunadeep - valence
Points
Ajunadeep
MSE: 1308295.87
RMSE: 1143.81
MAE: 778.35
R-squared: -157.31



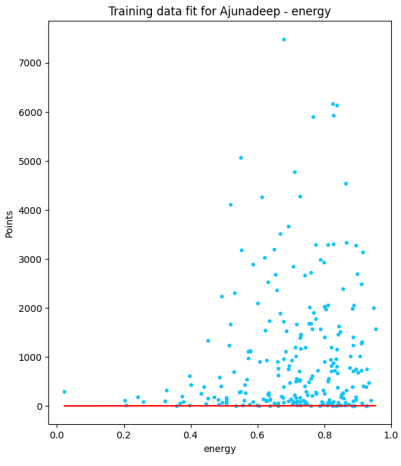

MSE: 2774020.62
RMSE: 1665.54
MAE: 1017.76
R-squared: -391.59



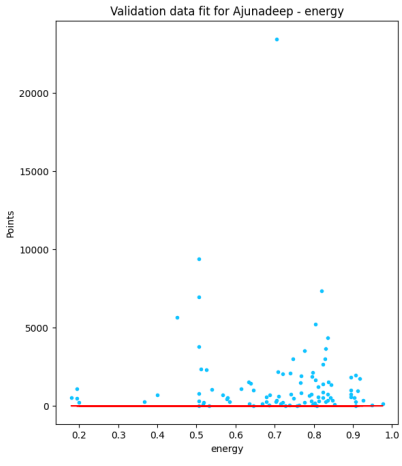

MSE: 9732750.90
RMSE: 3119.74
MAE: 1442.95
R-squared: -124.94



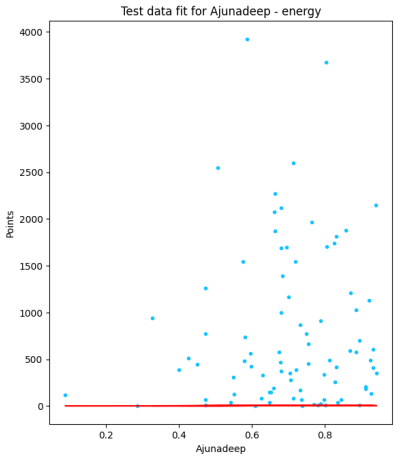

MSE: 1307057.60
RMSE: 1143.27
MAE: 777.56
R-squared: -157.16



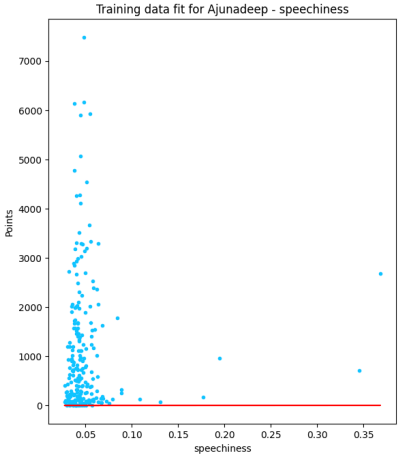

MSE: 2773891.06
RMSE: 1665.50
MAE: 1017.70
R-squared: -391.57



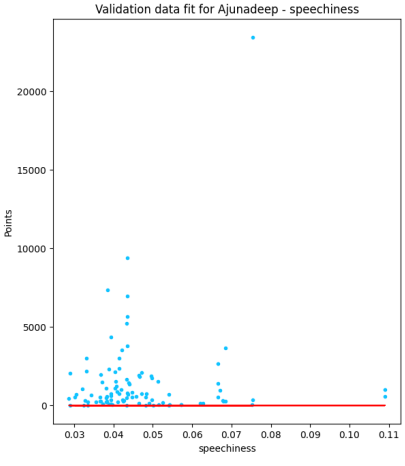

MSE: 9732386.24
RMSE: 3119.68
MAE: 1442.83
R-squared: -124.93



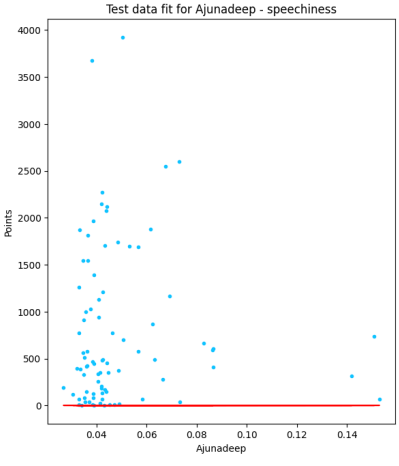

MSE: 1306857.65
RMSE: 1143.18
MAE: 777.43
R-squared: -157.13



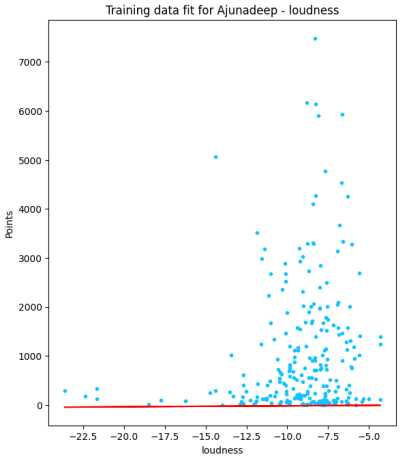

MSE: 2776579.32
RMSE: 1666.31
MAE: 1018.99
R-squared: -391.95



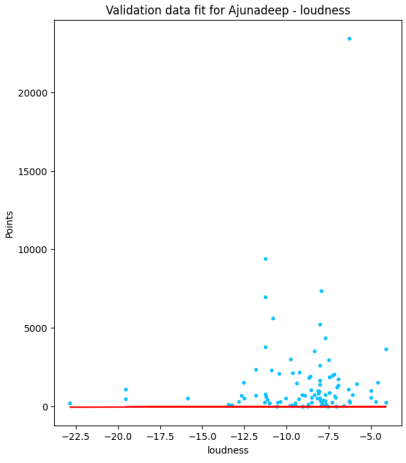

MSE: 9736217.53
RMSE: 3120.29
MAE: 1444.09
R-squared: -124.98



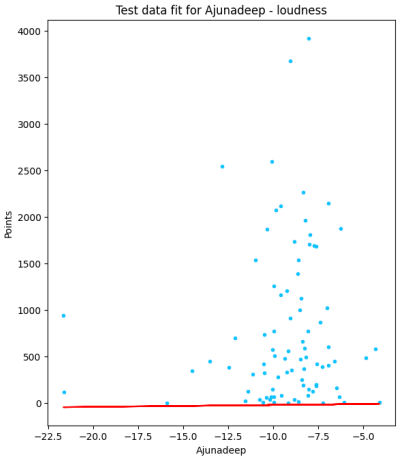

MSE: 1308895.43
RMSE: 1144.07
MAE: 778.74
R-squared: -157.38



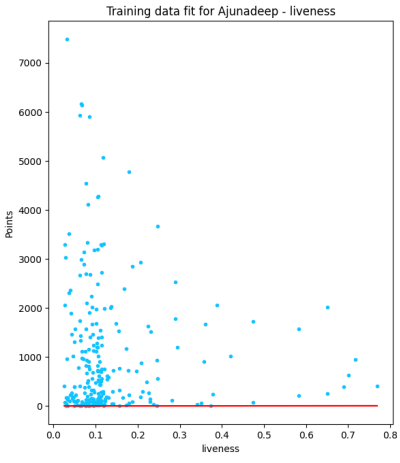

MSE: 2814245.99
RMSE: 1677.57
MAE: 1037.29
R-squared: -397.28



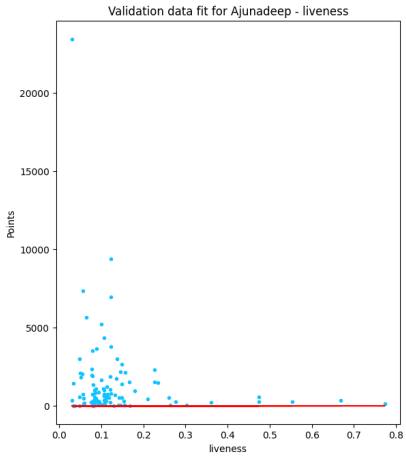

MSE: 9789490.37
RMSE: 3128.82
MAE: 1462.41
R-squared: -125.67



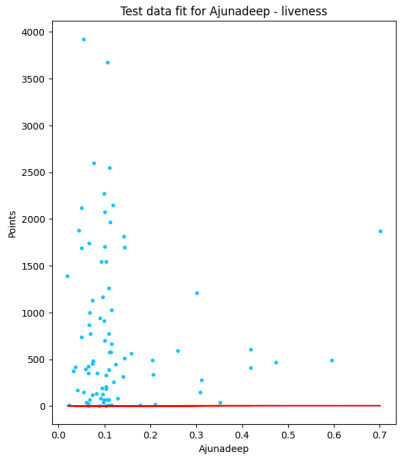

MSE: 1338333.39
RMSE: 1156.86
MAE: 797.40
R-squared: -160.94



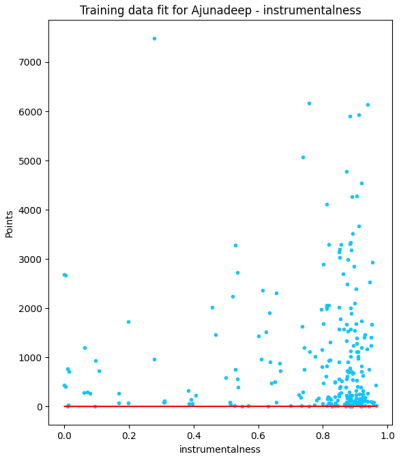

MSE: 2776230.66
RMSE: 1666.20
MAE: 1018.82
R-squared: -391.90



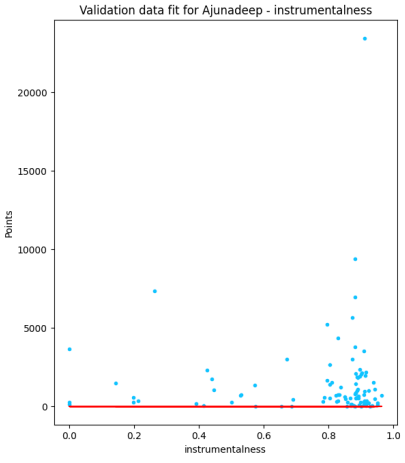

MSE: 9735684.91
RMSE: 3120.21
MAE: 1443.92
R-squared: -124.98



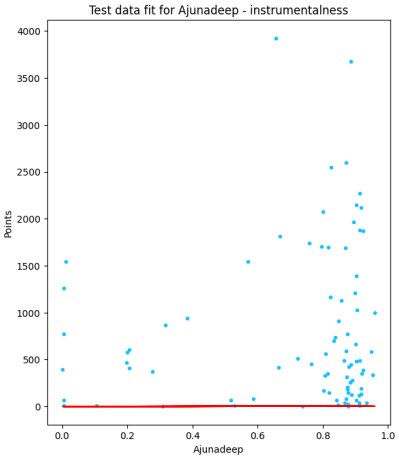

MSE: 1308634.25
RMSE: 1143.96
MAE: 778.57
R-squared: -157.35



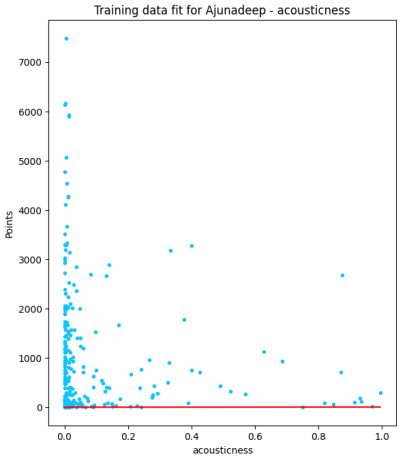

MSE: 2773702.75
RMSE: 1665.44
MAE: 1017.61
R-squared: -391.54



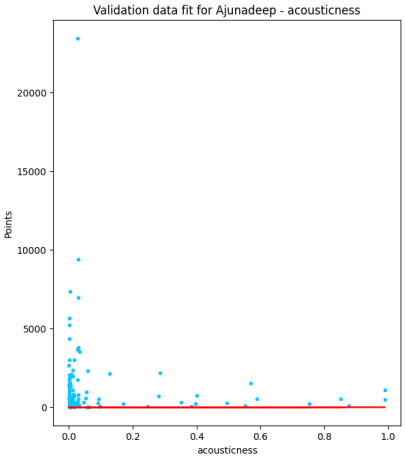

MSE: 9732029.20
RMSE: 3119.62
MAE: 1442.71
R-squared: -124.93



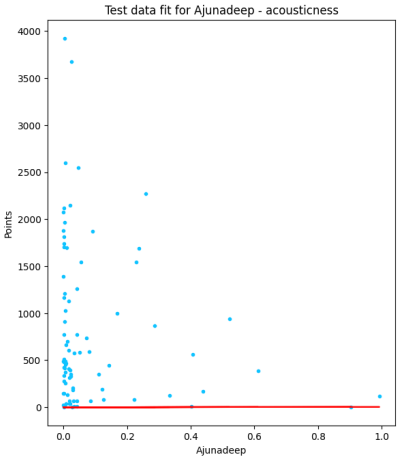

MSE: 1306804.47
RMSE: 1143.16
MAE: 777.39
R-squared: -157.13



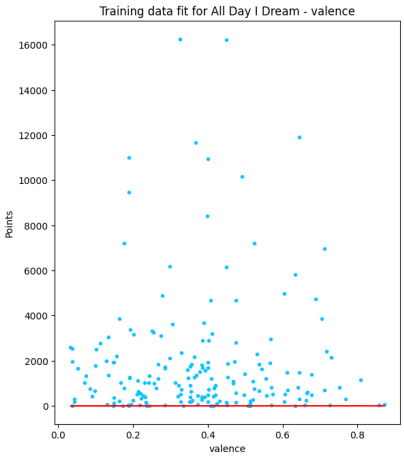

MSE: 2776398.20
RMSE: 1666.25
MAE: 1018.90
R-squared: -391.92



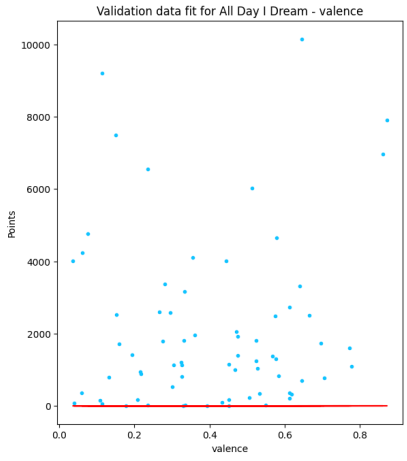

MSE: 9735863.32
RMSE: 3120.23
MAE: 1443.98
R-squared: -124.98



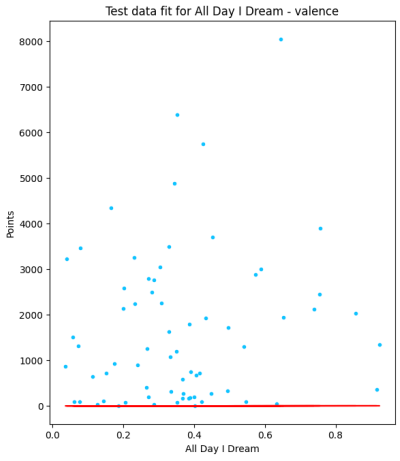

MSE: 1308765.66
RMSE: 1144.01
MAE: 778.65
R-squared: -157.36



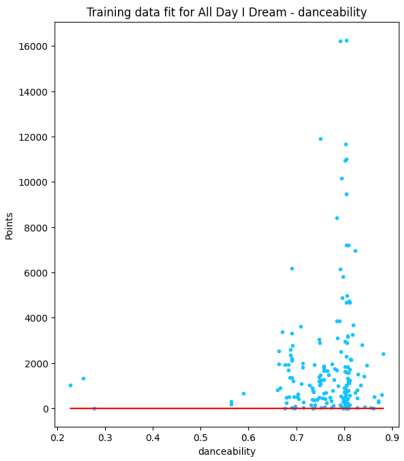

MSE: 10794405.64
RMSE: 3285.48
MAE: 1850.00
R-squared: -275.74



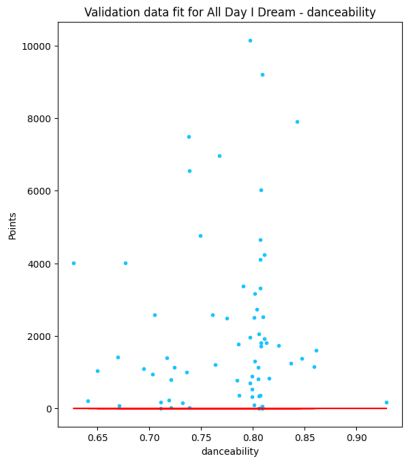

MSE: 9588381.20
RMSE: 3096.51
MAE: 2073.26
R-squared: -125.88



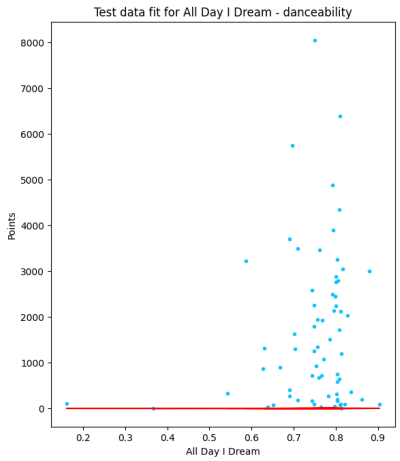

MSE: 5492132.78
RMSE: 2343.53
MAE: 1640.42
R-squared: -132.33



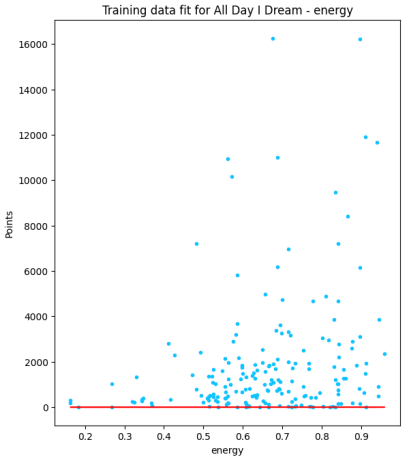

MSE: 10791544.68
RMSE: 3285.05
MAE: 1849.23
R-squared: -275.67



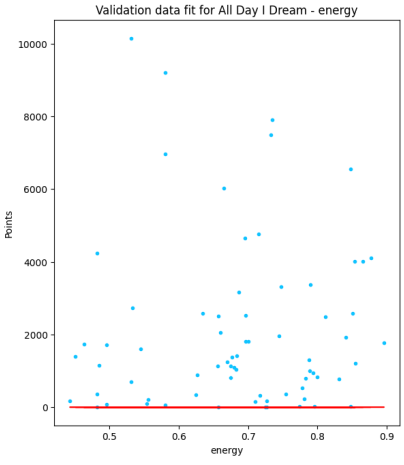

MSE: 9585293.67
RMSE: 3096.01
MAE: 2072.52
R-squared: -125.84



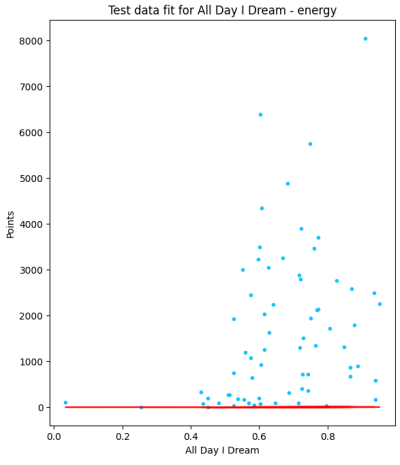

MSE: 5489598.25
RMSE: 2342.99
MAE: 1639.65
R-squared: -132.26



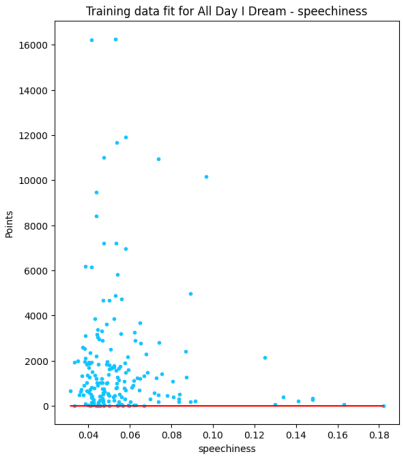

MSE: 10792280.96
RMSE: 3285.16
MAE: 1849.43
R-squared: -275.69



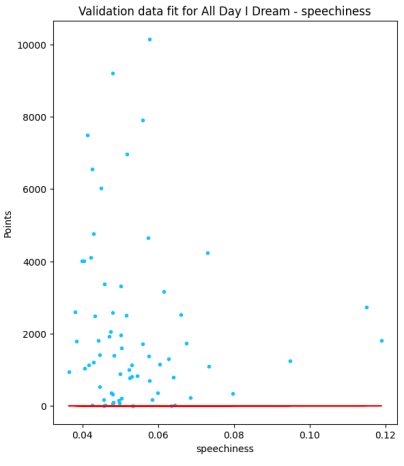

MSE: 9586044.71
RMSE: 3096.13
MAE: 2072.70
R-squared: -125.85



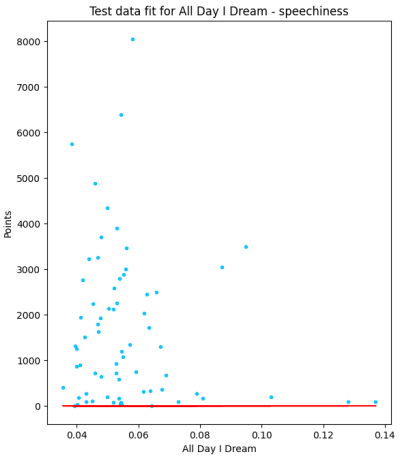

MSE: 5490150.81
RMSE: 2343.11
MAE: 1639.82
R-squared: -132.28



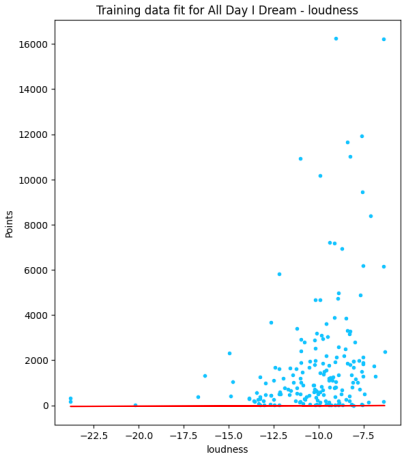

MSE: 10796765.91
RMSE: 3285.84
MAE: 1850.63
R-squared: -275.80



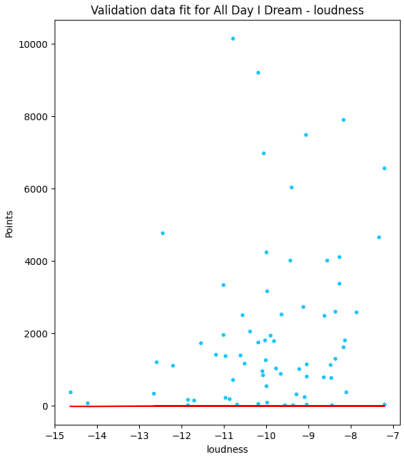

MSE: 9591277.69
RMSE: 3096.98
MAE: 2073.96
R-squared: -125.92



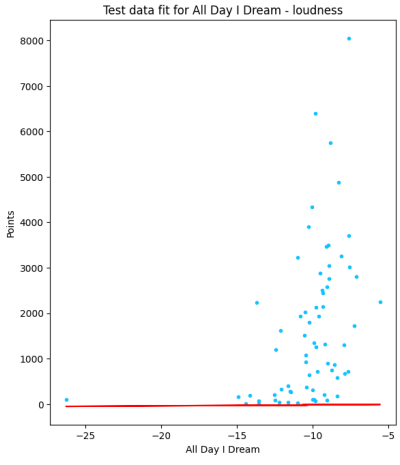

MSE: 5494138.21
RMSE: 2343.96
MAE: 1641.03
R-squared: -132.37



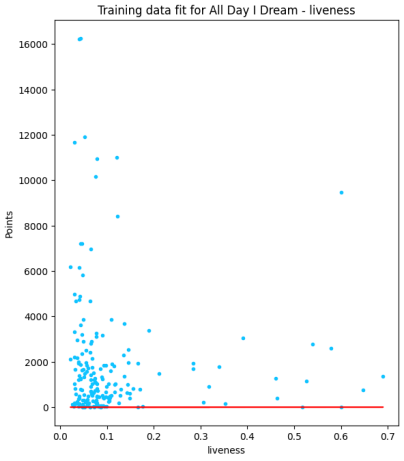

MSE: 10873435.15
RMSE: 3297.49
MAE: 1871.22
R-squared: -277.77



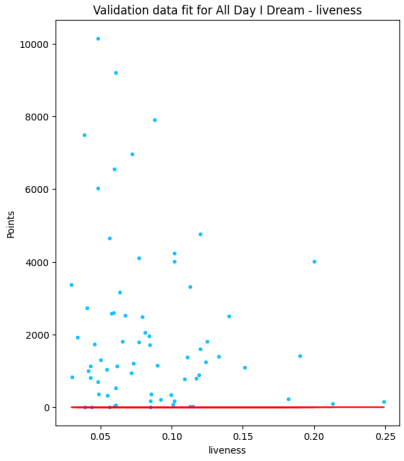

MSE: 9674304.26
RMSE: 3110.35
MAE: 2093.88
R-squared: -127.02



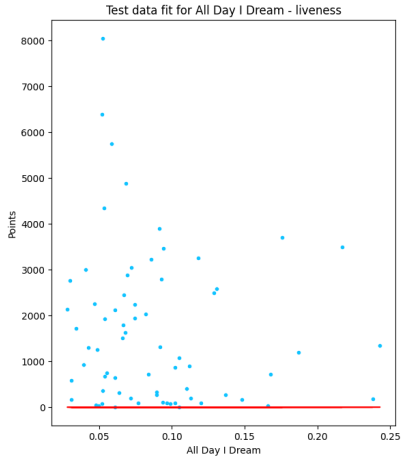

MSE: 5562395.21
RMSE: 2358.47
MAE: 1661.69
R-squared: -134.03



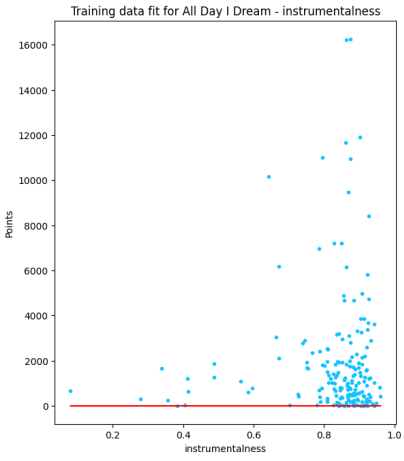

MSE: 10796365.95
RMSE: 3285.78
MAE: 1850.52
R-squared: -275.79



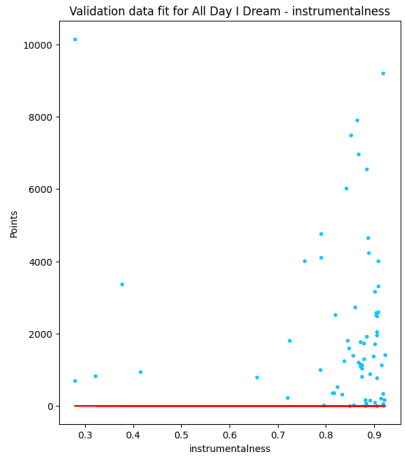

MSE: 9591009.25
RMSE: 3096.94
MAE: 2073.90
R-squared: -125.91



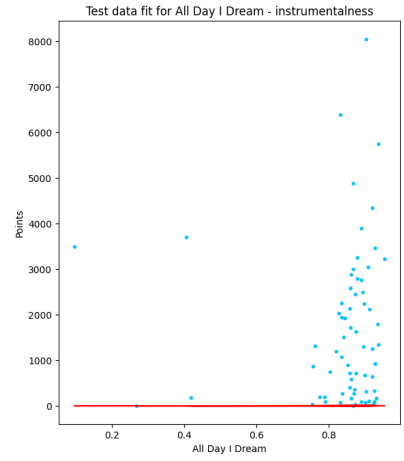

MSE: 5493931.73
RMSE: 2343.91
MAE: 1640.97
R-squared: -132.37



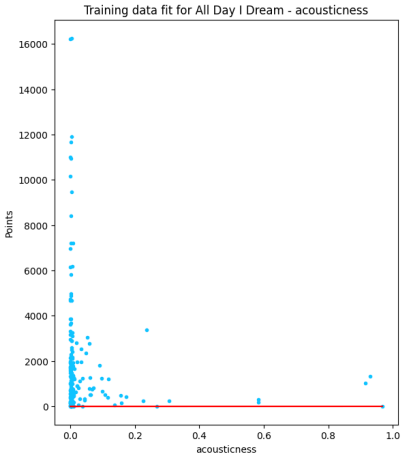

MSE: 10791031.13
RMSE: 3284.97
MAE: 1849.10
R-squared: -275.66



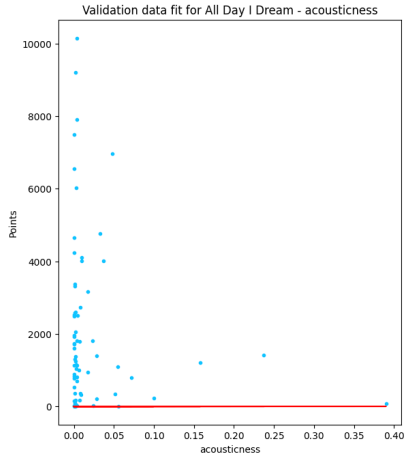

MSE: 9584866.46
RMSE: 3095.94
MAE: 2072.42
R-squared: -125.83



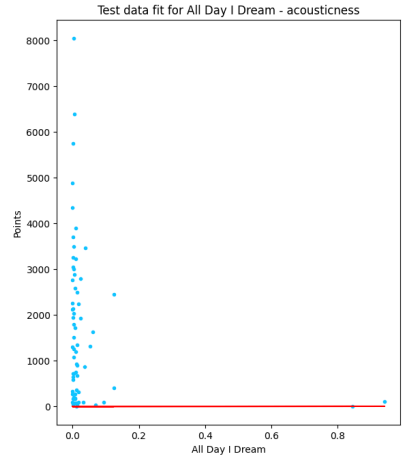

MSE: 5489014.58
RMSE: 2342.86
MAE: 1639.47
R-squared: -132.25



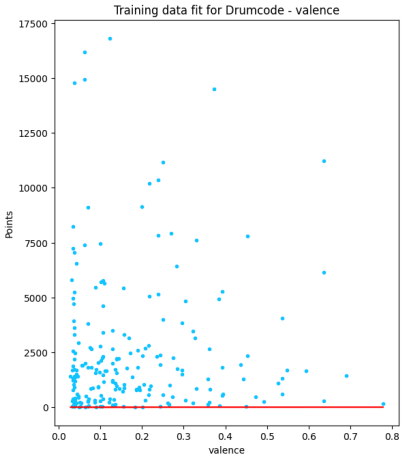

MSE: 10796881.23
RMSE: 3285.86
MAE: 1850.66
R-squared: -275.81



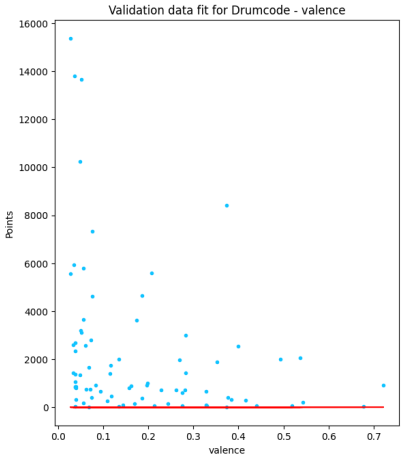

MSE: 9591547.92
RMSE: 3097.02
MAE: 2074.03
R-squared: -125.92



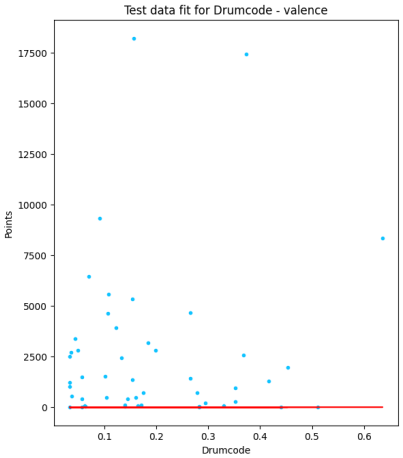

MSE: 5494240.43
RMSE: 2343.98
MAE: 1641.06
R-squared: -132.38



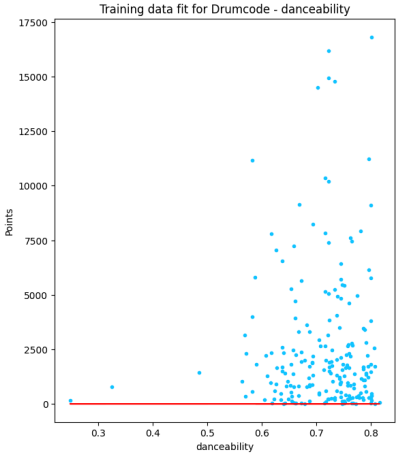

MSE: 15854203.16
RMSE: 3981.73
MAE: 2472.78
R-squared: -342.47



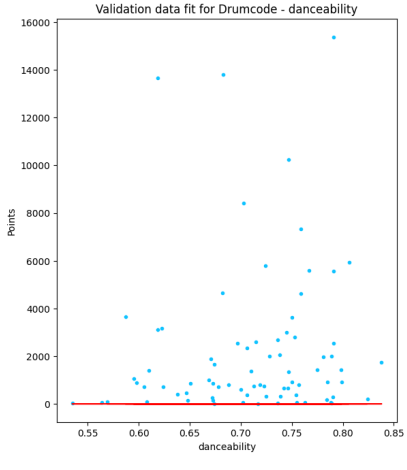

MSE: 14882067.79
RMSE: 3857.73
MAE: 2213.96
R-squared: -115.31



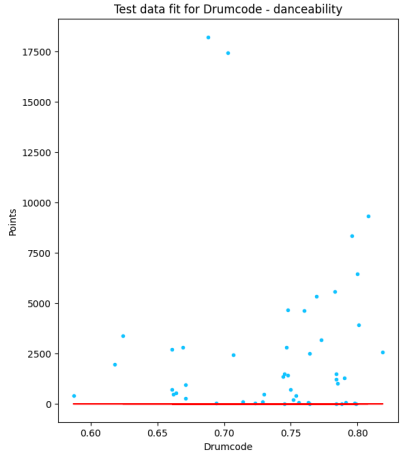

MSE: 21140013.95
RMSE: 4597.83
MAE: 2516.78
R-squared: -68.96



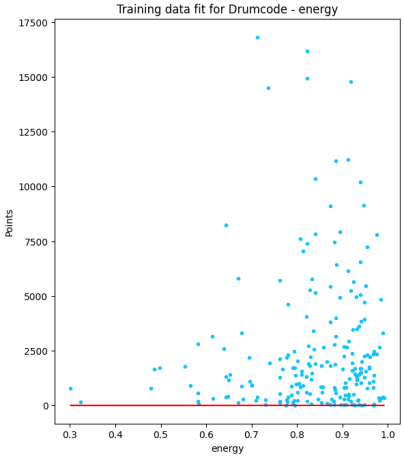

MSE: 15848873.18
RMSE: 3981.06
MAE: 2471.71
R-squared: -342.35



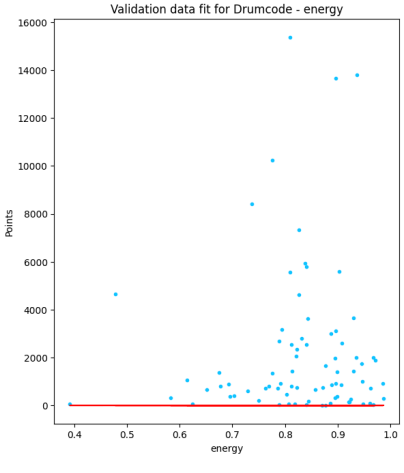

MSE: 14877426.40
RMSE: 3857.13
MAE: 2212.91
R-squared: -115.27



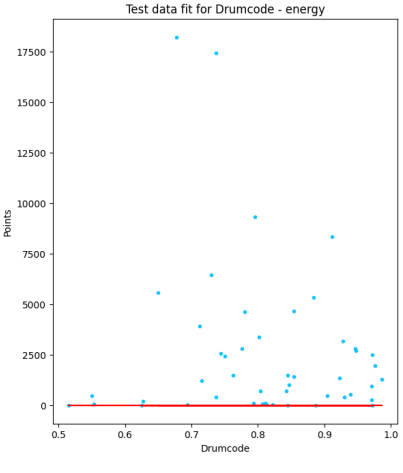

MSE: 21134489.01
RMSE: 4597.23
MAE: 2515.69
R-squared: -68.94



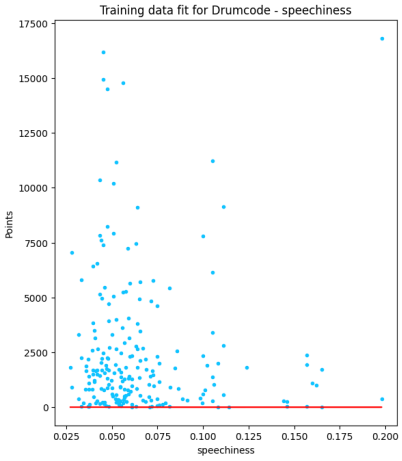

MSE: 15847618.12
RMSE: 3980.91
MAE: 2471.45
R-squared: -342.33



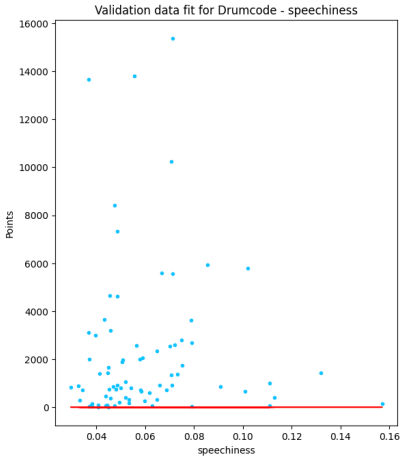

MSE: 14876377.69
RMSE: 3856.99
MAE: 2212.68
R-squared: -115.26



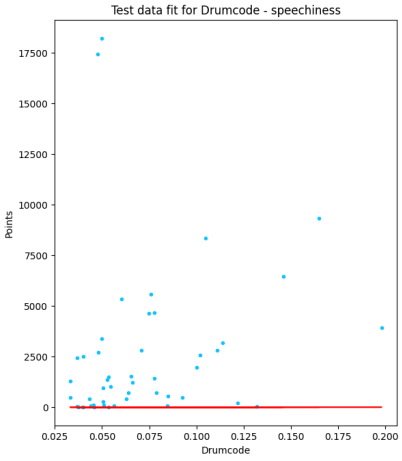

MSE: 21133756.13
RMSE: 4597.15
MAE: 2515.54
R-squared: -68.94



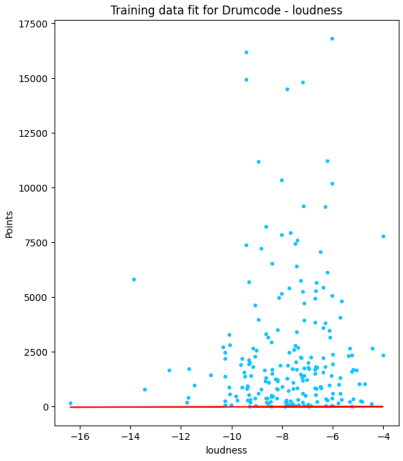

MSE: 15855277.83
RMSE: 3981.87
MAE: 2473.00
R-squared: -342.49



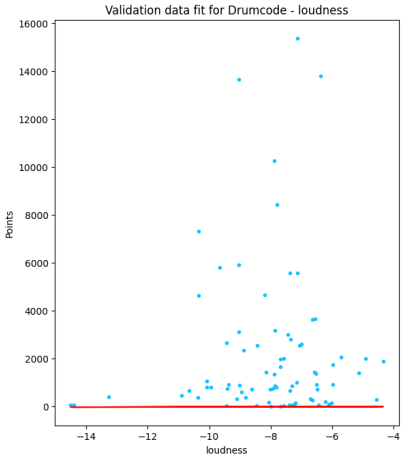

MSE: 14883187.88
RMSE: 3857.87
MAE: 2214.21
R-squared: -115.32



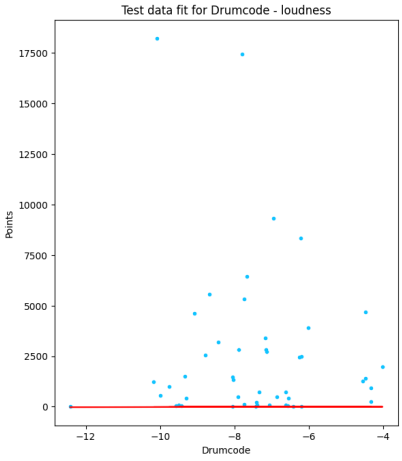

MSE: 21141205.61
RMSE: 4597.96
MAE: 2517.02
R-squared: -68.97



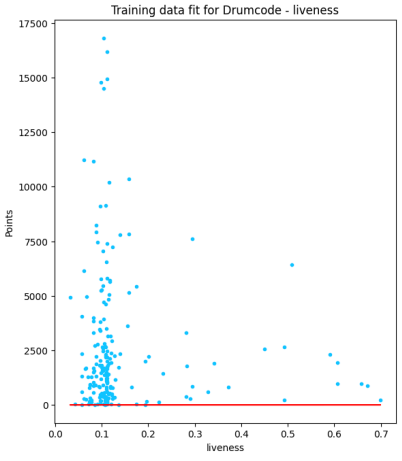

MSE: 15932940.35
RMSE: 3991.61
MAE: 2488.65
R-squared: -344.17



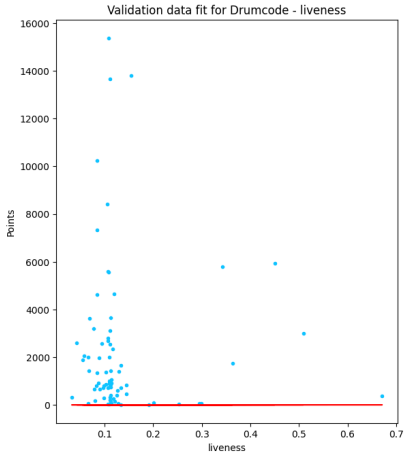

MSE: 14954948.52
RMSE: 3867.16
MAE: 2230.36
R-squared: -115.88



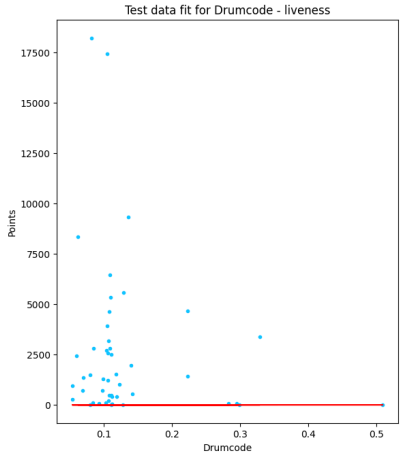

MSE: 21217608.40
RMSE: 4606.26
MAE: 2532.15
R-squared: -69.22




MSE: 15854564.24
RMSE: 3981.78
MAE: 2472.85
R-squared: -342.48

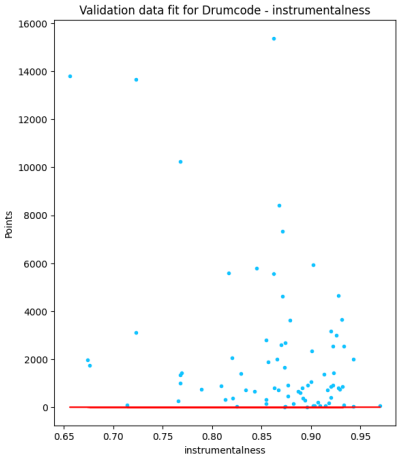



MSE: 14882518.44
RMSE: 3857.79
MAE: 2214.06
R-squared: -115.31

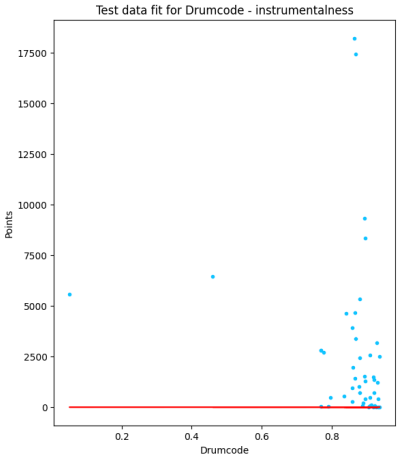



MSE: 21140596.24
RMSE: 4597.89
MAE: 2516.90
R-squared: -68.97

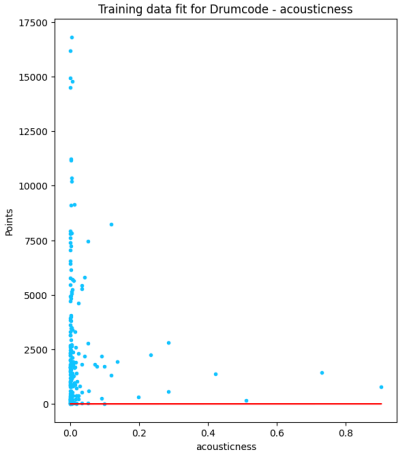

MSE: 15847709.63
RMSE: 3980.92
MAE: 2471.47
R-squared: -342.33



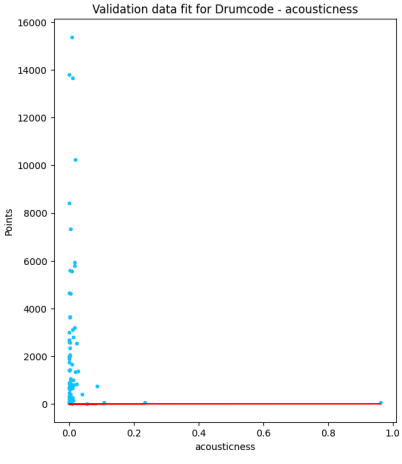

MSE: 14876074.81
RMSE: 3856.95
MAE: 2212.61
R-squared: -115.26



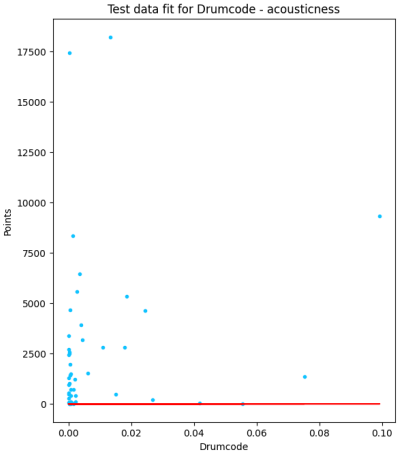

MSE: 21133321.99
RMSE: 4597.10
MAE: 2515.46
R-squared: -68.94



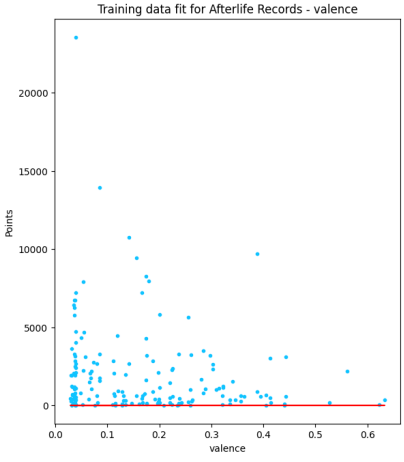

MSE: 15855657.38
RMSE: 3981.92
MAE: 2473.07
R-squared: -342.50



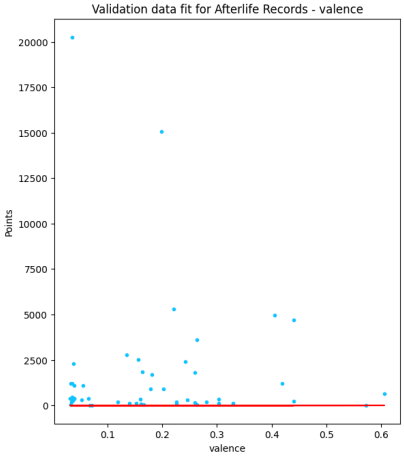

MSE: 14883511.20
RMSE: 3857.92
MAE: 2214.29
R-squared: -115.32



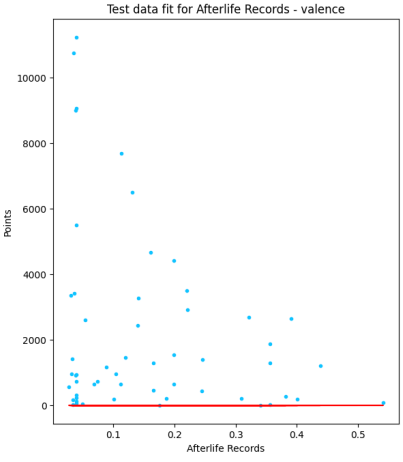

MSE: 21141828.12
RMSE: 4598.02
MAE: 2517.15
R-squared: -68.97



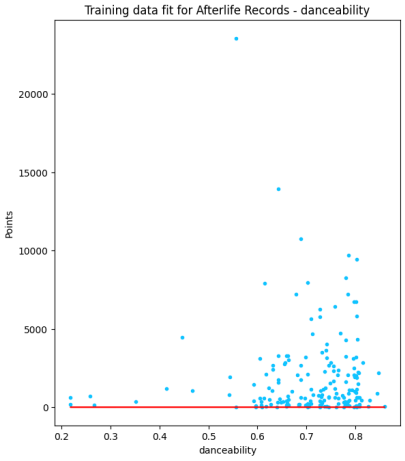

MSE: 11709907.93
RMSE: 3421.97
MAE: 1909.00
R-squared: -254.52



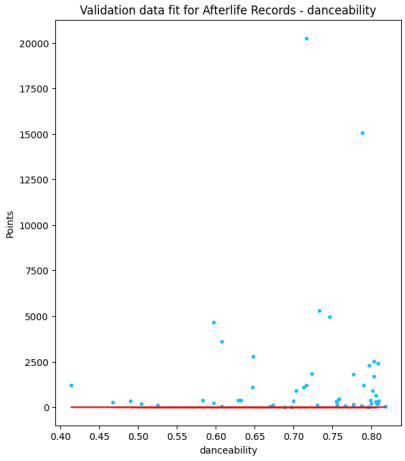

MSE: 15101182.62
RMSE: 3886.02
MAE: 1640.40
R-squared: -61.06



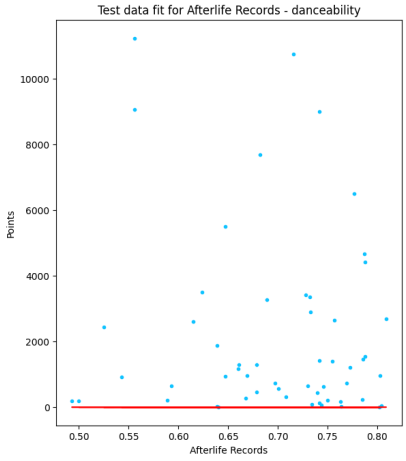

MSE: 12441504.15
RMSE: 3527.25
MAE: 2165.74
R-squared: -87.28



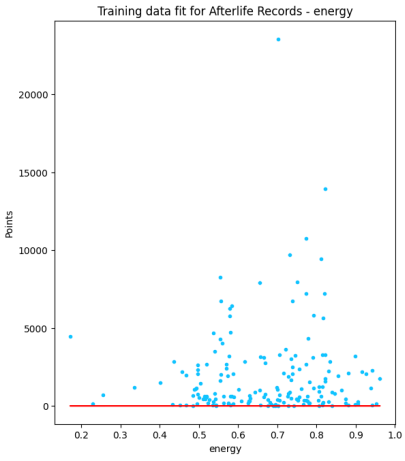

MSE: 11705771.52
RMSE: 3421.37
MAE: 1907.94
R-squared: -254.43



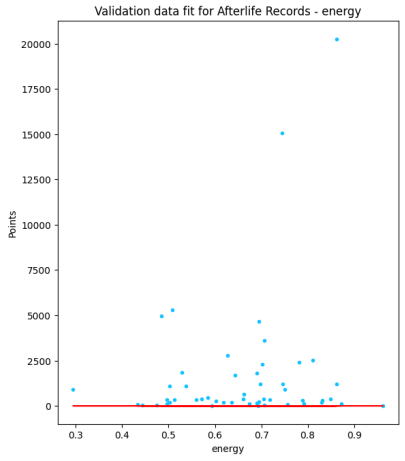

MSE: 15097724.77
RMSE: 3885.58
MAE: 1639.35
R-squared: -61.04



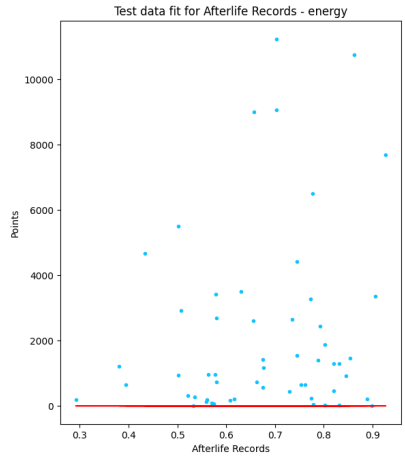

MSE: 12436808.65
RMSE: 3526.59
MAE: 2164.67
R-squared: -87.25



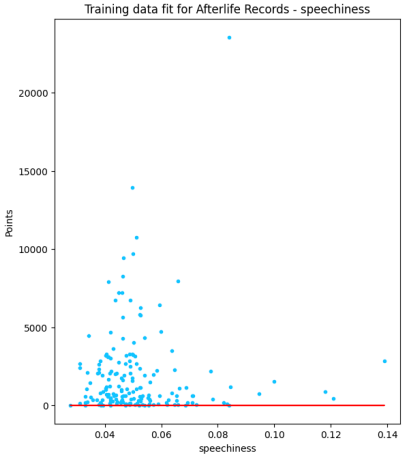

MSE: 11706031.69
RMSE: 3421.41
MAE: 1908.00
R-squared: -254.44



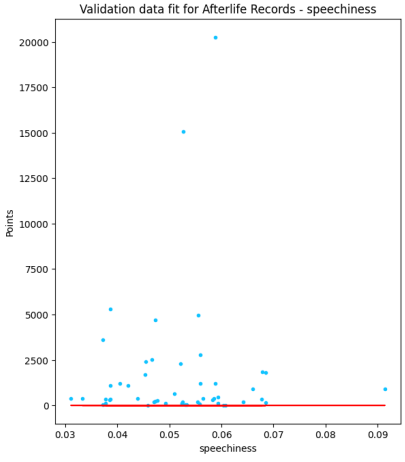

MSE: 15098052.17
RMSE: 3885.62
MAE: 1639.45
R-squared: -61.05



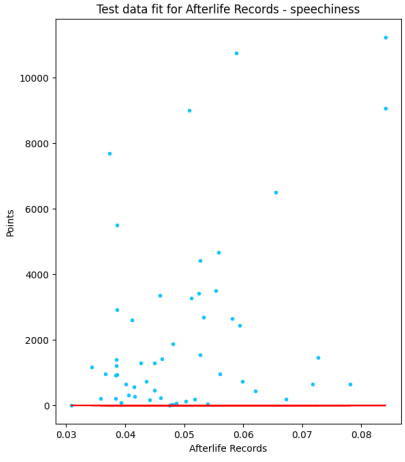

MSE: 12436988.18
RMSE: 3526.61
MAE: 2164.71
R-squared: -87.25



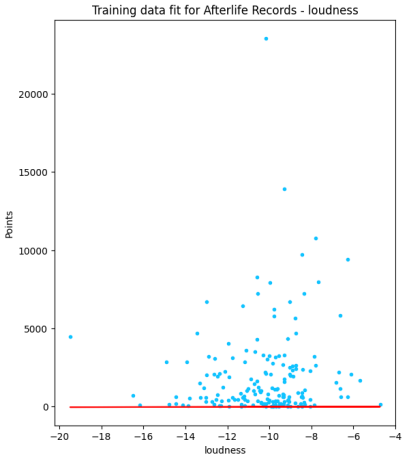

MSE: 11710781.60
RMSE: 3422.10
MAE: 1909.23
R-squared: -254.54



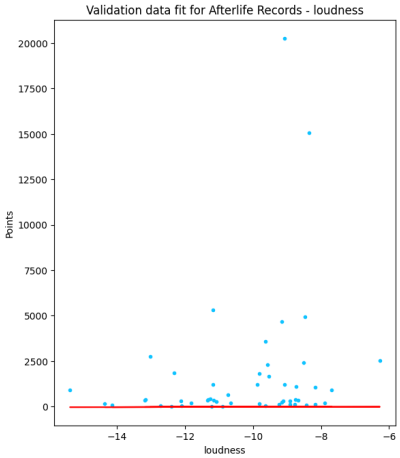

MSE: 15102032.17
RMSE: 3886.13
MAE: 1640.66
R-squared: -61.06



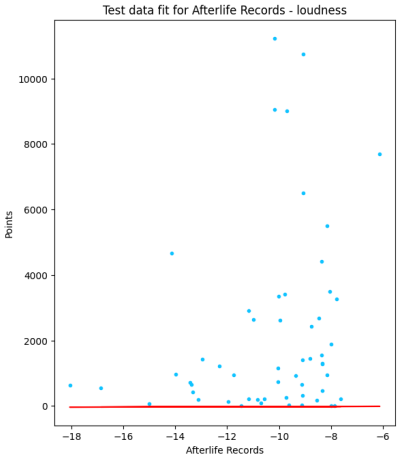

MSE: 12442421.44
RMSE: 3527.38
MAE: 2165.95
R-squared: -87.29



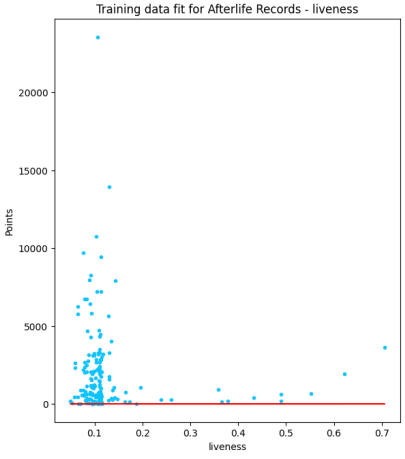

MSE: 11790092.16
RMSE: 3433.67
MAE: 1929.89
R-squared: -256.27



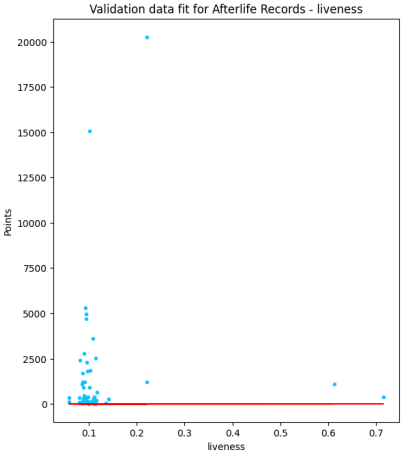

MSE: 15170501.12
RMSE: 3894.93
MAE: 1661.39
R-squared: -61.34



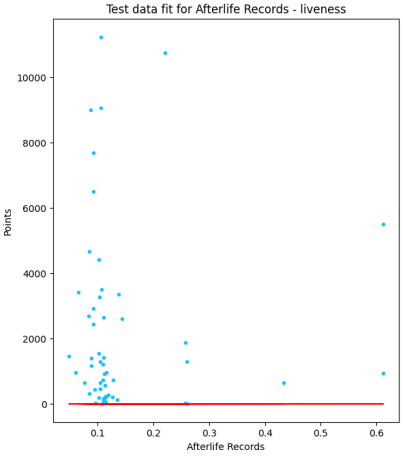

MSE: 12532543.49
RMSE: 3540.13
MAE: 2186.65
R-squared: -87.93



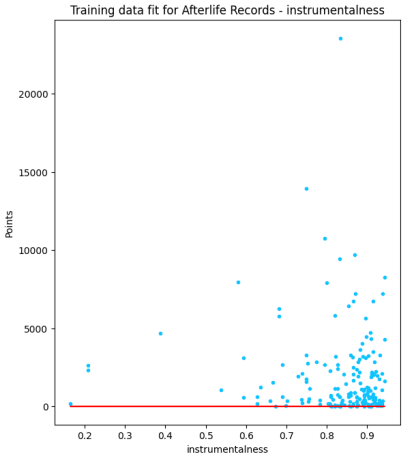

MSE: 11710227.16
RMSE: 3422.02
MAE: 1909.09
R-squared: -254.53



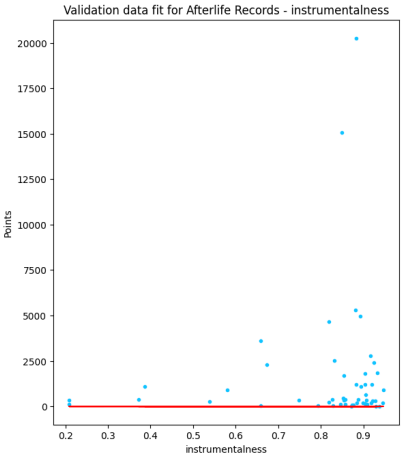

MSE: 15101548.16
RMSE: 3886.07
MAE: 1640.51
R-squared: -61.06



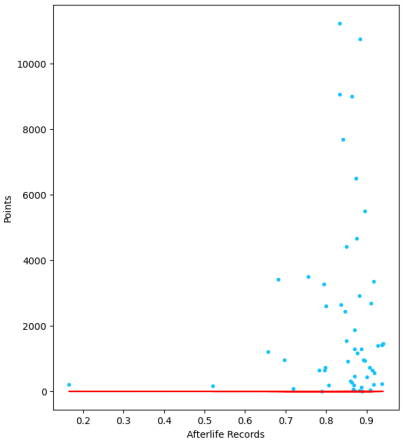

MSE: 12441633.32
RMSE: 3527.27
MAE: 2165.77
R-squared: -87.28



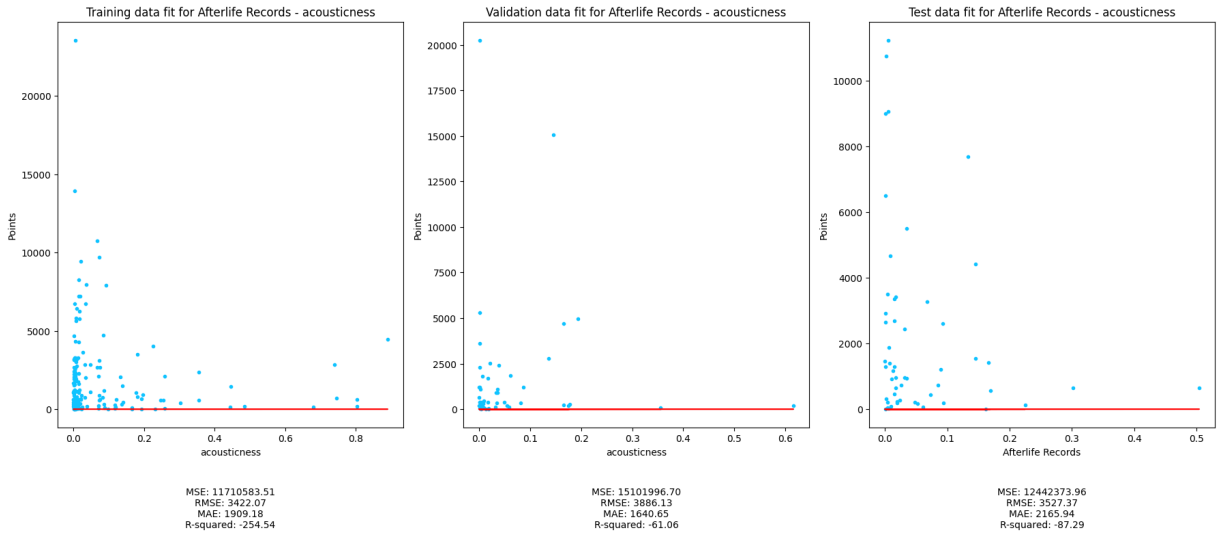

MSE: 11704814.08
RMSE: 3421.23
MAE: 1907.70
R-squared: -254.41




MSE: 15097059.00
RMSE: 3885.49
MAE: 1639.14
R-squared: -61.04




MSE: 12435615.75
RMSE: 3526.42
MAE: 2164.41
R-squared: -87.24




# Running log normalized data

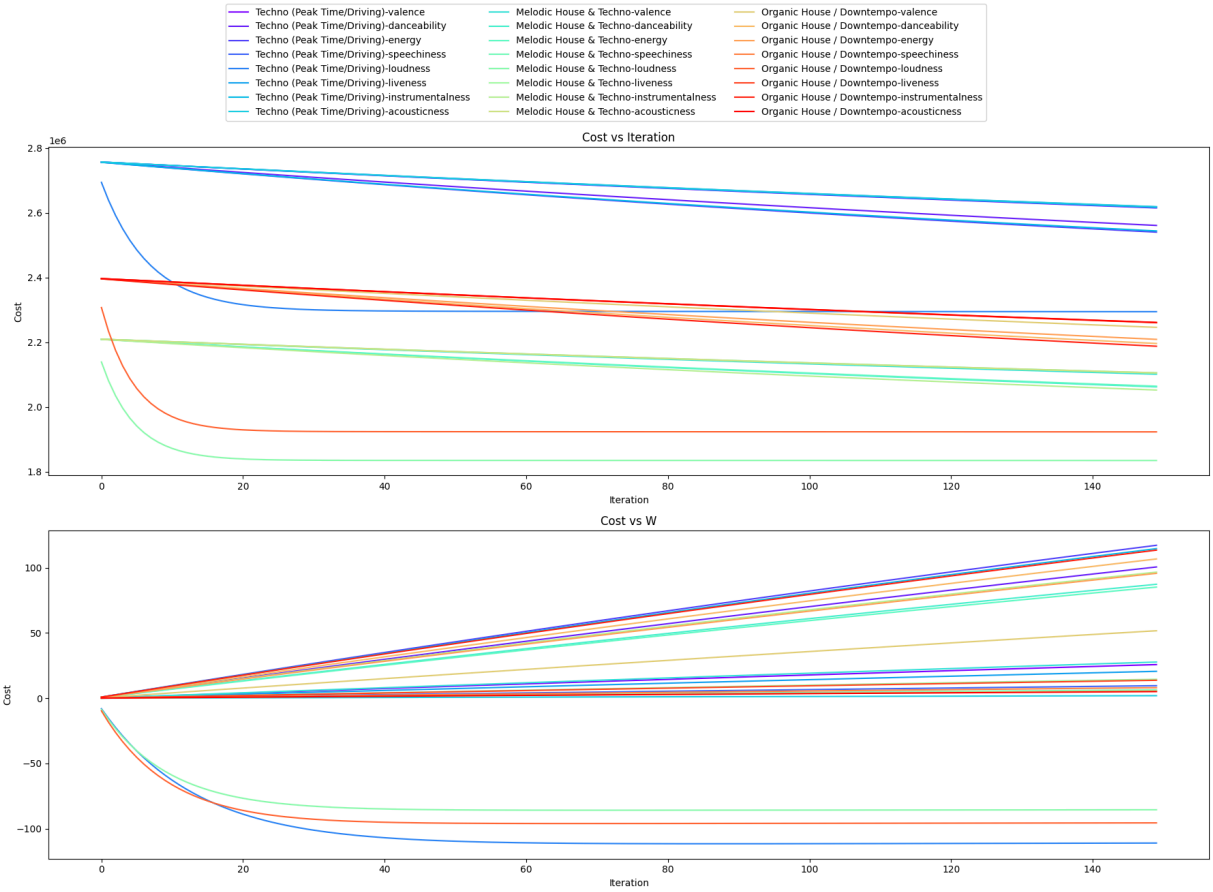
## Genre function

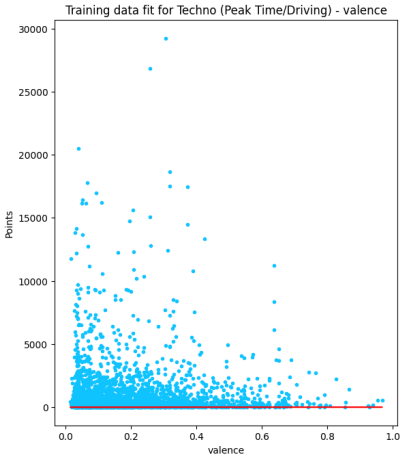
In [32]:
```
run_all_calculations_genre(log_train, log_validate, log_test)
```

```
Cost for Techno (Peak Time/Driving) and valence at initial w: 2756182.561454
402
Gradient at initial w & b for Techno (Peak Time/Driving) and valence: [-186.
38003769] -1035.4833504977687
Optimal w & b found by gradient descent for valence for Techno (Peak Time/Dr
iving): [25.7572645] 143.94703318586474
Cost for Techno (Peak Time/Driving) and danceability at initial w: 2755060.4
68024341
Gradient at initial w & b for Techno (Peak Time/Driving) and danceability:
[-748.40639341] -1035.4833504977687
Optimal w & b found by gradient descent for danceability for Techno (Peak Ti
me/Driving): [100.56557556] 139.01647516185307
Cost for Techno (Peak Time/Driving) and energy at initial w: 2754789.1012330
84
Gradient at initial w & b for Techno (Peak Time/Driving) and energy: [-884.4
1745108] -1035.4833504977687
Optimal w & b found by gradient descent for energy for Techno (Peak Time/Dri
ving): [117.17798244] 137.03003585766362
Cost for Techno (Peak Time/Driving) and speechiness at initial w: 2756417.59
19090556
Gradient at initial w & b for Techno (Peak Time/Driving) and speechiness: [-
68.68632612] -1035.4833504977687
Optimal w & b found by gradient descent for speechiness for Techno (Peak Tim
e/Driving): [9.56293594] 144.25501974254271
Cost for Techno (Peak Time/Driving) and loudness at initial w: 2772824.71671
1207
Gradient at initial w & b for Techno (Peak Time/Driving) and loudness: [807
2.37247442] -1035.4833504977687
Optimal w & b found by gradient descent for loudness for Techno (Peak Time/D
riving): [-111.01249844] 28.083004812890366
Cost for Techno (Peak Time/Driving) and liveness at initial w: 2756258.80501
5503
Gradient at initial w & b for Techno (Peak Time/Driving) and liveness: [-14
8.18583402] -1035.4833504977687
Optimal w & b found by gradient descent for liveness for Techno (Peak Time/D
riving): [20.5953207] 144.08684299564197
Cost for Techno (Peak Time/Driving) and instrumentalness at initial w: 27548
23.043724489
Gradient at initial w & b for Techno (Peak Time/Driving) and instrumentalnes
s: [-867.46312582] -1035.4833504977687
Optimal w & b found by gradient descent for instrumentalness for Techno (Pea
k Time/Driving): [114.60726016] 137.1357641157438
Cost for Techno (Peak Time/Driving) and acousticness at initial w: 2756526.2
268052096
Gradient at initial w & b for Techno (Peak Time/Driving) and acousticness:
[-14.32263319] -1035.4833504977687
Optimal w & b found by gradient descent for acousticness for Techno (Peak Ti
me/Driving): [1.91095185] 144.29842820771134
Cost for Melodic House & Techno and valence at initial w: 2208666.9941293644
Gradient at initial w & b for Melodic House & Techno and valence: [-200.6900
0748] -898.2544184134813
Optimal w & b found by gradient descent for valence for Melodic House & Tech
no: [27.71533071] 124.71156318353071
Cost for Melodic House & Techno and danceability at initial w: 2207767.16195
4749
Gradient at initial w & b for Melodic House & Techno and danceability: [-65
```

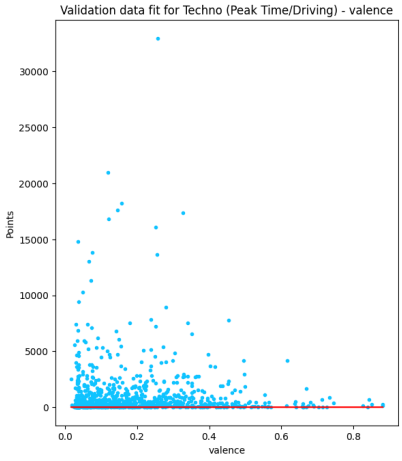
```
1.5541566] -898.2544184134813
Optimal w & b found by gradient descent for danceability for Melodic House &
Techno: [87.29899157] 120.49008216648747
Cost for Melodic House & Techno and energy at initial w: 2207799.834141317
Gradient at initial w & b for Melodic House & Techno and energy: [-635.18925
236] -898.2544184134813
Optimal w & b found by gradient descent for energy for Melodic House & Techn
o: [85.16207986] 120.71190521843569
Cost for Melodic House & Techno and speechiness at initial w: 2208976.083519
792
Gradient at initial w & b for Melodic House & Techno and speechiness: [-45.8
8196917] -898.2544184134813
Optimal w & b found by gradient descent for speechiness for Melodic House &
Techno: [6.37862502] 125.15332929718093
Cost for Melodic House & Techno and loudness at initial w: 2226476.299193209
Gradient at initial w & b for Melodic House & Techno and loudness: [8615.039
92109] -898.2544184134813
Optimal w & b found by gradient descent for loudness for Melodic House & Tec
hno: [-85.51871552] 15.883148985339075
Cost for Melodic House & Techno and liveness at initial w: 2208858.929269747
3
Gradient at initial w & b for Melodic House & Techno and liveness: [-104.547
81089] -898.2544184134813
Optimal w & b found by gradient descent for liveness for Melodic House & Tec
hno: [14.50292013] 125.05032509673134
Cost for Melodic House & Techno and instrumentalness at initial w: 2207613.4
04055469
Gradient at initial w & b for Melodic House & Techno and instrumentalness:
[-728.66629155] -898.2544184134813
Optimal w & b found by gradient descent for instrumentalness for Melodic Hou
se & Techno: [96.60893611] 119.37805941849585
Cost for Melodic House & Techno and acousticness at initial w: 2208989.47865
53043
Gradient at initial w & b for Melodic House & Techno and acousticness: [-39.
17319122] -898.2544184134813
Optimal w & b found by gradient descent for acousticness for Melodic House &
Techno: [5.53238229] 125.16347083925184
Cost for Organic House / Downtempo and valence at initial w: 2395641.6159838
946
Gradient at initial w & b for Organic House / Downtempo and valence: [-374.5
9608092] -1025.2286585365853
Optimal w & b found by gradient descent for valence for Organic House / Down
tempo: [51.69080791] 141.53858299564664
Cost for Organic House / Downtempo and danceability at initial w: 2394795.74
40202613
Gradient at initial w & b for Organic House / Downtempo and danceability: [-
798.37285105] -1025.2286585365853
Optimal w & b found by gradient descent for danceability for Organic House /
Downtempo: [106.70648574] 136.83871556568855
Cost for Organic House / Downtempo and energy at initial w: 2394971.17988757
46
Gradient at initial w & b for Organic House / Downtempo and energy: [-710.46
58135] -1025.2286585365853
Optimal w & b found by gradient descent for energy for Organic House / Downt
empo: [95.66352051] 138.0645991151515
Cost for Organic House / Downtempo and speechiness at initial w: 2396274.267
```

```
3216546
Gradient at initial w & b for Organic House / Downtempo and speechiness: [-5
7.81166037] -1025.2286585365853
Optimal w & b found by gradient descent for speechiness for Organic House /
Downtempo: [8.03957203] 142.83865529376598
Cost for Organic House / Downtempo and loudness at initial w: 2415980.511782
142
Gradient at initial w & b for Organic House / Downtempo and loudness: [9705.
56227962] -1025.2286585365853
Optimal w & b found by gradient descent for loudness for Organic House / Dow
ntempo: [-95.5510624] 20.478974553468245
Cost for Organic House / Downtempo and liveness at initial w: 2396190.718320
9276
Gradient at initial w & b for Organic House / Downtempo and liveness: [-99.6
5712186] -1025.2286585365853
Optimal w & b found by gradient descent for liveness for Organic House / Dow
ntempo: [13.72412802] 142.76216628503985
Cost for Organic House / Downtempo and instrumentalness at initial w: 239468
1.2133202143
Gradient at initial w & b for Organic House / Downtempo and instrumentalnes
s: [-855.80401] -1025.2286585365853
Optimal w & b found by gradient descent for instrumentalness for Organic Hou
se / Downtempo: [113.46602199] 135.94062117964387
Cost for Organic House / Downtempo and acousticness at initial w: 2396315.02
37648906
Gradient at initial w & b for Organic House / Downtempo and acousticness: [-
37.43836206] -1025.2286585365853
Optimal w & b found by gradient descent for acousticness for Organic House /
Downtempo: [5.05120229] 142.85339805730862
```

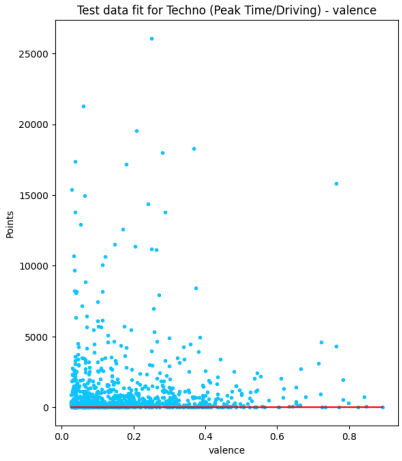




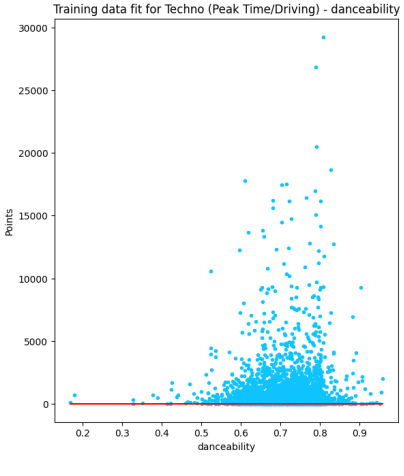

MSE: 5512328.85
RMSE: 2347.83
MAE: 1034.11
R-squared: -3613.13



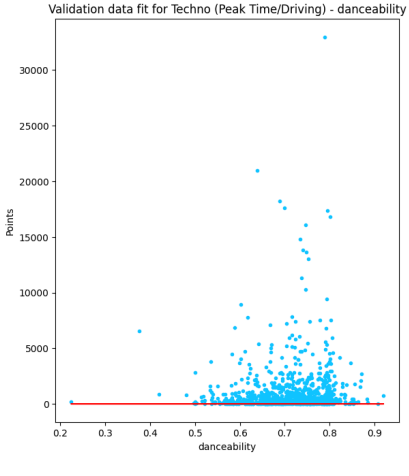

MSE: 6577144.55
RMSE: 2564.59
MAE: 1044.17
R-squared: -1167.74



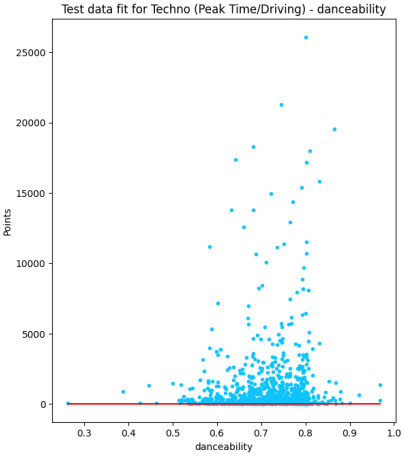

MSE: 7847608.90
RMSE: 2801.36
MAE: 1149.87
R-squared: -1164.34



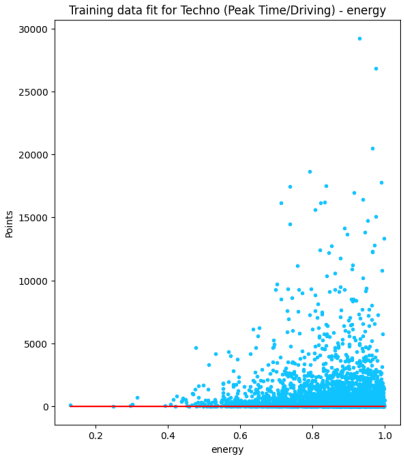

MSE: 5510156.05
RMSE: 2347.37
MAE: 1033.09
R-squared: -3611.71



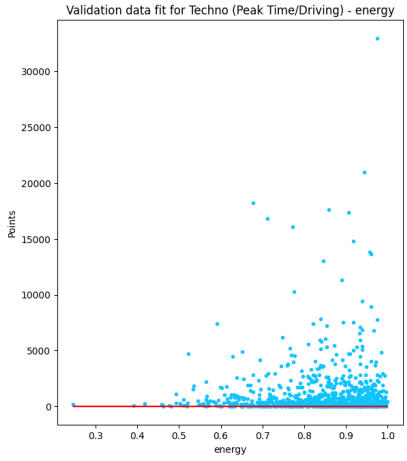

MSE: 6574943.91
RMSE: 2564.17
MAE: 1043.14
R-squared: -1167.35



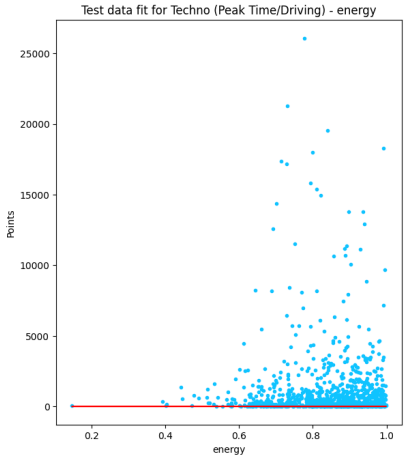

MSE: 7845159.38
RMSE: 2800.92
MAE: 1148.85
R-squared: -1163.98



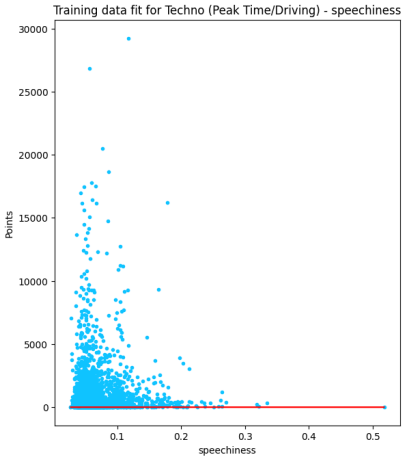

MSE: 5509638.60
RMSE: 2347.26
MAE: 1032.85
R-squared: -3611.37



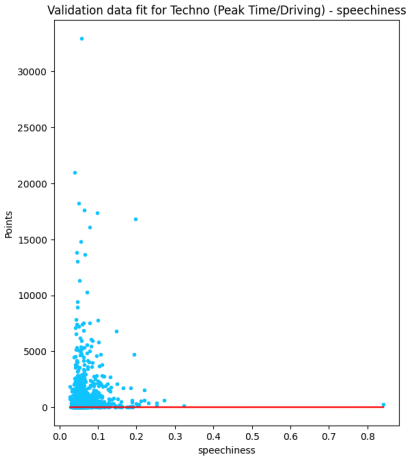

MSE: 6574412.90
RMSE: 2564.06
MAE: 1042.90
R-squared: -1167.25



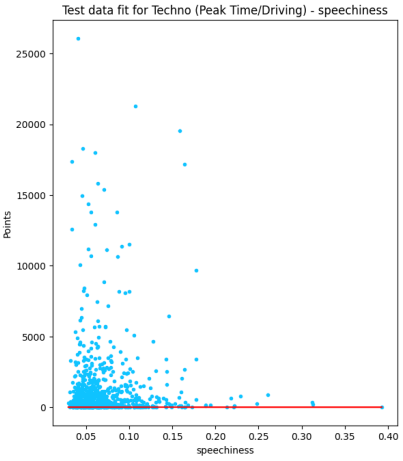

MSE: 7844605.00
RMSE: 2800.82
MAE: 1148.62
R-squared: -1163.89



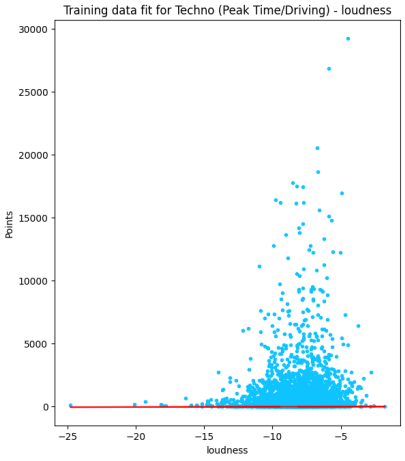

MSE: 5512833.35
RMSE: 2347.94
MAE: 1034.35
R-squared: -3613.46



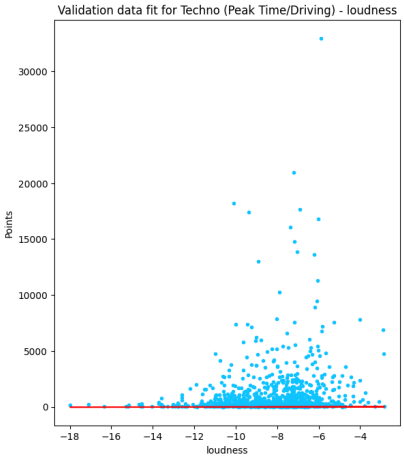

MSE: 6577639.20
RMSE: 2564.69
MAE: 1044.40
R-squared: -1167.83



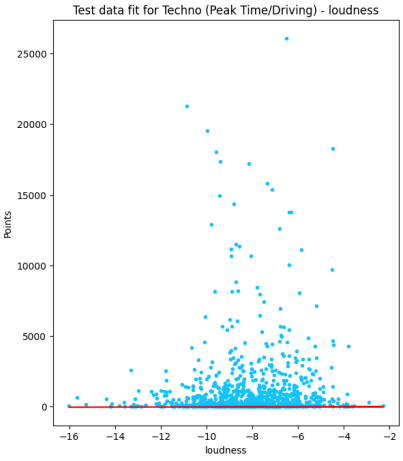

MSE: 7848147.88
RMSE: 2801.45
MAE: 1150.10
R-squared: -1164.42



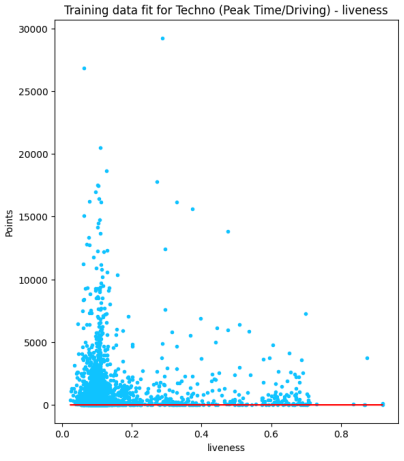

MSE: 5547219.21
RMSE: 2355.25
MAE: 1050.83
R-squared: -3636.01



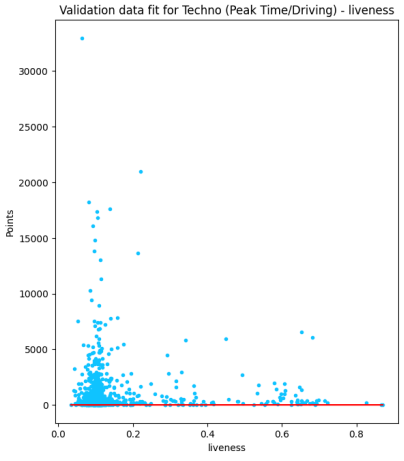

MSE: 6612719.66
RMSE: 2571.52
MAE: 1061.06
R-squared: -1174.06



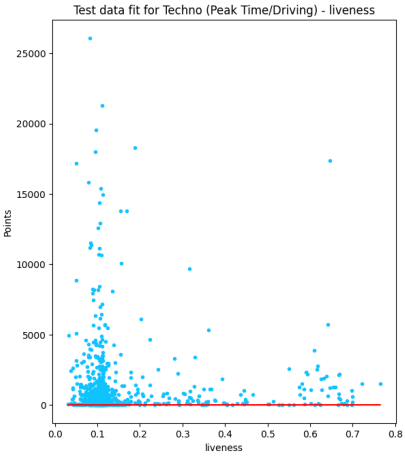

MSE: 7886225.38
RMSE: 2808.24
MAE: 1166.53
R-squared: -1170.07



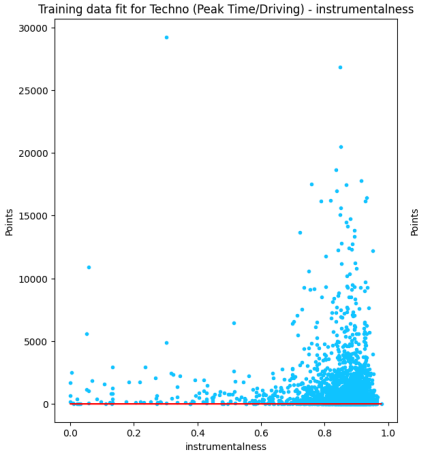

MSE: 5512517.17
RMSE: 2347.88
MAE: 1034.20
R-squared: -3613.25



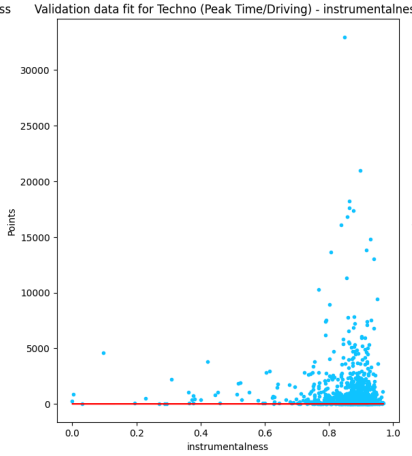

MSE: 6577301.88
RMSE: 2564.63
MAE: 1044.25
R-squared: -1167.77



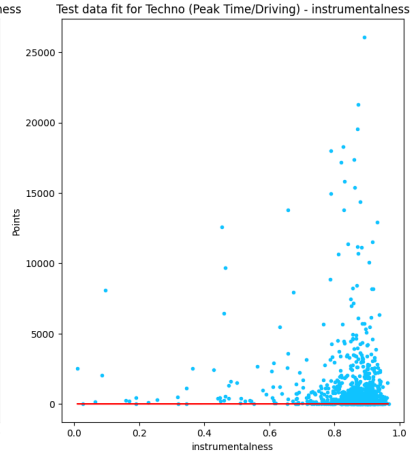

MSE: 7847777.10
RMSE: 2801.39
MAE: 1149.95
R-squared: -1164.36



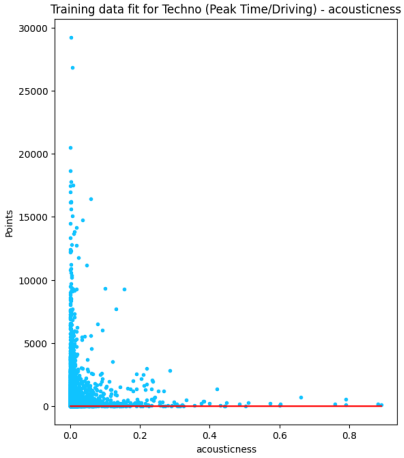

MSE: 5509615.72
RMSE: 2347.26
MAE: 1032.84
R-squared: -3611.35



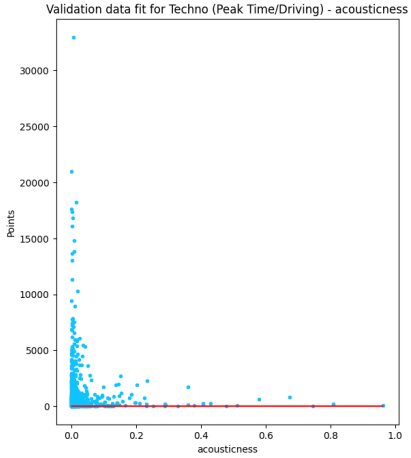

MSE: 6574357.26
RMSE: 2564.05
MAE: 1042.87
R-squared: -1167.24



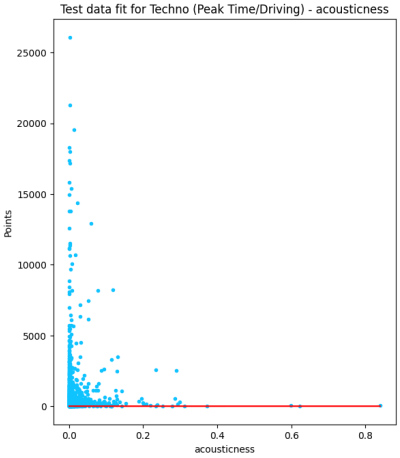

MSE: 7844574.02
RMSE: 2800.82
MAE: 1148.61
R-squared: -1163.89



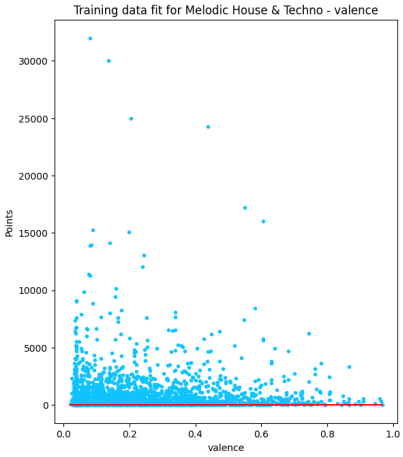

MSE: 5513020.78
RMSE: 2347.98
MAE: 1034.44
R-squared: -3613.58



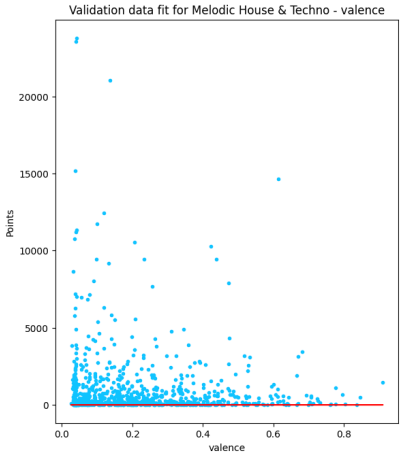

MSE: 6577820.47
RMSE: 2564.73
MAE: 1044.49
R-squared: -1167.86



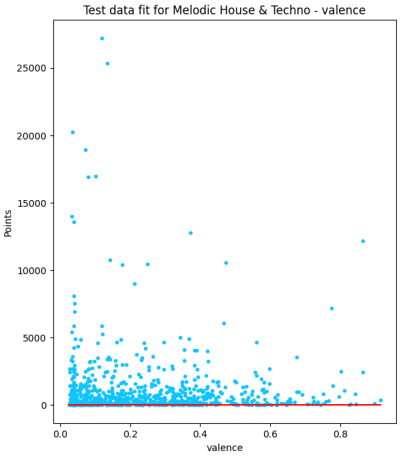

MSE: 7848376.08
RMSE: 2801.50
MAE: 1150.20
R-squared: -1164.45



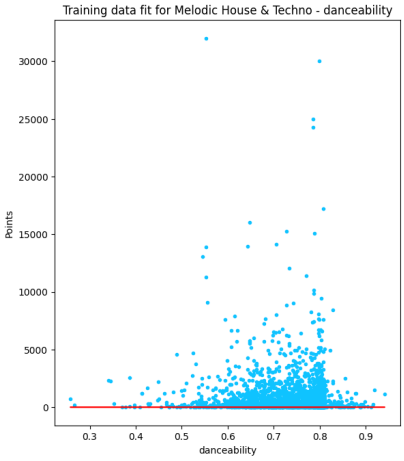

MSE: 4417308.01
RMSE: 2101.74
MAE: 896.80
R-squared: -2973.56



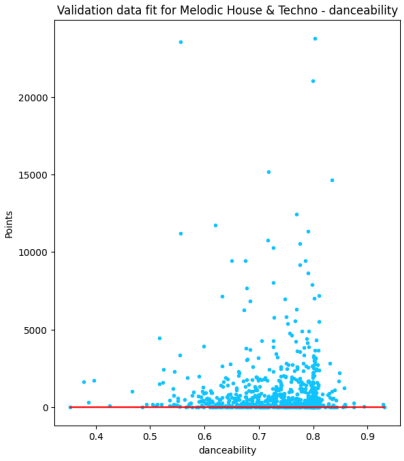

MSE: 6135673.99
RMSE: 2477.03
MAE: 1048.64
R-squared: -999.27



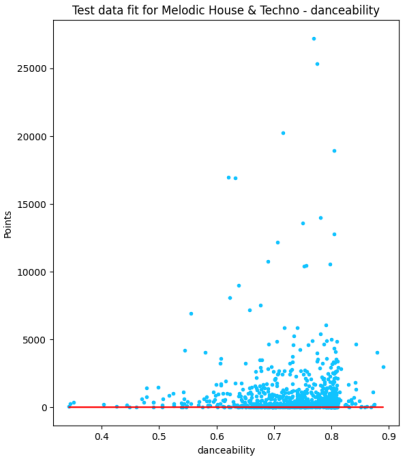

MSE: 6653576.66
RMSE: 2579.45
MAE: 1033.48
R-squared: -976.99



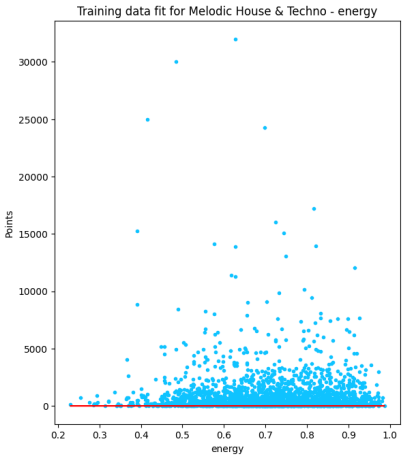

MSE: 4415520.80
RMSE: 2101.31
MAE: 895.84
R-squared: -2972.36



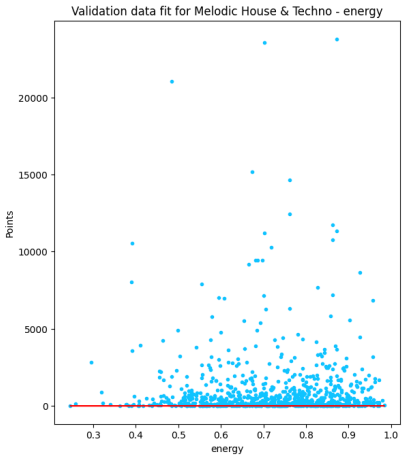

MSE: 6133544.19
RMSE: 2476.60
MAE: 1047.64
R-squared: -998.92



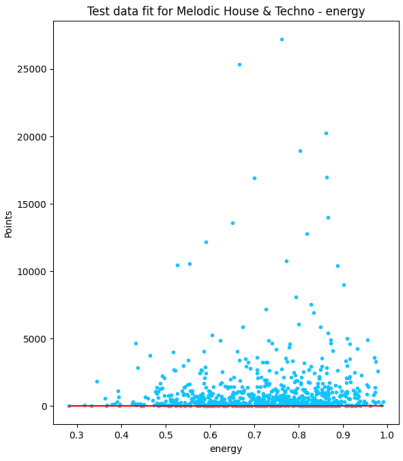

MSE: 6651540.60
RMSE: 2579.06
MAE: 1032.52
R-squared: -976.69



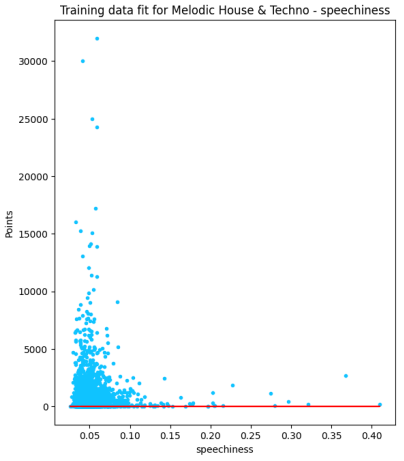

MSE: 4415581.50
RMSE: 2101.33
MAE: 895.87
R-squared: -2972.40



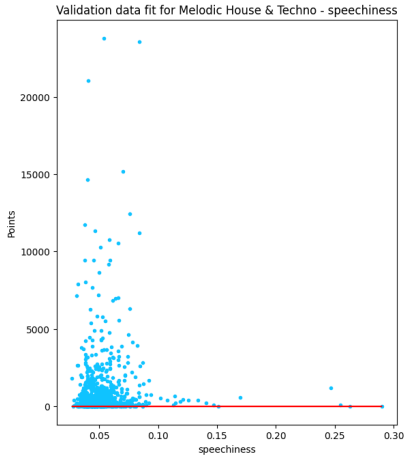

MSE: 6133599.69
RMSE: 2476.61
MAE: 1047.67
R-squared: -998.93



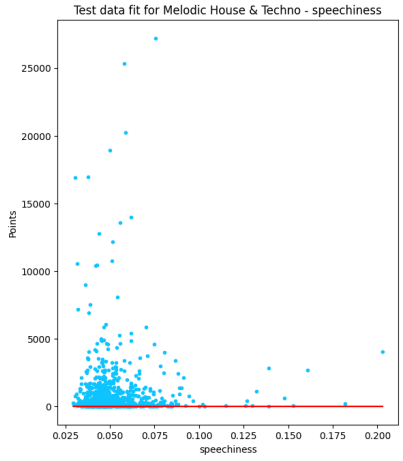

MSE: 6651577.28
RMSE: 2579.07
MAE: 1032.54
R-squared: -976.70



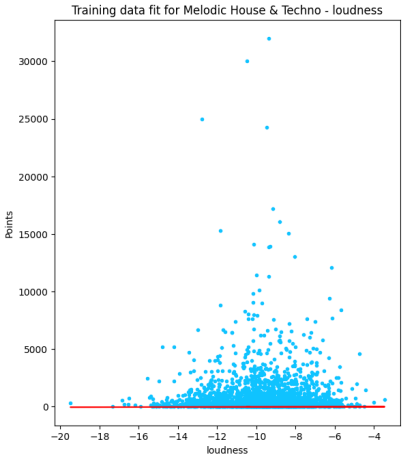

MSE: 4417947.03
RMSE: 2101.89
MAE: 897.15
R-squared: -2974.00



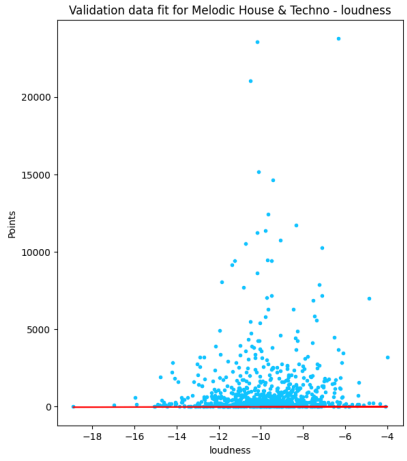

MSE: 6136380.32
RMSE: 2477.17
MAE: 1048.97
R-squared: -999.38



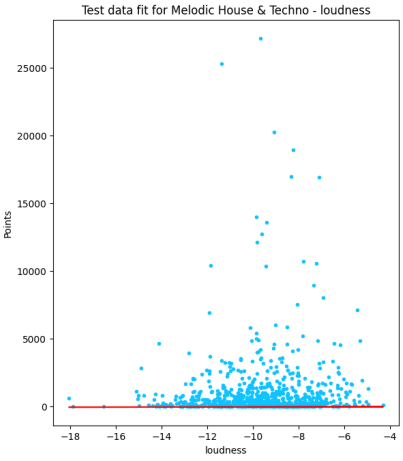

MSE: 6654347.23
RMSE: 2579.60
MAE: 1033.85
R-squared: -977.11



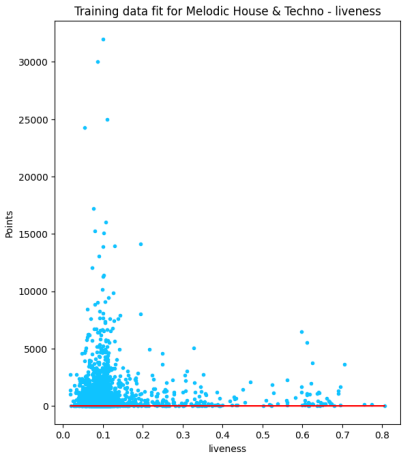

MSE: 4453501.12
RMSE: 2110.33
MAE: 916.74
R-squared: -2997.94



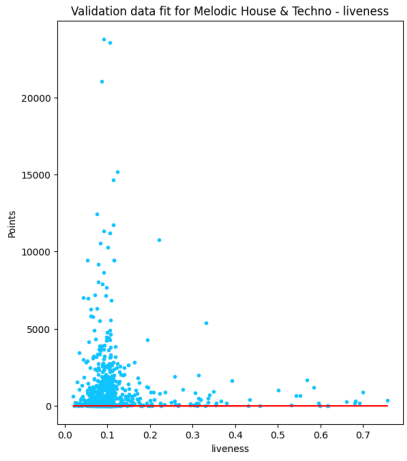

MSE: 6177896.00
RMSE: 2485.54
MAE: 1068.57
R-squared: -1006.15



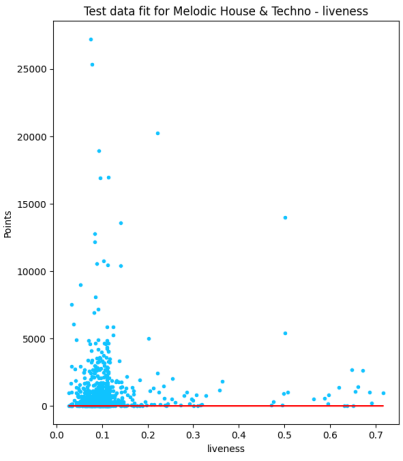

MSE: 6694713.15
RMSE: 2587.41
MAE: 1053.18
R-squared: -983.04



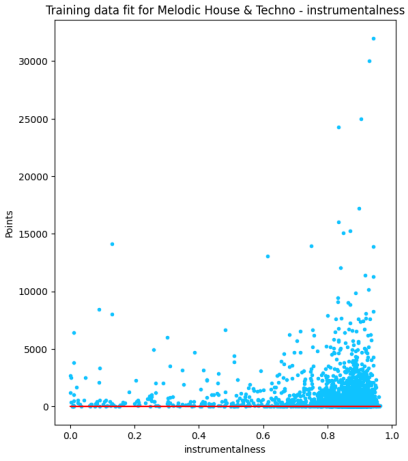

MSE: 4417702.65
RMSE: 2101.83
MAE: 897.02
R-squared: -2973.83



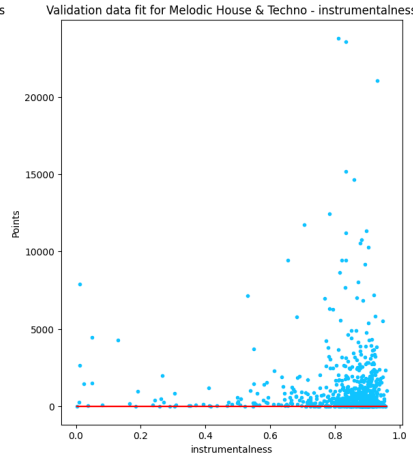

MSE: 6136108.57
RMSE: 2477.12
MAE: 1048.84
R-squared: -999.34



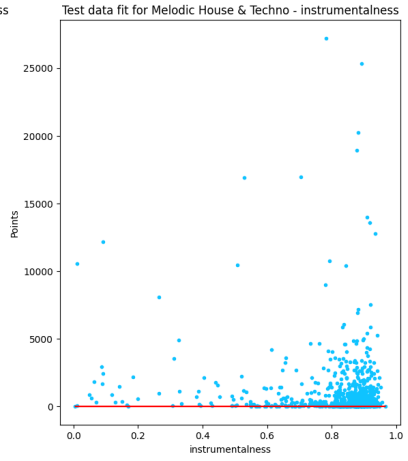

MSE: 6654088.42
RMSE: 2579.55
MAE: 1033.72
R-squared: -977.07



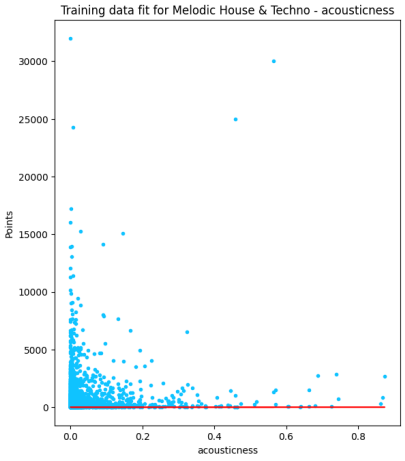

MSE: 4415222.76
RMSE: 2101.24
MAE: 895.68
R-squared: -2972.16



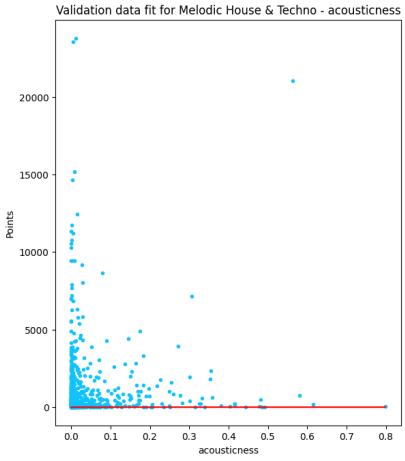

MSE: 6133141.05
RMSE: 2476.52
MAE: 1047.46
R-squared: -998.85



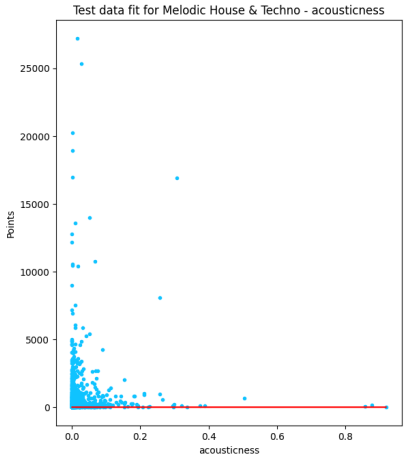

MSE: 6651182.35
RMSE: 2578.99
MAE: 1032.36
R-squared: -976.64



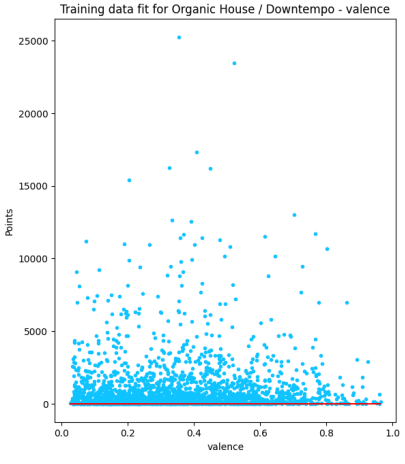

MSE: 4418007.98
RMSE: 2101.91
MAE: 897.18
R-squared: -2974.04



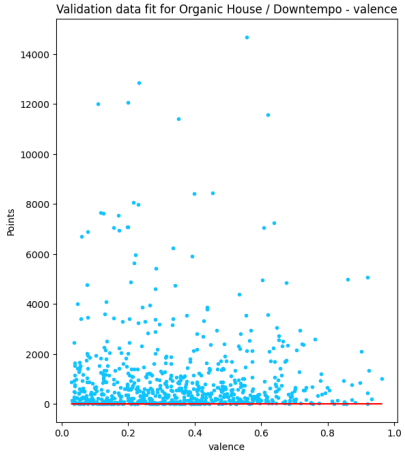

MSE: 6136447.49
RMSE: 2477.19
MAE: 1049.00
R-squared: -999.39



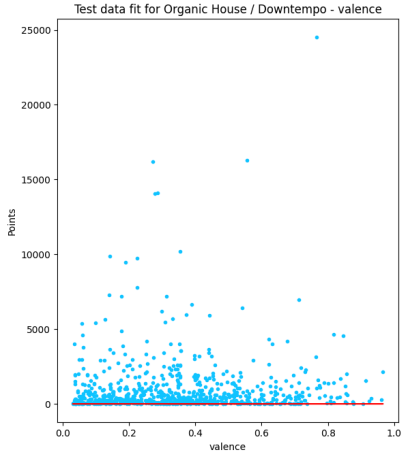

MSE: 6654449.18
RMSE: 2579.62
MAE: 1033.89
R-squared: -977.12



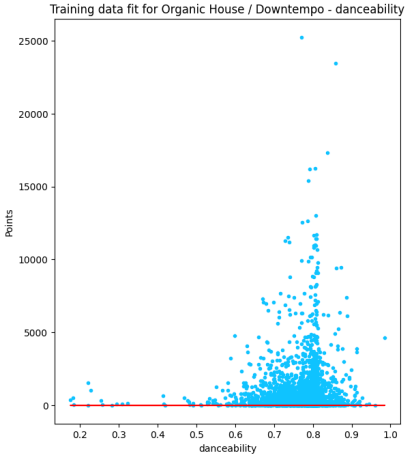

MSE: 4791330.73
RMSE: 2188.91
MAE: 1023.54
R-squared: -2937.48



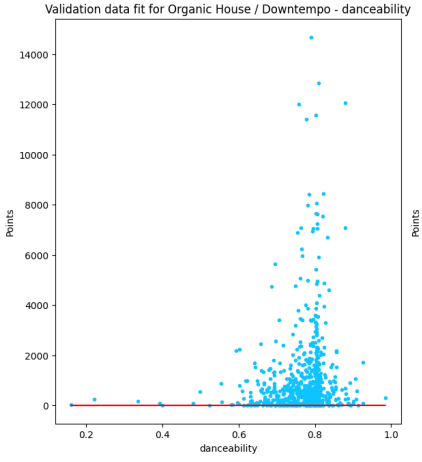

MSE: 3931608.58
RMSE: 1982.83
MAE: 981.07
R-squared: -998.74



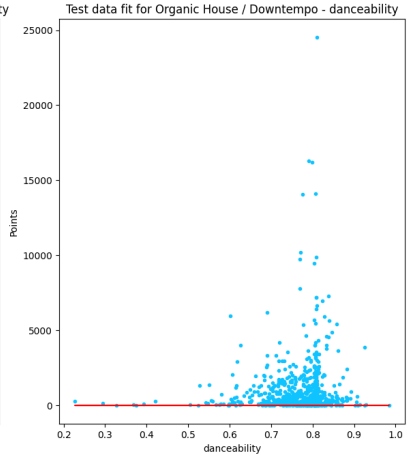

MSE: 4562511.82
RMSE: 2136.00
MAE: 967.63
R-squared: -962.78



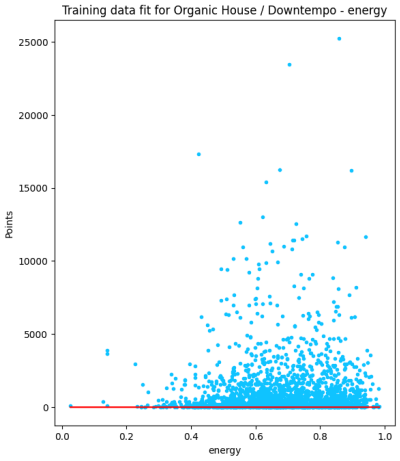

MSE: 4789638.79
RMSE: 2188.52
MAE: 1022.74
R-squared: -2936.44



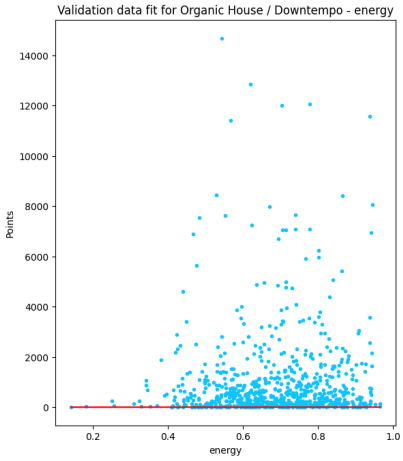

MSE: 3929964.70
RMSE: 1982.41
MAE: 980.25
R-squared: -998.32



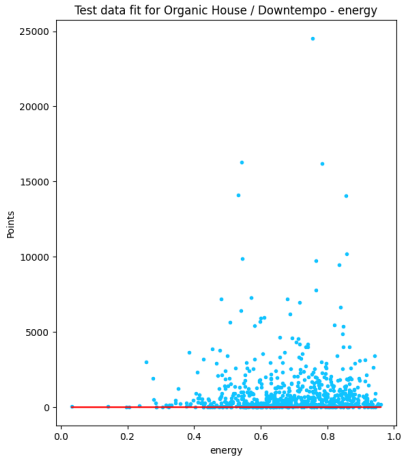

MSE: 4560892.21
RMSE: 2135.62
MAE: 966.81
R-squared: -962.44



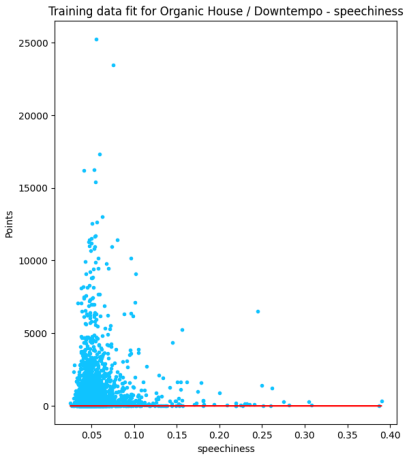

MSE: 4789981.07
RMSE: 2188.60
MAE: 1022.90
R-squared: -2936.65



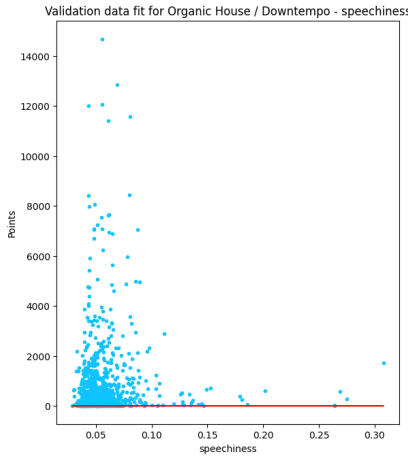

MSE: 3930298.86
RMSE: 1982.50
MAE: 980.42
R-squared: -998.41



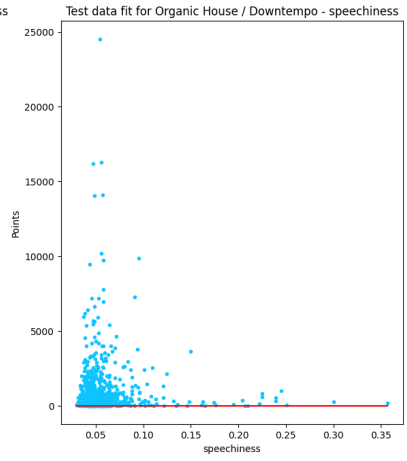

MSE: 4561223.28
RMSE: 2135.70
MAE: 966.97
R-squared: -962.51



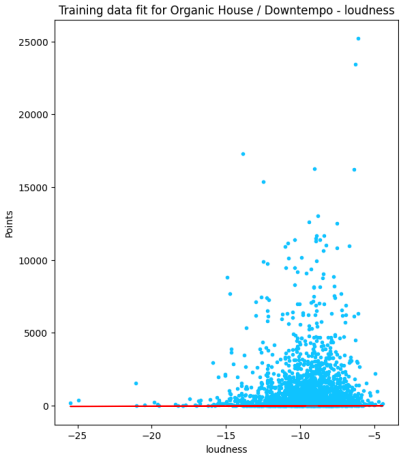

MSE: 4792543.19
RMSE: 2189.19
MAE: 1024.12
R-squared: -2938.22



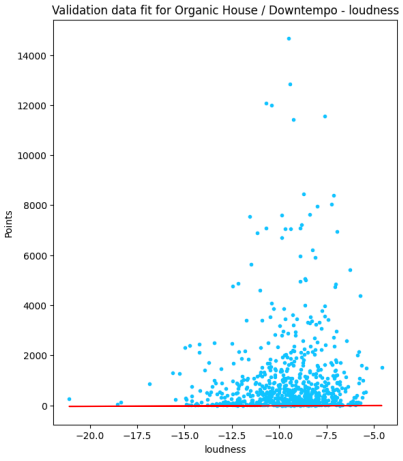

MSE: 3932754.09
RMSE: 1983.12
MAE: 981.64
R-squared: -999.03



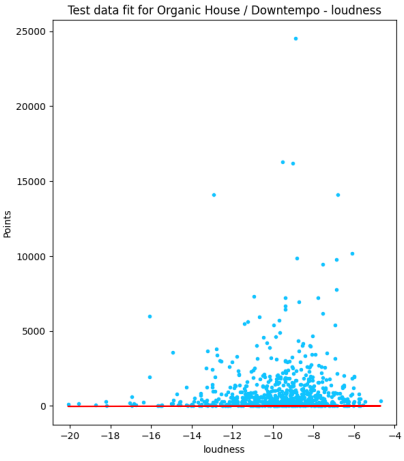

MSE: 4563630.81
RMSE: 2136.27
MAE: 968.20
R-squared: -963.02



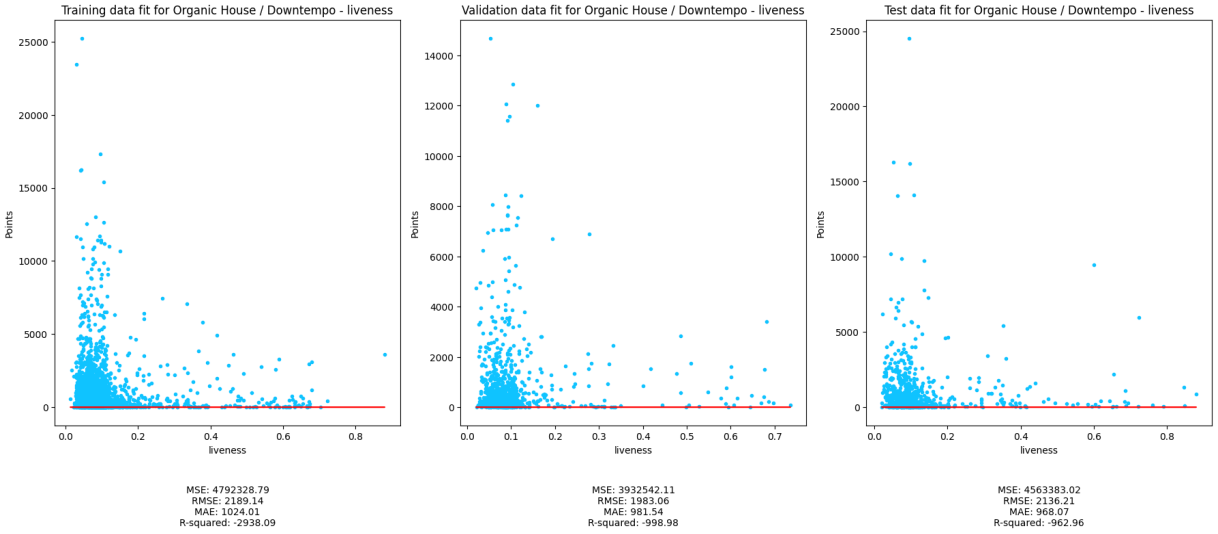

MSE: 4833121.02
RMSE: 2198.44
MAE: 1043.73
R-squared: -2963.11

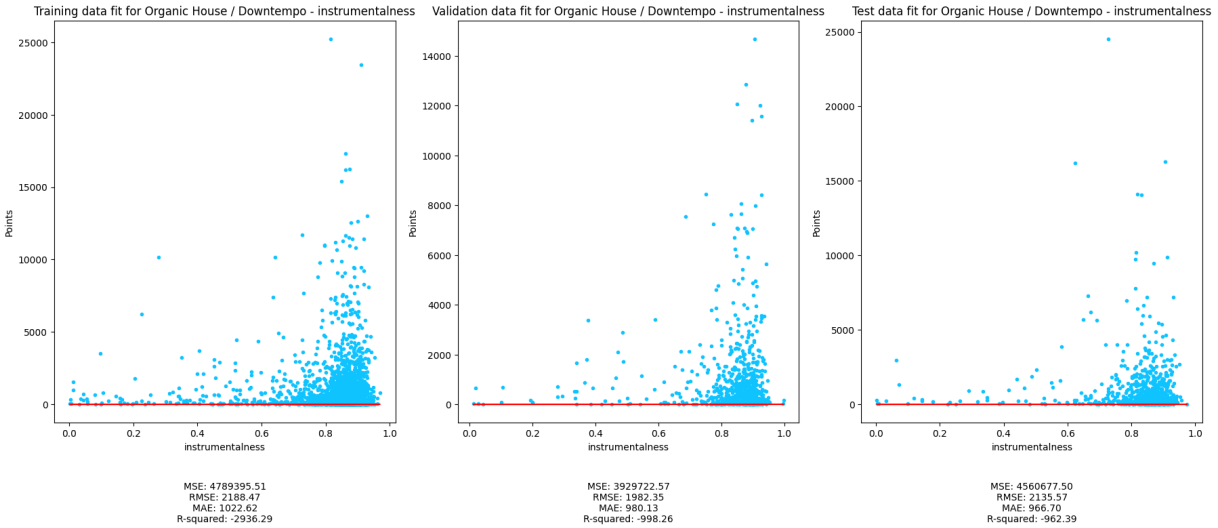


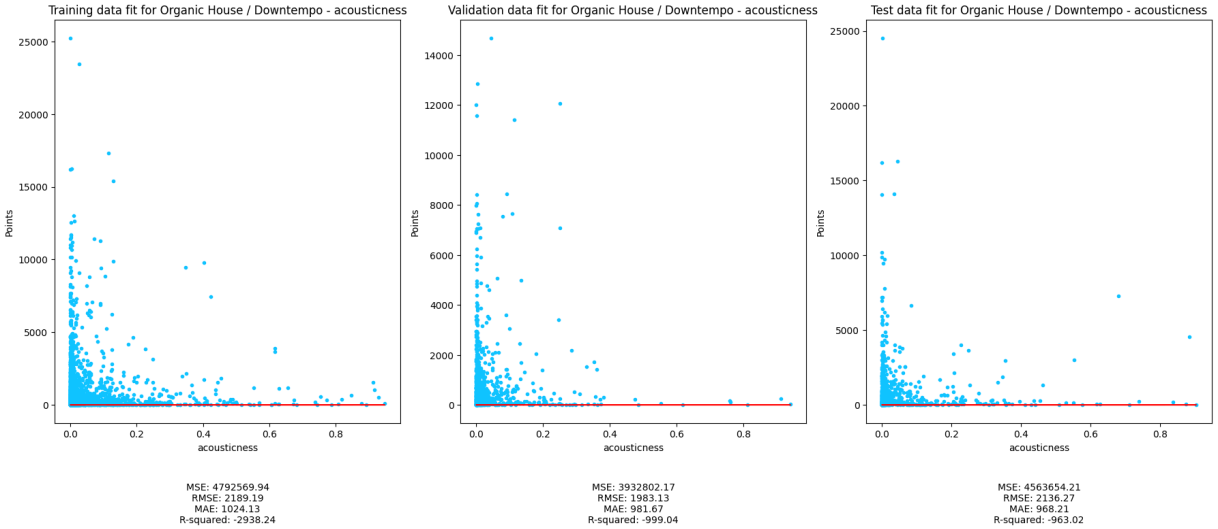

MSE: 3971114.46
RMSE: 1992.77
MAE: 1000.98
R-squared: -1008.79




MSE: 4602142.02
RMSE: 2145.26
MAE: 987.87
R-squared: -971.15



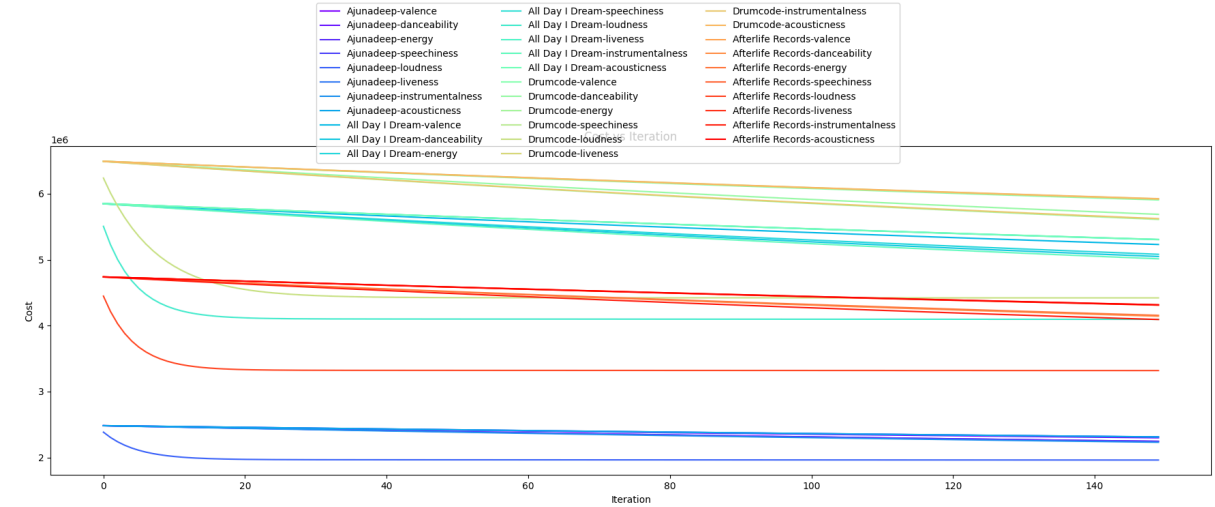

## Label function

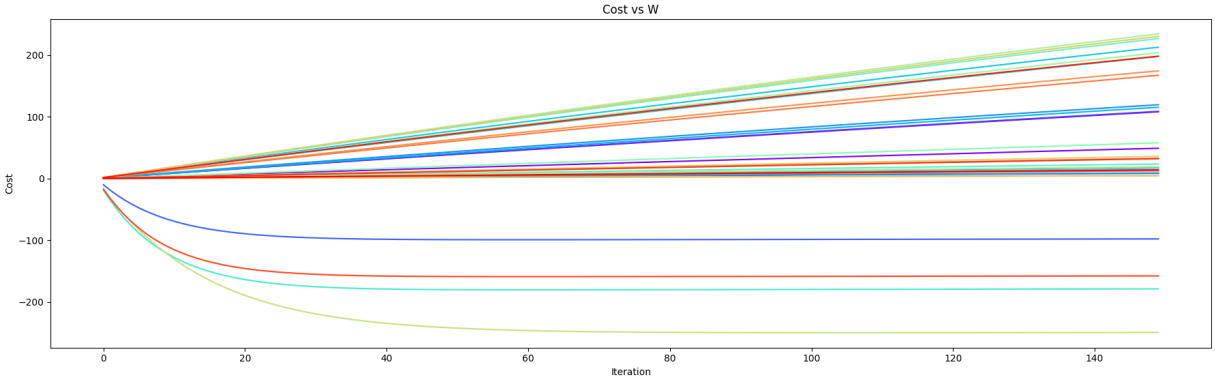
In [33]:
```
run_all_calculations_label(log_train, log_validate, log_test)
```

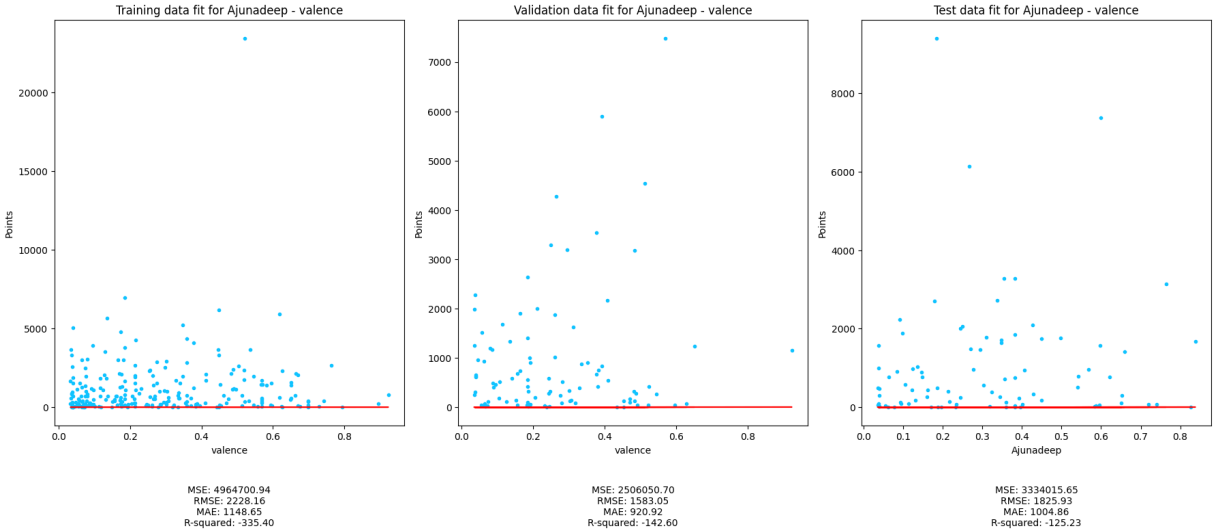
```
Cost for Ajunadeep and valence at initial w: 2482291.6808831836
Gradient at initial w & b for Ajunadeep and valence: [-352.66906721] -1150.2
024291497976
Optimal w & b found by gradient descent for Ajunadeep and valence [49.001058
51] 159.28617218380964
Cost for Ajunadeep and danceability at initial w: 2481404.08909744
Gradient at initial w & b for Ajunadeep and danceability: [-797.15543198] -1
150.2024291497976
Optimal w & b found by gradient descent for Ajunadeep and danceability [108.
05625785] 155.1353003301318
Cost for Ajunadeep and energy at initial w: 2481381.2213108004
Gradient at initial w & b for Ajunadeep and energy: [-808.69391174] -1150.20
24291497976
Optimal w & b found by gradient descent for Ajunadeep and energy [108.741838
82] 154.76336978533809
Cost for Ajunadeep and speechiness at initial w: 2482876.6673188126
Gradient at initial w & b for Ajunadeep and speechiness: [-59.82435304] -115
0.2024291497976
Optimal w & b found by gradient descent for Ajunadeep and speechiness [8.364
71286] 160.25830405665096
Cost for Ajunadeep and loudness at initial w: 2503565.7591824373
Gradient at initial w & b for Ajunadeep and loudness: [10193.61417814] -115
0.2024291497976
Optimal w & b found by gradient descent for Ajunadeep and loudness [-97.9269
1721] 36.82684514176162
Cost for Ajunadeep and liveness at initial w: 2482730.4306121147
Gradient at initial w & b for Ajunadeep and liveness: [-133.0632996] -1150.2
024291497976
Optimal w & b found by gradient descent for Ajunadeep and liveness [18.20825
202] 160.10335945056426
Cost for Ajunadeep and instrumentalness at initial w: 2481205.6847204524
Gradient at initial w & b for Ajunadeep and instrumentalness: [-896.6621707
1] -1150.2024291497976
Optimal w & b found by gradient descent for Ajunadeep and instrumentalness
[119.50439011] 153.59124022788768
Cost for Ajunadeep and acousticness at initial w: 2482858.1909034043
Gradient at initial w & b for Ajunadeep and acousticness: [-69.19695734] -11
50.2024291497976
Optimal w & b found by gradient descent for Ajunadeep and acousticness [8.91
513289] 160.21148335136579
Cost for All Day I Dream and valence at initial w: 5851049.324015055
Gradient at initial w & b for All Day I Dream and valence: [-835.13090483] -
2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and valence [11
5.45702791] 283.1273496567125
Cost for All Day I Dream and danceability at initial w: 5849540.390361996
Gradient at initial w & b for All Day I Dream and danceability: [-1590.39049
758] -2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and danceability
[212.63462531] 274.3721223115662
Cost for All Day I Dream and energy at initial w: 5849799.181401683
Gradient at initial w & b for All Day I Dream and energy: [-1460.78399034] -
2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and energy [197.
61443785] 276.60374916610584
Cost for All Day I Dream and speechiness at initial w: 5852497.004969041
```

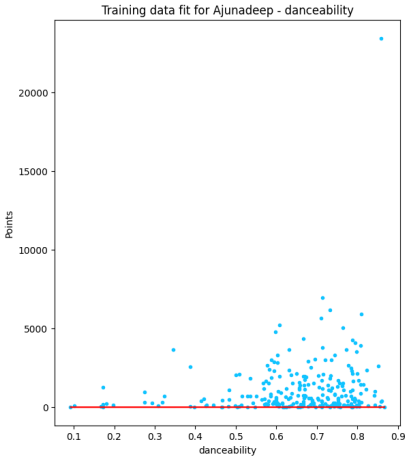
```
Gradient at initial w & b for All Day I Dream and speechiness: [-110.7934009
7] -2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and speechiness
[15.38982726] 286.24595744771057
Cost for All Day I Dream and loudness at initial w: 5891044.848144917
Gradient at initial w & b for All Day I Dream and loudness: [19069.23218841]
-2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and loudness [-1
78.81637766] 49.95920340860521
Cost for All Day I Dream and liveness at initial w: 5852377.780963319
Gradient at initial w & b for All Day I Dream and liveness: [-170.46548792]
-2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and liveness [2
3.35480356] 286.1387501807525
Cost for All Day I Dream and instrumentalness at initial w: 5849293.88918285
8
Gradient at initial w & b for All Day I Dream and instrumentalness: [-1713.8
3855024] -2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and instrumental
ness [226.81443464] 272.3362070885108
Cost for All Day I Dream and acousticness at initial w: 5852648.511361868
Gradient at initial w & b for All Day I Dream and acousticness: [-35.0321000
2] -2054.5024154589373
Optimal w & b found by gradient descent for All Day I Dream and acousticness
[4.47349533] 286.2966325324307
Cost for Drumcode and valence at initial w: 6494744.2484447695
Gradient at initial w & b for Drumcode and valence: [-414.6785805] -2101.35
Optimal w & b found by gradient descent for Drumcode and valence [57.7667961
9] 292.0561599749995
Cost for Drumcode and danceability at initial w: 6492542.452783395
Gradient at initial w & b for Drumcode and danceability: [-1516.563475] -210
1.35
Optimal w & b found by gradient descent for Drumcode and danceability [203.6
4992923] 282.1046251206058
Cost for Drumcode and energy at initial w: 6492041.367747831
Gradient at initial w & b for Drumcode and energy: [-1767.406265] -2101.35
Optimal w & b found by gradient descent for Drumcode and energy [234.2772097
7] 278.4559111247335
Cost for Drumcode and speechiness at initial w: 6495300.438388205
Gradient at initial w & b for Drumcode and speechiness: [-136.405918] -2101.
35
Optimal w & b found by gradient descent for Drumcode and speechiness [19.015
1876] 292.7484285311374
Cost for Drumcode and loudness at initial w: 6528157.434658233
Gradient at initial w & b for Drumcode and loudness: [16236.31015] -2101.35
Optimal w & b found by gradient descent for Drumcode and loudness [-249.5231
7467] 47.690018031861094
Cost for Drumcode and liveness at initial w: 6495051.695839176
Gradient at initial w & b for Drumcode and liveness: [-260.8628145] -2101.35
Optimal w & b found by gradient descent for Drumcode and liveness [36.164274
08] 292.49710713372076
Cost for Drumcode and instrumentalness at initial w: 6492089.488228487
Gradient at initial w & b for Drumcode and instrumentalness: [-1743.376086]
-2101.35
Optimal w & b found by gradient descent for Drumcode and instrumentalness [2
30.30131988] 278.5303511718433
```

```
Cost for Drumcode and acousticness at initial w: 6495506.212201014
Gradient at initial w & b for Drumcode and acousticness: [-33.48171919] -210
1.35
Optimal w & b found by gradient descent for Drumcode and acousticness [4.507
19497] 292.82945662098797
Cost for Afterlife Records and valence at initial w: 4740450.4905429445
Gradient at initial w & b for Afterlife Records and valence: [-240.79912573]
-1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and valence [3
3.03262327] 252.23881641041731
Cost for Afterlife Records and danceability at initial w: 4738339.496426246
Gradient at initial w & b for Afterlife Records and danceability: [-1297.314
46784] -1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and danceabili
ty [174.40567462] 243.55995975605907
Cost for Afterlife Records and energy at initial w: 4738454.573225952
Gradient at initial w & b for Afterlife Records and energy: [-1239.70226901]
-1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and energy [16
7.19209824] 244.38089351734246
Cost for Afterlife Records and speechiness at initial w: 4740752.044420955
Gradient at initial w & b for Afterlife Records and speechiness: [-89.879604
68] -1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and speechines
s [12.51059751] 252.56566183310738
Cost for Afterlife Records and loudness at initial w: 4776271.8581066765
Gradient at initial w & b for Afterlife Records and loudness: [17571.4294035
1] -1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and loudness
[-157.85479312] 39.19596640167284
Cost for Afterlife Records and liveness at initial w: 4740469.712163823
Gradient at initial w & b for Afterlife Records and liveness: [-231.1499625
7] -1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and liveness
[32.1022336] 252.30910089370715
Cost for Afterlife Records and instrumentalness at initial w: 4737938.645842
023
Gradient at initial w & b for Afterlife Records and instrumentalness: [-149
8.06740936] -1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and instrument
alness [198.18649451] 240.4099876813843
Cost for Afterlife Records and acousticness at initial w: 4740729.569170245
Gradient at initial w & b for Afterlife Records and acousticness: [-101.1601
954] -1812.6959064327486
Optimal w & b found by gradient descent for Afterlife Records and acousticne
ss [13.76737368] 252.53916854242638
```

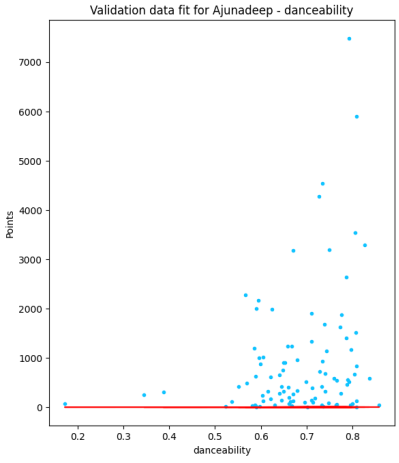

Ajunadeep-valence
Ajunadeep-danceability
Ajunadeep-energy
Ajunadeep-speechiness
Ajunadeep-loudness
Ajunadeep-liveness
Ajunadeep-instrumentalness
Ajunadeep-acousticness
All Day I Dream-valence
All Day I Dream-danceability
All Day I Dream-energy
All Day I Dream-speechiness
All Day I Dream-loudness
All Day I Dream-liveness
All Day I Dream-instrumentalness
All Day I Dream-acousticness
Drumcode-valence
Drumcode-danceability
Drumcode-energy
Drumcode-speechiness
Drumcode-loudness
Drumcode-liveness
Drumcode-instrumentalness
Drumcode-acousticness
Afterlife Records-valence
Afterlife Records-danceability
Afterlife Records-energy
Afterlife Records-speechiness
Afterlife Records-loudness
Afterlife Records-liveness
Afterlife Records-instrumentalness
Afterlife Records-acousticness
1e6
Cost
Iteration


Cost vs W
Cost
Iteration

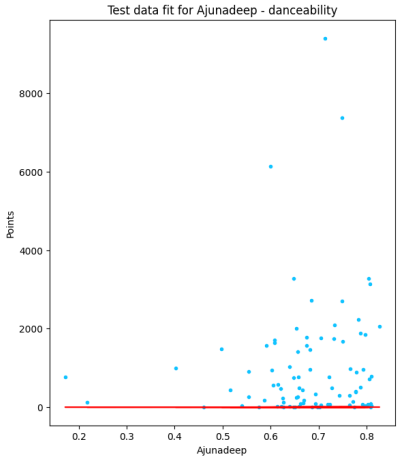

Training data fit for Ajunadeep - valence
Points
valence
MSE: 4964700.94
RMSE: 2228.16
MAE: 1148.65
R-squared: -335.40
Validation data fit for Ajunadeep - valence
Points
valence
MSE: 2506050.70
RMSE: 1583.05
MAE: 920.92
R-squared: -142.60
Test data fit for Ajunadeep - valence
Points
Ajunadeep
MSE: 3334015.65
RMSE: 1825.93
MAE: 1004.86
R-squared: -125.23



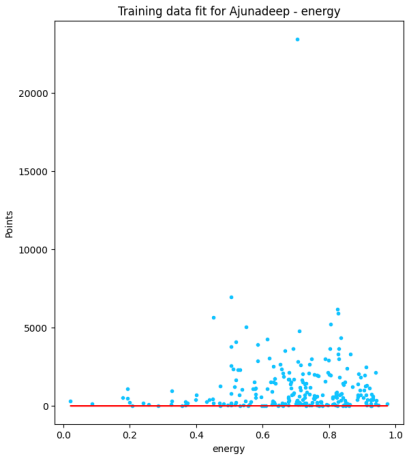

MSE: 4963006.00
RMSE: 2227.78
MAE: 1147.93
R-squared: -335.28



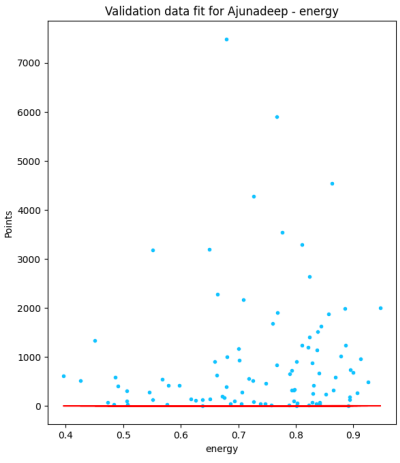

MSE: 2504495.27
RMSE: 1582.56
MAE: 920.08
R-squared: -142.51



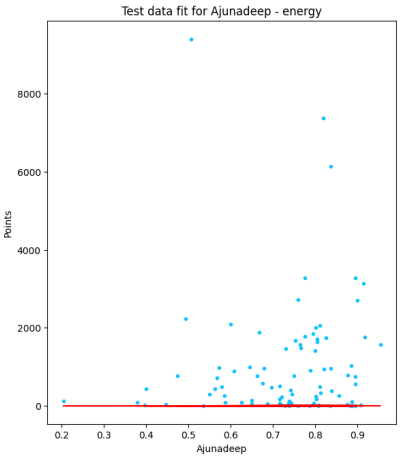

MSE: 3332539.60
RMSE: 1825.52
MAE: 1004.16
R-squared: -125.17



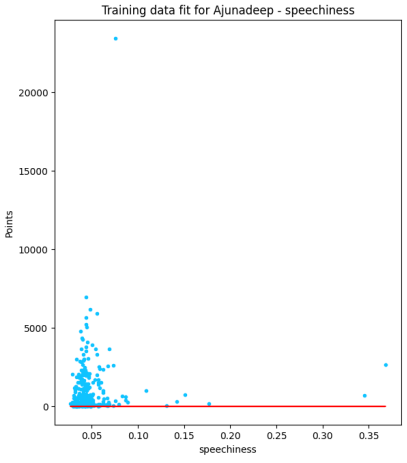

MSE: 4962819.34
RMSE: 2227.74
MAE: 1147.86
R-squared: -335.27



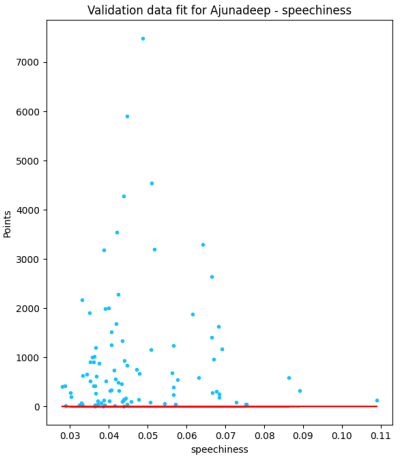

MSE: 2504308.96
RMSE: 1582.50
MAE: 919.97
R-squared: -142.50



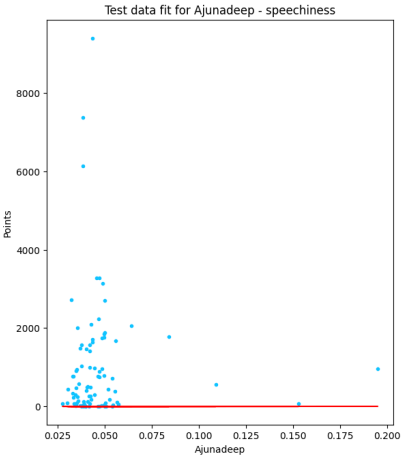

MSE: 3332325.17
RMSE: 1825.47
MAE: 1004.06
R-squared: -125.16



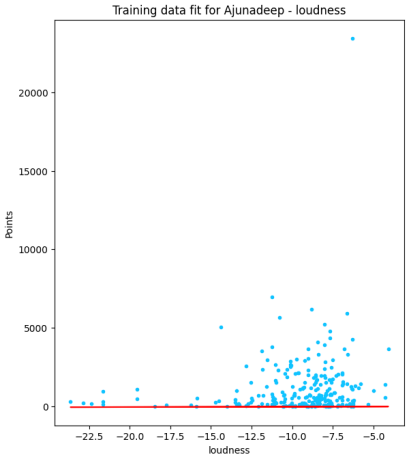

MSE: 4965765.27
RMSE: 2228.40
MAE: 1149.11
R-squared: -335.47



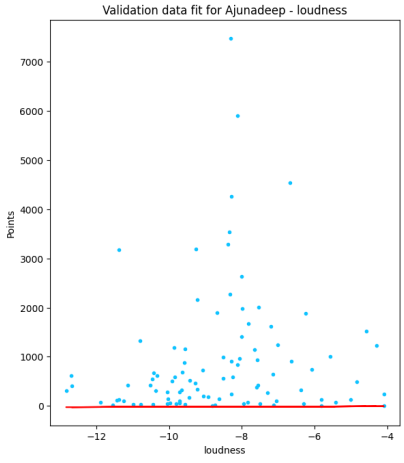

MSE: 2506837.03
RMSE: 1583.30
MAE: 921.35
R-squared: -142.64



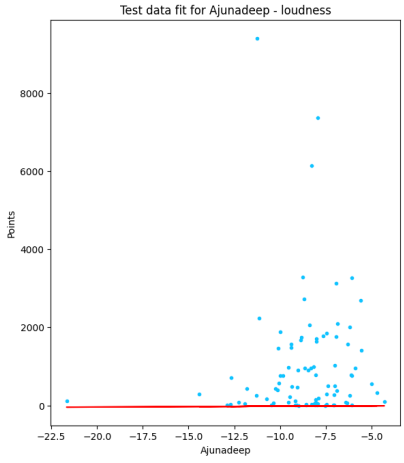

MSE: 3335069.29
RMSE: 1826.22
MAE: 1005.36
R-squared: -125.27



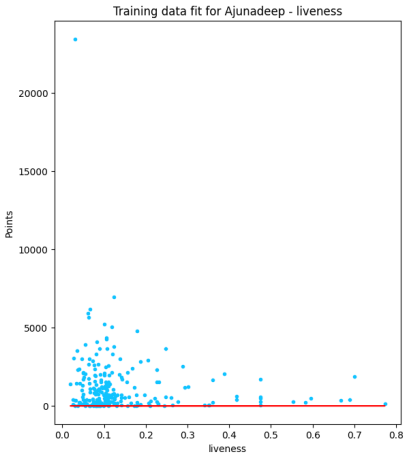

MSE: 5010175.14
RMSE: 2238.34
MAE: 1168.25
R-squared: -338.48



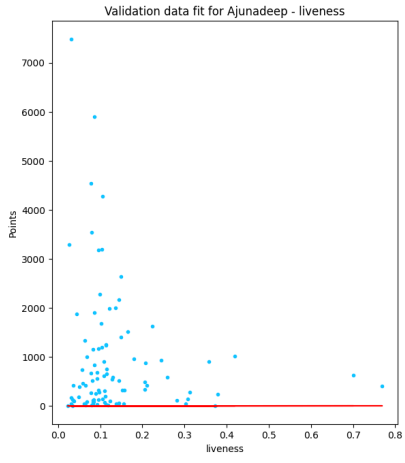

MSE: 2539106.10
RMSE: 1593.46
MAE: 938.69
R-squared: -144.49



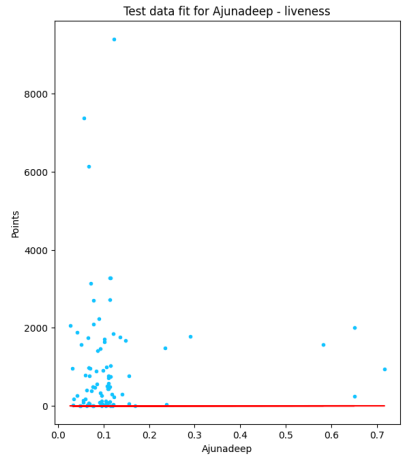

MSE: 3370242.09
RMSE: 1835.82
MAE: 1022.69
R-squared: -126.60



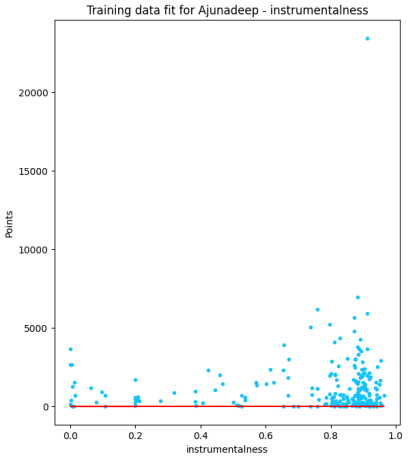

MSE: 4965354.19
RMSE: 2228.31
MAE: 1148.93
R-squared: -335.44



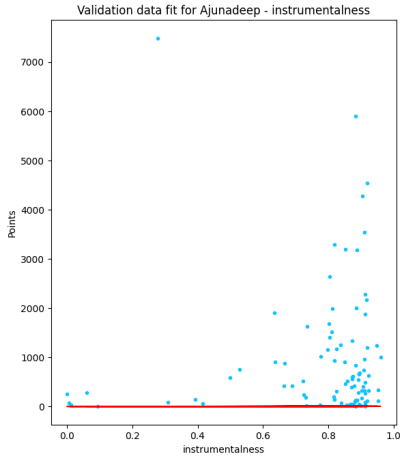

MSE: 2506503.18
RMSE: 1583.19
MAE: 921.17
R-squared: -142.62



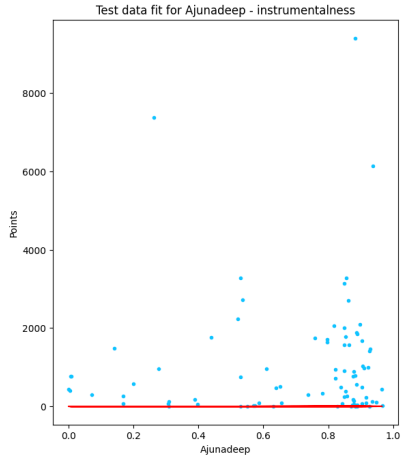

MSE: 3334767.58
RMSE: 1826.13
MAE: 1005.22
R-squared: -125.26



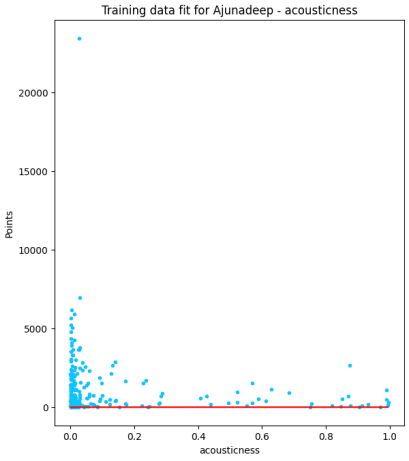

MSE: 4962503.13
RMSE: 2227.67
MAE: 1147.73
R-squared: -335.25



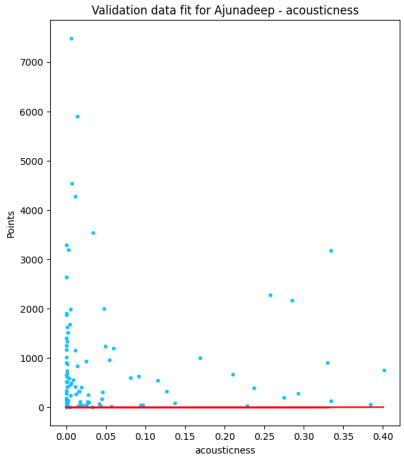

MSE: 2504124.03
RMSE: 1582.44
MAE: 919.87
R-squared: -142.49



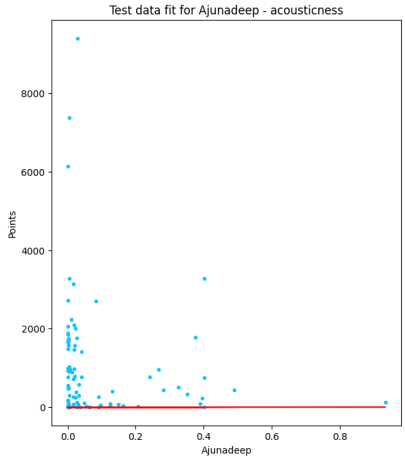

MSE: 3332454.15
RMSE: 1825.50
MAE: 1004.12
R-squared: -125.17



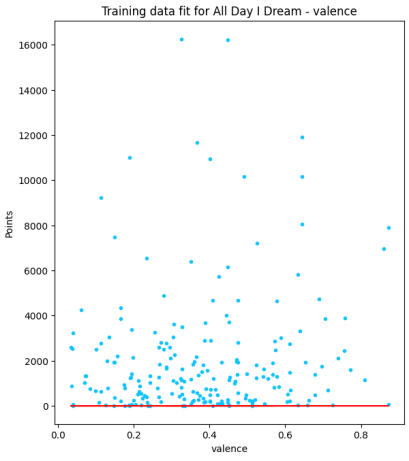

MSE: 4965462.31
RMSE: 2228.33
MAE: 1148.98
R-squared: -335.45



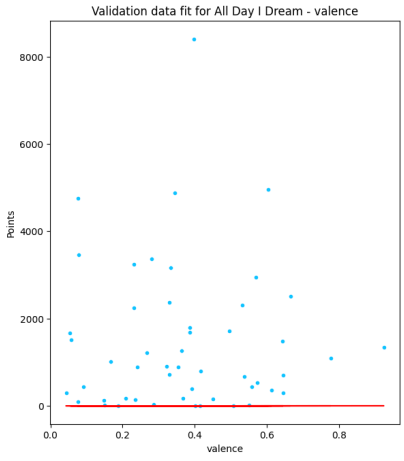

MSE: 2506799.74
RMSE: 1583.29
MAE: 921.33
R-squared: -142.64



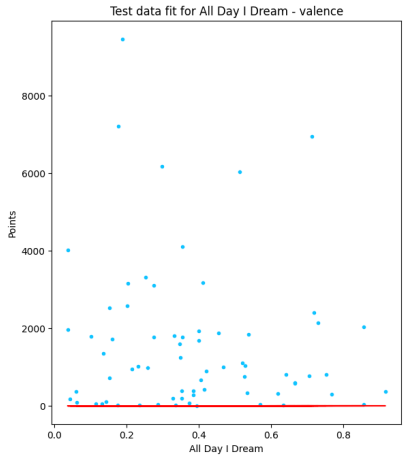

MSE: 3334918.36
RMSE: 1826.18
MAE: 1005.29
R-squared: -125.26



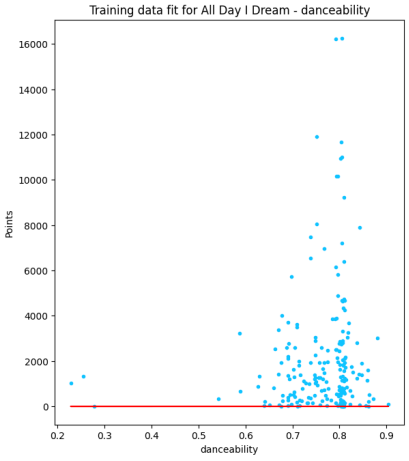

MSE: 11702332.58
RMSE: 3420.87
MAE: 2052.75
R-squared: -322.48



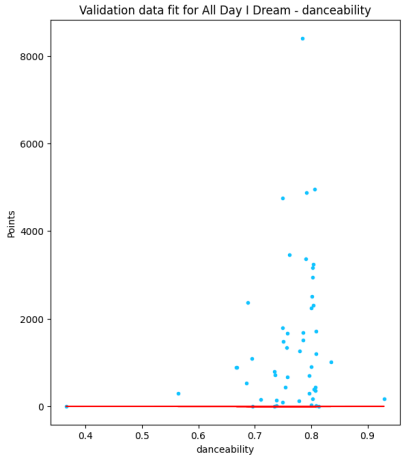

MSE: 4815676.02
RMSE: 2194.46
MAE: 1444.92
R-squared: -89.03



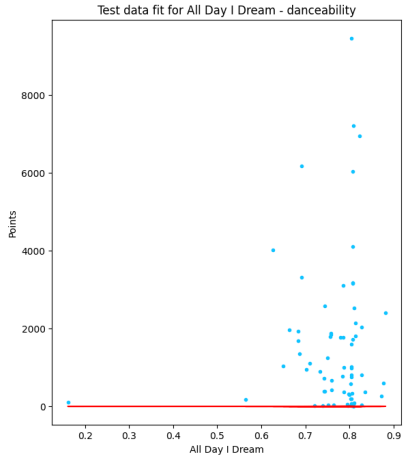

MSE: 6040790.96
RMSE: 2457.80
MAE: 1561.10
R-squared: -114.66



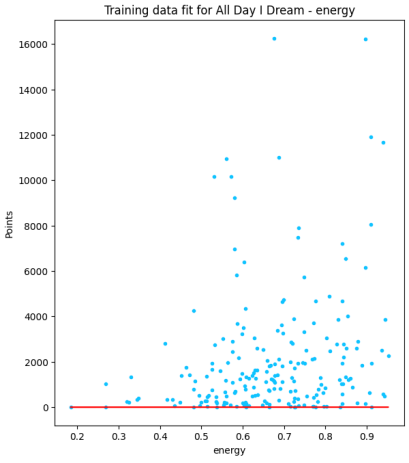

MSE: 11699182.47
RMSE: 3420.41
MAE: 2051.99
R-squared: -322.39



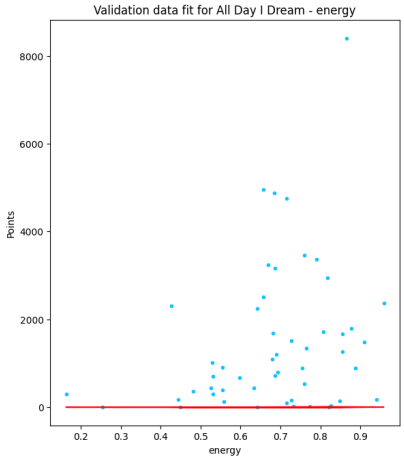

MSE: 4813433.68
RMSE: 2193.95
MAE: 1444.15
R-squared: -88.99



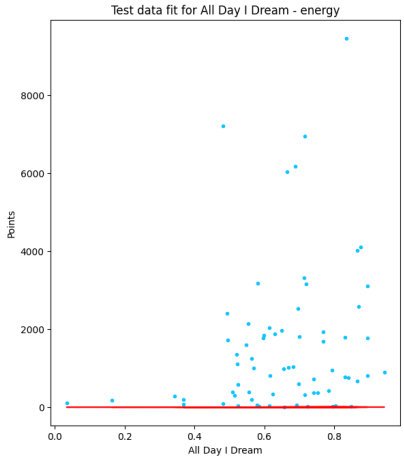

MSE: 6038420.19
RMSE: 2457.32
MAE: 1560.34
R-squared: -114.62



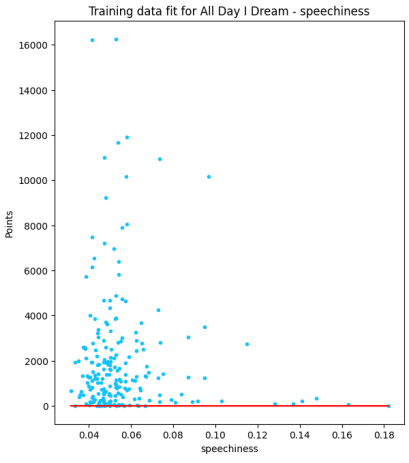

MSE: 11699943.76
RMSE: 3420.52
MAE: 2052.18
R-squared: -322.41



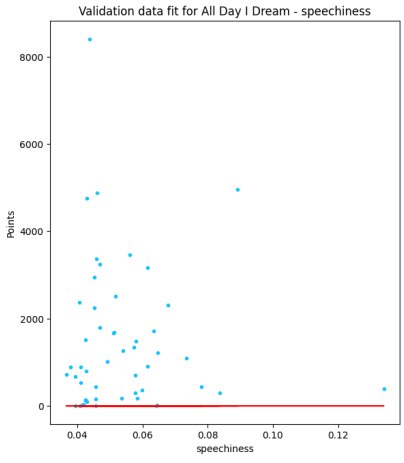

MSE: 4813880.16
RMSE: 2194.06
MAE: 1444.30
R-squared: -89.00



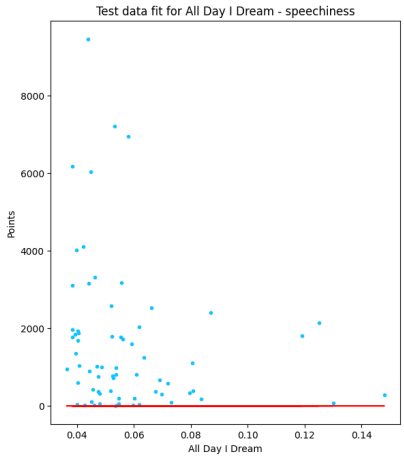

MSE: 6039120.53
RMSE: 2457.46
MAE: 1560.56
R-squared: -114.63



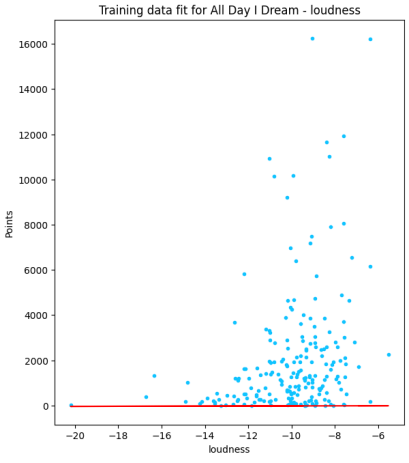

MSE: 11704978.22
RMSE: 3421.25
MAE: 2053.39
R-squared: -322.55



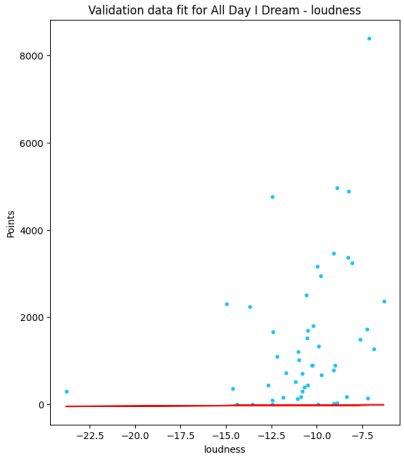

MSE: 4817513.66
RMSE: 2194.88
MAE: 1445.56
R-squared: -89.07



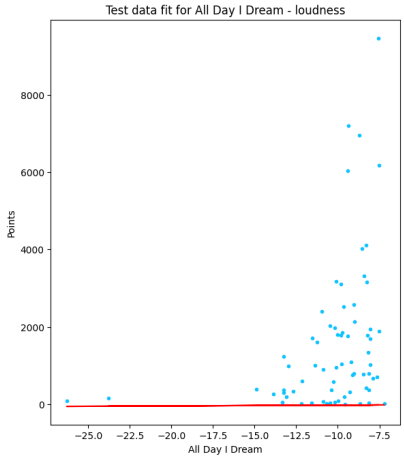

MSE: 6042836.40
RMSE: 2458.22
MAE: 1561.75
R-squared: -114.70



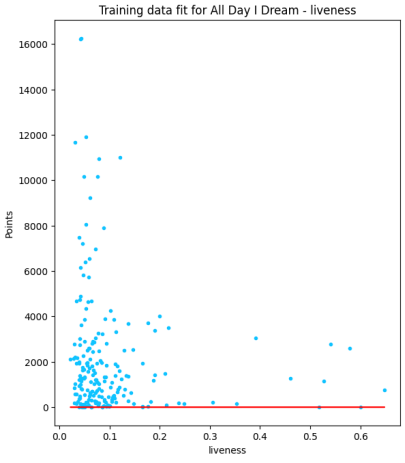

MSE: 11788024.02
RMSE: 3433.37
MAE: 2073.51
R-squared: -324.85



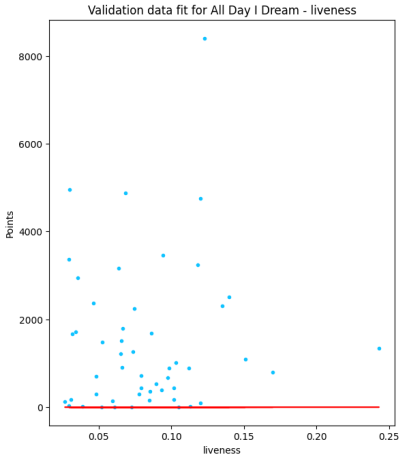

MSE: 4879678.90
RMSE: 2209.00
MAE: 1466.89
R-squared: -90.23



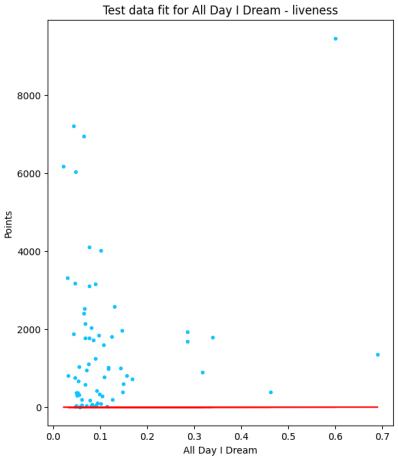

MSE: 6108421.65
RMSE: 2471.52
MAE: 1582.60
R-squared: -115.96



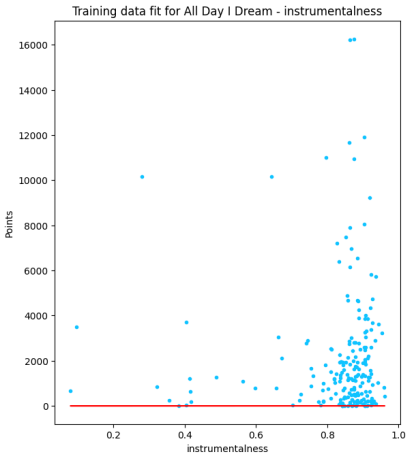

MSE: 11704620.46
RMSE: 3421.20
MAE: 2053.31
R-squared: -322.54



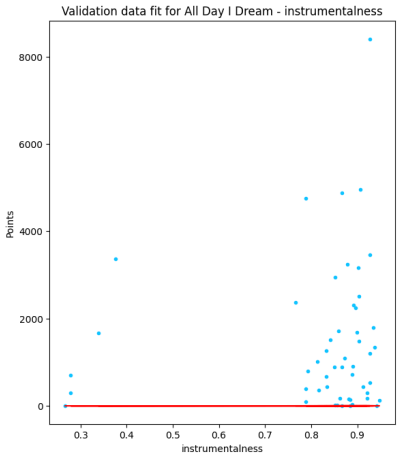

MSE: 4817347.91
RMSE: 2194.85
MAE: 1445.50
R-squared: -89.06



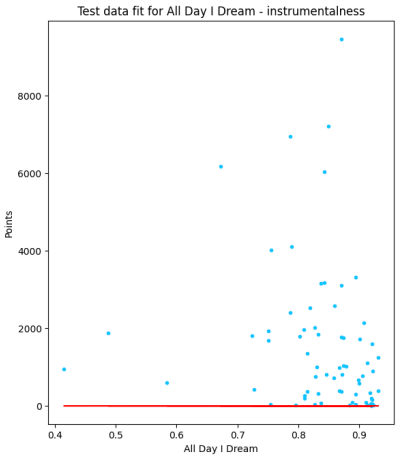

MSE: 6042461.80
RMSE: 2458.14
MAE: 1561.63
R-squared: -114.69



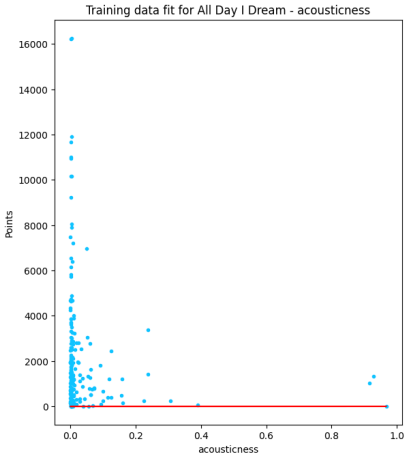

MSE: 11698596.82
RMSE: 3420.32
MAE: 2051.85
R-squared: -322.37



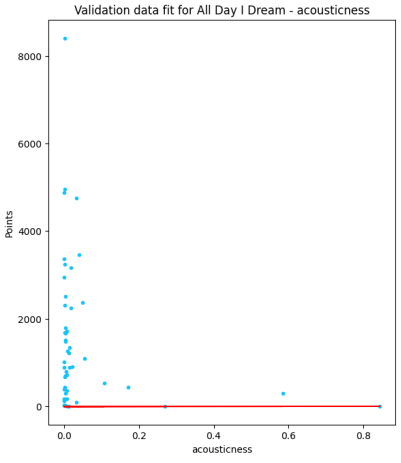

MSE: 4813091.50
RMSE: 2193.88
MAE: 1444.03
R-squared: -88.99



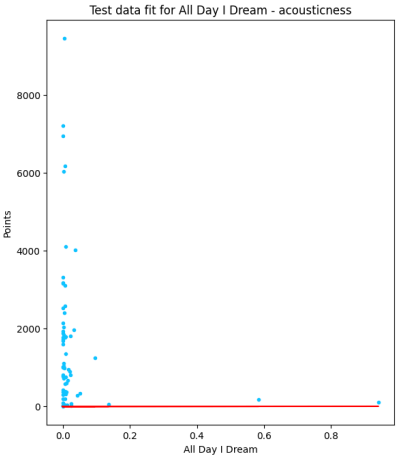

MSE: 6037968.84
RMSE: 2457.23
MAE: 1560.20
R-squared: -114.61



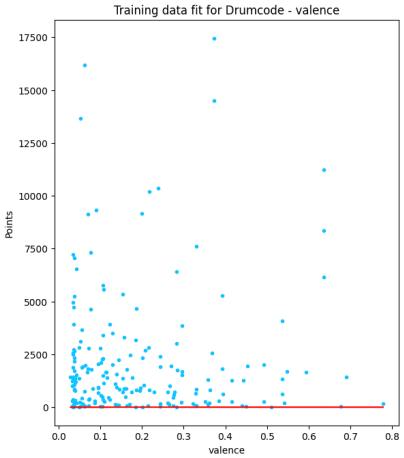

MSE: 11705146.05
RMSE: 3421.28
MAE: 2053.43
R-squared: -322.56



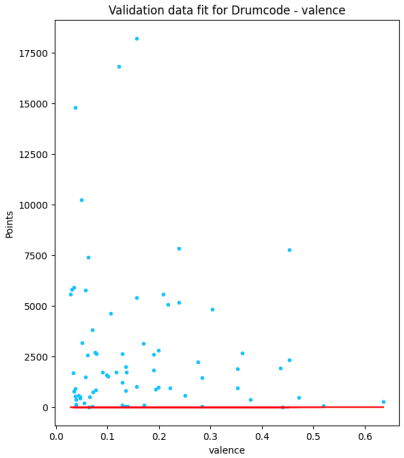

MSE: 4817551.77
RMSE: 2194.89
MAE: 1445.57
R-squared: -89.07



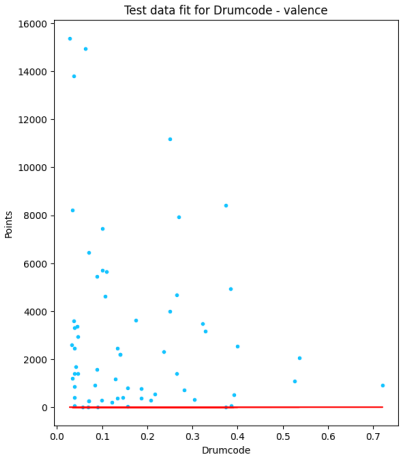

MSE: 6042991.91
RMSE: 2458.25
MAE: 1561.80
R-squared: -114.70



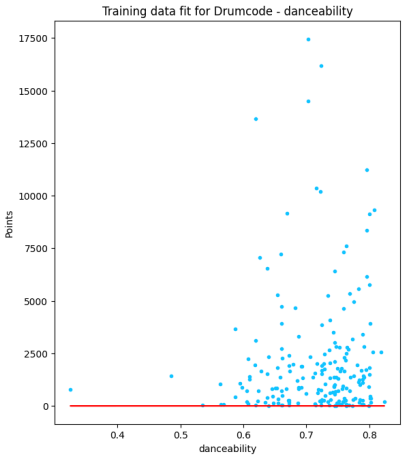

MSE: 12989584.44
RMSE: 3604.11
MAE: 2099.98
R-squared: -301.80



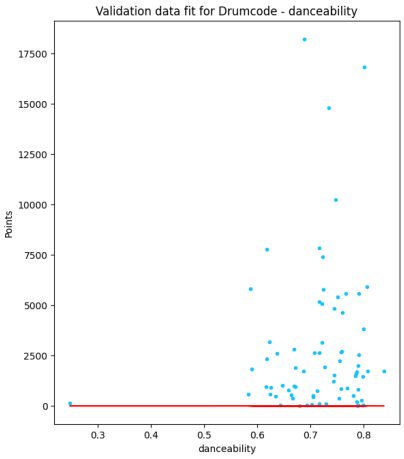

MSE: 20922654.74
RMSE: 4574.13
MAE: 2788.21
R-squared: -116.75



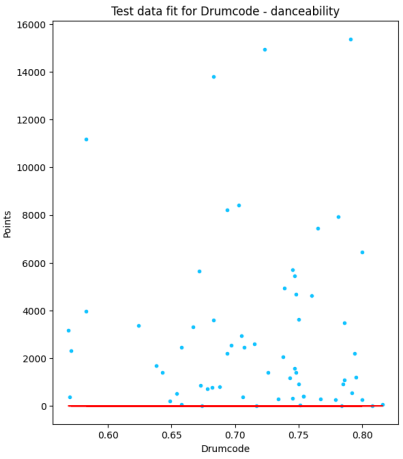

MSE: 21807956.43
RMSE: 4669.90
MAE: 2991.33
R-squared: -107.53



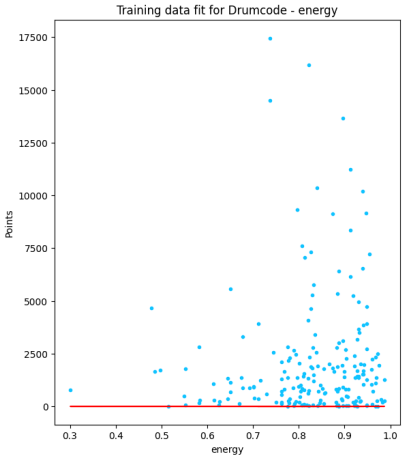

MSE: 12985131.79
RMSE: 3603.49
MAE: 2098.92
R-squared: -301.70



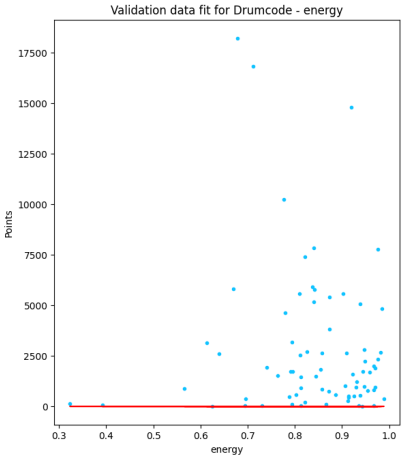

MSE: 20916541.62
RMSE: 4573.46
MAE: 2787.12
R-squared: -116.72



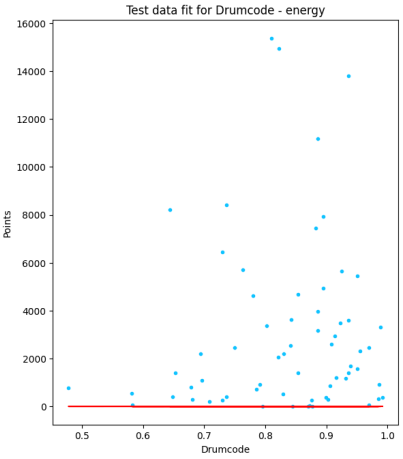

MSE: 21801432.71
RMSE: 4669.20
MAE: 2990.24
R-squared: -107.50



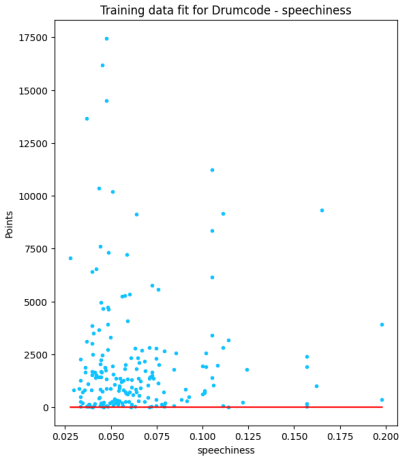

MSE: 12984170.86
RMSE: 3603.36
MAE: 2098.69
R-squared: -301.67



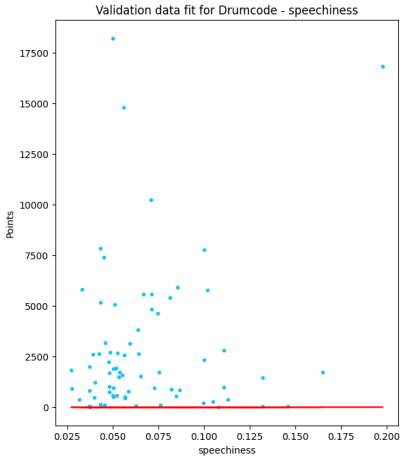

MSE: 20915151.63
RMSE: 4573.31
MAE: 2786.88
R-squared: -116.71



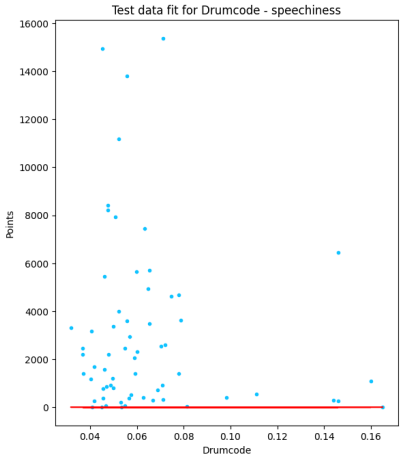

MSE: 21800035.24
RMSE: 4669.05
MAE: 2990.00
R-squared: -107.49



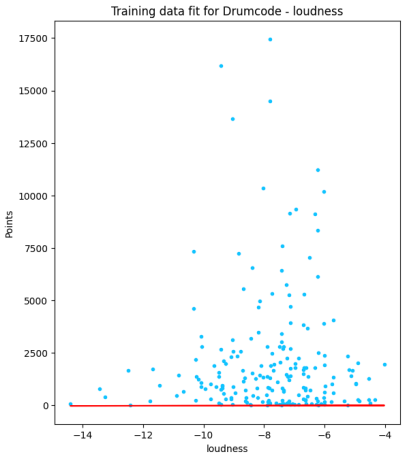

MSE: 12990607.70
RMSE: 3604.25
MAE: 2100.22
R-squared: -301.82



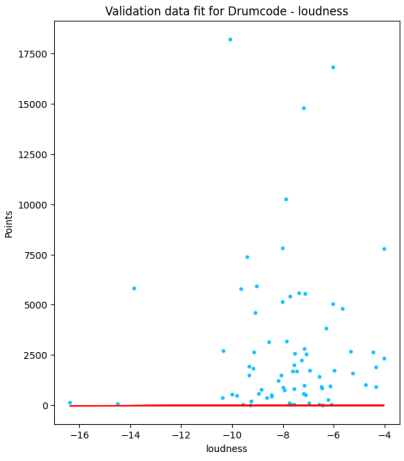

MSE: 20923767.17
RMSE: 4574.25
MAE: 2788.41
R-squared: -116.76



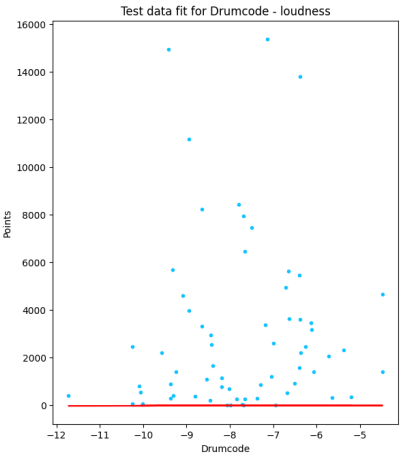

MSE: 21809236.34
RMSE: 4670.04
MAE: 2991.54
R-squared: -107.54



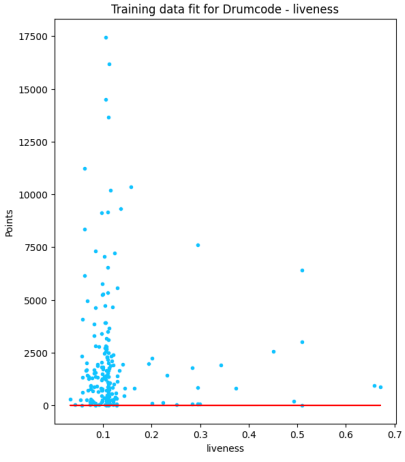

MSE: 13056762.74
RMSE: 3613.41
MAE: 2115.91
R-squared: -303.36



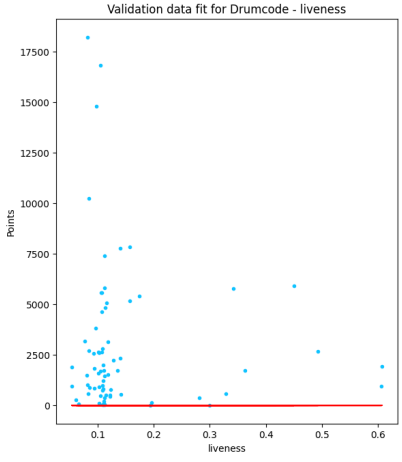

MSE: 21011851.42
RMSE: 4583.87
MAE: 2804.15
R-squared: -117.25



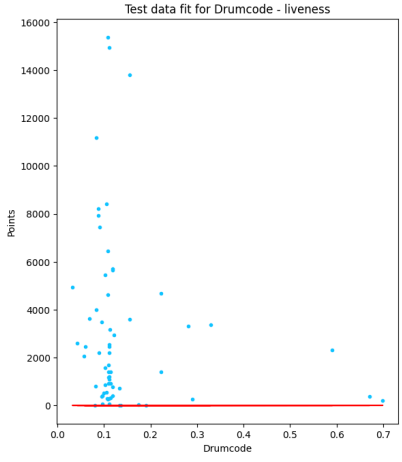

MSE: 21902949.98
RMSE: 4680.06
MAE: 3007.16
R-squared: -108.00



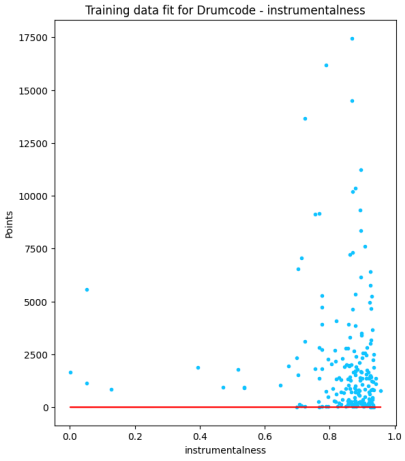

MSE: 12990061.96
RMSE: 3604.17
MAE: 2100.09
R-squared: -301.81



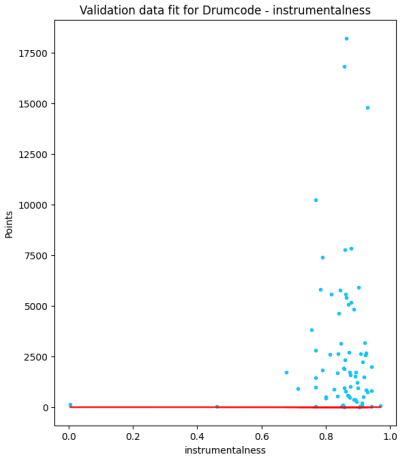

MSE: 20922869.72
RMSE: 4574.15
MAE: 2788.25
R-squared: -116.75



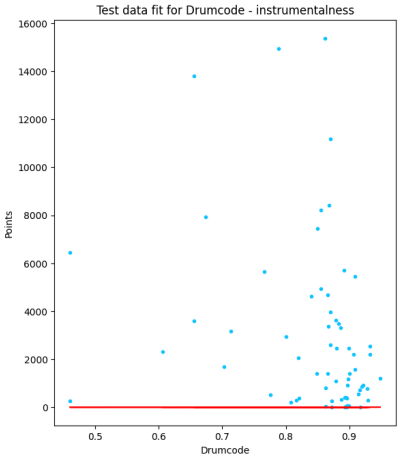

MSE: 21808276.11
RMSE: 4669.93
MAE: 2991.38
R-squared: -107.53



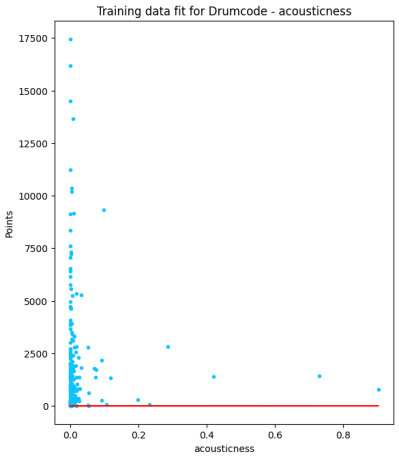

MSE: 12984095.40
RMSE: 3603.35
MAE: 2098.67
R-squared: -301.67



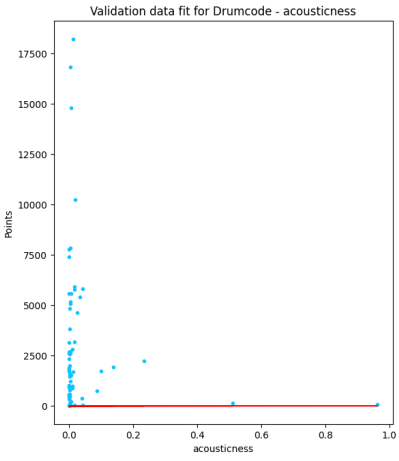

MSE: 20915097.80
RMSE: 4573.30
MAE: 2786.87
R-squared: -116.71



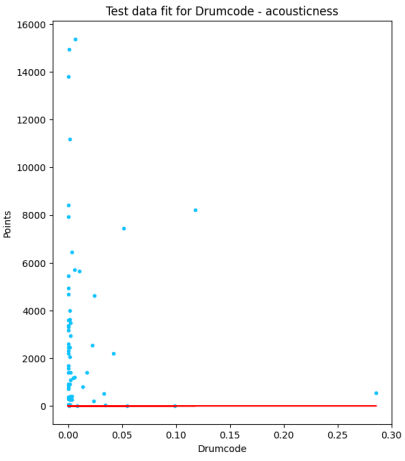

MSE: 21799909.09
RMSE: 4669.04
MAE: 2989.98
R-squared: -107.49



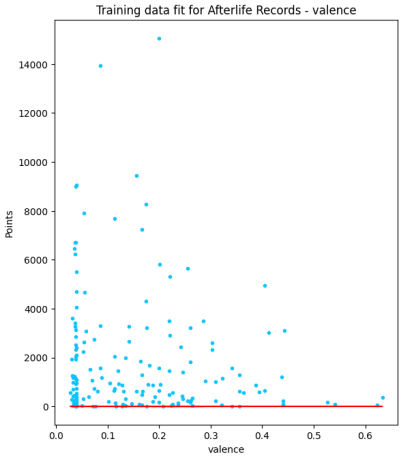

MSE: 12990954.40
RMSE: 3604.30
MAE: 2100.30
R-squared: -301.83



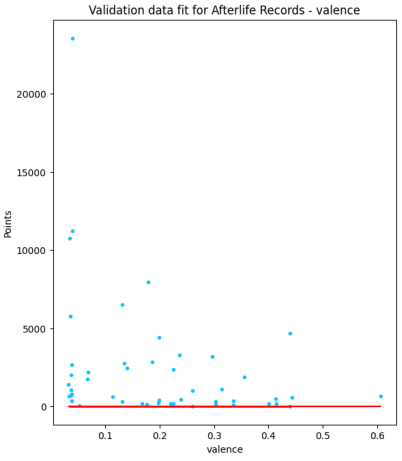

MSE: 20924145.08
RMSE: 4574.29
MAE: 2788.47
R-squared: -116.76



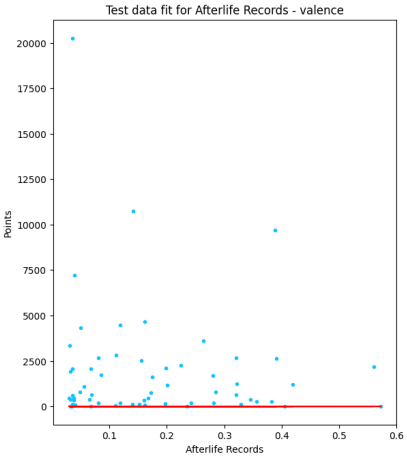

MSE: 21809836.87
RMSE: 4670.10
MAE: 2991.64
R-squared: -107.54



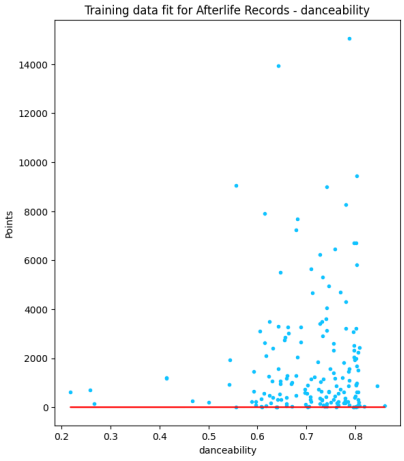

MSE: 9480745.84
RMSE: 3079.08
MAE: 1811.39
R-squared: -260.50



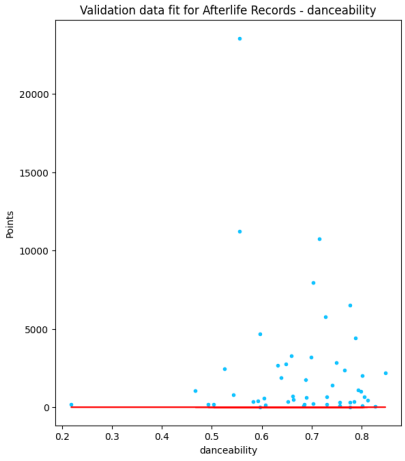

MSE: 21126983.99
RMSE: 4596.41
MAE: 2309.05
R-squared: -65.88



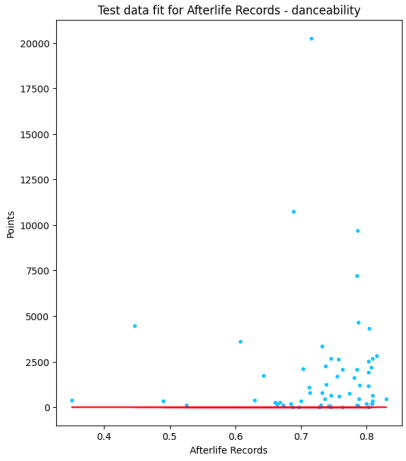

MSE: 13734844.12
RMSE: 3706.06
MAE: 1861.67
R-squared: -80.59



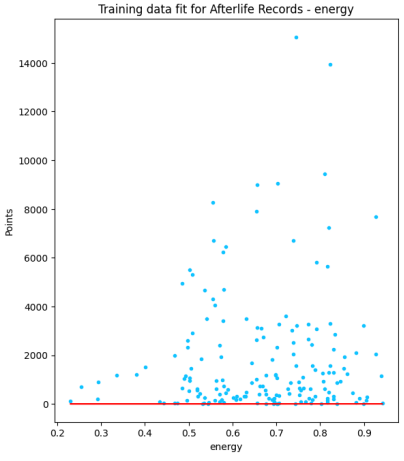

MSE: 9476752.91
RMSE: 3078.43
MAE: 1810.30
R-squared: -260.39



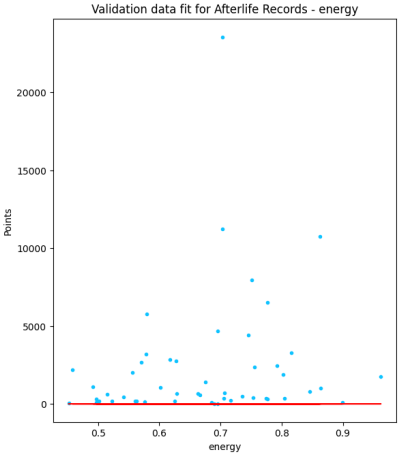

MSE: 21122543.19
RMSE: 4595.93
MAE: 2308.10
R-squared: -65.86



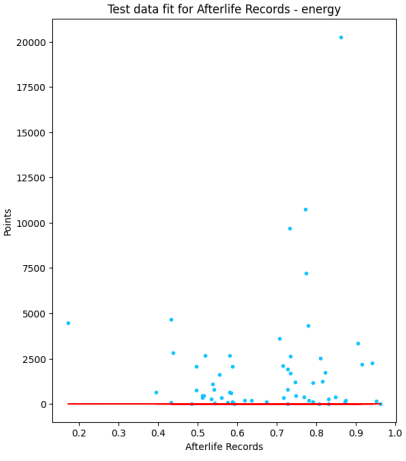

MSE: 13730724.14
RMSE: 3705.50
MAE: 1860.57
R-squared: -80.56



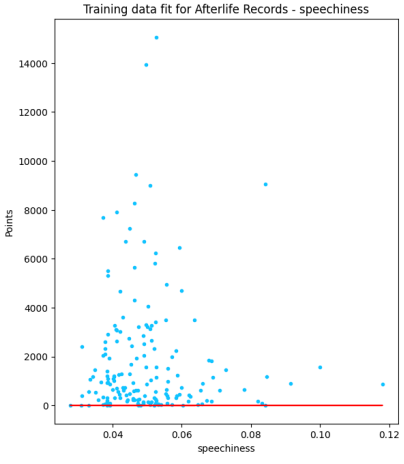

MSE: 9477010.80
RMSE: 3078.48
MAE: 1810.37
R-squared: -260.40



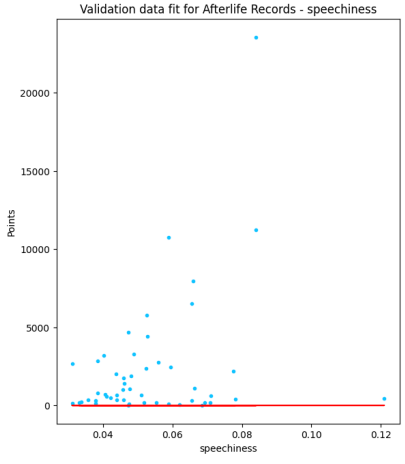

MSE: 21122606.53
RMSE: 4595.93
MAE: 2308.11
R-squared: -65.86



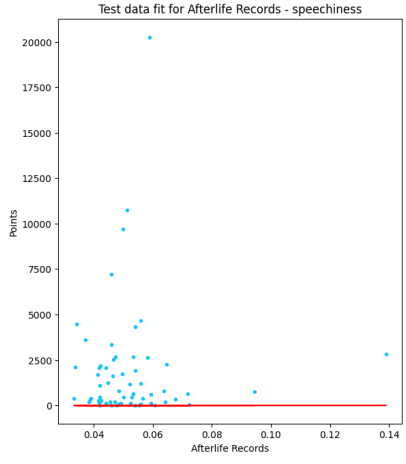

MSE: 13731121.10
RMSE: 3705.55
MAE: 1860.68
R-squared: -80.57



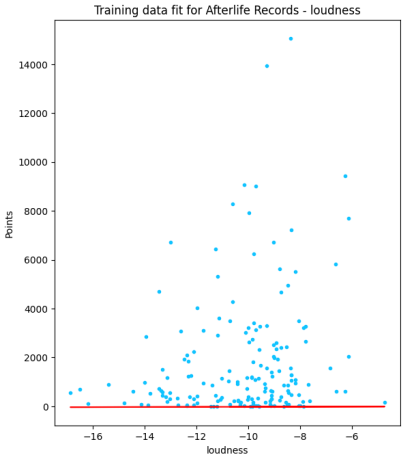

MSE: 9481500.93
RMSE: 3079.20
MAE: 1811.60
R-squared: -260.52



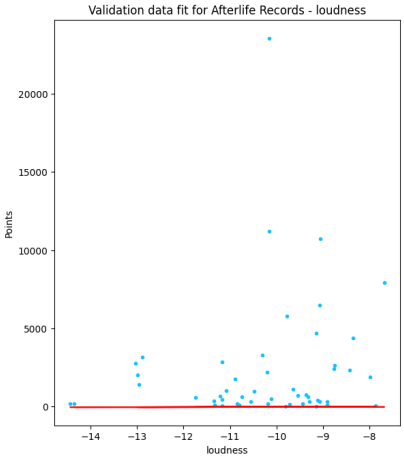

MSE: 21128313.04
RMSE: 4596.55
MAE: 2309.33
R-squared: -65.88



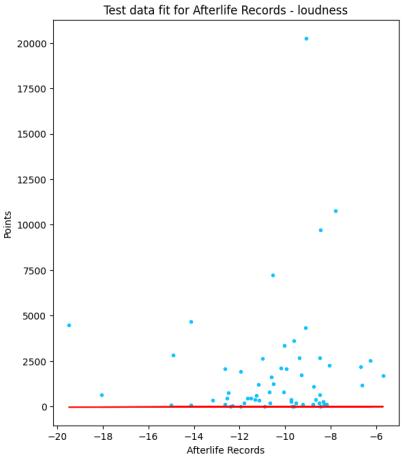

MSE: 13735765.60
RMSE: 3706.18
MAE: 1861.91
R-squared: -80.59



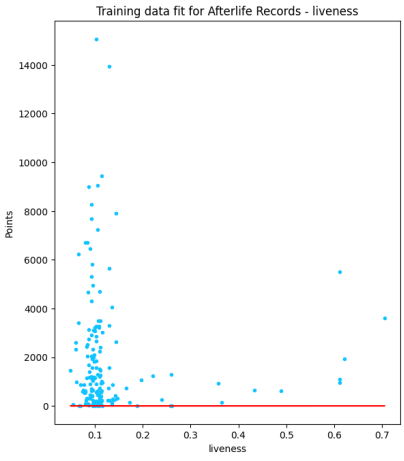

MSE: 9556344.16
RMSE: 3091.33
MAE: 1832.13
R-squared: -262.59



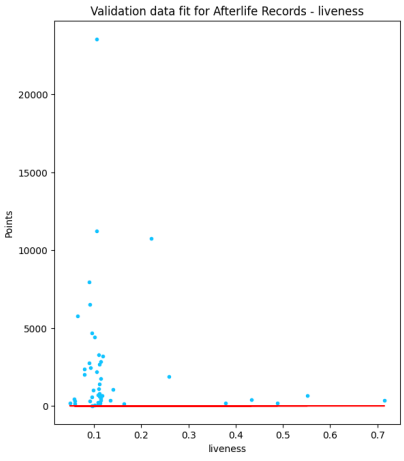

MSE: 21223929.48
RMSE: 4606.94
MAE: 2329.94
R-squared: -66.18



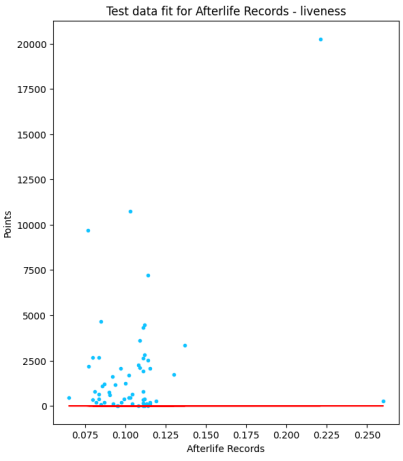

MSE: 13814920.65
RMSE: 3716.84
MAE: 1883.04
R-squared: -81.06



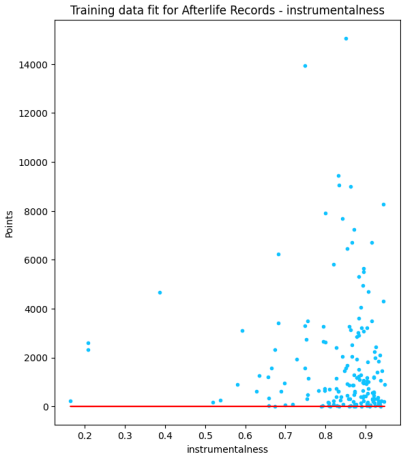

MSE: 9480926.58
RMSE: 3079.11
MAE: 1811.44
R-squared: -260.51



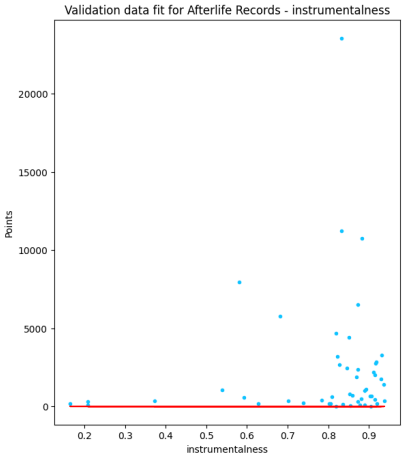

MSE: 21127439.39
RMSE: 4596.46
MAE: 2309.14
R-squared: -65.88



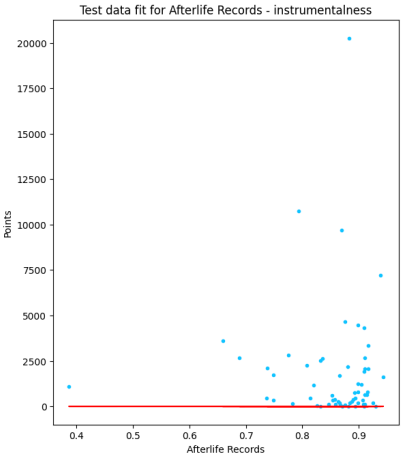

MSE: 13735367.61
RMSE: 3706.13
MAE: 1861.81
R-squared: -80.59



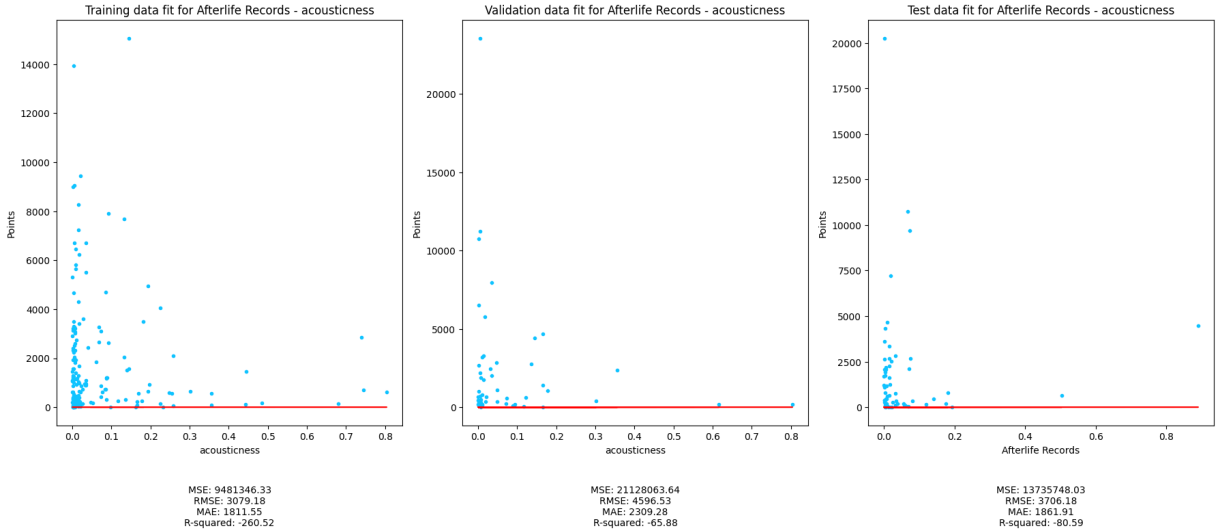

MSE: 9475833.54
RMSE: 3078.28
MAE: 1810.05
R-squared: -260.37




MSE: 21121529.98
RMSE: 4595.82
MAE: 2307.89
R-squared: -65.86




MSE: 13729792.33
RMSE: 3705.37
MAE: 1860.33
R-squared: -80.56


Training data fit for Afterlife Records - acousticness
Points
acousticness
MSE: 9481346.33
RMSE: 3079.18
MAE: 1811.55
R-squared: -260.52
Validation data fit for Afterlife Records - acousticness
Points
acousticness
MSE: 21128063.64
RMSE: 4596.53
MAE: 2309.28
R-squared: -65.88
Test data fit for Afterlife Records - acousticness
Points
Afterlife Records
MSE: 13735748.03
RMSE: 3706.18
MAE: 1861.91
R-squared: -80.59

In [ ]: