Earthquake_prediction_2

May 28, 2019

1 LANL Earthquake Prediction

1.0.1 1.1 Description

Forecasting earthquakes is one of the most important problems in Earth science because of their devastating consequences. Current scientific studies related to earthquake forecasting focus on three key points: when the event will occur, where it will occur, and how large it will be.

The goal of the challenge is to capture the physical state of the laboratory fault and how close it is from failure from a snapshot of the seismic data it is emitting. You will have to build a model that predicts the time remaining before failure from a chunk of seismic data, like we have done in our first paper above on easier data.

The input is a chunk of 0.0375 seconds of seismic data (ordered in time), which is recorded at 4MHz, hence 150′000 data points, and the output is time remaining until the following lab earthquake, in seconds.

The seismic data is recorded using a piezoceramic sensor, which outputs a voltage upon deformation by incoming seismic waves. The seismic data of the input is this recorded voltage, in integers.

Both the training and the testing set come from the same experiment. There is no overlap between the training and testing sets, that are contiguous in time.

Time to failure is based on a measure of fault strength (shear stress, not part of the data for the competition). When a labquake occurs this stress drops unambiguously.

The data is recorded in bins of 4096 samples. Within those bins seismic data is recorded at 4MHz, but there is a 12 microseconds gap between each bin, an artifact of the recording device.

1.1 Problem Statement:

To predict the time remaining before laboratory earthquakes occur from real-time seismic data.

1.2 Sources https://www.kaggle.com/c/LANL-Earthquake-Prediction https://www.kaggle.com/c/LANL-Earthquake-Prediction/discussion

2. Machine Learning problem

1.1.1 2.1 Data

train.csv - A single, continuous training segment of experimental data.

1.1.2 2.1.1 Data Overview

train.csv contains 2 columns: acoustic_data - the seismic signal [int16] time_to_failure - the time (in seconds) until the next laboratory earthquake [float64] Number of rows in Train.csv = 629145480

1.1.3 2.2.1 Type of Machine Leaning Problem

It is a Regression problem, for a given chunk of seismic data we need to predict the time remaining before laboratory earthquakes occur

2.2.2 Performance Metric Source: https://www.kaggle.com/c/LANL-Earthquake-Prediction#evaluation Metric(s): Mean Absolute Error

```
In [1]: from tqdm import tqdm_notebook

        import matplotlib.pyplot as plt

        import os

        from scipy.stats import skew
        from scipy.stats import norm
        from sklearn.linear_model import LinearRegression

        from scipy.signal import lfilter
        import scipy.signal

        from sklearn.model_selection import GridSearchCV
        from sklearn.decomposition import TruncatedSVD


        from catboost import CatBoostRegressor,Pool


        import os
        import time
        import warnings
        import traceback
        import numpy as np
        import pandas as pd
        from scipy import stats
        import scipy.signal as sg
        import multiprocessing as mp
        from scipy.signal import hann
        from scipy.signal import hilbert
        from scipy.signal import convolve
        from sklearn.linear_model import LinearRegression
        from sklearn.preprocessing import StandardScaler
        from sklearn.model_selection import GridSearchCV
        from tsfresh.feature_extraction import feature_calculators

        import scipy as sp
        import xgboost as xgb
        import lightgbm as lgb
        from sklearn.model_selection import KFold
        from sklearn.metrics import mean_squared_error
        from sklearn.metrics import mean_absolute_error

        from tqdm import tqdm
        warnings.filterwarnings("ignore")
```

2 Exploratory Data Analysis

I have used several kernels from kaggle and ideas from discussion threads . https://www.kaggle.com/vettejeep/masters-final-project-model-lb-1-392 https://www.kaggle.com/allunia/shaking-earth https://www.kaggle.com/gpreda/lanl-earthquake-eda-and-prediction

```
In [8]: train = pd.read_csv('train.csv', dtype={'acoustic_data': np.int16, 'time_to_failure': 
```

```
In [99]: train.shape
```

```
Out[99]: (629145480, 2)
```

There are 6.2 billion datapoints

```
In [98]: # to show all the decimal points
         pd.options.display.precision = 15
         train.head()
```

```
Out[98]:    acoustic_data  time_to_failure
         0             12     1.4690999832
         1              6     1.4690999821
         2              8     1.4690999810
         3              5     1.4690999799
         4              8     1.4690999788
```

We can see that for each sample the time to failure decreases by 1.1e-9

```
In [42]: train.describe()
```

```
Out[42]:        acoustic_data  time_to_failure
         count   6.291455e+08     6.291455e+08
         mean    4.519468e+00     5.678292e+00
         std     1.073571e+01     3.672697e+00
         min    -5.515000e+03     9.550396e-05
         25%     2.000000e+00     2.625997e+00
```

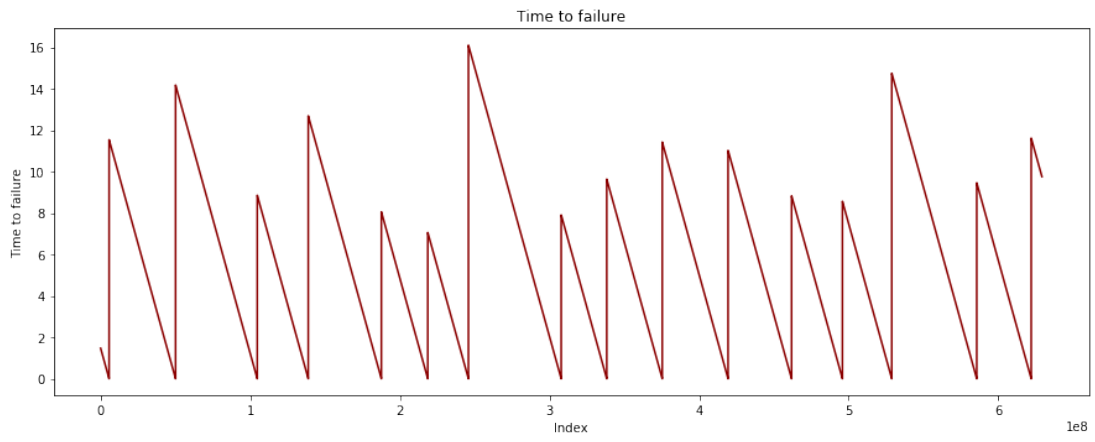
```
50%     5.000000e+00     5.349798e+00
75%     7.000000e+00     8.173396e+00
max     5.444000e+03     1.610740e+01
```

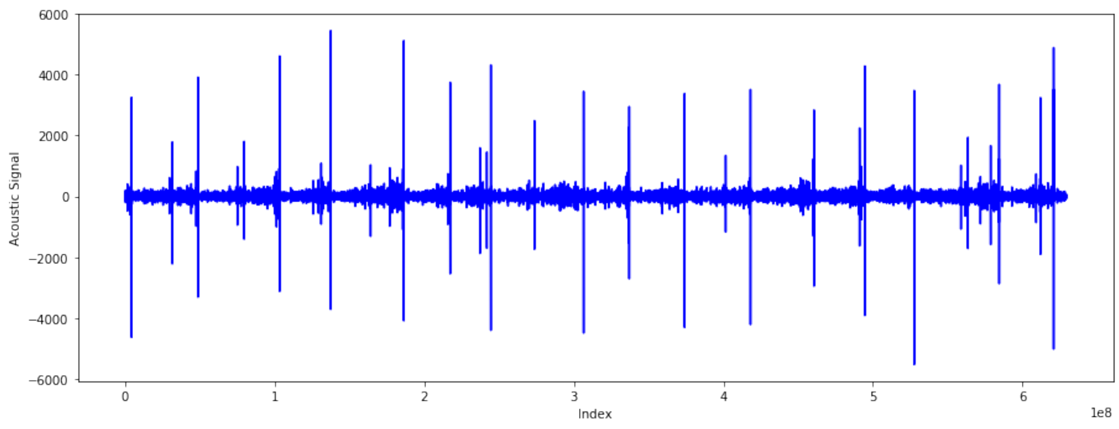
75% of the acoustic data is below 7 and the max value is 5.4e+03, i e only few values are approximately 5.4e+03

2.0.1 Visualizing Train data

2.0.2 Number of occurences of Earthquake

```
In [5]: #plotting the train data
        fig, ax = plt.subplots(2,1, figsize=(15,12))
        ax[0].plot(train.index.values, train.time_to_failure.values, c="darkred")
        ax[0].set_title("Time to failure")
        ax[0].set_xlabel("Index")
        ax[0].set_ylabel("Time to failure");
        ax[1].plot(train.index.values, train.acoustic_data.values, c="blue")
        #ax[1].set_title("Index")
        ax[1].set_xlabel("Index")
        ax[1].set_ylabel("Acoustic Signal")
        plt.show()
```

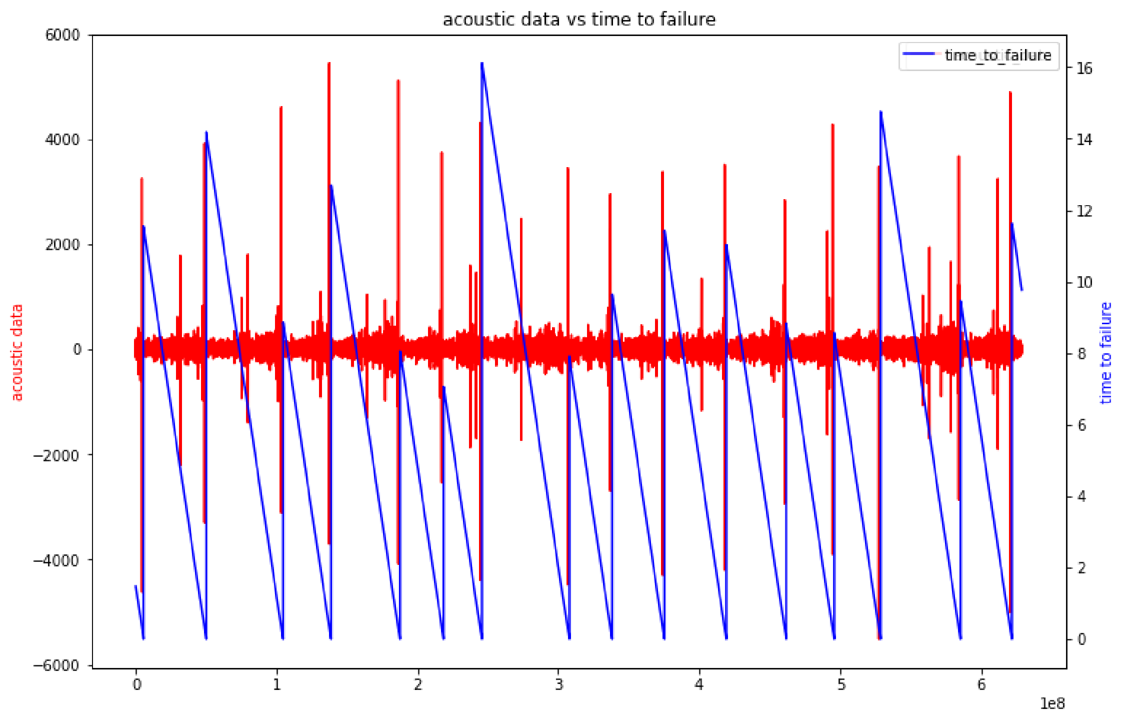




It is given that the earthquake occurs when the time_to_failure hits 0, hence we can count that there are 16 occurences of earthquake in the whole training data

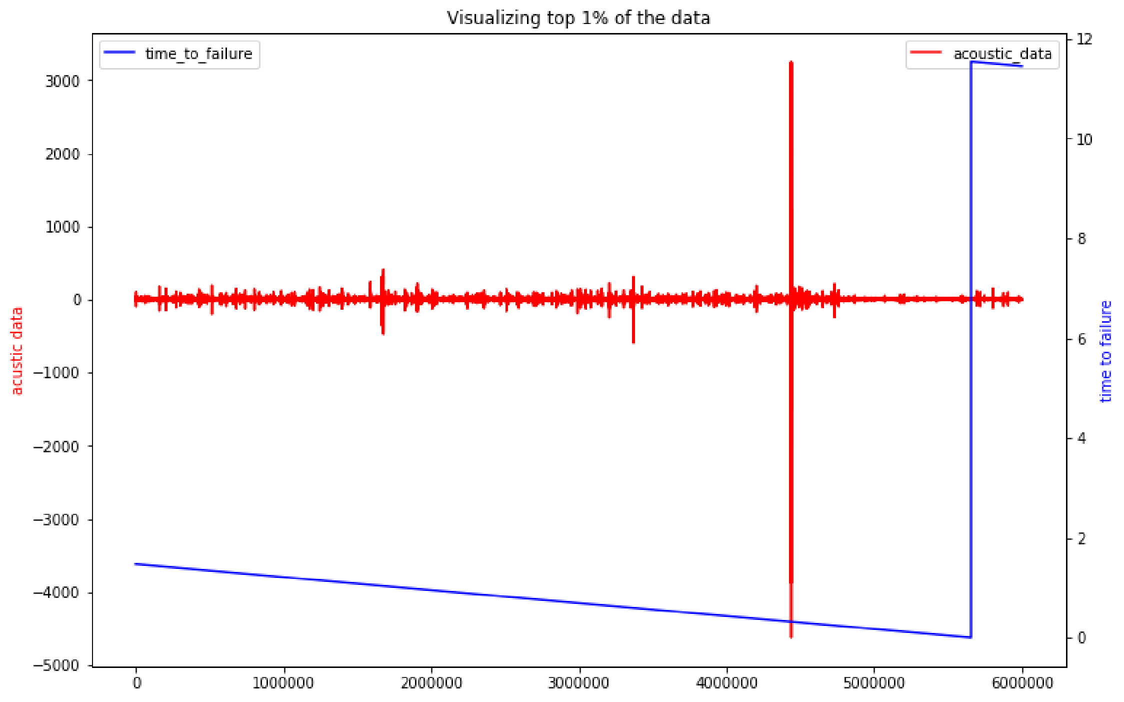
2.0.3 Relationship between time to failure and acoustic data

```
In [4]: #plotting acoustic data vs time to failure
        fig, ax1 = plt.subplots(figsize=(12, 8))
        plt.plot(train.acoustic_data,color='r')
        plt.legend()
        ax1.set_ylabel('acoustic data',color='r')
        ax2=ax1.twinx()
        ax2.set_ylabel('time to failure',color='b')
        plt.plot(train.time_to_failure,color='b')
        plt.title('acoustic data vs time to failure')
        plt.legend()
        plt.show()
```



The acoustic data has a peak just before time to failure hits zero. We can verify it by zooming into the plot.

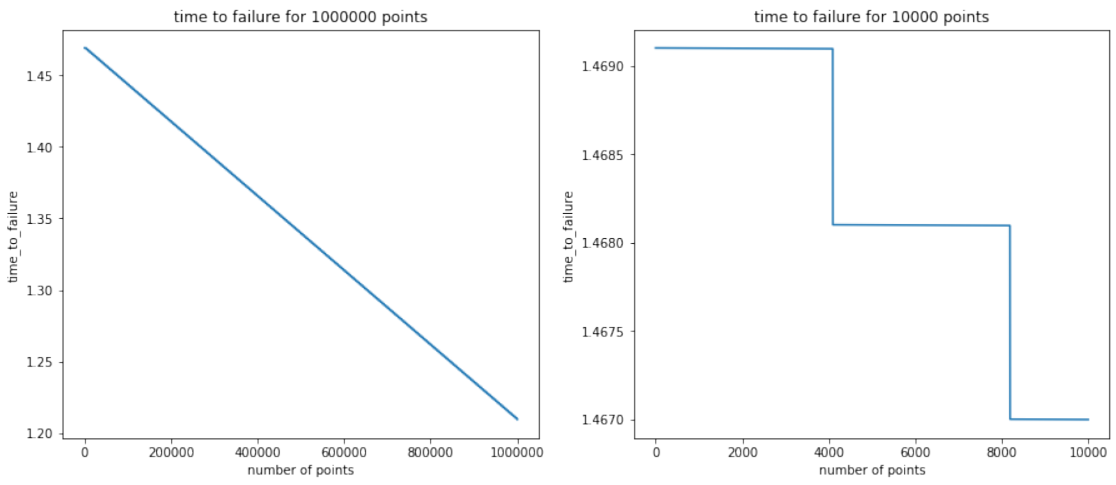
```
In [5]: #plotting only top 1% of the value
        fig, ax1 = plt.subplots(figsize=(12, 8))
        plt.plot(train.acoustic_data[0:6000000],color='r')
        plt.legend()
        ax1.set_ylabel('acustic data',color='r')
        ax2=ax1.twinx()
        ax2.set_ylabel('time to failure',color='b')
        plt.plot(train.time_to_failure[0:6000000],color='b')
        plt.title('Visualizing top 1% of the data')
        plt.legend()
        plt.show()
```



If we zoom into the data we can see that the acoustic data has a peak just before the earthquake occurs and the whole training data follows the same pattern

2.0.4 Is time to failure continously Decreasing

```
In [3]: #plotting time to failure for fewer data
        fig = plt.figure(figsize=(15, 6))
        plt.subplot(1,2,1)

        plt.plot(train.time_to_failure[0:1000000])
        plt.title('time to failure for 1000000 points')
        plt.xlabel('number of points')
        plt.ylabel('time_to_failure')

        plt.subplot(1,2,2)
        plt.plot(train.time_to_failure[0:10000])
        plt.xlabel('number of points')
        plt.ylabel('time_to_failure')
        plt.title('time to failure for 10000 points')

        plt.show()
```



If we plot the data for 1000000 points we can see that the graph is continously decreasing but if we zoom into it we can see that the time_to_failure stops decreasing for a while when it reaches ~4000 samples. It is due to the fact that the data is recorded in bins of 4096 samples and the recording device stops for 12 microseconds after each bin.

2.0.5 Visualizing Test Data

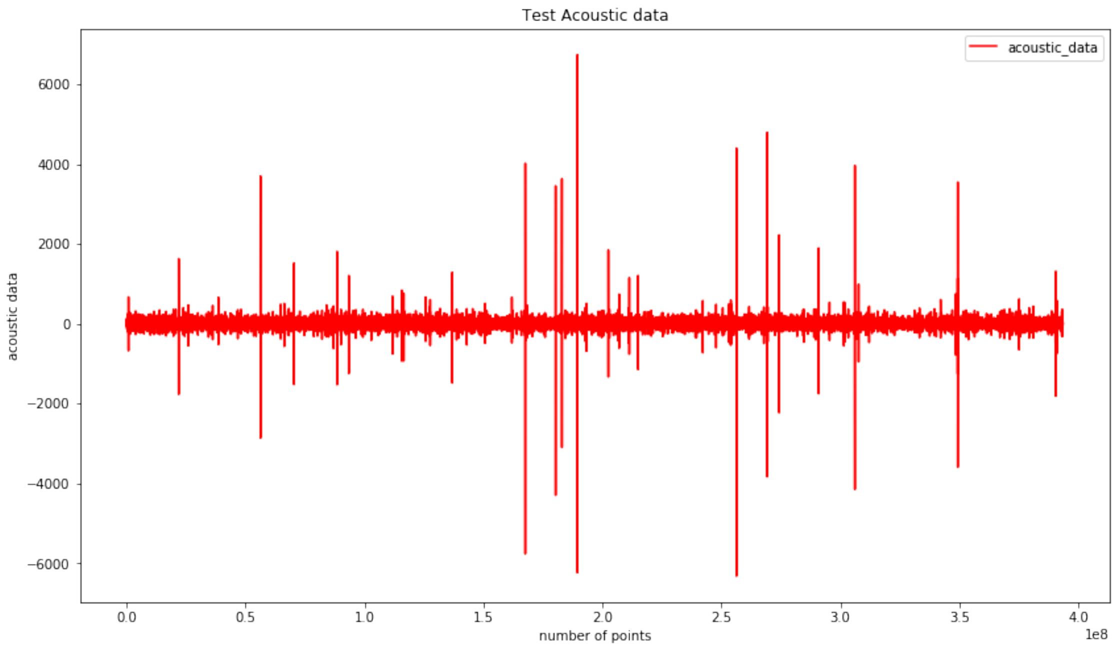
```
In [36]: #Reading the test data
         from tqdm.auto import tqdm
         submission = pd.read_csv('sample_submission.csv', index_col='seg_id')
         test = pd.DataFrame(dtype=np.float64, index=submission.index)
         whole_test=[]
         for seg_id in tqdm(test.index):
             seg = pd.read_csv('Untitled Folder/' + seg_id + '.csv')
             for i in seg['acoustic_data'].values:
                 whole_test.append(i)
```

```
HBox(children=(IntProgress(value=0, max=2624), HTML(value='')))
```

```
In [40]: #total number of datapoints in test
         len(whole_test)

Out[40]: 393600000
```

```
In [39]: #plotting test data
         fig = plt.figure(figsize=(14, 8))
```

```
plt.plot(whole_test,color='r',label='acoustic_data')
plt.title('Test Acoustic data')
plt.xlabel('number of points')
plt.ylabel('acoustic data')
plt.legend()
plt.show()
```



Checking for Null values

```
In [41]: train.isnull().any().any()
```

```
Out[41]: False
```

There are no null values in the whole training data

3 Featurization

3.0.1 Feature set 1

```
In [2]: OUTPUT_DIR = ''  # set for local environment
        DATA_DIR = ''  # set for local environment

        SIG_LEN = 150000
        NUM_SEG_PER_PROC = 4000
        NUM_THREADS = 6

        NY_FREQ_IDX = 75000  # the test signals are 150k samples long, Nyquist is thus 75k.
        CUTOFF = 18000
        MAX_FREQ_IDX = 20000
        FREQ_STEP = 2500
```

```
In [ ]: # into 6 slices
        def split_raw_data():
            df = pd.read_csv(os.path.join(DATA_DIR, 'train.csv'))

            max_start_index = len(df.index) - SIG_LEN
            slice_len = int(max_start_index / 6)

            for i in tqdm(range(NUM_THREADS)):
                print('working', i)
                df0 = df.iloc[slice_len * i: (slice_len * (i + 1)) + SIG_LEN]
                df0.to_csv(os.path.join(DATA_DIR, 'raw_data_%d.csv' % i), index=False)
                del df0

            del df
```

```
In [ ]: #building random indices
        def build_rnd_idxs():
            rnd_idxs = np.zeros(shape=(NUM_THREADS, NUM_SEG_PER_PROC), dtype=np.int32)
            max_start_idx = 100000000

            for i in range(NUM_THREADS):
                np.random.seed(5591 + i)
                start_indices = np.random.randint(0, max_start_idx, size=NUM_SEG_PER_PROC, dty
                rnd_idxs[i, :] = start_indices

            for i in range(NUM_THREADS):
                print(rnd_idxs[i, :8])
                print(rnd_idxs[i, -8:])
                print(min(rnd_idxs[i,:]), max(rnd_idxs[i,:]))

            np.savetxt(fname=os.path.join(OUTPUT_DIR, 'start_indices_4k.csv'), X=np.transpose(
```

```
In [5]: #finding the slope
        def add_trend_feature(arr, abs_values=False):
            idx = np.array(range(len(arr)))
            if abs_values:
                arr = np.abs(arr)
            lr = LinearRegression()
            lr.fit(idx.reshape(-1, 1), arr)
            return lr.coef_[0]
        def classic_sta_lta(x, length_sta, length_lta):
            sta = np.cumsum(x ** 2)
            # Convert to float
            sta = np.require(sta, dtype=np.float)
            # Copy for LTA
            lta = sta.copy()
            # Compute the STA and the LTA
            sta[length_sta:] = sta[length_sta:] - sta[:-length_sta]
            sta /= length_sta
            lta[length_lta:] = lta[length_lta:] - lta[:-length_lta]
            lta /= length_lta
            # Pad zeros
            sta[:length_lta - 1] = 0
            # Avoid division by zero by setting zero values to tiny float
            dtiny = np.finfo(0.0).tiny
            idx = lta < dtiny
            lta[idx] = dtiny
            return sta / lta
```

```
In [6]: def des_bw_filter_lp(cutoff=CUTOFF):  # low pass filter
            b, a = sg.butter(4, Wn=cutoff/NY_FREQ_IDX)
            return b, a

        def des_bw_filter_hp(cutoff=CUTOFF):  # high pass filter
            b, a = sg.butter(4, Wn=cutoff/NY_FREQ_IDX, btype='highpass')
            return b, a

        def des_bw_filter_bp(low, high):  # band pass filter
            b, a = sg.butter(4, Wn=(low/NY_FREQ_IDX, high/NY_FREQ_IDX), btype='bandpass')
            return b, a
```

```
In [4]: # a function to create features
        def create_features(seg_id, seg, X, st, end):
            try:
                X.loc[seg_id, 'seg_id'] = np.int32(seg_id)
                X.loc[seg_id, 'seg_start'] = np.int32(st)
                X.loc[seg_id, 'seg_end'] = np.int32(end)
            except:
                pass

            xc = pd.Series(seg['acoustic_data'].values)
            xcdm = xc - np.mean(xc)

            b, a = des_bw_filter_lp(cutoff=18000)
            xcz = sg.lfilter(b, a, xcdm)

            zc = np.fft.fft(xcz)
            zc = zc[:MAX_FREQ_IDX]
```

```
# FFT transform values
realFFT = np.real(zc)
imagFFT = np.imag(zc)

freq_bands = [x for x in range(0, MAX_FREQ_IDX, FREQ_STEP)]
magFFT = np.sqrt(realFFT ** 2 + imagFFT ** 2)
phzFFT = np.arctan(imagFFT / realFFT)
phzFFT[phzFFT == -np.inf] = -np.pi / 2.0
phzFFT[phzFFT == np.inf] = np.pi / 2.0
phzFFT = np.nan_to_num(phzFFT)

for freq in freq_bands:
    X.loc[seg_id, 'FFT_Mag_01q%d' % freq] = np.quantile(magFFT[freq: freq + FREQ_S
    X.loc[seg_id, 'FFT_Mag_10q%d' % freq] = np.quantile(magFFT[freq: freq + FREQ_S
    X.loc[seg_id, 'FFT_Mag_90q%d' % freq] = np.quantile(magFFT[freq: freq + FREQ_S
    X.loc[seg_id, 'FFT_Mag_99q%d' % freq] = np.quantile(magFFT[freq: freq + FREQ_S
    X.loc[seg_id, 'FFT_Mag_mean%d' % freq] = np.mean(magFFT[freq: freq + FREQ_STEP
    X.loc[seg_id, 'FFT_Mag_std%d' % freq] = np.std(magFFT[freq: freq + FREQ_STEP])
    X.loc[seg_id, 'FFT_Mag_max%d' % freq] = np.max(magFFT[freq: freq + FREQ_STEP])

    X.loc[seg_id, 'FFT_Phz_mean%d' % freq] = np.mean(phzFFT[freq: freq + FREQ_STEP
    X.loc[seg_id, 'FFT_Phz_std%d' % freq] = np.std(phzFFT[freq: freq + FREQ_STEP])

X.loc[seg_id, 'FFT_Rmean'] = realFFT.mean()
X.loc[seg_id, 'FFT_Rstd'] = realFFT.std()
X.loc[seg_id, 'FFT_Rmax'] = realFFT.max()
X.loc[seg_id, 'FFT_Rmin'] = realFFT.min()
X.loc[seg_id, 'FFT_Imean'] = imagFFT.mean()
X.loc[seg_id, 'FFT_Istd'] = imagFFT.std()
X.loc[seg_id, 'FFT_Imax'] = imagFFT.max()
X.loc[seg_id, 'FFT_Imin'] = imagFFT.min()

X.loc[seg_id, 'FFT_Rmean_first_6000'] = realFFT[:6000].mean()
X.loc[seg_id, 'FFT_Rstd__first_6000'] = realFFT[:6000].std()
X.loc[seg_id, 'FFT_Rmax_first_6000'] = realFFT[:6000].max()
X.loc[seg_id, 'FFT_Rmin_first_6000'] = realFFT[:6000].min()
X.loc[seg_id, 'FFT_Rmean_first_18000'] = realFFT[:18000].mean()
X.loc[seg_id, 'FFT_Rstd_first_18000'] = realFFT[:18000].std()
X.loc[seg_id, 'FFT_Rmax_first_18000'] = realFFT[:18000].max()
X.loc[seg_id, 'FFT_Rmin_first_18000'] = realFFT[:18000].min()

del xcz
del zc

b, a = des_bw_filter_lp(cutoff=2500)
xc0 = sg.lfilter(b, a, xcdm)
```

```
b, a = des_bw_filter_bp(low=2500, high=5000)
xc1 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=5000, high=7500)
xc2 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=7500, high=10000)
xc3 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=10000, high=12500)
xc4 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=12500, high=15000)
xc5 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=15000, high=17500)
xc6 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=17500, high=20000)
xc7 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_hp(cutoff=20000)
xc8 = sg.lfilter(b, a, xcdm)

sigs = [xc, pd.Series(xc0), pd.Series(xc1), pd.Series(xc2), pd.Series(xc3),
        pd.Series(xc4), pd.Series(xc5), pd.Series(xc6), pd.Series(xc7), pd.Series(

for i, sig in enumerate(sigs):
    X.loc[seg_id, 'mean_%d' % i] = sig.mean()
    X.loc[seg_id, 'std_%d' % i] = sig.std()
    X.loc[seg_id, 'max_%d' % i] = sig.max()
    X.loc[seg_id, 'min_%d' % i] = sig.min()

    X.loc[seg_id, 'mean_change_abs_%d' % i] = np.mean(np.diff(sig))
    X.loc[seg_id, 'mean_change_rate_%d' % i] = np.mean(np.nonzero((np.diff(sig) /
    X.loc[seg_id, 'abs_max_%d' % i] = np.abs(sig).max()
    X.loc[seg_id, 'abs_min_%d' % i] = np.abs(sig).min()

    X.loc[seg_id, 'std_first_50000_%d' % i] = sig[:50000].std()
    X.loc[seg_id, 'std_last_50000_%d' % i] = sig[-50000:].std()
    X.loc[seg_id, 'std_first_10000_%d' % i] = sig[:10000].std()
    X.loc[seg_id, 'std_last_10000_%d' % i] = sig[-10000:].std()

    X.loc[seg_id, 'avg_first_50000_%d' % i] = sig[:50000].mean()
    X.loc[seg_id, 'avg_last_50000_%d' % i] = sig[-50000:].mean()
    X.loc[seg_id, 'avg_first_10000_%d' % i] = sig[:10000].mean()
    X.loc[seg_id, 'avg_last_10000_%d' % i] = sig[-10000:].mean()
```

```
X.loc[seg_id, 'min_first_50000_%d' % i] = sig[:50000].min()
X.loc[seg_id, 'min_last_50000_%d' % i] = sig[-50000:].min()
X.loc[seg_id, 'min_first_10000_%d' % i] = sig[:10000].min()
X.loc[seg_id, 'min_last_10000_%d' % i] = sig[-10000:].min()

X.loc[seg_id, 'max_first_50000_%d' % i] = sig[:50000].max()
X.loc[seg_id, 'max_last_50000_%d' % i] = sig[-50000:].max()
X.loc[seg_id, 'max_first_10000_%d' % i] = sig[:10000].max()
X.loc[seg_id, 'max_last_10000_%d' % i] = sig[-10000:].max()

X.loc[seg_id, 'max_to_min_%d' % i] = sig.max() / np.abs(sig.min())
X.loc[seg_id, 'max_to_min_diff_%d' % i] = sig.max() - np.abs(sig.min())
X.loc[seg_id, 'count_big_%d' % i] = len(sig[np.abs(sig) > 500])
X.loc[seg_id, 'sum_%d' % i] = sig.sum()

X.loc[seg_id, 'mean_change_rate_first_50000_%d' % i] = np.mean(np.nonzero((np.
X.loc[seg_id, 'mean_change_rate_last_50000_%d' % i] = np.mean(np.nonzero((np.d
X.loc[seg_id, 'mean_change_rate_first_10000_%d' % i] = np.mean(np.nonzero((np.
X.loc[seg_id, 'mean_change_rate_last_10000_%d' % i] = np.mean(np.nonzero((np.d

X.loc[seg_id, 'q95_%d' % i] = np.quantile(sig, 0.95)
X.loc[seg_id, 'q99_%d' % i] = np.quantile(sig, 0.99)
X.loc[seg_id, 'q05_%d' % i] = np.quantile(sig, 0.05)
X.loc[seg_id, 'q01_%d' % i] = np.quantile(sig, 0.01)

X.loc[seg_id, 'abs_q95_%d' % i] = np.quantile(np.abs(sig), 0.95)
X.loc[seg_id, 'abs_q99_%d' % i] = np.quantile(np.abs(sig), 0.99)
X.loc[seg_id, 'abs_q05_%d' % i] = np.quantile(np.abs(sig), 0.05)
X.loc[seg_id, 'abs_q01_%d' % i] = np.quantile(np.abs(sig), 0.01)

X.loc[seg_id, 'trend_%d' % i] = add_trend_feature(sig)
X.loc[seg_id, 'abs_trend_%d' % i] = add_trend_feature(sig, abs_values=True)
X.loc[seg_id, 'abs_mean_%d' % i] = np.abs(sig).mean()
X.loc[seg_id, 'abs_std_%d' % i] = np.abs(sig).std()

X.loc[seg_id, 'mad_%d' % i] = sig.mad()
X.loc[seg_id, 'kurt_%d' % i] = sig.kurtosis()
X.loc[seg_id, 'skew_%d' % i] = sig.skew()
X.loc[seg_id, 'med_%d' % i] = sig.median()

X.loc[seg_id, 'Hilbert_mean_%d' % i] = np.abs(hilbert(sig)).mean()
X.loc[seg_id, 'Hann_window_mean'] = (convolve(xc, hann(150), mode='same') / su

X.loc[seg_id, 'classic_sta_lta1_mean_%d' % i] = classic_sta_lta(sig, 500, 1000
X.loc[seg_id, 'classic_sta_lta2_mean_%d' % i] = classic_sta_lta(sig, 5000, 100
X.loc[seg_id, 'classic_sta_lta3_mean_%d' % i] = classic_sta_lta(sig, 3333, 666
X.loc[seg_id, 'classic_sta_lta4_mean_%d' % i] = classic_sta_lta(sig, 10000, 25
        X.loc[seg_id, 'Moving_average_700_mean_%d' % i] = sig.rolling(window=700).mean
        X.loc[seg_id, 'Moving_average_1500_mean_%d' % i] = sig.rolling(window=1500).mea
        X.loc[seg_id, 'Moving_average_3000_mean_%d' % i] = sig.rolling(window=3000).mea
        X.loc[seg_id, 'Moving_average_6000_mean_%d' % i] = sig.rolling(window=6000).mea

        ewma = pd.Series.ewm
        X.loc[seg_id, 'exp_Moving_average_300_mean_%d' % i] = ewma(sig, span=300).mean
        X.loc[seg_id, 'exp_Moving_average_3000_mean_%d' % i] = ewma(sig, span=3000).mea
        X.loc[seg_id, 'exp_Moving_average_30000_mean_%d' % i] = ewma(sig, span=6000).me

        no_of_std = 2
        X.loc[seg_id, 'MA_700MA_std_mean_%d' % i] = sig.rolling(window=700).std().mean
        X.loc[seg_id, 'MA_700MA_BB_high_mean_%d' % i] = (
                    X.loc[seg_id, 'Moving_average_700_mean_%d' % i] + no_of_std * X.loc
        X.loc[seg_id, 'MA_700MA_BB_low_mean_%d' % i] = (
                    X.loc[seg_id, 'Moving_average_700_mean_%d' % i] - no_of_std * X.loc
        X.loc[seg_id, 'MA_400MA_std_mean_%d' % i] = sig.rolling(window=400).std().mean
        X.loc[seg_id, 'MA_400MA_BB_high_mean_%d' % i] = (
                    X.loc[seg_id, 'Moving_average_700_mean_%d' % i] + no_of_std * X.loc
        X.loc[seg_id, 'MA_400MA_BB_low_mean_%d' % i] = (
                    X.loc[seg_id, 'Moving_average_700_mean_%d' % i] - no_of_std * X.loc
        X.loc[seg_id, 'MA_1000MA_std_mean_%d' % i] = sig.rolling(window=1000).std().mea

        X.loc[seg_id, 'iqr_%d' % i] = np.subtract(*np.percentile(sig, [75, 25]))
        X.loc[seg_id, 'q999_%d' % i] = np.quantile(sig, 0.999)
        X.loc[seg_id, 'q001_%d' % i] = np.quantile(sig, 0.001)
        X.loc[seg_id, 'ave10_%d' % i] = stats.trim_mean(sig, 0.1)

    for windows in [10, 100, 1000]:
        x_roll_std = xc.rolling(windows).std().dropna().values
        x_roll_mean = xc.rolling(windows).mean().dropna().values

        X.loc[seg_id, 'ave_roll_std_' + str(windows)] = x_roll_std.mean()
        X.loc[seg_id, 'std_roll_std_' + str(windows)] = x_roll_std.std()
        X.loc[seg_id, 'max_roll_std_' + str(windows)] = x_roll_std.max()
        X.loc[seg_id, 'min_roll_std_' + str(windows)] = x_roll_std.min()
        X.loc[seg_id, 'q01_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0.01)
        X.loc[seg_id, 'q05_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0.05)
        X.loc[seg_id, 'q95_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0.95)
        X.loc[seg_id, 'q99_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0.99)
        X.loc[seg_id, 'av_change_abs_roll_std_' + str(windows)] = np.mean(np.diff(x_rol
        X.loc[seg_id, 'av_change_rate_roll_std_' + str(windows)] = np.mean(
            np.nonzero((np.diff(x_roll_std) / x_roll_std[:-1]))[0])
        X.loc[seg_id, 'abs_max_roll_std_' + str(windows)] = np.abs(x_roll_std).max()

        X.loc[seg_id, 'ave_roll_mean_' + str(windows)] = x_roll_mean.mean()
        X.loc[seg_id, 'std_roll_mean_' + str(windows)] = x_roll_mean.std()
        X.loc[seg_id, 'max_roll_mean_' + str(windows)] = x_roll_mean.max()
```

```
                X.loc[seg_id, 'min_roll_mean_' + str(windows)] = x_roll_mean.min()
                X.loc[seg_id, 'q01_roll_mean_' + str(windows)] = np.quantile(x_roll_mean, 0.01
                X.loc[seg_id, 'q05_roll_mean_' + str(windows)] = np.quantile(x_roll_mean, 0.05
                X.loc[seg_id, 'q95_roll_mean_' + str(windows)] = np.quantile(x_roll_mean, 0.95
                X.loc[seg_id, 'q99_roll_mean_' + str(windows)] = np.quantile(x_roll_mean, 0.99
                X.loc[seg_id, 'av_change_abs_roll_mean_' + str(windows)] = np.mean(np.diff(x_r
                X.loc[seg_id, 'av_change_rate_roll_mean_' + str(windows)] = np.mean(
                    np.nonzero((np.diff(x_roll_mean) / x_roll_mean[:-1]))[0])
                X.loc[seg_id, 'abs_max_roll_mean_' + str(windows)] = np.abs(x_roll_mean).max()

            return X

In [8]: def build_fields(proc_id):
            success = 1
            count = 0
            try:
                seg_st = int(NUM_SEG_PER_PROC * proc_id)
                train_df = pd.read_csv(os.path.join(DATA_DIR, 'raw_data_%d.csv' % proc_id), dty
                len_df = len(train_df.index)
                start_indices = (np.loadtxt(fname=os.path.join(OUTPUT_DIR, 'start_indices_4k.c
                train_X = pd.DataFrame(dtype=np.float64)
                train_y = pd.DataFrame(dtype=np.float64, columns=['time_to_failure'])
                t0 = time.time()

                for seg_id, start_idx in zip(range(seg_st, seg_st + NUM_SEG_PER_PROC), start_i
                    end_idx = np.int32(start_idx + 150000)
                    print('working: %d, %d, %d to %d of %d' % (proc_id, seg_id, start_idx, end
                    seg = train_df.iloc[start_idx: end_idx]
                    # train_X = create_features_pk_det(seg_id, seg, train_X, start_idx, end_id
                    train_X = create_features(seg_id, seg, train_X, start_idx, end_idx)
                    train_y.loc[seg_id, 'time_to_failure'] = seg['time_to_failure'].values[-1]

                    if count == 10:
                        print('saving: %d, %d to %d' % (seg_id, start_idx, end_idx))
                        train_X.to_csv('train_x_%d.csv' % proc_id, index=False)

                        train_y.to_csv('train_y_%d.csv' % proc_id, index=False)

                    count += 1

                print('final_save, process id: %d, loop time: %.2f for %d iterations' % (proc_
                train_X.to_csv(os.path.join(OUTPUT_DIR, 'train_x_%d.csv' % proc_id), index=Fal
                train_y.to_csv(os.path.join(OUTPUT_DIR, 'train_y_%d.csv' % proc_id), index=Fal

            except:
                print(traceback.format_exc())
                success = 0
            return success  # 1 on success, 0 if fail

In [9]: #for multiprocessing
        def run_mp_build():
            t0 = time.time()
            num_proc = NUM_THREADS
            pool = mp.Pool(processes=num_proc)
            results = [pool.apply_async(build_fields, args=(pid, )) for pid in range(NUM_THREAI
            output = [p.get() for p in results]
            num_built = sum(output)
            pool.close()
            pool.join()
            print(num_built)
            print('Run time: %.2f' % (time.time() - t0))

In [10]: def join_mp_build():
             df0 = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_x_%d.csv' % 0))
             df1 = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_y_%d.csv' % 0))

             for i in range(1, NUM_THREADS):
                 print('working %d' % i)
                 temp = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_x_%d.csv' % i))
                 df0 = df0.append(temp)

                 temp = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_y_%d.csv' % i))
                 df1 = df1.append(temp)

             df0.to_csv(os.path.join(OUTPUT_DIR, 'train_x.csv'), index=False)
             df1.to_csv(os.path.join(OUTPUT_DIR, 'train_y.csv'), index=False)

In [11]: from tqdm.auto import tqdm
         def build_test_fields():
             train_X = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_x.csv'))
             try:
                 train_X.drop(labels=['seg_id', 'seg_start', 'seg_end'], axis=1, inplace=True)
             except:
                 pass

             submission = pd.read_csv(os.path.join(DATA_DIR, 'sample_submission.csv'), index_c
             test_X = pd.DataFrame(columns=train_X.columns, dtype=np.float64, index=submission

             print('start for loop')
             count = 0
             for seg_id in tqdm(test_X.index):  # just tqdm in IDE
                 seg = pd.read_csv(os.path.join(DATA_DIR, 'Untitled Folder/', str(seg_id) + '.
                 # train_X = create_features_pk_det(seg_id, seg, train_X, start_idx, end_idx)
                 test_X = create_features(seg_id, seg, test_X, 0, 0)
                 if count % 100 == 0:
                     print('working', seg_id)
                 count += 1

             test_X.to_csv(os.path.join(OUTPUT_DIR, 'test_x.csv'), index=False)
```

```
In [12]: #standardization
         def scale_fields(fn_train='train_x.csv', fn_test='test_x.csv',
                          fn_out_train='scaled_train_X.csv' , fn_out_test='scaled_test_X.csv')
             train_X = pd.read_csv(os.path.join(OUTPUT_DIR, fn_train))
             try:
                 train_X.drop(labels=['seg_id', 'seg_start', 'seg_end'], axis=1, inplace=True)
             except:
                 pass
             test_X = pd.read_csv(os.path.join(OUTPUT_DIR, fn_test))

             print('start scaler')
             scaler = StandardScaler()
             scaler.fit(train_X)
             scaled_train_X = pd.DataFrame(scaler.transform(train_X), columns=train_X.columns)
             scaled_test_X = pd.DataFrame(scaler.transform(test_X), columns=test_X.columns)

             scaled_train_X.to_csv(os.path.join(OUTPUT_DIR, fn_out_train), index=False)
             scaled_test_X.to_csv(os.path.join(OUTPUT_DIR, fn_out_test), index=False)
```

```
In [ ]: split_raw_data()
```

```
In [20]: build_rnd_idxs()
```

```
[10804991 40754581 61152051 51046969 26130885 37920772 36775305  7675825]
[68761251 51632120 86559696 90282599 60663556 85061082 95027462 23825753]
67619 99994297
[16837712 49519822 86613139  3210689 98148542 31101347  1090339 72122324]
[79426720 43809532 43249236 23265647 44502411 86787131 90136975 34661131]
26666 99956067
[63044133  2442657 90777691 16268569 63311688 90814034 75756302 37813113]
[91763196  7353084 29675563 12721978 64093656 39100415  2453472 56466376]
19628 99986179
[61673177 33536021 43935586 94121751  3158245 18377637 64912898 52164547]
[98328338 73239137 19836471 25502780 59800782 58627599 55588218 24985417]
2272 99981324
[59879818 56182569 67051701 16143352 53734196 57460600 55941981 67579513]
[11170509 67106840 93093344  4809245 73117841 87221360   829083 51383467]
26823 99942141
[26548514  5175447 39498226 33934210 76764021 34939489 82316461 79515410]
[67651567 76925054 97654318 99863711 49392180 70557795 10896601 75562170]
71058 99981201
```

```
In [ ]: run_mp_build()

In [30]: join_mp_build()

working 1
working 2
working 3
working 4
working 5


In [ ]: build_test_fields()

In [14]: scale_fields()

start scaler
```

3.0.2 Featurizing original data

```
In [8]: rows = 150000
        segments = int(np.floor(train.shape[0] / rows))
        X_check = pd.DataFrame(index=range(segments), dtype=np.float64)
        y_check = pd.DataFrame(index=range(segments), dtype=np.float64,
                               columns=['time_to_failure'])

In [9]: #building features for original 4194 features to plot and check the predictions
        import warnings
        warnings.filterwarnings("ignore")

        from tqdm.auto import tqdm
        for seg_id in tqdm(range(segments)):
            seg = train.iloc[seg_id*rows:seg_id*rows+rows]
            xc = pd.Series(seg['acoustic_data'].values)
            xcdm = xc - np.mean(xc)
            y = seg['time_to_failure'].values[-1]
            y_check.loc[seg_id, 'time_to_failure'] = y

            b, a = des_bw_filter_lp(cutoff=18000)
            xcz = sg.lfilter(b, a, xcdm)

            zc = np.fft.fft(xcz)
            zc = zc[:MAX_FREQ_IDX]

            # FFT transform values
            realFFT = np.real(zc)
            imagFFT = np.imag(zc)

            freq_bands = [x for x in range(0, MAX_FREQ_IDX, FREQ_STEP)]
```

```
magFFT = np.sqrt(realFFT ** 2 + imagFFT ** 2)
phzFFT = np.arctan(imagFFT / realFFT)
phzFFT[phzFFT == -np.inf] = -np.pi / 2.0
phzFFT[phzFFT == np.inf] = np.pi / 2.0
phzFFT = np.nan_to_num(phzFFT)

for freq in freq_bands:
    X_check.loc[seg_id, 'FFT_Mag_01q%d' % freq] = np.quantile(magFFT[freq: freq + F
    X_check.loc[seg_id, 'FFT_Mag_10q%d' % freq] = np.quantile(magFFT[freq: freq + F
    X_check.loc[seg_id, 'FFT_Mag_90q%d' % freq] = np.quantile(magFFT[freq: freq + F
    X_check.loc[seg_id, 'FFT_Mag_99q%d' % freq] = np.quantile(magFFT[freq: freq + F
    X_check.loc[seg_id, 'FFT_Mag_mean%d' % freq] = np.mean(magFFT[freq: freq + FREQ
    X_check.loc[seg_id, 'FFT_Mag_std%d' % freq] = np.std(magFFT[freq: freq + FREQ_S
    X_check.loc[seg_id, 'FFT_Mag_max%d' % freq] = np.max(magFFT[freq: freq + FREQ_S

    X_check.loc[seg_id, 'FFT_Phz_mean%d' % freq] = np.mean(phzFFT[freq: freq + FREQ
    X_check.loc[seg_id, 'FFT_Phz_std%d' % freq] = np.std(phzFFT[freq: freq + FREQ_S

X_check.loc[seg_id, 'FFT_Rmean'] = realFFT.mean()
X_check.loc[seg_id, 'FFT_Rstd'] = realFFT.std()
X_check.loc[seg_id, 'FFT_Rmax'] = realFFT.max()
X_check.loc[seg_id, 'FFT_Rmin'] = realFFT.min()
X_check.loc[seg_id, 'FFT_Imean'] = imagFFT.mean()
X_check.loc[seg_id, 'FFT_Istd'] = imagFFT.std()
X_check.loc[seg_id, 'FFT_Imax'] = imagFFT.max()
X_check.loc[seg_id, 'FFT_Imin'] = imagFFT.min()

X_check.loc[seg_id, 'FFT_Rmean_first_6000'] = realFFT[:6000].mean()
X_check.loc[seg_id, 'FFT_Rstd__first_6000'] = realFFT[:6000].std()
X_check.loc[seg_id, 'FFT_Rmax_first_6000'] = realFFT[:6000].max()
X_check.loc[seg_id, 'FFT_Rmin_first_6000'] = realFFT[:6000].min()
X_check.loc[seg_id, 'FFT_Rmean_first_18000'] = realFFT[:18000].mean()
X_check.loc[seg_id, 'FFT_Rstd_first_18000'] = realFFT[:18000].std()
X_check.loc[seg_id, 'FFT_Rmax_first_18000'] = realFFT[:18000].max()
X_check.loc[seg_id, 'FFT_Rmin_first_18000'] = realFFT[:18000].min()

del xcz
del zc

b, a = des_bw_filter_lp(cutoff=2500)
xc0 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=2500, high=5000)
xc1 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=5000, high=7500)
xc2 = sg.lfilter(b, a, xcdm)
```

```
b, a = des_bw_filter_bp(low=7500, high=10000)
xc3 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=10000, high=12500)
xc4 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=12500, high=15000)
xc5 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=15000, high=17500)
xc6 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_bp(low=17500, high=20000)
xc7 = sg.lfilter(b, a, xcdm)

b, a = des_bw_filter_hp(cutoff=20000)
xc8 = sg.lfilter(b, a, xcdm)

sigs = [xc, pd.Series(xc0), pd.Series(xc1), pd.Series(xc2), pd.Series(xc3),
        pd.Series(xc4), pd.Series(xc5), pd.Series(xc6), pd.Series(xc7), pd.Series(

for i, sig in enumerate(sigs):
    X_check.loc[seg_id, 'mean_%d' % i] = sig.mean()
    X_check.loc[seg_id, 'std_%d' % i] = sig.std()
    X_check.loc[seg_id, 'max_%d' % i] = sig.max()
    X_check.loc[seg_id, 'min_%d' % i] = sig.min()

    X_check.loc[seg_id, 'mean_change_abs_%d' % i] = np.mean(np.diff(sig))
    X_check.loc[seg_id, 'mean_change_rate_%d' % i] = np.mean(np.nonzero((np.diff(s
    X_check.loc[seg_id, 'abs_max_%d' % i] = np.abs(sig).max()
    X_check.loc[seg_id, 'abs_min_%d' % i] = np.abs(sig).min()

    X_check.loc[seg_id, 'std_first_50000_%d' % i] = sig[:50000].std()
    X_check.loc[seg_id, 'std_last_50000_%d' % i] = sig[-50000:].std()
    X_check.loc[seg_id, 'std_first_10000_%d' % i] = sig[:10000].std()
    X_check.loc[seg_id, 'std_last_10000_%d' % i] = sig[-10000:].std()

    X_check.loc[seg_id, 'avg_first_50000_%d' % i] = sig[:50000].mean()
    X_check.loc[seg_id, 'avg_last_50000_%d' % i] = sig[-50000:].mean()
    X_check.loc[seg_id, 'avg_first_10000_%d' % i] = sig[:10000].mean()
    X_check.loc[seg_id, 'avg_last_10000_%d' % i] = sig[-10000:].mean()

    X_check.loc[seg_id, 'min_first_50000_%d' % i] = sig[:50000].min()
    X_check.loc[seg_id, 'min_last_50000_%d' % i] = sig[-50000:].min()
    X_check.loc[seg_id, 'min_first_10000_%d' % i] = sig[:10000].min()
    X_check.loc[seg_id, 'min_last_10000_%d' % i] = sig[-10000:].min()

    X_check.loc[seg_id, 'max_first_50000_%d' % i] = sig[:50000].max()
    X_check.loc[seg_id, 'max_last_50000_%d' % i] = sig[-50000:].max()
    X_check.loc[seg_id, 'max_first_10000_%d' % i] = sig[:10000].max()
    X_check.loc[seg_id, 'max_last_10000_%d' % i] = sig[-10000:].max()

    X_check.loc[seg_id, 'max_to_min_%d' % i] = sig.max() / np.abs(sig.min())
    X_check.loc[seg_id, 'max_to_min_diff_%d' % i] = sig.max() - np.abs(sig.min())
    X_check.loc[seg_id, 'count_big_%d' % i] = len(sig[np.abs(sig) > 500])
    X_check.loc[seg_id, 'sum_%d' % i] = sig.sum()

    X_check.loc[seg_id, 'mean_change_rate_first_50000_%d' % i] = np.mean(np.nonzer
    X_check.loc[seg_id, 'mean_change_rate_last_50000_%d' % i] = np.mean(np.nonzero
    X_check.loc[seg_id, 'mean_change_rate_first_10000_%d' % i] = np.mean(np.nonzer
    X_check.loc[seg_id, 'mean_change_rate_last_10000_%d' % i] = np.mean(np.nonzero

    X_check.loc[seg_id, 'q95_%d' % i] = np.quantile(sig, 0.95)
    X_check.loc[seg_id, 'q99_%d' % i] = np.quantile(sig, 0.99)
    X_check.loc[seg_id, 'q05_%d' % i] = np.quantile(sig, 0.05)
    X_check.loc[seg_id, 'q01_%d' % i] = np.quantile(sig, 0.01)

    X_check.loc[seg_id, 'abs_q95_%d' % i] = np.quantile(np.abs(sig), 0.95)
    X_check.loc[seg_id, 'abs_q99_%d' % i] = np.quantile(np.abs(sig), 0.99)
    X_check.loc[seg_id, 'abs_q05_%d' % i] = np.quantile(np.abs(sig), 0.05)
    X_check.loc[seg_id, 'abs_q01_%d' % i] = np.quantile(np.abs(sig), 0.01)

    X_check.loc[seg_id, 'trend_%d' % i] = add_trend_feature(sig)
    X_check.loc[seg_id, 'abs_trend_%d' % i] = add_trend_feature(sig, abs_values=Tr
    X_check.loc[seg_id, 'abs_mean_%d' % i] = np.abs(sig).mean()
    X_check.loc[seg_id, 'abs_std_%d' % i] = np.abs(sig).std()

    X_check.loc[seg_id, 'mad_%d' % i] = sig.mad()
    X_check.loc[seg_id, 'kurt_%d' % i] = sig.kurtosis()
    X_check.loc[seg_id, 'skew_%d' % i] = sig.skew()
    X_check.loc[seg_id, 'med_%d' % i] = sig.median()

    X_check.loc[seg_id, 'Hilbert_mean_%d' % i] = np.abs(hilbert(sig)).mean()
    X_check.loc[seg_id, 'Hann_window_mean'] = (convolve(xc, hann(150), mode='same'

    X_check.loc[seg_id, 'classic_sta_lta1_mean_%d' % i] = classic_sta_lta(sig, 500
    X_check.loc[seg_id, 'classic_sta_lta2_mean_%d' % i] = classic_sta_lta(sig, 500
    X_check.loc[seg_id, 'classic_sta_lta3_mean_%d' % i] = classic_sta_lta(sig, 333
    X_check.loc[seg_id, 'classic_sta_lta4_mean_%d' % i] = classic_sta_lta(sig, 100

    X_check.loc[seg_id, 'Moving_average_700_mean_%d' % i] = sig.rolling(window=700
    X_check.loc[seg_id, 'Moving_average_1500_mean_%d' % i] = sig.rolling(window=15
    X_check.loc[seg_id, 'Moving_average_3000_mean_%d' % i] = sig.rolling(window=30
    X_check.loc[seg_id, 'Moving_average_6000_mean_%d' % i] = sig.rolling(window=60

    ewma = pd.Series.ewm
```

```
        X_check.loc[seg_id, 'exp_Moving_average_300_mean_%d' % i] = ewma(sig, span=300)
        X_check.loc[seg_id, 'exp_Moving_average_3000_mean_%d' % i] = ewma(sig, span=300
        X_check.loc[seg_id, 'exp_Moving_average_30000_mean_%d' % i] = ewma(sig, span=60

        no_of_std = 2
        X_check.loc[seg_id, 'MA_700MA_std_mean_%d' % i] = sig.rolling(window=700).std()
        X_check.loc[seg_id, 'MA_700MA_BB_high_mean_%d' % i] = (
                    X_check.loc[seg_id, 'Moving_average_700_mean_%d' % i] + no_of_std *
        X_check.loc[seg_id, 'MA_700MA_BB_low_mean_%d' % i] = (
                    X_check.loc[seg_id, 'Moving_average_700_mean_%d' % i] - no_of_std *
        X_check.loc[seg_id, 'MA_400MA_std_mean_%d' % i] = sig.rolling(window=400).std()
        X_check.loc[seg_id, 'MA_400MA_BB_high_mean_%d' % i] = (
                    X_check.loc[seg_id, 'Moving_average_700_mean_%d' % i] + no_of_std *
        X_check.loc[seg_id, 'MA_400MA_BB_low_mean_%d' % i] = (
                    X_check.loc[seg_id, 'Moving_average_700_mean_%d' % i] - no_of_std *
        X_check.loc[seg_id, 'MA_1000MA_std_mean_%d' % i] = sig.rolling(window=1000).std

        X_check.loc[seg_id, 'iqr_%d' % i] = np.subtract(*np.percentile(sig, [75, 25]))
        X_check.loc[seg_id, 'q999_%d' % i] = np.quantile(sig, 0.999)
        X_check.loc[seg_id, 'q001_%d' % i] = np.quantile(sig, 0.001)
        X_check.loc[seg_id, 'ave10_%d' % i] = stats.trim_mean(sig, 0.1)

    for windows in [10, 100, 1000]:
        x_roll_std = xc.rolling(windows).std().dropna().values
        x_roll_mean = xc.rolling(windows).mean().dropna().values

        X_check.loc[seg_id, 'ave_roll_std_' + str(windows)] = x_roll_std.mean()
        X_check.loc[seg_id, 'std_roll_std_' + str(windows)] = x_roll_std.std()
        X_check.loc[seg_id, 'max_roll_std_' + str(windows)] = x_roll_std.max()
        X_check.loc[seg_id, 'min_roll_std_' + str(windows)] = x_roll_std.min()
        X_check.loc[seg_id, 'q01_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0
        X_check.loc[seg_id, 'q05_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0
        X_check.loc[seg_id, 'q95_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0
        X_check.loc[seg_id, 'q99_roll_std_' + str(windows)] = np.quantile(x_roll_std, 0
        X_check.loc[seg_id, 'av_change_abs_roll_std_' + str(windows)] = np.mean(np.dif
        X_check.loc[seg_id, 'av_change_rate_roll_std_' + str(windows)] = np.mean(
            np.nonzero((np.diff(x_roll_std) / x_roll_std[:-1]))[0])
        X_check.loc[seg_id, 'abs_max_roll_std_' + str(windows)] = np.abs(x_roll_std).ma

        X_check.loc[seg_id, 'ave_roll_mean_' + str(windows)] = x_roll_mean.mean()
        X_check.loc[seg_id, 'std_roll_mean_' + str(windows)] = x_roll_mean.std()
        X_check.loc[seg_id, 'max_roll_mean_' + str(windows)] = x_roll_mean.max()
        X_check.loc[seg_id, 'min_roll_mean_' + str(windows)] = x_roll_mean.min()
        X_check.loc[seg_id, 'q01_roll_mean_' + str(windows)] = np.quantile(x_roll_mean
        X_check.loc[seg_id, 'q05_roll_mean_' + str(windows)] = np.quantile(x_roll_mean
        X_check.loc[seg_id, 'q95_roll_mean_' + str(windows)] = np.quantile(x_roll_mean
        X_check.loc[seg_id, 'q99_roll_mean_' + str(windows)] = np.quantile(x_roll_mean
        X_check.loc[seg_id, 'av_change_abs_roll_mean_' + str(windows)] = np.mean(np.di
        X_check.loc[seg_id, 'av_change_rate_roll_mean_' + str(windows)] = np.mean(
            np.nonzero((np.diff(x_roll_mean) / x_roll_mean[:-1]))[0])
        X_check.loc[seg_id, 'abs_max_roll_mean_' + str(windows)] = np.abs(x_roll_mean)
```

```
HBox(children=(IntProgress(value=0, max=4194), HTML(value='')))
```

```
In [11]: scaler = StandardScaler()
         scaler.fit(X_check)
         scaled_check_X = pd.DataFrame(scaler.transform(X_check), columns=X_check.columns)

In [13]: scaled_check_X.to_csv(os.path.join(OUTPUT_DIR, 'scaled_check_X.csv'), index=False)

In [14]: scaled_check_X=pd.read_csv('scaled_check_X.csv')
         scaled_check_X.head()
```

```
Out[14]:    FFT_Mag_01q0  FFT_Mag_10q0  FFT_Mag_90q0  FFT_Mag_99q0  FFT_Mag_mean0  \
         0     -0.060473     -0.085420     -0.137072     -0.152365      -0.121781
         1     -0.102473     -0.036434      0.028897      0.014212       0.004478
         2     -0.049900     -0.039858     -0.003688     -0.003713      -0.014248
         3      0.013489     -0.024736      0.154143      0.301586       0.108312
         4     -0.066028     -0.029450     -0.015406      0.031369      -0.017600

            FFT_Mag_std0  FFT_Mag_max0  FFT_Phz_mean0  FFT_Phz_std0  FFT_Mag_01q2500  \
         0     -0.126132      0.218116       0.950155     -1.864500        -0.140295
         1      0.014293     -0.302923      -0.030805      0.967366         0.023442
         2     -0.003003      0.015514      -1.484378     -0.137772         0.131602
         3      0.238146      0.053061       0.986414     -0.535139         0.037916
         4     -0.006492     -0.272360      -0.609263     -0.875324         0.050317

            ...  std_roll_mean_1000  max_roll_mean_1000  min_roll_mean_1000  \
         0  ...            0.268470           -0.004742            0.178278
         1  ...           -0.141264            0.007341           -0.025387
         2  ...            0.085078            0.099556            0.245184
         3  ...            0.083085            0.068076            0.105059
         4  ...           -0.164151            0.138032            0.187535

            q01_roll_mean_1000  q05_roll_mean_1000  q95_roll_mean_1000  \
         0            0.287332            0.965402            1.509153
         1            0.622391            0.842747            0.522428
         2            0.634878            1.207106            1.530919
         3            0.770151            1.160208            1.432972
         4            1.040695            1.557034            1.393068

            q99_roll_mean_1000  av_change_abs_roll_mean_1000  \
         0            0.885262                     -0.631300
```

```
1            0.294357                        -0.912054
2            0.889790                         0.441128
3            0.815078                        -0.949994
4            0.901110                         0.595416

   av_change_rate_roll_mean_1000  abs_max_roll_mean_1000
0                      -1.832422               -0.004742
1                      -0.890022                0.007341
2                       0.639209                0.099556
3                      -1.097513                0.068076
4                      -0.465464                0.138032

[5 rows x 865 columns]
```

4 Feature set 2

```
In [5]: #http://gilestrolab.github.io/pyrem/pyrem.univariate.html
        #http://pyeeg.sourceforge.net/
        #returns acivity, mobility and complexity of the signal
        def hjorth(a):

            first_deriv = np.diff(a)
            second_deriv = np.diff(a,2)

            var_zero = np.mean(a ** 2)
            var_d1 = np.mean(first_deriv ** 2)
            var_d2 = np.mean(second_deriv ** 2)

            activity = var_zero
            mobility = np.sqrt(var_d1 / var_zero)
            complexity = np.sqrt(var_d2 / var_d1) / mobility

            return activity, mobility, complexity
```

```
In [6]: import scipy
        def create_features_set2(seg_id, seg, X, st, end):
            try:
                X.loc[seg_id, 'seg_id'] = np.int32(seg_id)
                X.loc[seg_id, 'seg_start'] = np.int32(st)
                X.loc[seg_id, 'seg_end'] = np.int32(end)
            except:
                pass

            x = seg['acoustic_data'].values

            X.loc[seg_id,'kstat_1'] = sp.stats.kstat(x, 1)
            X.loc[seg_id,'kstat_2'] = sp.stats.kstat(x, 2)
```

```
X.loc[seg_id,'kstat_3'] = sp.stats.kstat(x, 3)
X.loc[seg_id,'kstat_4'] = sp.stats.kstat(x, 4)
X.loc[seg_id,'moment_1'] = sp.stats.moment(x, 1)
X.loc[seg_id,'moment_2'] = sp.stats.moment(x, 2)
X.loc[seg_id,'moment_3'] = sp.stats.moment(x, 3)
X.loc[seg_id,'moment_4'] = sp.stats.moment(x, 4)

X.loc[seg_id,'abs_energy'] = feature_calculators.abs_energy(x)
X.loc[seg_id,'abs_sum_of_changes'] = feature_calculators.absolute_sum_of_changes(x
X.loc[seg_id,'count_above_mean'] = feature_calculators.count_above_mean(x)
X.loc[seg_id,'count_below_mean'] = feature_calculators.count_below_mean(x)
X.loc[seg_id,'mean_abs_change'] = feature_calculators.mean_abs_change(x)
X.loc[seg_id,'mean_change'] = feature_calculators.mean_change(x)
X.loc[seg_id,'var_larger_than_std_dev'] = feature_calculators.variance_larger_than
X.loc[seg_id,'range_minf_m4000'] = feature_calculators.range_count(x, -np.inf, -40

X.loc[seg_id,'range_m4000_m3000'] = feature_calculators.range_count(x, -4000, -300
X.loc[seg_id,'range_m3000_m2000'] = feature_calculators.range_count(x, -3000, -200
X.loc[seg_id,'range_m2000_m1000'] = feature_calculators.range_count(x, -2000, -100
X.loc[seg_id,'range_m1000_0'] = feature_calculators.range_count(x, -1000, 0)
X.loc[seg_id,'range_0_p1000'] = feature_calculators.range_count(x, 0, 1000)
X.loc[seg_id,'range_p1000_p2000'] = feature_calculators.range_count(x, 1000, 2000)
X.loc[seg_id,'range_p2000_p3000'] = feature_calculators.range_count(x, 2000, 3000)
X.loc[seg_id,'range_p3000_p4000'] = feature_calculators.range_count(x, 3000, 4000)
X.loc[seg_id,'range_p4000_pinf'] = feature_calculators.range_count(x, 4000, np.inf

X.loc[seg_id,'ratio_unique_values'] = feature_calculators.ratio_value_number_to_ti
X.loc[seg_id,'first_loc_min'] = feature_calculators.first_location_of_minimum(x)
X.loc[seg_id,'first_loc_max'] = feature_calculators.first_location_of_maximum(x)
X.loc[seg_id,'last_loc_min'] = feature_calculators.last_location_of_minimum(x)
X.loc[seg_id,'last_loc_max'] = feature_calculators.last_location_of_maximum(x)
X.loc[seg_id,'time_rev_asym_stat_10'] = feature_calculators.time_reversal_asymmetr
X.loc[seg_id,'time_rev_asym_stat_100'] = feature_calculators.time_reversal_asymmet
X.loc[seg_id,'time_rev_asym_stat_1000'] = feature_calculators.time_reversal_asymme
X.loc[seg_id,'autocorrelation_5'] = feature_calculators.autocorrelation(x, 5)
X.loc[seg_id,'autocorrelation_10'] = feature_calculators.autocorrelation(x, 10)
X.loc[seg_id,'autocorrelation_50'] = feature_calculators.autocorrelation(x, 50)
X.loc[seg_id,'autocorrelation_100'] = feature_calculators.autocorrelation(x, 100)
X.loc[seg_id,'autocorrelation_1000'] = feature_calculators.autocorrelation(x, 1000
X.loc[seg_id,'c3_5'] = feature_calculators.c3(x, 5)
X.loc[seg_id,'c3_10'] = feature_calculators.c3(x, 10)
X.loc[seg_id,'c3_100'] = feature_calculators.c3(x, 100)
X.loc[seg_id,'long_strk_above_mean'] = feature_calculators.longest_strike_above_me
X.loc[seg_id,'long_strk_below_mean'] = feature_calculators.longest_strike_below_me
X.loc[seg_id,'cid_ce_0'] = feature_calculators.cid_ce(x, 0)
X.loc[seg_id,'cid_ce_1'] = feature_calculators.cid_ce(x, 1)
X.loc[seg_id,'binned_entropy_5'] = feature_calculators.binned_entropy(x, 5)
X.loc[seg_id,'binned_entropy_10'] = feature_calculators.binned_entropy(x, 10)
```

```
            X.loc[seg_id,'binned_entropy_20'] = feature_calculators.binned_entropy(x, 20)
            X.loc[seg_id,'binned_entropy_50'] = feature_calculators.binned_entropy(x, 50)
            X.loc[seg_id,'binned_entropy_80'] = feature_calculators.binned_entropy(x, 80)
            X.loc[seg_id,'binned_entropy_100'] = feature_calculators.binned_entropy(x, 100)
            X.loc[seg_id,'num_crossing_0'] = feature_calculators.number_crossing_m(x, 0)
            X.loc[seg_id,'num_peaks_10'] = feature_calculators.number_peaks(x, 10)
            X.loc[seg_id,'num_peaks_50'] = feature_calculators.number_peaks(x, 50)
            X.loc[seg_id,'num_peaks_100'] = feature_calculators.number_peaks(x, 100)
            X.loc[seg_id,'num_peaks_500'] = feature_calculators.number_peaks(x, 500)
            X.loc[seg_id,'spkt_welch_density_1'] = list(feature_calculators.spkt_welch_density
            X.loc[seg_id,'spkt_welch_density_10'] = list(feature_calculators.spkt_welch_density
            X.loc[seg_id,'spkt_welch_density_50'] = list(feature_calculators.spkt_welch_density
            X.loc[seg_id,'spkt_welch_density_100'] = list(feature_calculators.spkt_welch_densi

            X.loc[seg_id,'time_rev_asym_stat_1'] = feature_calculators.time_reversal_asymmetry
            X.loc[seg_id,'time_rev_asym_stat_10'] = feature_calculators.time_reversal_asymmetr
            X.loc[seg_id,'time_rev_asym_stat_100'] = feature_calculators.time_reversal_asymmet



            X.loc[seg_id, 'hjorth_0'] =hjorth(x)[0]
            X.loc[seg_id, 'hjorth_1'] =hjorth(x)[1]
            X.loc[seg_id, 'hjorth_2'] =hjorth(x)[2]
            #X.loc[seg_id, 'dfa'] =dfa(x, Ave=None, L=None)

            #returns the peak of the signal
            peaks=scipy.signal.find_peaks(x,100)[1]['peak_heights']
            X.loc[seg_id, 'peak_count']=len(peaks)
            X.loc[seg_id, 'peak_std']=np.std(peaks)
            X.loc[seg_id, 'peak_mean']=np.mean(peaks)

            return X

In [7]: def build_fields2(proc_id):
            success = 1
            count = 0
            try:
                seg_st = int(NUM_SEG_PER_PROC * proc_id)
                train_df = pd.read_csv(os.path.join(DATA_DIR, 'raw_data_%d.csv' % proc_id), dty
                len_df = len(train_df.index)
                start_indices = (np.loadtxt(fname=os.path.join(OUTPUT_DIR, 'start_indices_4k.c
                train_X = pd.DataFrame(dtype=np.float64)
                train_y = pd.DataFrame(dtype=np.float64, columns=['time_to_failure'])
                t0 = time.time()

                for seg_id, start_idx in zip(range(seg_st, seg_st + NUM_SEG_PER_PROC), start_i
                    end_idx = np.int32(start_idx + 150000)
                    print('working: %d, %d, %d to %d of %d' % (proc_id, seg_id, start_idx, end
                    seg = train_df.iloc[start_idx: end_idx]
                    # train_X = create_features_pk_det(seg_id, seg, train_X, start_idx, end_id
                    train_X = create_features_set2(seg_id, seg, train_X, start_idx, end_idx)
                    train_y.loc[seg_id, 'time_to_failure'] = seg['time_to_failure'].values[-1]

                    #if count == 10:
                    #    print('saving: %d, %d to %d' % (seg_id, start_idx, end_idx))
                    #    train_X.to_csv('train_x_%d.csv' % proc_id, index=False)
                    #    train_y.to_csv('train_y_%d.csv' % proc_id, index=False)

                    #count += 1

                print('final_save, process id: %d, loop time: %.2f for %d iterations' % (proc_
                train_X.to_csv(os.path.join(OUTPUT_DIR, 'train_x2_%d.csv' % proc_id), index=Fal
                #train_y.to_csv(os.path.join(OUTPUT_DIR, 'train_y_%d.csv' % proc_id), index=Fa

            except:
                print(traceback.format_exc())
                success = 0

            return success  # 1 on success, 0 if fail
```

```
In [13]: def run_mp_build2():
             t0 = time.time()
             num_proc = NUM_THREADS
             pool = mp.Pool(processes=num_proc)
             results = [pool.apply_async(build_fields2, args=(pid, )) for pid in range(NUM_THR
             output = [p.get() for p in results]
             num_built = sum(output)
             pool.close()
             pool.join()
             print(num_built)
             print('Run time: %.2f' % (time.time() - t0))
```

```
In [14]: def join_mp_build2():
             df0 = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_x2_%d.csv' % 0))
             df1 = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_y_%d.csv' % 0))

             for i in range(1, NUM_THREADS):
                 print('working %d' % i)
                 temp = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_x2_%d.csv' % i))
                 df0 = df0.append(temp)

                 temp = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_y_%d.csv' % i))
                 df1 = df1.append(temp)

             df0.to_csv(os.path.join(OUTPUT_DIR, 'train_x2.csv'), index=False)
             #df1.to_csv(os.path.join(OUTPUT_DIR, 'train_y.csv'), index=False)
```

```
In [20]: from tqdm.auto import tqdm
         def build_test_fields2():
             train_X = pd.read_csv(os.path.join(OUTPUT_DIR, 'train_x2.csv'))
             try:
                 train_X.drop(labels=['seg_id', 'seg_start', 'seg_end'], axis=1, inplace=True)
             except:
                 pass

             submission = pd.read_csv(os.path.join(DATA_DIR, 'sample_submission.csv'), index_co
             test_X = pd.DataFrame(columns=train_X.columns, dtype=np.float64, index=submission

             print('start for loop')
             count = 0
             for seg_id in tqdm(test_X.index):  # just tqdm in IDE
                 seg = pd.read_csv(os.path.join(DATA_DIR, 'Untitled Folder/', str(seg_id) + '.c
                 # train_X = create_features_pk_det(seg_id, seg, train_X, start_idx, end_idx)
                 test_X = create_features_set2(seg_id, seg, test_X, 0, 0)

                 if count % 100 == 0:
                     print('working', seg_id)
                 count += 1

             test_X.to_csv(os.path.join(OUTPUT_DIR, 'test_x2.csv'), index=False)
```

```
In [16]: def scale_fields2(fn_train='train_x2.csv', fn_test='test_x2.csv',
                           fn_out_train='scaled_train_X2.csv' , fn_out_test='scaled_test_X2.csv
             train_X = pd.read_csv(os.path.join(OUTPUT_DIR, fn_train))
             try:
                 train_X.drop(labels=['seg_id', 'seg_start', 'seg_end'], axis=1, inplace=True)
             except:
                 pass
             test_X = pd.read_csv(os.path.join(OUTPUT_DIR, fn_test))

             print('start scaler')
             scaler = StandardScaler()
             scaler.fit(train_X)
             scaled_train_X = pd.DataFrame(scaler.transform(train_X), columns=train_X.columns)
             scaled_test_X = pd.DataFrame(scaler.transform(test_X), columns=test_X.columns)

             scaled_train_X.to_csv(os.path.join(OUTPUT_DIR, fn_out_train), index=False)
             scaled_test_X.to_csv(os.path.join(OUTPUT_DIR, fn_out_test), index=False)
```

```
In [ ]: run_mp_build2()
```

```
In [18]: join_mp_build2()
```

```
working 1
working 2
working 3
working 4
working 5
```

```
In [ ]: build_test_fields2()
```

```
In [22]: scale_fields2()
```

```
start scaler
```

5 Machine Learning Models

```
In [9]: def plot_op(y_predicted):
            plt.figure(figsize=(12,6))
            plt.plot(y_train,label='Time to Failure')
            plt.plot(y_predicted,label='Predicted Time to Failure')
            plt.xlabel('index')
            plt.ylabel('Time to failure')
            plt.legend()
            plt.title('Predictions')
            plt.show()

        #to plot feature importances of respective models
        def plot_importance(clf):
            fig, ax = plt.subplots(figsize=(15, 10))
            X_train=pd.read_csv('scaled_train_X.csv')
            my_dict={}
            #getting feature names and score
            for a,b in zip(X_train.columns,clf.feature_importances_):
                my_dict[a]=b
            import collections
            #to get top 10 features
            c = collections.Counter(my_dict)
            g=c.most_common(10)
            keys=[]
            values=[]
            for i in range(len(g)):
                keys.append(g[i][0])
                values.append(g[i][1])
            plt.bar(keys,values)
            plt.title('feature importances')
            plt.xlabel('features')
            plt.show()
```

5.1 LGBM

```
In [4]: #Since CV is not reliable i have used some default and approx values
        params = {'num_leaves': 21,
                 'min_data_in_leaf': 20,
                 'objective':'gamma',
                 'learning_rate': 0.001,
                 'max_depth': 108,
                 "boosting": "gbdt",
                 "feature_fraction": 0.91,
                 "bagging_freq": 1,
                 "bagging_fraction": 0.91,
                 "bagging_seed": 42,
                 "metric": 'mae',
                 "lambda_l1": 0.1,
                 "verbosity": -1,
                 "random_state": 42}


        def lgb_base_model():
            maes = []
            rmses = []
            submission = pd.read_csv(os.path.join(DATA_DIR, 'sample_submission.csv'), index_co
            scaled_train_X = pd.read_csv('scaled_train_X.csv')
            scaled_test_X = pd.read_csv('scaled_test_X.csv')
            scaled_check_X = pd.read_csv('scaled_check_X.csv')
            train_y = pd.read_csv('train_y.csv')
            predictions = np.zeros(len(scaled_test_X))
            predictions_check = np.zeros(len(scaled_check_X))
            predictions_train = np.zeros(len(scaled_train_X))

            n_fold = 8
            folds = KFold(n_splits=n_fold, shuffle=True, random_state=42)

            fold_importance_df = pd.DataFrame()
            fold_importance_df["Feature"] = scaled_train_X.columns

            for fold_, (trn_idx, val_idx) in enumerate(folds.split(scaled_train_X, train_y.valu
                print('working fold %d' % fold_)
                strLog = "fold {}".format(fold_)
                print(strLog)

                X_tr, X_val = scaled_train_X.iloc[trn_idx], scaled_train_X.iloc[val_idx]
                y_tr, y_val = train_y.iloc[trn_idx], train_y.iloc[val_idx]

                model = lgb.LGBMRegressor(**params, n_estimators=80000, n_jobs=-1)
                model.fit(X_tr, y_tr,
                          eval_set=[(X_tr, y_tr), (X_val, y_val)], eval_metric='mae',
                          verbose=1000, early_stopping_rounds=200)

                # predictions
                preds = model.predict(scaled_test_X, num_iteration=model.best_iteration_)
```

```
                predictions += preds / folds.n_splits
                preds = model.predict(X_val, num_iteration=model.best_iteration_)

                preds2=model.predict(scaled_train_X, num_iteration=model.best_iteration_)
                predictions_train += preds2 / folds.n_splits
                preds2 = model.predict(X_val, num_iteration=model.best_iteration_)

                preds3=model.predict(scaled_check_X, num_iteration=model.best_iteration_)
                predictions_check += preds3 / folds.n_splits
                preds3 = model.predict(X_val, num_iteration=model.best_iteration_)


                # mean absolute error
                mae = mean_absolute_error(y_val, preds)
                print('MAE: %.6f' % mae)
                maes.append(mae)

                # root mean squared error
                rmse = mean_squared_error(y_val, preds)
                print('RMSE: %.6f' % rmse)
                rmses.append(rmse)

                fold_importance_df['importance_%d' % fold_] = model.feature_importances_[:len(s

            print('MAEs', maes)
            print('MAE mean: %.6f' % np.mean(maes))
            print('RMSEs', rmses)
            print('RMSE mean: %.6f' % np.mean(rmses))

            submission.time_to_failure = predictions
            submission.to_csv('latest_lgb_80000.csv', index=False)
            fold_importance_df.to_csv('fold_imp_lgb_8_80k_108dpk.csv')
            return model,predictions,predictions_train,predictions_check
```

```
In [5]: clf1,predictions,predictions_train,predictions_check=lgb_base_model()
```

```
working fold 0
fold 0
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.04205        valid_1's l1: 2.0658
[2000]        training's l1: 1.85568        valid_1's l1: 1.90821
[3000]        training's l1: 1.78033        valid_1's l1: 1.85463
[4000]        training's l1: 1.72237        valid_1's l1: 1.81275
[5000]        training's l1: 1.67465        valid_1's l1: 1.77696
[6000]        training's l1: 1.63059        valid_1's l1: 1.74507
[7000]        training's l1: 1.58972        valid_1's l1: 1.71523
[8000]        training's l1: 1.55069        valid_1's l1: 1.68785
[9000]        training's l1: 1.5136        valid_1's l1: 1.66211
```

```
[10000]         training's l1: 1.47859        valid_1's l1: 1.63896
[11000]         training's l1: 1.44565        valid_1's l1: 1.61872
[12000]         training's l1: 1.41446        valid_1's l1: 1.59919
[13000]         training's l1: 1.38513        valid_1's l1: 1.58177
[14000]         training's l1: 1.35638        valid_1's l1: 1.56434
[15000]         training's l1: 1.32905        valid_1's l1: 1.54875
[16000]         training's l1: 1.30284        valid_1's l1: 1.53389
[17000]         training's l1: 1.27758        valid_1's l1: 1.51991
[18000]         training's l1: 1.2539        valid_1's l1: 1.50715
[19000]         training's l1: 1.23063        valid_1's l1: 1.49497
[20000]         training's l1: 1.20805        valid_1's l1: 1.48284
[21000]         training's l1: 1.18625        valid_1's l1: 1.47127
[22000]         training's l1: 1.16531        valid_1's l1: 1.46104
[23000]         training's l1: 1.14483        valid_1's l1: 1.45023
[24000]         training's l1: 1.12501        valid_1's l1: 1.44034
[25000]         training's l1: 1.10542        valid_1's l1: 1.43029
[26000]         training's l1: 1.08657        valid_1's l1: 1.42111
[27000]         training's l1: 1.06822        valid_1's l1: 1.41222
[28000]         training's l1: 1.05038        valid_1's l1: 1.40396
[29000]         training's l1: 1.03296        valid_1's l1: 1.39564
[30000]         training's l1: 1.01582        valid_1's l1: 1.38746
[31000]         training's l1: 0.999237        valid_1's l1: 1.37975
[32000]         training's l1: 0.982697        valid_1's l1: 1.37191
[33000]         training's l1: 0.966926        valid_1's l1: 1.36505
[34000]         training's l1: 0.951181        valid_1's l1: 1.35772
[35000]         training's l1: 0.936115        valid_1's l1: 1.35118
[36000]         training's l1: 0.92157        valid_1's l1: 1.34494
[37000]         training's l1: 0.907055        valid_1's l1: 1.33844
[38000]         training's l1: 0.892816        valid_1's l1: 1.33233
[39000]         training's l1: 0.878979        valid_1's l1: 1.32654
[40000]         training's l1: 0.865642        valid_1's l1: 1.32109
[41000]         training's l1: 0.852398        valid_1's l1: 1.31566
[42000]         training's l1: 0.839318        valid_1's l1: 1.31021
[43000]         training's l1: 0.826634        valid_1's l1: 1.30529
[44000]         training's l1: 0.814261        valid_1's l1: 1.30046
[45000]         training's l1: 0.802096        valid_1's l1: 1.29549
[46000]         training's l1: 0.790099        valid_1's l1: 1.29068
[47000]         training's l1: 0.778274        valid_1's l1: 1.28578
[48000]         training's l1: 0.766811        valid_1's l1: 1.28156
[49000]         training's l1: 0.755492        valid_1's l1: 1.27723
[50000]         training's l1: 0.744478        valid_1's l1: 1.27287
[51000]         training's l1: 0.733624        valid_1's l1: 1.26873
[52000]         training's l1: 0.72287        valid_1's l1: 1.26476
[53000]         training's l1: 0.712266        valid_1's l1: 1.26058
[54000]         training's l1: 0.70197        valid_1's l1: 1.25675
[55000]         training's l1: 0.692023        valid_1's l1: 1.25326
[56000]         training's l1: 0.682144        valid_1's l1: 1.24967
[57000]         training's l1: 0.672654        valid_1's l1: 1.24627
```

```
[58000]        training's l1: 0.663146        valid_1's l1: 1.24262
[59000]        training's l1: 0.653861        valid_1's l1: 1.23931
[60000]        training's l1: 0.644644        valid_1's l1: 1.23593
[61000]        training's l1: 0.635584        valid_1's l1: 1.23251
[62000]        training's l1: 0.626758        valid_1's l1: 1.22953
[63000]        training's l1: 0.618064        valid_1's l1: 1.22649
[64000]        training's l1: 0.609556        valid_1's l1: 1.22353
[65000]        training's l1: 0.601187        valid_1's l1: 1.22066
[66000]        training's l1: 0.59294        valid_1's l1: 1.21773
[67000]        training's l1: 0.584928        valid_1's l1: 1.21507
[68000]        training's l1: 0.577004        valid_1's l1: 1.21248
[69000]        training's l1: 0.569321        valid_1's l1: 1.21005
[70000]        training's l1: 0.561675        valid_1's l1: 1.20732
[71000]        training's l1: 0.554293        valid_1's l1: 1.20497
[72000]        training's l1: 0.547072        valid_1's l1: 1.20272
[73000]        training's l1: 0.539844        valid_1's l1: 1.20055
[74000]        training's l1: 0.532772        valid_1's l1: 1.19834
[75000]        training's l1: 0.525834        valid_1's l1: 1.19608
[76000]        training's l1: 0.519051        valid_1's l1: 1.19414
[77000]        training's l1: 0.512208        valid_1's l1: 1.19181
[78000]        training's l1: 0.505532        valid_1's l1: 1.18973
[79000]        training's l1: 0.499029        valid_1's l1: 1.18781
[80000]        training's l1: 0.492627        valid_1's l1: 1.18591
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.492627        valid_1's l1: 1.18591
MAE: 1.185912
RMSE: 2.867659
working fold 1
fold 1
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.02943        valid_1's l1: 2.10519
[2000]        training's l1: 1.84727        valid_1's l1: 1.9456
[3000]        training's l1: 1.7748        valid_1's l1: 1.89097
[4000]        training's l1: 1.71527        valid_1's l1: 1.852
[5000]        training's l1: 1.66659        valid_1's l1: 1.81751
[6000]        training's l1: 1.62196        valid_1's l1: 1.7868
[7000]        training's l1: 1.57995        valid_1's l1: 1.75825
[8000]        training's l1: 1.54066        valid_1's l1: 1.73326
[9000]        training's l1: 1.50319        valid_1's l1: 1.70971
[10000]        training's l1: 1.46904        valid_1's l1: 1.68857
[11000]        training's l1: 1.43607        valid_1's l1: 1.66879
[12000]        training's l1: 1.40497        valid_1's l1: 1.65087
[13000]        training's l1: 1.37551        valid_1's l1: 1.63371
[14000]        training's l1: 1.34751        valid_1's l1: 1.61831
[15000]        training's l1: 1.32035        valid_1's l1: 1.60298
[16000]        training's l1: 1.29444        valid_1's l1: 1.58886
[17000]        training's l1: 1.26942        valid_1's l1: 1.57563
[18000]        training's l1: 1.24578        valid_1's l1: 1.56316
```

```
[19000]         training's l1: 1.22276       valid_1's l1: 1.55079
[20000]         training's l1: 1.20004       valid_1's l1: 1.53853
[21000]         training's l1: 1.1787       valid_1's l1: 1.52784
[22000]         training's l1: 1.1577       valid_1's l1: 1.51728
[23000]         training's l1: 1.13719       valid_1's l1: 1.50667
[24000]         training's l1: 1.117       valid_1's l1: 1.49667
[25000]         training's l1: 1.0978       valid_1's l1: 1.48758
[26000]         training's l1: 1.07901       valid_1's l1: 1.47891
[27000]         training's l1: 1.06089       valid_1's l1: 1.4704
[28000]         training's l1: 1.04325       valid_1's l1: 1.46239
[29000]         training's l1: 1.02589       valid_1's l1: 1.45454
[30000]         training's l1: 1.00891       valid_1's l1: 1.44686
[31000]         training's l1: 0.992317       valid_1's l1: 1.4393
[32000]         training's l1: 0.97611       valid_1's l1: 1.43192
[33000]         training's l1: 0.960456       valid_1's l1: 1.42504
[34000]         training's l1: 0.944739       valid_1's l1: 1.41781
[35000]         training's l1: 0.929675       valid_1's l1: 1.4113
[36000]         training's l1: 0.915266       valid_1's l1: 1.40502
[37000]         training's l1: 0.900461       valid_1's l1: 1.39818
[38000]         training's l1: 0.886003       valid_1's l1: 1.39181
[39000]         training's l1: 0.872176       valid_1's l1: 1.38571
[40000]         training's l1: 0.858913       valid_1's l1: 1.38031
[41000]         training's l1: 0.845587       valid_1's l1: 1.37466
[42000]         training's l1: 0.83264       valid_1's l1: 1.36908
[43000]         training's l1: 0.82003       valid_1's l1: 1.36402
[44000]         training's l1: 0.807345       valid_1's l1: 1.35846
[45000]         training's l1: 0.795224       valid_1's l1: 1.35339
[46000]         training's l1: 0.782948       valid_1's l1: 1.34805
[47000]         training's l1: 0.771254       valid_1's l1: 1.3435
[48000]         training's l1: 0.759721       valid_1's l1: 1.33864
[49000]         training's l1: 0.748568       valid_1's l1: 1.33418
[50000]         training's l1: 0.737786       valid_1's l1: 1.32995
[51000]         training's l1: 0.726953       valid_1's l1: 1.32539
[52000]         training's l1: 0.716435       valid_1's l1: 1.32114
[53000]         training's l1: 0.706021       valid_1's l1: 1.31682
[54000]         training's l1: 0.695886       valid_1's l1: 1.31286
[55000]         training's l1: 0.686094       valid_1's l1: 1.30926
[56000]         training's l1: 0.676115       valid_1's l1: 1.30535
[57000]         training's l1: 0.666584       valid_1's l1: 1.30168
[58000]         training's l1: 0.657297       valid_1's l1: 1.29823
[59000]         training's l1: 0.648107       valid_1's l1: 1.2948
[60000]         training's l1: 0.639112       valid_1's l1: 1.29147
[61000]         training's l1: 0.630238       valid_1's l1: 1.28828
[62000]         training's l1: 0.62151       valid_1's l1: 1.28498
[63000]         training's l1: 0.613052       valid_1's l1: 1.28191
[64000]         training's l1: 0.604605       valid_1's l1: 1.27878
[65000]         training's l1: 0.596381       valid_1's l1: 1.27579
[66000]         training's l1: 0.588298       valid_1's l1: 1.27285
```

```
[67000]        training's l1: 0.58022        valid_1's l1: 1.27001
[68000]        training's l1: 0.572472        valid_1's l1: 1.26725
[69000]        training's l1: 0.564815        valid_1's l1: 1.26457
[70000]        training's l1: 0.557375        valid_1's l1: 1.26203
[71000]        training's l1: 0.549957        valid_1's l1: 1.25946
[72000]        training's l1: 0.542673        valid_1's l1: 1.25698
[73000]        training's l1: 0.535502        valid_1's l1: 1.25458
[74000]        training's l1: 0.528407        valid_1's l1: 1.25221
[75000]        training's l1: 0.521443        valid_1's l1: 1.2499
[76000]        training's l1: 0.514628        valid_1's l1: 1.2478
[77000]        training's l1: 0.508044        valid_1's l1: 1.24573
[78000]        training's l1: 0.501433        valid_1's l1: 1.2436
[79000]        training's l1: 0.494967        valid_1's l1: 1.24158
[80000]        training's l1: 0.488545        valid_1's l1: 1.23955
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.488545        valid_1's l1: 1.23955
MAE: 1.239551
RMSE: 3.181445
working fold 2
fold 2
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.04706        valid_1's l1: 2.03109
[2000]        training's l1: 1.86426        valid_1's l1: 1.88386
[3000]        training's l1: 1.79088        valid_1's l1: 1.83429
[4000]        training's l1: 1.7305        valid_1's l1: 1.79724
[5000]        training's l1: 1.67943        valid_1's l1: 1.76558
[6000]        training's l1: 1.634        valid_1's l1: 1.73673
[7000]        training's l1: 1.59165        valid_1's l1: 1.71071
[8000]        training's l1: 1.55229        valid_1's l1: 1.68748
[9000]        training's l1: 1.51579        valid_1's l1: 1.66528
[10000]        training's l1: 1.48125        valid_1's l1: 1.64507
[11000]        training's l1: 1.4485        valid_1's l1: 1.62631
[12000]        training's l1: 1.41736        valid_1's l1: 1.60828
[13000]        training's l1: 1.38736        valid_1's l1: 1.59149
[14000]        training's l1: 1.35906        valid_1's l1: 1.57585
[15000]        training's l1: 1.33198        valid_1's l1: 1.56128
[16000]        training's l1: 1.30622        valid_1's l1: 1.54809
[17000]        training's l1: 1.28141        valid_1's l1: 1.53541
[18000]        training's l1: 1.25737        valid_1's l1: 1.52314
[19000]        training's l1: 1.2341        valid_1's l1: 1.51137
[20000]        training's l1: 1.21197        valid_1's l1: 1.50041
[21000]        training's l1: 1.1901        valid_1's l1: 1.48958
[22000]        training's l1: 1.16881        valid_1's l1: 1.47903
[23000]        training's l1: 1.14824        valid_1's l1: 1.46874
[24000]        training's l1: 1.12817        valid_1's l1: 1.45943
[25000]        training's l1: 1.10867        valid_1's l1: 1.45018
[26000]        training's l1: 1.08963        valid_1's l1: 1.4412
[27000]        training's l1: 1.07117        valid_1's l1: 1.43286
```

```
[28000]         training's l1: 1.05305       valid_1's l1: 1.42475
[29000]         training's l1: 1.0353        valid_1's l1: 1.41623
[30000]         training's l1: 1.01803       valid_1's l1: 1.40849
[31000]         training's l1: 1.001         valid_1's l1: 1.40079
[32000]         training's l1: 0.984446      valid_1's l1: 1.39302
[33000]         training's l1: 0.96848       valid_1's l1: 1.38576
[34000]         training's l1: 0.953022      valid_1's l1: 1.379
[35000]         training's l1: 0.937412      valid_1's l1: 1.37209
[36000]         training's l1: 0.922328      valid_1's l1: 1.36558
[37000]         training's l1: 0.907859      valid_1's l1: 1.35944
[38000]         training's l1: 0.893497      valid_1's l1: 1.3532
[39000]         training's l1: 0.879706      valid_1's l1: 1.34738
[40000]         training's l1: 0.865862      valid_1's l1: 1.34145
[41000]         training's l1: 0.852338      valid_1's l1: 1.33562
[42000]         training's l1: 0.839451      valid_1's l1: 1.33003
[43000]         training's l1: 0.826709      valid_1's l1: 1.32459
[44000]         training's l1: 0.814192      valid_1's l1: 1.31984
[45000]         training's l1: 0.802181      valid_1's l1: 1.3151
[46000]         training's l1: 0.790106      valid_1's l1: 1.31058
[47000]         training's l1: 0.778283      valid_1's l1: 1.30604
[48000]         training's l1: 0.766819      valid_1's l1: 1.30168
[49000]         training's l1: 0.75541       valid_1's l1: 1.29714
[50000]         training's l1: 0.744374      valid_1's l1: 1.2929
[51000]         training's l1: 0.73345       valid_1's l1: 1.28881
[52000]         training's l1: 0.722919      valid_1's l1: 1.28486
[53000]         training's l1: 0.712452      valid_1's l1: 1.28081
[54000]         training's l1: 0.702285      valid_1's l1: 1.27707
[55000]         training's l1: 0.692327      valid_1's l1: 1.27316
[56000]         training's l1: 0.682573      valid_1's l1: 1.26959
[57000]         training's l1: 0.67278       valid_1's l1: 1.26574
[58000]         training's l1: 0.663401      valid_1's l1: 1.26229
[59000]         training's l1: 0.654145      valid_1's l1: 1.25904
[60000]         training's l1: 0.645178      valid_1's l1: 1.25586
[61000]         training's l1: 0.636183      valid_1's l1: 1.25243
[62000]         training's l1: 0.627325      valid_1's l1: 1.24924
[63000]         training's l1: 0.618595      valid_1's l1: 1.2462
[64000]         training's l1: 0.610056      valid_1's l1: 1.24307
[65000]         training's l1: 0.601601      valid_1's l1: 1.24003
[66000]         training's l1: 0.593379      valid_1's l1: 1.23726
[67000]         training's l1: 0.585229      valid_1's l1: 1.23436
[68000]         training's l1: 0.577283      valid_1's l1: 1.23167
[69000]         training's l1: 0.569534      valid_1's l1: 1.22909
[70000]         training's l1: 0.56186       valid_1's l1: 1.22653
[71000]         training's l1: 0.554411      valid_1's l1: 1.22413
[72000]         training's l1: 0.547054      valid_1's l1: 1.22175
[73000]         training's l1: 0.53981       valid_1's l1: 1.21933
[74000]         training's l1: 0.532709      valid_1's l1: 1.217
[75000]         training's l1: 0.525721      valid_1's l1: 1.21455
```

```
[76000]         training's l1: 0.518794        valid_1's l1: 1.21242
[77000]         training's l1: 0.511977        valid_1's l1: 1.21006
[78000]         training's l1: 0.505435        valid_1's l1: 1.20806
[79000]         training's l1: 0.498945        valid_1's l1: 1.20607
[80000]         training's l1: 0.492502        valid_1's l1: 1.20409
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.492502        valid_1's l1: 1.20409
MAE: 1.204090
RMSE: 3.016129
working fold 3
fold 3
Training until validation scores don't improve for 200 rounds.
[1000]         training's l1: 2.03722        valid_1's l1: 2.09205
[2000]         training's l1: 1.85463        valid_1's l1: 1.93213
[3000]         training's l1: 1.78076        valid_1's l1: 1.87763
[4000]         training's l1: 1.72119        valid_1's l1: 1.83624
[5000]         training's l1: 1.67122        valid_1's l1: 1.80078
[6000]         training's l1: 1.62608        valid_1's l1: 1.76941
[7000]         training's l1: 1.58458        valid_1's l1: 1.742
[8000]         training's l1: 1.54594        valid_1's l1: 1.71664
[9000]         training's l1: 1.50939        valid_1's l1: 1.69319
[10000]         training's l1: 1.47503        valid_1's l1: 1.67189
[11000]         training's l1: 1.4422        valid_1's l1: 1.6515
[12000]         training's l1: 1.41172        valid_1's l1: 1.63419
[13000]         training's l1: 1.38263        valid_1's l1: 1.61794
[14000]         training's l1: 1.35452        valid_1's l1: 1.60182
[15000]         training's l1: 1.32797        valid_1's l1: 1.58784
[16000]         training's l1: 1.30214        valid_1's l1: 1.57388
[17000]         training's l1: 1.2771        valid_1's l1: 1.56066
[18000]         training's l1: 1.25307        valid_1's l1: 1.54847
[19000]         training's l1: 1.22955        valid_1's l1: 1.53627
[20000]         training's l1: 1.20693        valid_1's l1: 1.52483
[21000]         training's l1: 1.18464        valid_1's l1: 1.51354
[22000]         training's l1: 1.16316        valid_1's l1: 1.50296
[23000]         training's l1: 1.14266        valid_1's l1: 1.49324
[24000]         training's l1: 1.12242        valid_1's l1: 1.48318
[25000]         training's l1: 1.10255        valid_1's l1: 1.47348
[26000]         training's l1: 1.08344        valid_1's l1: 1.46452
[27000]         training's l1: 1.06492        valid_1's l1: 1.45569
[28000]         training's l1: 1.04682        valid_1's l1: 1.44708
[29000]         training's l1: 1.02936        valid_1's l1: 1.43907
[30000]         training's l1: 1.01209        valid_1's l1: 1.43082
[32000]         training's l1: 0.979083        valid_1's l1: 1.41594
[33000]         training's l1: 0.962741        valid_1's l1: 1.40828
[34000]         training's l1: 0.94709        valid_1's l1: 1.40175
[35000]         training's l1: 0.931633        valid_1's l1: 1.39478
[36000]         training's l1: 0.916613        valid_1's l1: 1.38808
[37000]         training's l1: 0.902106        valid_1's l1: 1.38195
```

```
[38000]         training's l1: 0.887818        valid_1's l1: 1.37564
[39000]         training's l1: 0.873995        valid_1's l1: 1.36988
[40000]         training's l1: 0.860454        valid_1's l1: 1.36427
[41000]         training's l1: 0.847414        valid_1's l1: 1.35909
[42000]         training's l1: 0.834478        valid_1's l1: 1.35393
[43000]         training's l1: 0.821478        valid_1's l1: 1.34836
[44000]         training's l1: 0.808897        valid_1's l1: 1.34352
[45000]         training's l1: 0.796606        valid_1's l1: 1.33867
[46000]         training's l1: 0.784598        valid_1's l1: 1.33394
[47000]         training's l1: 0.772876        valid_1's l1: 1.32934
[48000]         training's l1: 0.761371        valid_1's l1: 1.32516
[49000]         training's l1: 0.750091        valid_1's l1: 1.32095
[50000]         training's l1: 0.739105        valid_1's l1: 1.31677
[51000]         training's l1: 0.728194        valid_1's l1: 1.31278
[52000]         training's l1: 0.717558        valid_1's l1: 1.30891
[53000]         training's l1: 0.707143        valid_1's l1: 1.3051
[54000]         training's l1: 0.696853        valid_1's l1: 1.30121
[55000]         training's l1: 0.686722        valid_1's l1: 1.29751
[56000]         training's l1: 0.676737        valid_1's l1: 1.29379
[57000]         training's l1: 0.666973        valid_1's l1: 1.29038
[58000]         training's l1: 0.65735        valid_1's l1: 1.28682
[59000]         training's l1: 0.648017        valid_1's l1: 1.28341
[60000]         training's l1: 0.638886        valid_1's l1: 1.28027
[61000]         training's l1: 0.629893        valid_1's l1: 1.27705
[62000]         training's l1: 0.620993        valid_1's l1: 1.27387
[63000]         training's l1: 0.61233        valid_1's l1: 1.27096
[64000]         training's l1: 0.603709        valid_1's l1: 1.26785
[65000]         training's l1: 0.595348        valid_1's l1: 1.26496
[66000]         training's l1: 0.587129        valid_1's l1: 1.26189
[67000]         training's l1: 0.579189        valid_1's l1: 1.25921
[68000]         training's l1: 0.571223        valid_1's l1: 1.25642
[69000]         training's l1: 0.563357        valid_1's l1: 1.25382
[70000]         training's l1: 0.555664        valid_1's l1: 1.25113
[71000]         training's l1: 0.548221        valid_1's l1: 1.24863
[72000]         training's l1: 0.540799        valid_1's l1: 1.24596
[73000]         training's l1: 0.533626        valid_1's l1: 1.24369
[74000]         training's l1: 0.526494        valid_1's l1: 1.24129
[75000]         training's l1: 0.519477        valid_1's l1: 1.23899
[76000]         training's l1: 0.512655        valid_1's l1: 1.23688
[77000]         training's l1: 0.505848        valid_1's l1: 1.2347
[78000]         training's l1: 0.499186        valid_1's l1: 1.23245
[79000]         training's l1: 0.492671        valid_1's l1: 1.23046
[80000]         training's l1: 0.486183        valid_1's l1: 1.22835
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.486183        valid_1's l1: 1.22835
MAE: 1.228351
RMSE: 3.212806
working fold 4
```

```
fold 4
Training until validation scores don't improve for 200 rounds.
[1000]       training's l1: 2.03628       valid_1's l1: 2.08161
[2000]       training's l1: 1.85469       valid_1's l1: 1.91147
[3000]       training's l1: 1.78214       valid_1's l1: 1.85216
[4000]       training's l1: 1.72441       valid_1's l1: 1.81094
[5000]       training's l1: 1.67424       valid_1's l1: 1.77484
[6000]       training's l1: 1.62958       valid_1's l1: 1.74446
[7000]       training's l1: 1.5875       valid_1's l1: 1.71698
[8000]       training's l1: 1.54779       valid_1's l1: 1.69153
[9000]       training's l1: 1.51104       valid_1's l1: 1.66847
[10000]       training's l1: 1.47619       valid_1's l1: 1.6478
[11000]       training's l1: 1.44294       valid_1's l1: 1.62846
[12000]       training's l1: 1.41161       valid_1's l1: 1.61069
[13000]       training's l1: 1.3817       valid_1's l1: 1.59358
[14000]       training's l1: 1.35329       valid_1's l1: 1.57757
[15000]       training's l1: 1.32576       valid_1's l1: 1.5634
[16000]       training's l1: 1.29968       valid_1's l1: 1.54964
[17000]       training's l1: 1.27465       valid_1's l1: 1.5369
[18000]       training's l1: 1.25052       valid_1's l1: 1.52478
[19000]       training's l1: 1.22708       valid_1's l1: 1.51273
[20000]       training's l1: 1.20487       valid_1's l1: 1.50184
[21000]       training's l1: 1.18363       valid_1's l1: 1.49194
[22000]       training's l1: 1.16271       valid_1's l1: 1.48235
[23000]       training's l1: 1.1423       valid_1's l1: 1.47321
[24000]       training's l1: 1.12209       valid_1's l1: 1.46381
[25000]       training's l1: 1.10264       valid_1's l1: 1.45486
[26000]       training's l1: 1.08388       valid_1's l1: 1.44612
[27000]       training's l1: 1.06547       valid_1's l1: 1.43788
[28000]       training's l1: 1.0475       valid_1's l1: 1.42993
[29000]       training's l1: 1.02983       valid_1's l1: 1.42216
[30000]       training's l1: 1.01275       valid_1's l1: 1.41463
[31000]       training's l1: 0.996047       valid_1's l1: 1.40748
[32000]       training's l1: 0.979418       valid_1's l1: 1.40015
[33000]       training's l1: 0.963774       valid_1's l1: 1.39362
[34000]       training's l1: 0.948643       valid_1's l1: 1.38734
[35000]       training's l1: 0.933389       valid_1's l1: 1.38072
[36000]       training's l1: 0.918543       valid_1's l1: 1.37446
[37000]       training's l1: 0.904237       valid_1's l1: 1.3688
[38000]       training's l1: 0.889757       valid_1's l1: 1.36252
[39000]       training's l1: 0.876069       valid_1's l1: 1.35705
[40000]       training's l1: 0.862242       valid_1's l1: 1.35123
[41000]       training's l1: 0.848728       valid_1's l1: 1.34604
[42000]       training's l1: 0.835769       valid_1's l1: 1.34095
[43000]       training's l1: 0.822951       valid_1's l1: 1.33589
[44000]       training's l1: 0.81064       valid_1's l1: 1.33108
[45000]       training's l1: 0.798571       valid_1's l1: 1.32652
[46000]       training's l1: 0.786658       valid_1's l1: 1.32182
```

```
[47000]        training's l1: 0.774903       valid_1's l1: 1.31748
[48000]        training's l1: 0.763561       valid_1's l1: 1.31321
[49000]        training's l1: 0.752415       valid_1's l1: 1.30928
[50000]        training's l1: 0.741606       valid_1's l1: 1.3056
[51000]        training's l1: 0.731043       valid_1's l1: 1.30215
[52000]        training's l1: 0.720372       valid_1's l1: 1.29826
[53000]        training's l1: 0.710128       valid_1's l1: 1.29461
[54000]        training's l1: 0.699896       valid_1's l1: 1.29095
[55000]        training's l1: 0.690029       valid_1's l1: 1.28776
[56000]        training's l1: 0.680259       valid_1's l1: 1.2843
[57000]        training's l1: 0.670717       valid_1's l1: 1.28108
[58000]        training's l1: 0.661396       valid_1's l1: 1.27795
[59000]        training's l1: 0.652088       valid_1's l1: 1.27494
[60000]        training's l1: 0.642933       valid_1's l1: 1.27193
[61000]        training's l1: 0.634087       valid_1's l1: 1.26914
[62000]        training's l1: 0.625477       valid_1's l1: 1.26633
[63000]        training's l1: 0.6169       valid_1's l1: 1.26354
[64000]        training's l1: 0.608402       valid_1's l1: 1.2606
[65000]        training's l1: 0.600031       valid_1's l1: 1.25791
[66000]        training's l1: 0.591862       valid_1's l1: 1.25517
[67000]        training's l1: 0.583923       valid_1's l1: 1.2528
[68000]        training's l1: 0.576051       valid_1's l1: 1.25019
[69000]        training's l1: 0.568279       valid_1's l1: 1.24774
[70000]        training's l1: 0.560809       valid_1's l1: 1.24557
[71000]        training's l1: 0.55335       valid_1's l1: 1.24321
[72000]        training's l1: 0.546088       valid_1's l1: 1.24108
[73000]        training's l1: 0.538721       valid_1's l1: 1.2387
[74000]        training's l1: 0.531657       valid_1's l1: 1.23638
[75000]        training's l1: 0.524609       valid_1's l1: 1.23426
[76000]        training's l1: 0.517809       valid_1's l1: 1.23226
[77000]        training's l1: 0.511152       valid_1's l1: 1.23012
[78000]        training's l1: 0.504508       valid_1's l1: 1.22803
[79000]        training's l1: 0.49802       valid_1's l1: 1.22619
[80000]        training's l1: 0.491676       valid_1's l1: 1.22444
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.491676       valid_1's l1: 1.22444
MAE: 1.224438
RMSE: 3.216581
working fold 5
fold 5
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.03981       valid_1's l1: 2.07427
[2000]        training's l1: 1.85702       valid_1's l1: 1.92205
[3000]        training's l1: 1.78465       valid_1's l1: 1.87187
[4000]        training's l1: 1.72396       valid_1's l1: 1.83221
[5000]        training's l1: 1.67367       valid_1's l1: 1.79875
[6000]        training's l1: 1.62989       valid_1's l1: 1.77198
[7000]        training's l1: 1.58813       valid_1's l1: 1.74502
```

```
[8000]          training's l1: 1.54885        valid_1's l1: 1.72052
[9000]          training's l1: 1.51171        valid_1's l1: 1.69817
[10000]          training's l1: 1.47673        valid_1's l1: 1.67814
[11000]          training's l1: 1.44335        valid_1's l1: 1.65839
[12000]          training's l1: 1.41182        valid_1's l1: 1.64055
[13000]          training's l1: 1.38158        valid_1's l1: 1.62385
[14000]          training's l1: 1.35283        valid_1's l1: 1.60858
[15000]          training's l1: 1.32545        valid_1's l1: 1.59395
[16000]          training's l1: 1.29929        valid_1's l1: 1.58039
[17000]          training's l1: 1.27378        valid_1's l1: 1.56739
[18000]          training's l1: 1.24897        valid_1's l1: 1.55489
[19000]          training's l1: 1.22533        valid_1's l1: 1.54322
[20000]          training's l1: 1.20269        valid_1's l1: 1.53236
[21000]          training's l1: 1.1806        valid_1's l1: 1.52182
[22000]          training's l1: 1.15933        valid_1's l1: 1.51212
[23000]          training's l1: 1.13866        valid_1's l1: 1.50278
[24000]          training's l1: 1.11857        valid_1's l1: 1.49389
[25000]          training's l1: 1.09894        valid_1's l1: 1.48492
[26000]          training's l1: 1.07981        valid_1's l1: 1.47635
[27000]          training's l1: 1.06152        valid_1's l1: 1.46858
[28000]          training's l1: 1.04338        valid_1's l1: 1.46117
[29000]          training's l1: 1.02593        valid_1's l1: 1.454
[30000]          training's l1: 1.00866        valid_1's l1: 1.44671
[31000]          training's l1: 0.991892        valid_1's l1: 1.43988
[32000]          training's l1: 0.97573        valid_1's l1: 1.43333
[33000]          training's l1: 0.959918        valid_1's l1: 1.42711
[34000]          training's l1: 0.944588        valid_1's l1: 1.42085
[35000]          training's l1: 0.929626        valid_1's l1: 1.41488
[36000]          training's l1: 0.915119        valid_1's l1: 1.40948
[37000]          training's l1: 0.900543        valid_1's l1: 1.40381
[38000]          training's l1: 0.886398        valid_1's l1: 1.39835
[39000]          training's l1: 0.872667        valid_1's l1: 1.39301
[40000]          training's l1: 0.858962        valid_1's l1: 1.3877
[41000]          training's l1: 0.845528        valid_1's l1: 1.38248
[42000]          training's l1: 0.832624        valid_1's l1: 1.37747
[43000]          training's l1: 0.819954        valid_1's l1: 1.37288
[44000]          training's l1: 0.807583        valid_1's l1: 1.36823
[45000]          training's l1: 0.795392        valid_1's l1: 1.36381
[46000]          training's l1: 0.783397        valid_1's l1: 1.35948
[47000]          training's l1: 0.771797        valid_1's l1: 1.35542
[48000]          training's l1: 0.76011        valid_1's l1: 1.35112
[49000]          training's l1: 0.748787        valid_1's l1: 1.34682
[50000]          training's l1: 0.737553        valid_1's l1: 1.34259
[51000]          training's l1: 0.726624        valid_1's l1: 1.33866
[52000]          training's l1: 0.716036        valid_1's l1: 1.33489
[53000]          training's l1: 0.705621        valid_1's l1: 1.33097
[54000]          training's l1: 0.695485        valid_1's l1: 1.32728
[55000]          training's l1: 0.68561        valid_1's l1: 1.32365
```

```
[56000]         training's l1: 0.675854        valid_1's l1: 1.32024
[57000]         training's l1: 0.666518        valid_1's l1: 1.31689
[58000]         training's l1: 0.656997        valid_1's l1: 1.31344
[59000]         training's l1: 0.647659        valid_1's l1: 1.30967
[60000]         training's l1: 0.638513        valid_1's l1: 1.30625
[61000]         training's l1: 0.629471        valid_1's l1: 1.30306
[62000]         training's l1: 0.620815        valid_1's l1: 1.30018
[63000]         training's l1: 0.612128        valid_1's l1: 1.29709
[64000]         training's l1: 0.603767        valid_1's l1: 1.2942
[65000]         training's l1: 0.595427        valid_1's l1: 1.29125
[66000]         training's l1: 0.58731        valid_1's l1: 1.28834
[67000]         training's l1: 0.579416        valid_1's l1: 1.28575
[68000]         training's l1: 0.571668        valid_1's l1: 1.28318
[69000]         training's l1: 0.56398        valid_1's l1: 1.28048
[70000]         training's l1: 0.556504        valid_1's l1: 1.27824
[71000]         training's l1: 0.549065        valid_1's l1: 1.27571
[72000]         training's l1: 0.541795        valid_1's l1: 1.27338
[73000]         training's l1: 0.53452        valid_1's l1: 1.27084
[74000]         training's l1: 0.527428        valid_1's l1: 1.26841
[75000]         training's l1: 0.520444        valid_1's l1: 1.26605
[76000]         training's l1: 0.513626        valid_1's l1: 1.26378
[77000]         training's l1: 0.506934        valid_1's l1: 1.26166
[78000]         training's l1: 0.500189        valid_1's l1: 1.25942
[79000]         training's l1: 0.493627        valid_1's l1: 1.25727
[80000]         training's l1: 0.48726        valid_1's l1: 1.25508
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.48726        valid_1's l1: 1.25508
MAE: 1.255081
RMSE: 3.297988
working fold 6
fold 6
Training until validation scores don't improve for 200 rounds.
[1000]          training's l1: 2.03958        valid_1's l1: 2.07611
[2000]          training's l1: 1.85625        valid_1's l1: 1.90815
[3000]          training's l1: 1.78269        valid_1's l1: 1.85319
[4000]          training's l1: 1.72027        valid_1's l1: 1.81181
[5000]          training's l1: 1.66911        valid_1's l1: 1.77879
[6000]          training's l1: 1.62396        valid_1's l1: 1.74948
[7000]          training's l1: 1.5826        valid_1's l1: 1.72237
[8000]          training's l1: 1.54309        valid_1's l1: 1.69651
[9000]          training's l1: 1.50598        valid_1's l1: 1.67254
[10000]         training's l1: 1.47063        valid_1's l1: 1.64985
[11000]         training's l1: 1.43756        valid_1's l1: 1.62929
[12000]         training's l1: 1.40581        valid_1's l1: 1.60908
[13000]         training's l1: 1.37625        valid_1's l1: 1.59118
[14000]         training's l1: 1.34773        valid_1's l1: 1.57366
[15000]         training's l1: 1.32068        valid_1's l1: 1.55779
[16000]         training's l1: 1.29489        valid_1's l1: 1.54271
```

```
[17000]         training's l1: 1.26962        valid_1's l1: 1.52828
[18000]         training's l1: 1.24562        valid_1's l1: 1.51446
[19000]         training's l1: 1.22233        valid_1's l1: 1.5014
[20000]         training's l1: 1.19983        valid_1's l1: 1.48978
[21000]         training's l1: 1.17818        valid_1's l1: 1.47855
[22000]         training's l1: 1.15715        valid_1's l1: 1.46759
[23000]         training's l1: 1.13687        valid_1's l1: 1.45771
[24000]         training's l1: 1.11665        valid_1's l1: 1.4474
[25000]         training's l1: 1.09706        valid_1's l1: 1.43804
[26000]         training's l1: 1.07796        valid_1's l1: 1.42874
[27000]         training's l1: 1.05982        valid_1's l1: 1.42045
[28000]         training's l1: 1.04145        valid_1's l1: 1.4113
[29000]         training's l1: 1.02399        valid_1's l1: 1.40341
[30000]         training's l1: 1.00708        valid_1's l1: 1.39574
[31000]         training's l1: 0.990569        valid_1's l1: 1.38851
[32000]         training's l1: 0.974486        valid_1's l1: 1.38159
[33000]         training's l1: 0.958678        valid_1's l1: 1.37463
[34000]         training's l1: 0.943088        valid_1's l1: 1.36789
[35000]         training's l1: 0.927994        valid_1's l1: 1.3611
[36000]         training's l1: 0.913359        valid_1's l1: 1.35492
[37000]         training's l1: 0.898974        valid_1's l1: 1.34877
[38000]         training's l1: 0.885133        valid_1's l1: 1.34314
[39000]         training's l1: 0.871373        valid_1's l1: 1.33752
[40000]         training's l1: 0.857895        valid_1's l1: 1.33173
[41000]         training's l1: 0.844584        valid_1's l1: 1.32626
[42000]         training's l1: 0.831687        valid_1's l1: 1.32106
[43000]         training's l1: 0.819154        valid_1's l1: 1.31616
[44000]         training's l1: 0.806707        valid_1's l1: 1.31121
[45000]         training's l1: 0.794644        valid_1's l1: 1.30671
[46000]         training's l1: 0.782805        valid_1's l1: 1.30213
[47000]         training's l1: 0.771031        valid_1's l1: 1.29755
[48000]         training's l1: 0.759396        valid_1's l1: 1.29287
[49000]         training's l1: 0.748066        valid_1's l1: 1.28859
[50000]         training's l1: 0.737226        valid_1's l1: 1.28468
[51000]         training's l1: 0.726502        valid_1's l1: 1.28097
[52000]         training's l1: 0.715775        valid_1's l1: 1.27676
[53000]         training's l1: 0.705298        valid_1's l1: 1.27277
[54000]         training's l1: 0.69524        valid_1's l1: 1.26899
[55000]         training's l1: 0.685187        valid_1's l1: 1.26533
[56000]         training's l1: 0.675208        valid_1's l1: 1.26171
[57000]         training's l1: 0.665602        valid_1's l1: 1.25824
[58000]         training's l1: 0.656275        valid_1's l1: 1.25483
[59000]         training's l1: 0.647014        valid_1's l1: 1.25139
[60000]         training's l1: 0.637901        valid_1's l1: 1.2482
[61000]         training's l1: 0.628914        valid_1's l1: 1.24493
[62000]         training's l1: 0.620035        valid_1's l1: 1.24158
[63000]         training's l1: 0.611435        valid_1's l1: 1.2385
[64000]         training's l1: 0.603136        valid_1's l1: 1.23575
```

```
[65000]         training's l1: 0.594832        valid_1's l1: 1.23294
[66000]         training's l1: 0.586561        valid_1's l1: 1.23005
[67000]         training's l1: 0.57846        valid_1's l1: 1.22718
[68000]         training's l1: 0.570439        valid_1's l1: 1.22434
[69000]         training's l1: 0.562664        valid_1's l1: 1.22175
[70000]         training's l1: 0.555084        valid_1's l1: 1.21913
[71000]         training's l1: 0.547672        valid_1's l1: 1.21672
[72000]         training's l1: 0.540348        valid_1's l1: 1.21432
[73000]         training's l1: 0.533192        valid_1's l1: 1.212
[74000]         training's l1: 0.52606        valid_1's l1: 1.20969
[75000]         training's l1: 0.519087        valid_1's l1: 1.20736
[76000]         training's l1: 0.512143        valid_1's l1: 1.20513
[77000]         training's l1: 0.505341        valid_1's l1: 1.20293
[78000]         training's l1: 0.498572        valid_1's l1: 1.20062
[79000]         training's l1: 0.492113        valid_1's l1: 1.19868
[80000]         training's l1: 0.485741        valid_1's l1: 1.19675
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.485741        valid_1's l1: 1.19675
MAE: 1.196754
RMSE: 2.935862
working fold 7
fold 7
Training until validation scores don't improve for 200 rounds.
[1000]         training's l1: 2.03864        valid_1's l1: 2.0768
[2000]         training's l1: 1.8571        valid_1's l1: 1.92485
[3000]         training's l1: 1.78443        valid_1's l1: 1.87322
[4000]         training's l1: 1.72647        valid_1's l1: 1.83085
[5000]         training's l1: 1.67803        valid_1's l1: 1.79541
[6000]         training's l1: 1.63429        valid_1's l1: 1.76407
[7000]         training's l1: 1.59337        valid_1's l1: 1.73442
[8000]         training's l1: 1.55401        valid_1's l1: 1.70655
[9000]         training's l1: 1.51614        valid_1's l1: 1.67995
[10000]         training's l1: 1.48065        valid_1's l1: 1.65554
[11000]         training's l1: 1.44666        valid_1's l1: 1.63341
[12000]         training's l1: 1.41531        valid_1's l1: 1.61369
[13000]         training's l1: 1.38518        valid_1's l1: 1.59494
[14000]         training's l1: 1.35641        valid_1's l1: 1.57766
[15000]         training's l1: 1.32894        valid_1's l1: 1.56199
[16000]         training's l1: 1.30274        valid_1's l1: 1.54714
[17000]         training's l1: 1.27755        valid_1's l1: 1.53341
[18000]         training's l1: 1.25322        valid_1's l1: 1.52012
[19000]         training's l1: 1.22973        valid_1's l1: 1.50784
[20000]         training's l1: 1.20689        valid_1's l1: 1.49588
[21000]         training's l1: 1.18467        valid_1's l1: 1.48455
[22000]         training's l1: 1.16311        valid_1's l1: 1.47391
[23000]         training's l1: 1.14226        valid_1's l1: 1.46394
[24000]         training's l1: 1.12212        valid_1's l1: 1.45385
[25000]         training's l1: 1.10232        valid_1's l1: 1.44438
```

```
[26000]         training's l1: 1.08303        valid_1's l1: 1.43495
[27000]         training's l1: 1.06413        valid_1's l1: 1.42585
[28000]         training's l1: 1.04624        valid_1's l1: 1.41782
[29000]         training's l1: 1.02857        valid_1's l1: 1.4099
[30000]         training's l1: 1.01132        valid_1's l1: 1.40225
[31000]         training's l1: 0.994693        valid_1's l1: 1.39507
[32000]         training's l1: 0.97847        valid_1's l1: 1.388
[33000]         training's l1: 0.962592        valid_1's l1: 1.38106
[34000]         training's l1: 0.947034        valid_1's l1: 1.37465
[35000]         training's l1: 0.931904        valid_1's l1: 1.36855
[36000]         training's l1: 0.917006        valid_1's l1: 1.3621
[37000]         training's l1: 0.902602        valid_1's l1: 1.35637
[38000]         training's l1: 0.888455        valid_1's l1: 1.35065
[39000]         training's l1: 0.874755        valid_1's l1: 1.34506
[40000]         training's l1: 0.861242        valid_1's l1: 1.33962
[41000]         training's l1: 0.848059        valid_1's l1: 1.33436
[42000]         training's l1: 0.835141        valid_1's l1: 1.32934
[43000]         training's l1: 0.822489        valid_1's l1: 1.3244
[44000]         training's l1: 0.810013        valid_1's l1: 1.31924
[45000]         training's l1: 0.797895        valid_1's l1: 1.31477
[46000]         training's l1: 0.785986        valid_1's l1: 1.3104
[47000]         training's l1: 0.774443        valid_1's l1: 1.30628
[48000]         training's l1: 0.763094        valid_1's l1: 1.30231
[49000]         training's l1: 0.751851        valid_1's l1: 1.29805
[50000]         training's l1: 0.740958        valid_1's l1: 1.29399
[51000]         training's l1: 0.730152        valid_1's l1: 1.29013
[52000]         training's l1: 0.719564        valid_1's l1: 1.28641
[53000]         training's l1: 0.709177        valid_1's l1: 1.28262
[54000]         training's l1: 0.698975        valid_1's l1: 1.27897
[55000]         training's l1: 0.688978        valid_1's l1: 1.27553
[56000]         training's l1: 0.67913        valid_1's l1: 1.27231
[57000]         training's l1: 0.669566        valid_1's l1: 1.26905
[58000]         training's l1: 0.660263        valid_1's l1: 1.26607
[59000]         training's l1: 0.651129        valid_1's l1: 1.26301
[60000]         training's l1: 0.642109        valid_1's l1: 1.26002
[61000]         training's l1: 0.632968        valid_1's l1: 1.25682
[62000]         training's l1: 0.624181        valid_1's l1: 1.25402
[63000]         training's l1: 0.615604        valid_1's l1: 1.25127
[64000]         training's l1: 0.607286        valid_1's l1: 1.24849
[65000]         training's l1: 0.59901        valid_1's l1: 1.24586
[66000]         training's l1: 0.59083        valid_1's l1: 1.24307
[67000]         training's l1: 0.582825        valid_1's l1: 1.24053
[68000]         training's l1: 0.574928        valid_1's l1: 1.23815
[69000]         training's l1: 0.567195        valid_1's l1: 1.23567
[70000]         training's l1: 0.559568        valid_1's l1: 1.23334
[71000]         training's l1: 0.55208        valid_1's l1: 1.23106
[72000]         training's l1: 0.544643        valid_1's l1: 1.22872
[73000]         training's l1: 0.53736        valid_1's l1: 1.22642
```

```
[74000]       training's l1: 0.530227       valid_1's l1: 1.22414
[75000]       training's l1: 0.523313       valid_1's l1: 1.22204
[76000]       training's l1: 0.516415       valid_1's l1: 1.21997
[77000]       training's l1: 0.509672       valid_1's l1: 1.21794
[78000]       training's l1: 0.503058       valid_1's l1: 1.21596
[79000]       training's l1: 0.496599       valid_1's l1: 1.21416
[80000]       training's l1: 0.490173       valid_1's l1: 1.21238
Did not meet early stopping. Best iteration is:
[80000]       training's l1: 0.490173       valid_1's l1: 1.21238
MAE: 1.212378
RMSE: 3.031687
MAEs [1.18591200098456, 1.239550552895232, 1.2040904177485894, 1.2283506663053225, 1.224438255
MAE mean: 1.218319
RMSEs [2.867658747707904, 3.1814452457139724, 3.0161289448298327, 3.2128057321593606, 3.216581
RMSE mean: 3.095020
```

```
In [7]: pd.DataFrame(predictions_train).to_csv("predictions_train_xgb.csv", header=None, index=
        pd.DataFrame(predictions_check).to_csv("predictions_check_xgb.csv", header=None, index=
        #predictions_train.to_csv('train_pred_lgb.csv', index=False)
        #predictions_check.to_csv('check_pred_lgb.csv', index=False)
```

```
In [8]: predictions_check[0:10]
```

```
Out[8]: array([5.09622617, 2.3245475 , 2.2125402 , 3.775534  , 2.32544473,
               1.82324896, 2.77731835, 4.16331317, 1.66960789, 1.38710654])
```

```
In [50]: #predictions_train=pd.read_csv('predictions_train.csv')
         predictions_train.to_csv('predictions_train_lgb.csv')
         #predictions_check=pd.read_csv('predictions_check.csv')
         predictions_check.to_csv('predictions_check_lgb.csv')
```

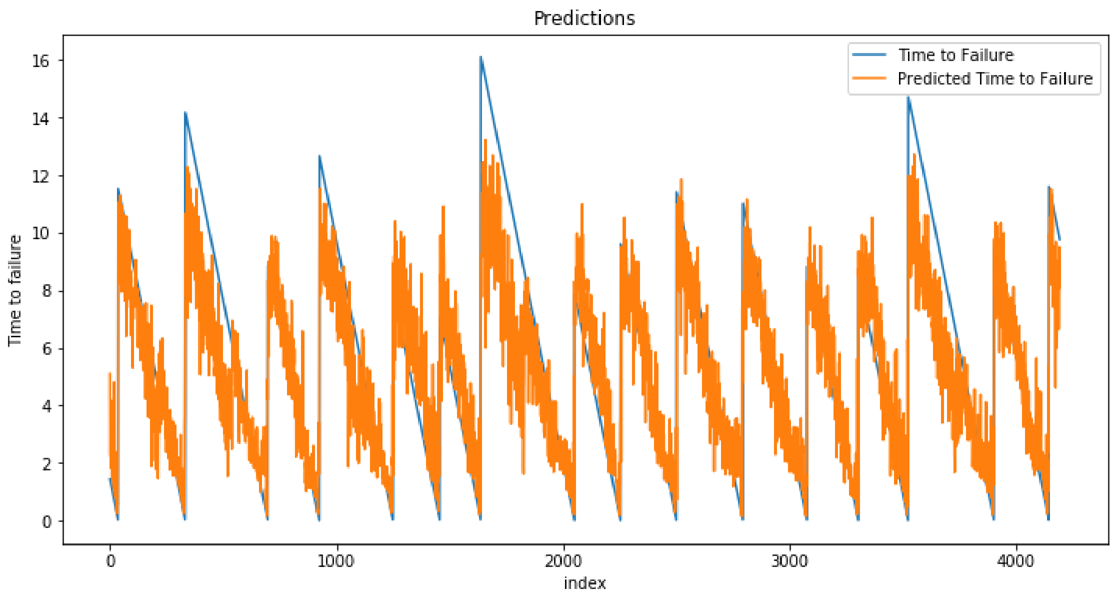
```
In [10]: y = train.time_to_failure
```

```
In [11]: rows = 150000
         segments = int(np.floor(train.shape[0] / rows))
         y_train = pd.DataFrame(index=range(segments), dtype=np.float64,
                                columns=['time_to_failure'])
         for segment in tqdm(range(segments)):

             seg = train.iloc[segment*rows:segment*rows+rows]
             y = seg['time_to_failure'].values[-1]
             y_train.loc[segment, 'time_to_failure'] = y
```

```
100%|| 4194/4194 [00:01<00:00, 2262.95it/s]
```

```
In [36]: y_train.to_csv('y_train_original.csv',index=None)
```

```
In [34]: plot_op(predictions_check)
```



We can see that the model is able to detect most the earthquakes

5.2 LGBM with feature set 1 and 2

```
In [6]: predictions_check[0:10]

Out[6]: array([5.44475999, 5.01630807, 4.3423958 , 4.72229591, 4.7615773 ,
               4.38402993, 4.59371988, 5.86153722, 2.96471365, 3.55317467])

In [5]: scaled_train_X2=pd.read_csv('scaled_train_X2.csv')
        scaled_train_X=pd.read_csv('scaled_train_X.csv')
        scaled_test_X=pd.read_csv('scaled_test_X.csv')
        scaled_test_X2=pd.read_csv('scaled_test_X2.csv')
        scaled_train_X=pd.concat([scaled_train_X,scaled_train_X2],axis=1)
        scaled_test_X=pd.concat([scaled_test_X,scaled_test_X2],axis=1)

In [6]: print(scaled_test_X2.shape)
        scaled_train_X.shape

(2624, 67)


Out[6]: (24000, 932)

In [7]: #with feature set 1 and 2
        #1.351
        params = {'num_leaves': 21,
                  'min_data_in_leaf': 20,
                  'objective':'gamma',
                 'learning_rate': 0.001,
                 'max_depth': 108,
                 "boosting": "gbdt",
                 "feature_fraction": 0.91,
                 "bagging_freq": 1,
                 "bagging_fraction": 0.91,
                 "bagging_seed": 42,
                 "metric": 'mae',
                 "lambda_l1": 0.1,
                 "verbosity": -1,
                 "random_state": 42}


        def lgb_f2_model():
            maes = []
            rmses = []
            submission = pd.read_csv(os.path.join(DATA_DIR, 'sample_submission.csv'), index_col
            #scaled_train_X = scaled_train_X
            #scaled_test_X = scaled_test_X
            train_y = pd.read_csv('train_y.csv')
            predictions = np.zeros(len(scaled_test_X))

            n_fold = 8
            folds = KFold(n_splits=n_fold, shuffle=True, random_state=42)

            fold_importance_df = pd.DataFrame()
            fold_importance_df["Feature"] = scaled_train_X.columns

            for fold_, (trn_idx, val_idx) in enumerate(folds.split(scaled_train_X, train_y.valu
                print('working fold %d' % fold_)
                strLog = "fold {}".format(fold_)
                print(strLog)

                X_tr, X_val = scaled_train_X.iloc[trn_idx], scaled_train_X.iloc[val_idx]
                y_tr, y_val = train_y.iloc[trn_idx], train_y.iloc[val_idx]

                model = lgb.LGBMRegressor(**params, n_estimators=80000, n_jobs=-1)
                model.fit(X_tr, y_tr,
                          eval_set=[(X_tr, y_tr), (X_val, y_val)], eval_metric='mae',
                          verbose=1000, early_stopping_rounds=200)

                # predictions
                preds = model.predict(scaled_test_X, num_iteration=model.best_iteration_)
                predictions += preds / folds.n_splits
                preds = model.predict(X_val, num_iteration=model.best_iteration_)

                # mean absolute error
                mae = mean_absolute_error(y_val, preds)
```

```
                print('MAE: %.6f' % mae)
                maes.append(mae)

                # root mean squared error
                rmse = mean_squared_error(y_val, preds)
                print('RMSE: %.6f' % rmse)
                rmses.append(rmse)

                fold_importance_df['importance_%d' % fold_] = model.feature_importances_[:len(

            print('MAEs', maes)
            print('MAE mean: %.6f' % np.mean(maes))
            print('RMSEs', rmses)
            print('RMSE mean: %.6f' % np.mean(rmses))

            submission.time_to_failure = predictions
            submission.to_csv('submission_lgb_2featureset.csv', index=False)
            fold_importance_df.to_csv('fold_imp_lgb_8_80k_108dp.csv')
```

```
In [8]: lgb_f2_model()
```

```
working fold 0
fold 0
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.04174        valid_1's l1: 2.06447
[2000]        training's l1: 1.85357        valid_1's l1: 1.90978
[3000]        training's l1: 1.77847        valid_1's l1: 1.85564
[4000]        training's l1: 1.71851        valid_1's l1: 1.8109
[5000]        training's l1: 1.67005        valid_1's l1: 1.77346
[6000]        training's l1: 1.62553        valid_1's l1: 1.74131
[7000]        training's l1: 1.58344        valid_1's l1: 1.71118
[8000]        training's l1: 1.54337        valid_1's l1: 1.68327
[9000]        training's l1: 1.50615        valid_1's l1: 1.65851
[10000]        training's l1: 1.47071        valid_1's l1: 1.63519
[11000]        training's l1: 1.4374        valid_1's l1: 1.61514
[12000]        training's l1: 1.40553        valid_1's l1: 1.59579
[13000]        training's l1: 1.37578        valid_1's l1: 1.57863
[14000]        training's l1: 1.34667        valid_1's l1: 1.56116
[15000]        training's l1: 1.31927        valid_1's l1: 1.54572
[16000]        training's l1: 1.29222        valid_1's l1: 1.53013
[17000]        training's l1: 1.26602        valid_1's l1: 1.51496
[18000]        training's l1: 1.24142        valid_1's l1: 1.50167
[19000]        training's l1: 1.21779        valid_1's l1: 1.48896
[20000]        training's l1: 1.19485        valid_1's l1: 1.47687
[21000]        training's l1: 1.17251        valid_1's l1: 1.46565
[22000]        training's l1: 1.15097        valid_1's l1: 1.45491
[23000]        training's l1: 1.12996        valid_1's l1: 1.44389
[24000]        training's l1: 1.10936        valid_1's l1: 1.43358
```

```
[25000]         training's l1: 1.08948        valid_1's l1: 1.42367
[26000]         training's l1: 1.07036        valid_1's l1: 1.4148
[27000]         training's l1: 1.05182        valid_1's l1: 1.40605
[28000]         training's l1: 1.03365        valid_1's l1: 1.39717
[29000]         training's l1: 1.01625        valid_1's l1: 1.38913
[30000]         training's l1: 0.998912        valid_1's l1: 1.38104
[31000]         training's l1: 0.982012        valid_1's l1: 1.37324
[32000]         training's l1: 0.965416        valid_1's l1: 1.36566
[33000]         training's l1: 0.949323        valid_1's l1: 1.35805
[34000]         training's l1: 0.933428        valid_1's l1: 1.35052
[35000]         training's l1: 0.918264        valid_1's l1: 1.34411
[36000]         training's l1: 0.902974        valid_1's l1: 1.33679
[37000]         training's l1: 0.888201        valid_1's l1: 1.33008
[38000]         training's l1: 0.873663        valid_1's l1: 1.32359
[39000]         training's l1: 0.859827        valid_1's l1: 1.31737
[40000]         training's l1: 0.846222        valid_1's l1: 1.3112
[41000]         training's l1: 0.832744        valid_1's l1: 1.30556
[42000]         training's l1: 0.819664        valid_1's l1: 1.29987
[43000]         training's l1: 0.806711        valid_1's l1: 1.29442
[44000]         training's l1: 0.794143        valid_1's l1: 1.28901
[45000]         training's l1: 0.781942        valid_1's l1: 1.28408
[46000]         training's l1: 0.769967        valid_1's l1: 1.27949
[47000]         training's l1: 0.758137        valid_1's l1: 1.27447
[48000]         training's l1: 0.746753        valid_1's l1: 1.2701
[49000]         training's l1: 0.735531        valid_1's l1: 1.26571
[50000]         training's l1: 0.724418        valid_1's l1: 1.26119
[51000]         training's l1: 0.71369        valid_1's l1: 1.25705
[52000]         training's l1: 0.702968        valid_1's l1: 1.25288
[53000]         training's l1: 0.692496        valid_1's l1: 1.24861
[54000]         training's l1: 0.682301        valid_1's l1: 1.24469
[55000]         training's l1: 0.672357        valid_1's l1: 1.24088
[56000]         training's l1: 0.662603        valid_1's l1: 1.2372
[57000]         training's l1: 0.653167        valid_1's l1: 1.23374
[58000]         training's l1: 0.643932        valid_1's l1: 1.23028
[59000]         training's l1: 0.634687        valid_1's l1: 1.22684
[60000]         training's l1: 0.625593        valid_1's l1: 1.22325
[61000]         training's l1: 0.616625        valid_1's l1: 1.21965
[62000]         training's l1: 0.60784        valid_1's l1: 1.21628
[63000]         training's l1: 0.599488        valid_1's l1: 1.21328
[64000]         training's l1: 0.591122        valid_1's l1: 1.2101
[65000]         training's l1: 0.582821        valid_1's l1: 1.20694
[66000]         training's l1: 0.574747        valid_1's l1: 1.20392
[67000]         training's l1: 0.566888        valid_1's l1: 1.20137
[68000]         training's l1: 0.55913        valid_1's l1: 1.19853
[69000]         training's l1: 0.551525        valid_1's l1: 1.19576
[70000]         training's l1: 0.543965        valid_1's l1: 1.19308
[71000]         training's l1: 0.536567        valid_1's l1: 1.19049
[72000]         training's l1: 0.529393        valid_1's l1: 1.1881
```

```
[73000]        training's l1: 0.522261       valid_1's l1: 1.18567
[74000]        training's l1: 0.51531       valid_1's l1: 1.18324
[75000]        training's l1: 0.508457       valid_1's l1: 1.18083
[76000]        training's l1: 0.501735       valid_1's l1: 1.17854
[77000]        training's l1: 0.495033       valid_1's l1: 1.17613
[78000]        training's l1: 0.488527       valid_1's l1: 1.1738
[79000]        training's l1: 0.482169       valid_1's l1: 1.1718
[80000]        training's l1: 0.475882       valid_1's l1: 1.16963
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.475882       valid_1's l1: 1.16963
MAE: 1.169632
RMSE: 2.799177
working fold 1
fold 1
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.02966       valid_1's l1: 2.10472
[2000]        training's l1: 1.84564       valid_1's l1: 1.9413
[3000]        training's l1: 1.77038       valid_1's l1: 1.88347
[4000]        training's l1: 1.71025       valid_1's l1: 1.84261
[5000]        training's l1: 1.66052       valid_1's l1: 1.80769
[6000]        training's l1: 1.614       valid_1's l1: 1.77651
[7000]        training's l1: 1.57033       valid_1's l1: 1.74776
[8000]        training's l1: 1.52985       valid_1's l1: 1.72197
[9000]        training's l1: 1.49177       valid_1's l1: 1.69738
[10000]        training's l1: 1.45659       valid_1's l1: 1.67521
[11000]        training's l1: 1.42349       valid_1's l1: 1.65441
[12000]        training's l1: 1.39184       valid_1's l1: 1.63525
[13000]        training's l1: 1.36175       valid_1's l1: 1.61719
[14000]        training's l1: 1.33318       valid_1's l1: 1.60082
[15000]        training's l1: 1.30593       valid_1's l1: 1.58538
[16000]        training's l1: 1.27968       valid_1's l1: 1.57098
[17000]        training's l1: 1.25461       valid_1's l1: 1.55768
[18000]        training's l1: 1.23045       valid_1's l1: 1.54499
[19000]        training's l1: 1.20712       valid_1's l1: 1.53286
[20000]        training's l1: 1.18435       valid_1's l1: 1.52088
[21000]        training's l1: 1.16266       valid_1's l1: 1.51027
[22000]        training's l1: 1.14166       valid_1's l1: 1.49984
[23000]        training's l1: 1.12136       valid_1's l1: 1.48965
[24000]        training's l1: 1.10147       valid_1's l1: 1.48
[25000]        training's l1: 1.08229       valid_1's l1: 1.47064
[26000]        training's l1: 1.06347       valid_1's l1: 1.46187
[27000]        training's l1: 1.04489       valid_1's l1: 1.45275
[28000]        training's l1: 1.02672       valid_1's l1: 1.44387
[29000]        training's l1: 1.00931       valid_1's l1: 1.43591
[30000]        training's l1: 0.992318       valid_1's l1: 1.42817
[31000]        training's l1: 0.975559       valid_1's l1: 1.42067
[32000]        training's l1: 0.959284       valid_1's l1: 1.41323
[33000]        training's l1: 0.943681       valid_1's l1: 1.40635
```

```
[34000]         training's l1: 0.928092        valid_1's l1: 1.39937
[35000]         training's l1: 0.913266        valid_1's l1: 1.39287
[36000]         training's l1: 0.898438        valid_1's l1: 1.38632
[37000]         training's l1: 0.883881        valid_1's l1: 1.37968
[38000]         training's l1: 0.869529        valid_1's l1: 1.37353
[39000]         training's l1: 0.855544        valid_1's l1: 1.36728
[40000]         training's l1: 0.842068        valid_1's l1: 1.36172
[41000]         training's l1: 0.82872        valid_1's l1: 1.35597
[42000]         training's l1: 0.815656        valid_1's l1: 1.35029
[43000]         training's l1: 0.803005        valid_1's l1: 1.34503
[44000]         training's l1: 0.790532        valid_1's l1: 1.33955
[45000]         training's l1: 0.778372        valid_1's l1: 1.33434
[46000]         training's l1: 0.76632        valid_1's l1: 1.32907
[47000]         training's l1: 0.754733        valid_1's l1: 1.32413
[48000]         training's l1: 0.743294        valid_1's l1: 1.31926
[49000]         training's l1: 0.73224        valid_1's l1: 1.31461
[50000]         training's l1: 0.721361        valid_1's l1: 1.3101
[51000]         training's l1: 0.710645        valid_1's l1: 1.30562
[52000]         training's l1: 0.699994        valid_1's l1: 1.30091
[53000]         training's l1: 0.689688        valid_1's l1: 1.29668
[54000]         training's l1: 0.679576        valid_1's l1: 1.2927
[55000]         training's l1: 0.669679        valid_1's l1: 1.28863
[56000]         training's l1: 0.659811        valid_1's l1: 1.28462
[57000]         training's l1: 0.650199        valid_1's l1: 1.28073
[58000]         training's l1: 0.641014        valid_1's l1: 1.27703
[59000]         training's l1: 0.631874        valid_1's l1: 1.27346
[60000]         training's l1: 0.622765        valid_1's l1: 1.26985
[61000]         training's l1: 0.613929        valid_1's l1: 1.26624
[62000]         training's l1: 0.605226        valid_1's l1: 1.26282
[63000]         training's l1: 0.596887        valid_1's l1: 1.25947
[64000]         training's l1: 0.588517        valid_1's l1: 1.25609
[65000]         training's l1: 0.580226        valid_1's l1: 1.25289
[66000]         training's l1: 0.572179        valid_1's l1: 1.24986
[67000]         training's l1: 0.564233        valid_1's l1: 1.24663
[68000]         training's l1: 0.556556        valid_1's l1: 1.24397
[69000]         training's l1: 0.548953        valid_1's l1: 1.24111
[70000]         training's l1: 0.541626        valid_1's l1: 1.23844
[71000]         training's l1: 0.534342        valid_1's l1: 1.23587
[72000]         training's l1: 0.527202        valid_1's l1: 1.23333
[73000]         training's l1: 0.520046        valid_1's l1: 1.23079
[74000]         training's l1: 0.513153        valid_1's l1: 1.22835
[75000]         training's l1: 0.506271        valid_1's l1: 1.22587
[76000]         training's l1: 0.49959        valid_1's l1: 1.22343
[77000]         training's l1: 0.493008        valid_1's l1: 1.2211
[78000]         training's l1: 0.486509        valid_1's l1: 1.21868
[79000]         training's l1: 0.48012        valid_1's l1: 1.21652
[80000]         training's l1: 0.473823        valid_1's l1: 1.21432
Did not meet early stopping. Best iteration is:
```

```
[80000]        training's l1: 0.473823       valid_1's l1: 1.21432
MAE: 1.214319
RMSE: 3.065115
working fold 2
fold 2
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.04528       valid_1's l1: 2.03237
[2000]        training's l1: 1.85908       valid_1's l1: 1.88365
[3000]        training's l1: 1.78371       valid_1's l1: 1.83037
[4000]        training's l1: 1.72373       valid_1's l1: 1.79049
[5000]        training's l1: 1.6732       valid_1's l1: 1.75905
[6000]        training's l1: 1.62685       valid_1's l1: 1.73124
[7000]        training's l1: 1.58439       valid_1's l1: 1.70505
[8000]        training's l1: 1.5435       valid_1's l1: 1.68058
[9000]        training's l1: 1.50565       valid_1's l1: 1.65808
[10000]        training's l1: 1.46976       valid_1's l1: 1.63721
[11000]        training's l1: 1.43576       valid_1's l1: 1.61736
[12000]        training's l1: 1.40437       valid_1's l1: 1.59978
[13000]        training's l1: 1.37393       valid_1's l1: 1.58303
[14000]        training's l1: 1.3451       valid_1's l1: 1.56717
[15000]        training's l1: 1.31741       valid_1's l1: 1.55165
[16000]        training's l1: 1.29105       valid_1's l1: 1.53755
[17000]        training's l1: 1.26583       valid_1's l1: 1.52494
[18000]        training's l1: 1.24128       valid_1's l1: 1.51196
[19000]        training's l1: 1.21781       valid_1's l1: 1.50011
[20000]        training's l1: 1.19539       valid_1's l1: 1.48881
[21000]        training's l1: 1.17323       valid_1's l1: 1.47799
[22000]        training's l1: 1.15188       valid_1's l1: 1.46807
[23000]        training's l1: 1.13106       valid_1's l1: 1.45771
[24000]        training's l1: 1.11108       valid_1's l1: 1.44829
[25000]        training's l1: 1.09163       valid_1's l1: 1.43935
[26000]        training's l1: 1.07245       valid_1's l1: 1.43042
[27000]        training's l1: 1.05377       valid_1's l1: 1.42159
[28000]        training's l1: 1.03532       valid_1's l1: 1.41332
[29000]        training's l1: 1.01745       valid_1's l1: 1.40519
[30000]        training's l1: 1.00025       valid_1's l1: 1.39741
[31000]        training's l1: 0.983597       valid_1's l1: 1.39015
[32000]        training's l1: 0.966968       valid_1's l1: 1.3828
[33000]        training's l1: 0.950837       valid_1's l1: 1.37577
[34000]        training's l1: 0.935008       valid_1's l1: 1.3686
[35000]        training's l1: 0.919716       valid_1's l1: 1.36178
[36000]        training's l1: 0.904627       valid_1's l1: 1.35514
[37000]        training's l1: 0.890044       valid_1's l1: 1.349
[38000]        training's l1: 0.87555       valid_1's l1: 1.3429
[39000]        training's l1: 0.861535       valid_1's l1: 1.3368
[40000]        training's l1: 0.847729       valid_1's l1: 1.33084
[41000]        training's l1: 0.834274       valid_1's l1: 1.32529
[42000]        training's l1: 0.821352       valid_1's l1: 1.31982
```

```
[43000]         training's l1: 0.808648        valid_1's l1: 1.31465
[44000]         training's l1: 0.796196        valid_1's l1: 1.30945
[45000]         training's l1: 0.784165        valid_1's l1: 1.30466
[46000]         training's l1: 0.772118        valid_1's l1: 1.29953
[47000]         training's l1: 0.760296        valid_1's l1: 1.29479
[48000]         training's l1: 0.748848        valid_1's l1: 1.29029
[49000]         training's l1: 0.737536        valid_1's l1: 1.28599
[50000]         training's l1: 0.726372        valid_1's l1: 1.28134
[51000]         training's l1: 0.71568        valid_1's l1: 1.2771
[52000]         training's l1: 0.705198        valid_1's l1: 1.27322
[53000]         training's l1: 0.694671        valid_1's l1: 1.26899
[54000]         training's l1: 0.684651        valid_1's l1: 1.26534
[55000]         training's l1: 0.674647        valid_1's l1: 1.26141
[56000]         training's l1: 0.66491        valid_1's l1: 1.25762
[57000]         training's l1: 0.655404        valid_1's l1: 1.25392
[58000]         training's l1: 0.646023        valid_1's l1: 1.25036
[59000]         training's l1: 0.63687        valid_1's l1: 1.24692
[60000]         training's l1: 0.627859        valid_1's l1: 1.24346
[61000]         training's l1: 0.618891        valid_1's l1: 1.24007
[62000]         training's l1: 0.610029        valid_1's l1: 1.23675
[63000]         training's l1: 0.601428        valid_1's l1: 1.23353
[64000]         training's l1: 0.593027        valid_1's l1: 1.23041
[65000]         training's l1: 0.584669        valid_1's l1: 1.22741
[66000]         training's l1: 0.576519        valid_1's l1: 1.22423
[67000]         training's l1: 0.568521        valid_1's l1: 1.2214
[68000]         training's l1: 0.560739        valid_1's l1: 1.21859
[69000]         training's l1: 0.552952        valid_1's l1: 1.2157
[70000]         training's l1: 0.545477        valid_1's l1: 1.21297
[71000]         training's l1: 0.537976        valid_1's l1: 1.21023
[72000]         training's l1: 0.530625        valid_1's l1: 1.2075
[73000]         training's l1: 0.523455        valid_1's l1: 1.20505
[74000]         training's l1: 0.516405        valid_1's l1: 1.20252
[75000]         training's l1: 0.509634        valid_1's l1: 1.20035
[76000]         training's l1: 0.502881        valid_1's l1: 1.19819
[77000]         training's l1: 0.496125        valid_1's l1: 1.19578
[78000]         training's l1: 0.489596        valid_1's l1: 1.19347
[79000]         training's l1: 0.48315        valid_1's l1: 1.19135
[80000]         training's l1: 0.476797        valid_1's l1: 1.18917
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.476797        valid_1's l1: 1.18917
MAE: 1.189172
RMSE: 2.927019
working fold 3
fold 3
Training until validation scores don't improve for 200 rounds.
[1000]          training's l1: 2.03764        valid_1's l1: 2.0931
[2000]          training's l1: 1.85316        valid_1's l1: 1.92982
[3000]          training's l1: 1.77751        valid_1's l1: 1.87336
```

```
[4000]        training's l1: 1.71685        valid_1's l1: 1.83318
[5000]        training's l1: 1.66618        valid_1's l1: 1.79928
[6000]        training's l1: 1.62006        valid_1's l1: 1.76798
[7000]        training's l1: 1.57721        valid_1's l1: 1.73874
[8000]        training's l1: 1.53663        valid_1's l1: 1.71169
[9000]        training's l1: 1.49955        valid_1's l1: 1.68803
[10000]        training's l1: 1.46473        valid_1's l1: 1.66636
[11000]        training's l1: 1.43172        valid_1's l1: 1.64634
[12000]        training's l1: 1.40088        valid_1's l1: 1.6283
[13000]        training's l1: 1.37164        valid_1's l1: 1.61175
[14000]        training's l1: 1.34289        valid_1's l1: 1.59542
[15000]        training's l1: 1.31549        valid_1's l1: 1.58028
[16000]        training's l1: 1.28913        valid_1's l1: 1.56611
[17000]        training's l1: 1.26363        valid_1's l1: 1.55209
[18000]        training's l1: 1.23937        valid_1's l1: 1.53929
[19000]        training's l1: 1.21546        valid_1's l1: 1.52656
[20000]        training's l1: 1.19249        valid_1's l1: 1.51478
[21000]        training's l1: 1.16983        valid_1's l1: 1.50262
[22000]        training's l1: 1.14814        valid_1's l1: 1.49122
[23000]        training's l1: 1.12755        valid_1's l1: 1.48107
[24000]        training's l1: 1.10696        valid_1's l1: 1.47087
[25000]        training's l1: 1.08724        valid_1's l1: 1.46136
[26000]        training's l1: 1.06786        valid_1's l1: 1.45219
[27000]        training's l1: 1.0491        valid_1's l1: 1.44323
[28000]        training's l1: 1.03097        valid_1's l1: 1.4347
[29000]        training's l1: 1.01319        valid_1's l1: 1.42656
[30000]        training's l1: 0.996078        valid_1's l1: 1.41873
[31000]        training's l1: 0.979041        valid_1's l1: 1.41063
[32000]        training's l1: 0.962681        valid_1's l1: 1.40351
[33000]        training's l1: 0.946551        valid_1's l1: 1.39628
[34000]        training's l1: 0.930822        valid_1's l1: 1.38932
[35000]        training's l1: 0.915509        valid_1's l1: 1.3824
[36000]        training's l1: 0.900473        valid_1's l1: 1.37584
[37000]        training's l1: 0.885973        valid_1's l1: 1.36968
[38000]        training's l1: 0.87136        valid_1's l1: 1.36321
[39000]        training's l1: 0.857633        valid_1's l1: 1.35754
[40000]        training's l1: 0.84415        valid_1's l1: 1.35216
[41000]        training's l1: 0.830865        valid_1's l1: 1.34683
[42000]        training's l1: 0.817883        valid_1's l1: 1.3417
[43000]        training's l1: 0.805259        valid_1's l1: 1.33654
[44000]        training's l1: 0.792783        valid_1's l1: 1.3316
[45000]        training's l1: 0.78052        valid_1's l1: 1.32638
[46000]        training's l1: 0.768468        valid_1's l1: 1.3215
[47000]        training's l1: 0.756656        valid_1's l1: 1.31667
[48000]        training's l1: 0.745269        valid_1's l1: 1.3123
[49000]        training's l1: 0.733928        valid_1's l1: 1.30801
[50000]        training's l1: 0.723056        valid_1's l1: 1.30416
[51000]        training's l1: 0.712202        valid_1's l1: 1.3001
```

```
[52000]        training's l1: 0.701741       valid_1's l1: 1.2964
[53000]        training's l1: 0.691265       valid_1's l1: 1.29229
[54000]        training's l1: 0.680959       valid_1's l1: 1.2884
[55000]        training's l1: 0.670851       valid_1's l1: 1.28481
[56000]        training's l1: 0.661006       valid_1's l1: 1.28116
[57000]        training's l1: 0.651261       valid_1's l1: 1.27767
[58000]        training's l1: 0.641713       valid_1's l1: 1.27408
[59000]        training's l1: 0.632542       valid_1's l1: 1.27084
[60000]        training's l1: 0.623554       valid_1's l1: 1.26774
[61000]        training's l1: 0.614636       valid_1's l1: 1.26457
[62000]        training's l1: 0.605873       valid_1's l1: 1.26157
[63000]        training's l1: 0.597016       valid_1's l1: 1.25832
[64000]        training's l1: 0.588469       valid_1's l1: 1.2554
[65000]        training's l1: 0.580287       valid_1's l1: 1.25262
[66000]        training's l1: 0.572132       valid_1's l1: 1.24967
[67000]        training's l1: 0.564057       valid_1's l1: 1.24688
[68000]        training's l1: 0.556186       valid_1's l1: 1.24419
[69000]        training's l1: 0.548409       valid_1's l1: 1.24161
[70000]        training's l1: 0.540864       valid_1's l1: 1.23914
[71000]        training's l1: 0.533502       valid_1's l1: 1.23677
[72000]        training's l1: 0.526152       valid_1's l1: 1.23426
[73000]        training's l1: 0.519107       valid_1's l1: 1.23207
[74000]        training's l1: 0.512067       valid_1's l1: 1.22955
[75000]        training's l1: 0.505087       valid_1's l1: 1.22717
[76000]        training's l1: 0.498235       valid_1's l1: 1.22475
[77000]        training's l1: 0.491486       valid_1's l1: 1.22245
[78000]        training's l1: 0.484947       valid_1's l1: 1.22031
[79000]        training's l1: 0.478404       valid_1's l1: 1.21802
[80000]        training's l1: 0.472073       valid_1's l1: 1.21601
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.472073       valid_1's l1: 1.21601
MAE: 1.216014
RMSE: 3.187613
working fold 4
fold 4
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.03814       valid_1's l1: 2.08542
[2000]        training's l1: 1.85461       valid_1's l1: 1.91203
[3000]        training's l1: 1.78045       valid_1's l1: 1.85034
[4000]        training's l1: 1.721       valid_1's l1: 1.80645
[5000]        training's l1: 1.66993       valid_1's l1: 1.77025
[6000]        training's l1: 1.62412       valid_1's l1: 1.73802
[7000]        training's l1: 1.58138       valid_1's l1: 1.70871
[8000]        training's l1: 1.54144       valid_1's l1: 1.68337
[9000]        training's l1: 1.50413       valid_1's l1: 1.65992
[10000]        training's l1: 1.46834       valid_1's l1: 1.63973
[11000]        training's l1: 1.43448       valid_1's l1: 1.62035
[12000]        training's l1: 1.40279       valid_1's l1: 1.60233
```

```
[13000]	training's l1: 1.37226	valid_1's l1: 1.58581
[14000]	training's l1: 1.34334	valid_1's l1: 1.57036
[15000]	training's l1: 1.31496	valid_1's l1: 1.55494
[16000]	training's l1: 1.28809	valid_1's l1: 1.54097
[17000]	training's l1: 1.26223	valid_1's l1: 1.528
[18000]	training's l1: 1.23768	valid_1's l1: 1.51572
[19000]	training's l1: 1.21391	valid_1's l1: 1.5035
[20000]	training's l1: 1.19072	valid_1's l1: 1.49202
[21000]	training's l1: 1.16878	valid_1's l1: 1.48136
[22000]	training's l1: 1.14748	valid_1's l1: 1.47134
[23000]	training's l1: 1.127	valid_1's l1: 1.46207
[24000]	training's l1: 1.10681	valid_1's l1: 1.45286
[25000]	training's l1: 1.08717	valid_1's l1: 1.44386
[26000]	training's l1: 1.0681	valid_1's l1: 1.43511
[27000]	training's l1: 1.04938	valid_1's l1: 1.42668
[28000]	training's l1: 1.03135	valid_1's l1: 1.41868
[29000]	training's l1: 1.01349	valid_1's l1: 1.41057
[30000]	training's l1: 0.996387	valid_1's l1: 1.40291
[31000]	training's l1: 0.979645	valid_1's l1: 1.39538
[32000]	training's l1: 0.963081	valid_1's l1: 1.38773
[33000]	training's l1: 0.947052	valid_1's l1: 1.38058
[34000]	training's l1: 0.931773	valid_1's l1: 1.37403
[35000]	training's l1: 0.916503	valid_1's l1: 1.3674
[36000]	training's l1: 0.901559	valid_1's l1: 1.36105
[37000]	training's l1: 0.88705	valid_1's l1: 1.35496
[38000]	training's l1: 0.87258	valid_1's l1: 1.34892
[39000]	training's l1: 0.858578	valid_1's l1: 1.34321
[40000]	training's l1: 0.844742	valid_1's l1: 1.33726
[41000]	training's l1: 0.831063	valid_1's l1: 1.33143
[42000]	training's l1: 0.817939	valid_1's l1: 1.32618
[43000]	training's l1: 0.8051	valid_1's l1: 1.32081
[44000]	training's l1: 0.792791	valid_1's l1: 1.31621
[45000]	training's l1: 0.780586	valid_1's l1: 1.31149
[46000]	training's l1: 0.768632	valid_1's l1: 1.30674
[47000]	training's l1: 0.756934	valid_1's l1: 1.30221
[48000]	training's l1: 0.745568	valid_1's l1: 1.2978
[49000]	training's l1: 0.73424	valid_1's l1: 1.29341
[50000]	training's l1: 0.723382	valid_1's l1: 1.28922
[51000]	training's l1: 0.71269	valid_1's l1: 1.28521
[52000]	training's l1: 0.70212	valid_1's l1: 1.28135
[53000]	training's l1: 0.691738	valid_1's l1: 1.27751
[54000]	training's l1: 0.681578	valid_1's l1: 1.27378
[55000]	training's l1: 0.671613	valid_1's l1: 1.27024
[56000]	training's l1: 0.661898	valid_1's l1: 1.26681
[57000]	training's l1: 0.652458	valid_1's l1: 1.26367
[58000]	training's l1: 0.643155	valid_1's l1: 1.26065
[59000]	training's l1: 0.633889	valid_1's l1: 1.25748
[60000]	training's l1: 0.624922	valid_1's l1: 1.2545
```

```
[61000]        training's l1: 0.616212       valid_1's l1: 1.25152
[62000]        training's l1: 0.607596       valid_1's l1: 1.24837
[63000]        training's l1: 0.599212       valid_1's l1: 1.24544
[64000]        training's l1: 0.590823       valid_1's l1: 1.24255
[65000]        training's l1: 0.58259       valid_1's l1: 1.23965
[66000]        training's l1: 0.57453       valid_1's l1: 1.23693
[67000]        training's l1: 0.566677       valid_1's l1: 1.23429
[68000]        training's l1: 0.558852       valid_1's l1: 1.23174
[69000]        training's l1: 0.551239       valid_1's l1: 1.22929
[70000]        training's l1: 0.543815       valid_1's l1: 1.22687
[71000]        training's l1: 0.536544       valid_1's l1: 1.22461
[72000]        training's l1: 0.529428       valid_1's l1: 1.22226
[73000]        training's l1: 0.522221       valid_1's l1: 1.21986
[74000]        training's l1: 0.515488       valid_1's l1: 1.21791
[75000]        training's l1: 0.508653       valid_1's l1: 1.21585
[76000]        training's l1: 0.501967       valid_1's l1: 1.21388
[77000]        training's l1: 0.495394       valid_1's l1: 1.21182
[78000]        training's l1: 0.488875       valid_1's l1: 1.20961
[79000]        training's l1: 0.482502       valid_1's l1: 1.2077
[80000]        training's l1: 0.476263       valid_1's l1: 1.20571
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.476263       valid_1's l1: 1.20571
MAE: 1.205707
RMSE: 3.128079
working fold 5
fold 5
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.04056       valid_1's l1: 2.07249
[2000]        training's l1: 1.85412       valid_1's l1: 1.91775
[3000]        training's l1: 1.77802       valid_1's l1: 1.86551
[4000]        training's l1: 1.71788       valid_1's l1: 1.82609
[5000]        training's l1: 1.6675       valid_1's l1: 1.79193
[6000]        training's l1: 1.6225       valid_1's l1: 1.76234
[7000]        training's l1: 1.57992       valid_1's l1: 1.73564
[8000]        training's l1: 1.53991       valid_1's l1: 1.71061
[9000]        training's l1: 1.50173       valid_1's l1: 1.68762
[10000]        training's l1: 1.46578       valid_1's l1: 1.6657
[11000]        training's l1: 1.43162       valid_1's l1: 1.64578
[12000]        training's l1: 1.39932       valid_1's l1: 1.62763
[13000]        training's l1: 1.36888       valid_1's l1: 1.61072
[14000]        training's l1: 1.33969       valid_1's l1: 1.59522
[15000]        training's l1: 1.31166       valid_1's l1: 1.5808
[16000]        training's l1: 1.28489       valid_1's l1: 1.5672
[17000]        training's l1: 1.25861       valid_1's l1: 1.55376
[18000]        training's l1: 1.2338       valid_1's l1: 1.54113
[19000]        training's l1: 1.20976       valid_1's l1: 1.52904
[20000]        training's l1: 1.18677       valid_1's l1: 1.51736
[21000]        training's l1: 1.16441       valid_1's l1: 1.50712
```

```
[22000]         training's l1: 1.14282        valid_1's l1: 1.49692
[23000]         training's l1: 1.12194        valid_1's l1: 1.48688
[24000]         training's l1: 1.1018        valid_1's l1: 1.4776
[25000]         training's l1: 1.08222        valid_1's l1: 1.46854
[26000]         training's l1: 1.06326        valid_1's l1: 1.46017
[27000]         training's l1: 1.04479        valid_1's l1: 1.4519
[28000]         training's l1: 1.02643        valid_1's l1: 1.44369
[29000]         training's l1: 1.00857        valid_1's l1: 1.43594
[30000]         training's l1: 0.991156        valid_1's l1: 1.42825
[31000]         training's l1: 0.974255        valid_1's l1: 1.42089
[32000]         training's l1: 0.958053        valid_1's l1: 1.41419
[33000]         training's l1: 0.942169        valid_1's l1: 1.40761
[34000]         training's l1: 0.926526        valid_1's l1: 1.40103
[35000]         training's l1: 0.911221        valid_1's l1: 1.39449
[36000]         training's l1: 0.896472        valid_1's l1: 1.38861
[37000]         training's l1: 0.881826        valid_1's l1: 1.38252
[38000]         training's l1: 0.867474        valid_1's l1: 1.37653
[39000]         training's l1: 0.853657        valid_1's l1: 1.37104
[40000]         training's l1: 0.839972        valid_1's l1: 1.36565
[41000]         training's l1: 0.826688        valid_1's l1: 1.36035
[42000]         training's l1: 0.81377        valid_1's l1: 1.35517
[43000]         training's l1: 0.801096        valid_1's l1: 1.35029
[44000]         training's l1: 0.78851        valid_1's l1: 1.3453
[45000]         training's l1: 0.776449        valid_1's l1: 1.3408
[46000]         training's l1: 0.764442        valid_1's l1: 1.33609
[47000]         training's l1: 0.75271        valid_1's l1: 1.33155
[48000]         training's l1: 0.741112        valid_1's l1: 1.32713
[49000]         training's l1: 0.729728        valid_1's l1: 1.32264
[50000]         training's l1: 0.718624        valid_1's l1: 1.3183
[51000]         training's l1: 0.707809        valid_1's l1: 1.31418
[52000]         training's l1: 0.697242        valid_1's l1: 1.31001
[53000]         training's l1: 0.687011        valid_1's l1: 1.30612
[54000]         training's l1: 0.676808        valid_1's l1: 1.30208
[55000]         training's l1: 0.666926        valid_1's l1: 1.29841
[56000]         training's l1: 0.657104        valid_1's l1: 1.2945
[57000]         training's l1: 0.647453        valid_1's l1: 1.2908
[58000]         training's l1: 0.63789        valid_1's l1: 1.28716
[59000]         training's l1: 0.628679        valid_1's l1: 1.28351
[60000]         training's l1: 0.619705        valid_1's l1: 1.28021
[61000]         training's l1: 0.610866        valid_1's l1: 1.27695
[62000]         training's l1: 0.602284        valid_1's l1: 1.27401
[63000]         training's l1: 0.593762        valid_1's l1: 1.27099
[64000]         training's l1: 0.585397        valid_1's l1: 1.2679
[65000]         training's l1: 0.577125        valid_1's l1: 1.2649
[66000]         training's l1: 0.569035        valid_1's l1: 1.26196
[67000]         training's l1: 0.561118        valid_1's l1: 1.25913
[68000]         training's l1: 0.553463        valid_1's l1: 1.25653
[69000]         training's l1: 0.54597        valid_1's l1: 1.25403
```

```
[70000]         training's l1: 0.538446        valid_1's l1: 1.25129
[71000]         training's l1: 0.531223        valid_1's l1: 1.24886
[72000]         training's l1: 0.524034        valid_1's l1: 1.2465
[73000]         training's l1: 0.516781        valid_1's l1: 1.24385
[74000]         training's l1: 0.509791        valid_1's l1: 1.24147
[75000]         training's l1: 0.502971        valid_1's l1: 1.23906
[76000]         training's l1: 0.49633        valid_1's l1: 1.2368
[77000]         training's l1: 0.489757        valid_1's l1: 1.23477
[78000]         training's l1: 0.483199        valid_1's l1: 1.23252
[79000]         training's l1: 0.476848        valid_1's l1: 1.23035
[80000]         training's l1: 0.470524        valid_1's l1: 1.2282
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.470524        valid_1's l1: 1.2282
MAE: 1.228195
RMSE: 3.172298
working fold 6
fold 6
Training until validation scores don't improve for 200 rounds.
[1000]         training's l1: 2.03981        valid_1's l1: 2.07768
[2000]         training's l1: 1.85478        valid_1's l1: 1.90824
[3000]         training's l1: 1.77917        valid_1's l1: 1.85093
[4000]         training's l1: 1.71677        valid_1's l1: 1.81056
[5000]         training's l1: 1.6646        valid_1's l1: 1.77741
[6000]         training's l1: 1.61812        valid_1's l1: 1.74724
[7000]         training's l1: 1.57447        valid_1's l1: 1.71931
[8000]         training's l1: 1.53397        valid_1's l1: 1.69353
[9000]         training's l1: 1.49592        valid_1's l1: 1.66955
[10000]         training's l1: 1.46043        valid_1's l1: 1.64682
[11000]         training's l1: 1.42717        valid_1's l1: 1.62623
[12000]         training's l1: 1.39533        valid_1's l1: 1.60667
[13000]         training's l1: 1.3652        valid_1's l1: 1.58846
[14000]         training's l1: 1.33646        valid_1's l1: 1.57032
[15000]         training's l1: 1.30891        valid_1's l1: 1.55351
[16000]         training's l1: 1.2827        valid_1's l1: 1.53795
[17000]         training's l1: 1.25731        valid_1's l1: 1.52323
[18000]         training's l1: 1.23304        valid_1's l1: 1.50953
[19000]         training's l1: 1.20959        valid_1's l1: 1.49676
[20000]         training's l1: 1.18688        valid_1's l1: 1.48446
[21000]         training's l1: 1.16508        valid_1's l1: 1.47264
[22000]         training's l1: 1.14377        valid_1's l1: 1.46139
[23000]         training's l1: 1.12327        valid_1's l1: 1.45077
[24000]         training's l1: 1.10314        valid_1's l1: 1.4403
[25000]         training's l1: 1.08367        valid_1's l1: 1.43073
[26000]         training's l1: 1.06471        valid_1's l1: 1.42131
[27000]         training's l1: 1.04629        valid_1's l1: 1.41255
[28000]         training's l1: 1.02803        valid_1's l1: 1.40342
[29000]         training's l1: 1.01047        valid_1's l1: 1.39505
[30000]         training's l1: 0.993383        valid_1's l1: 1.38715
```

```
[31000]         training's l1: 0.976741        valid_1's l1: 1.37965
[32000]         training's l1: 0.960582        valid_1's l1: 1.37253
[33000]         training's l1: 0.944818        valid_1's l1: 1.36571
[34000]         training's l1: 0.929355        valid_1's l1: 1.35901
[35000]         training's l1: 0.914356        valid_1's l1: 1.3523
[36000]         training's l1: 0.899196        valid_1's l1: 1.34527
[37000]         training's l1: 0.884686        valid_1's l1: 1.33895
[38000]         training's l1: 0.870611        valid_1's l1: 1.33315
[39000]         training's l1: 0.85675        valid_1's l1: 1.32728
[40000]         training's l1: 0.84316        valid_1's l1: 1.32158
[41000]         training's l1: 0.82994        valid_1's l1: 1.31605
[42000]         training's l1: 0.816832        valid_1's l1: 1.31065
[43000]         training's l1: 0.804001        valid_1's l1: 1.30519
[44000]         training's l1: 0.791649        valid_1's l1: 1.29998
[45000]         training's l1: 0.77936        valid_1's l1: 1.29491
[46000]         training's l1: 0.767603        valid_1's l1: 1.29024
[47000]         training's l1: 0.756016        valid_1's l1: 1.28541
[48000]         training's l1: 0.744367        valid_1's l1: 1.28089
[49000]         training's l1: 0.733006        valid_1's l1: 1.27626
[50000]         training's l1: 0.721939        valid_1's l1: 1.27206
[51000]         training's l1: 0.711137        valid_1's l1: 1.26807
[52000]         training's l1: 0.700528        valid_1's l1: 1.26387
[53000]         training's l1: 0.689973        valid_1's l1: 1.25967
[54000]         training's l1: 0.679873        valid_1's l1: 1.25589
[55000]         training's l1: 0.669694        valid_1's l1: 1.25187
[56000]         training's l1: 0.659883        valid_1's l1: 1.24819
[57000]         training's l1: 0.650495        valid_1's l1: 1.24462
[58000]         training's l1: 0.641022        valid_1's l1: 1.24103
[59000]         training's l1: 0.631788        valid_1's l1: 1.23746
[60000]         training's l1: 0.622773        valid_1's l1: 1.23413
[61000]         training's l1: 0.613771        valid_1's l1: 1.23082
[62000]         training's l1: 0.60506        valid_1's l1: 1.22754
[63000]         training's l1: 0.596426        valid_1's l1: 1.22432
[64000]         training's l1: 0.588093        valid_1's l1: 1.22139
[65000]         training's l1: 0.579873        valid_1's l1: 1.21855
[66000]         training's l1: 0.57176        valid_1's l1: 1.21557
[67000]         training's l1: 0.563716        valid_1's l1: 1.21259
[68000]         training's l1: 0.555857        valid_1's l1: 1.2097
[69000]         training's l1: 0.548158        valid_1's l1: 1.20699
[70000]         training's l1: 0.540679        valid_1's l1: 1.20443
[71000]         training's l1: 0.533299        valid_1's l1: 1.20192
[72000]         training's l1: 0.525997        valid_1's l1: 1.19937
[73000]         training's l1: 0.518925        valid_1's l1: 1.19672
[74000]         training's l1: 0.511886        valid_1's l1: 1.19434
[75000]         training's l1: 0.504978        valid_1's l1: 1.19211
[76000]         training's l1: 0.498186        valid_1's l1: 1.18984
[77000]         training's l1: 0.491485        valid_1's l1: 1.18767
[78000]         training's l1: 0.484919        valid_1's l1: 1.18549
```

```
[79000]         training's l1: 0.47846        valid_1's l1: 1.18336
[80000]         training's l1: 0.472124        valid_1's l1: 1.18124
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.472124        valid_1's l1: 1.18124
MAE: 1.181244
RMSE: 2.878627
working fold 7
fold 7
Training until validation scores don't improve for 200 rounds.
[1000]         training's l1: 2.03888        valid_1's l1: 2.07513
[2000]         training's l1: 1.85461        valid_1's l1: 1.91763
[3000]         training's l1: 1.78066        valid_1's l1: 1.86358
[4000]         training's l1: 1.72194        valid_1's l1: 1.82173
[5000]         training's l1: 1.67255        valid_1's l1: 1.7865
[6000]         training's l1: 1.62559        valid_1's l1: 1.75372
[7000]         training's l1: 1.58301        valid_1's l1: 1.72407
[8000]         training's l1: 1.54345        valid_1's l1: 1.69775
[9000]         training's l1: 1.50518        valid_1's l1: 1.67228
[10000]         training's l1: 1.46952        valid_1's l1: 1.64925
[11000]         training's l1: 1.43552        valid_1's l1: 1.62727
[12000]         training's l1: 1.40358        valid_1's l1: 1.60766
[13000]         training's l1: 1.37325        valid_1's l1: 1.58854
[14000]         training's l1: 1.34408        valid_1's l1: 1.57089
[15000]         training's l1: 1.31644        valid_1's l1: 1.55475
[16000]         training's l1: 1.28951        valid_1's l1: 1.53926
[17000]         training's l1: 1.26371        valid_1's l1: 1.52463
[18000]         training's l1: 1.23898        valid_1's l1: 1.51063
[19000]         training's l1: 1.21499        valid_1's l1: 1.49753
[20000]         training's l1: 1.19168        valid_1's l1: 1.48501
[21000]         training's l1: 1.16949        valid_1's l1: 1.47326
[22000]         training's l1: 1.14798        valid_1's l1: 1.4619
[23000]         training's l1: 1.12674        valid_1's l1: 1.45103
[24000]         training's l1: 1.10631        valid_1's l1: 1.44125
[25000]         training's l1: 1.08624        valid_1's l1: 1.43179
[26000]         training's l1: 1.0669        valid_1's l1: 1.42245
[27000]         training's l1: 1.04805        valid_1's l1: 1.41377
[28000]         training's l1: 1.02975        valid_1's l1: 1.4054
[29000]         training's l1: 1.01189        valid_1's l1: 1.3976
[30000]         training's l1: 0.994592        valid_1's l1: 1.38962
[31000]         training's l1: 0.977956        valid_1's l1: 1.3822
[32000]         training's l1: 0.96129        valid_1's l1: 1.37476
[33000]         training's l1: 0.944915        valid_1's l1: 1.36743
[34000]         training's l1: 0.929174        valid_1's l1: 1.36094
[35000]         training's l1: 0.913958        valid_1's l1: 1.35491
[36000]         training's l1: 0.898947        valid_1's l1: 1.34842
[37000]         training's l1: 0.884567        valid_1's l1: 1.34222
[38000]         training's l1: 0.870371        valid_1's l1: 1.33636
[39000]         training's l1: 0.856678        valid_1's l1: 1.33098
```

```
[40000]	training's l1: 0.842974	valid_1's l1: 1.32503
[41000]	training's l1: 0.829744	valid_1's l1: 1.31952
[42000]	training's l1: 0.816886	valid_1's l1: 1.31435
[43000]	training's l1: 0.804321	valid_1's l1: 1.30967
[44000]	training's l1: 0.791705	valid_1's l1: 1.30445
[45000]	training's l1: 0.779561	valid_1's l1: 1.29972
[46000]	training's l1: 0.767631	valid_1's l1: 1.29487
[47000]	training's l1: 0.75597	valid_1's l1: 1.29055
[48000]	training's l1: 0.74452	valid_1's l1: 1.28626
[49000]	training's l1: 0.733259	valid_1's l1: 1.2817
[50000]	training's l1: 0.722157	valid_1's l1: 1.2772
[51000]	training's l1: 0.711426	valid_1's l1: 1.27308
[52000]	training's l1: 0.700969	valid_1's l1: 1.26922
[53000]	training's l1: 0.690558	valid_1's l1: 1.26526
[54000]	training's l1: 0.680336	valid_1's l1: 1.26153
[55000]	training's l1: 0.670289	valid_1's l1: 1.25772
[56000]	training's l1: 0.660608	valid_1's l1: 1.25421
[57000]	training's l1: 0.651052	valid_1's l1: 1.25084
[58000]	training's l1: 0.641719	valid_1's l1: 1.24746
[59000]	training's l1: 0.63257	valid_1's l1: 1.24441
[60000]	training's l1: 0.62367	valid_1's l1: 1.24163
[61000]	training's l1: 0.614802	valid_1's l1: 1.23832
[62000]	training's l1: 0.606106	valid_1's l1: 1.23525
[63000]	training's l1: 0.597541	valid_1's l1: 1.23216
[64000]	training's l1: 0.589269	valid_1's l1: 1.22937
[65000]	training's l1: 0.58106	valid_1's l1: 1.22658
[66000]	training's l1: 0.572938	valid_1's l1: 1.2239
[67000]	training's l1: 0.564971	valid_1's l1: 1.22128
[68000]	training's l1: 0.557183	valid_1's l1: 1.21864
[69000]	training's l1: 0.549578	valid_1's l1: 1.21623
[70000]	training's l1: 0.542057	valid_1's l1: 1.21373
[71000]	training's l1: 0.534855	valid_1's l1: 1.21153
[72000]	training's l1: 0.52763	valid_1's l1: 1.20931
[73000]	training's l1: 0.520544	valid_1's l1: 1.20694
[74000]	training's l1: 0.513504	valid_1's l1: 1.20463
[75000]	training's l1: 0.506726	valid_1's l1: 1.20263
[76000]	training's l1: 0.499947	valid_1's l1: 1.20047
[77000]	training's l1: 0.493298	valid_1's l1: 1.19823
[78000]	training's l1: 0.486741	valid_1's l1: 1.19615
[79000]	training's l1: 0.480277	valid_1's l1: 1.19415
[80000]	training's l1: 0.4741	valid_1's l1: 1.19233
Did not meet early stopping. Best iteration is:
[80000]	training's l1: 0.4741	valid_1's l1: 1.19233
MAE: 1.192332
RMSE: 2.923887
MAEs [1.1696316311538906, 1.2143186072035126, 1.1891716654034454, 1.2160141266822453, 1.205707
MAE mean: 1.199577
RMSEs [2.799177168134087, 3.0651151358315993, 2.9270192104579733, 3.187612870861349, 3.1280786
```

```
RMSE mean: 3.010227
```

```
In [22]: print(scaled_train_X.isnull().any().any())
         scaled_train_X=scaled_train_X.fillna(0)
         scaled_test_X=scaled_test_X.fillna(0)
         print(scaled_train_X.isnull().any().any())
```

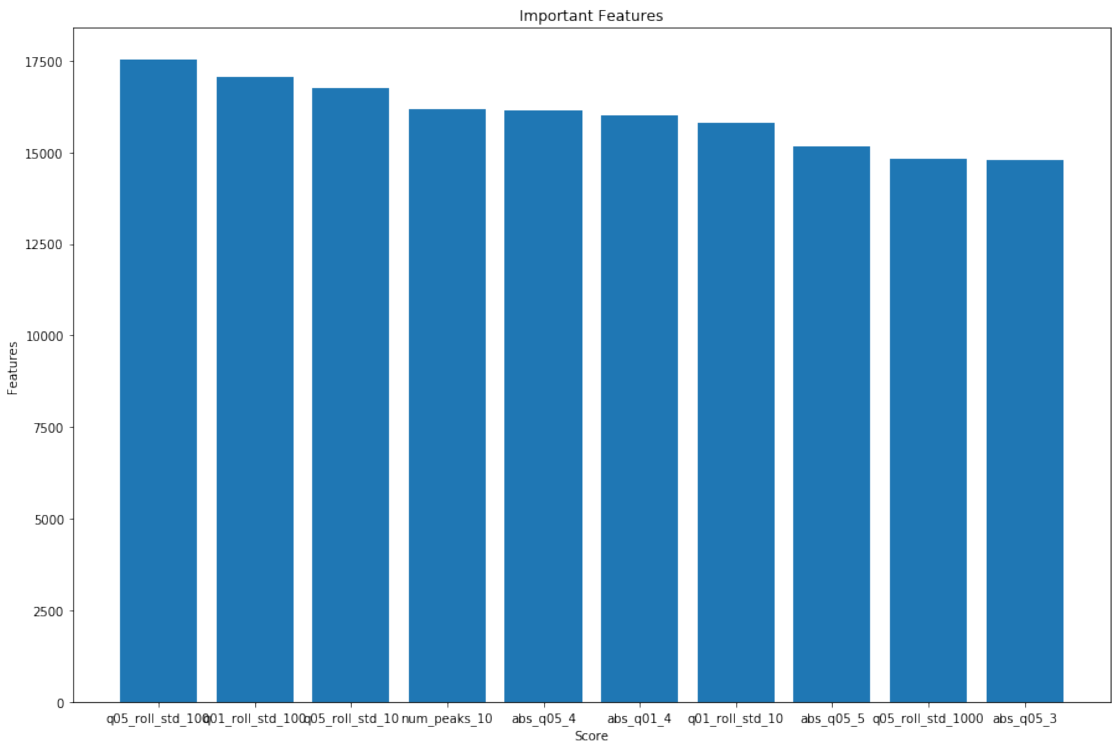
```
False
False
```

5.3 Feature Selection

```
In [32]: import sklearn
         #normalising, since sklearns selectkbest does not work with negative features
         scaler = sklearn.preprocessing.MinMaxScaler()
         X_train_norm=scaler.fit_transform(scaled_train_X)
         #converting to dataframe
         X_train_norm=pd.DataFrame(X_train_norm,columns=scaled_train_X.columns)
```

```
In [34]: #using sklearns selectkbest
         import matplotlib.pyplot as plt
         fig, ax = plt.subplots(figsize=(15, 10))
         from sklearn.feature_selection import SelectKBest
         from sklearn.feature_selection import chi2
         train_y = pd.read_csv('train_y.csv')
         X = X_train_norm
         y = train_y
         #  extracting top 10 features
         bestfeatures = SelectKBest(score_func=sklearn.feature_selection.f_regression, k=10)
         fit = bestfeatures.fit(X,y)
         scores_df = pd.DataFrame(fit.scores_)
         columns_df = pd.DataFrame(X.columns)
         #concat two dataframes for better visualization
         topfeatures = pd.concat([columns_df,scores_df],axis=1)
         topfeatures.columns = ['features','Score']

         topfeatures=topfeatures.sort_values(by='Score',ascending=False)
         print(topfeatures[0:10])

         print('-----------------------------------------------------------------------------
         print('-----------------------------------------------------------------------------
         plt.bar(topfeatures.features[0:10],topfeatures.Score[0:10])
         plt.ylabel('Features')
         plt.title('Important Features')
         plt.xlabel('Score')
         plt.show()
```

```
               features         Score
826   q05_roll_std_100  17523.290389
825   q01_roll_std_100  17044.985850
804    q05_roll_std_10  16761.917094
917       num_peaks_10  16189.163749
411          abs_q05_4  16141.192817
412          abs_q01_4  16003.954512
803    q01_roll_std_10  15792.506711
482          abs_q05_5  15175.374271
848  q05_roll_std_1000  14811.628390
340          abs_q05_3  14803.987410
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
```



```
In [55]: #considering top 300 features
         truncated_train=scaled_train_X[topfeatures[0:300]['features'].tolist()]

In [56]: truncated_test=scaled_test_X[topfeatures[0:300]['features'].tolist()]

In [62]: #with feature set 1 and 2
         params = {'num_leaves': 21,
                   'min_data_in_leaf': 20,
                   'objective':'gamma',
                   'learning_rate': 0.001,
                   'max_depth': 108,
                   "boosting": "gbdt",
                   "feature_fraction": 0.91,
                   "bagging_freq": 1,
                   "bagging_fraction": 0.91,
                   "bagging_seed": 42,
                   "metric": 'mae',
                   "lambda_l1": 0.1,
                   "verbosity": -1,
                   "random_state": 42}


         def lgb_truncated_model():
             maes = []
             rmses = []
             submission = pd.read_csv(os.path.join(DATA_DIR, 'sample_submission.csv'), index_co
             #scaled_train_X = scaled_train_X
             #scaled_test_X = scaled_test_X
             train_y = pd.read_csv('train_y.csv')
             predictions = np.zeros(len(scaled_test_X))

             n_fold = 8
             folds = KFold(n_splits=n_fold, shuffle=True, random_state=42)

             fold_importance_df = pd.DataFrame()
             fold_importance_df["Feature"] = truncated_train.columns

             for fold_, (trn_idx, val_idx) in enumerate(folds.split(truncated_train, train_y.va
                 print('working fold %d' % fold_)
                 strLog = "fold {}".format(fold_)
                 print(strLog)

                 X_tr, X_val = truncated_train.iloc[trn_idx], truncated_train.iloc[val_idx]
                 y_tr, y_val = train_y.iloc[trn_idx], train_y.iloc[val_idx]

                 model = lgb.LGBMRegressor(**params, n_estimators=80000, n_jobs=-1)
                 model.fit(X_tr, y_tr,
                           eval_set=[(X_tr, y_tr), (X_val, y_val)], eval_metric='mae',
                           verbose=1000, early_stopping_rounds=200)

                 # predictions
                 preds = model.predict(truncated_test, num_iteration=model.best_iteration_)
                 predictions += preds / folds.n_splits
                 preds = model.predict(X_val, num_iteration=model.best_iteration_)

                 # mean absolute error
                 mae = mean_absolute_error(y_val, preds)
```

```
            print('MAE: %.6f' % mae)
            maes.append(mae)

            # root mean squared error
            rmse = mean_squared_error(y_val, preds)
            print('RMSE: %.6f' % rmse)
            rmses.append(rmse)

            fold_importance_df['importance_%d' % fold_] = model.feature_importances_[:len

        print('MAEs', maes)
        print('MAE mean: %.6f' % np.mean(maes))
        print('RMSEs', rmses)
        print('RMSE mean: %.6f' % np.mean(rmses))

        submission.time_to_failure = predictions
        submission.to_csv('submission_lgb_truncated.csv', index=False)
        fold_importance_df.to_csv('fold_imp_lgb_8_80k_108dp.csv')
        return model
```

```
In [63]: clf7=lgb_truncated_model()
```

```
working fold 0
fold 0
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.05775        valid_1's l1: 2.08199
[2000]        training's l1: 1.87762        valid_1's l1: 1.93895
[3000]        training's l1: 1.8135        valid_1's l1: 1.89157
[4000]        training's l1: 1.76696        valid_1's l1: 1.85747
[5000]        training's l1: 1.72922        valid_1's l1: 1.8307
[6000]        training's l1: 1.69591        valid_1's l1: 1.80817
[7000]        training's l1: 1.66383        valid_1's l1: 1.78704
[8000]        training's l1: 1.63385        valid_1's l1: 1.76782
[9000]        training's l1: 1.60539        valid_1's l1: 1.74965
[10000]        training's l1: 1.57822        valid_1's l1: 1.73277
[11000]        training's l1: 1.55161        valid_1's l1: 1.71598
[12000]        training's l1: 1.52633        valid_1's l1: 1.70056
[13000]        training's l1: 1.50242        valid_1's l1: 1.68634
[14000]        training's l1: 1.47953        valid_1's l1: 1.67341
[15000]        training's l1: 1.45675        valid_1's l1: 1.66071
[16000]        training's l1: 1.43498        valid_1's l1: 1.64845
[17000]        training's l1: 1.41365        valid_1's l1: 1.63668
[18000]        training's l1: 1.39303        valid_1's l1: 1.62569
[19000]        training's l1: 1.37287        valid_1's l1: 1.61537
[20000]        training's l1: 1.35377        valid_1's l1: 1.60524
[21000]        training's l1: 1.33462        valid_1's l1: 1.59523
[22000]        training's l1: 1.31596        valid_1's l1: 1.58542
[23000]        training's l1: 1.29768        valid_1's l1: 1.57614
```

```
[24000]         training's l1: 1.27976        valid_1's l1: 1.56686
[25000]         training's l1: 1.26246        valid_1's l1: 1.55805
[26000]         training's l1: 1.24532        valid_1's l1: 1.54963
[27000]         training's l1: 1.22849        valid_1's l1: 1.54142
[28000]         training's l1: 1.21179        valid_1's l1: 1.53322
[29000]         training's l1: 1.19598        valid_1's l1: 1.52532
[30000]         training's l1: 1.18054        valid_1's l1: 1.51771
[31000]         training's l1: 1.1651        valid_1's l1: 1.51024
[32000]         training's l1: 1.15013        valid_1's l1: 1.50317
[33000]         training's l1: 1.13565        valid_1's l1: 1.49637
[34000]         training's l1: 1.12128        valid_1's l1: 1.48957
[35000]         training's l1: 1.1072        valid_1's l1: 1.48279
[36000]         training's l1: 1.09334        valid_1's l1: 1.4763
[37000]         training's l1: 1.07955        valid_1's l1: 1.47003
[38000]         training's l1: 1.06629        valid_1's l1: 1.46376
[39000]         training's l1: 1.0533        valid_1's l1: 1.45784
[40000]         training's l1: 1.04062        valid_1's l1: 1.45179
[41000]         training's l1: 1.0281        valid_1's l1: 1.44592
[42000]         training's l1: 1.01576        valid_1's l1: 1.44033
[43000]         training's l1: 1.00348        valid_1's l1: 1.43506
[44000]         training's l1: 0.99159        valid_1's l1: 1.4295
[45000]         training's l1: 0.979692        valid_1's l1: 1.42398
[46000]         training's l1: 0.968326        valid_1's l1: 1.41863
[47000]         training's l1: 0.956817        valid_1's l1: 1.41327
[48000]         training's l1: 0.945955        valid_1's l1: 1.40836
[49000]         training's l1: 0.935013        valid_1's l1: 1.4035
[50000]         training's l1: 0.924187        valid_1's l1: 1.39868
[51000]         training's l1: 0.913455        valid_1's l1: 1.39373
[52000]         training's l1: 0.903063        valid_1's l1: 1.38884
[53000]         training's l1: 0.892764        valid_1's l1: 1.38434
[54000]         training's l1: 0.882652        valid_1's l1: 1.37988
[55000]         training's l1: 0.872649        valid_1's l1: 1.37548
[56000]         training's l1: 0.863044        valid_1's l1: 1.37095
[57000]         training's l1: 0.853505        valid_1's l1: 1.3667
[58000]         training's l1: 0.843978        valid_1's l1: 1.36242
[59000]         training's l1: 0.834633        valid_1's l1: 1.35833
[60000]         training's l1: 0.825321        valid_1's l1: 1.35405
[61000]         training's l1: 0.816313        valid_1's l1: 1.35006
[62000]         training's l1: 0.807433        valid_1's l1: 1.34611
[63000]         training's l1: 0.798764        valid_1's l1: 1.3421
[64000]         training's l1: 0.790074        valid_1's l1: 1.33836
[65000]         training's l1: 0.781761        valid_1's l1: 1.33476
[66000]         training's l1: 0.77321        valid_1's l1: 1.33097
[67000]         training's l1: 0.764868        valid_1's l1: 1.32758
[68000]         training's l1: 0.756459        valid_1's l1: 1.32395
[69000]         training's l1: 0.748383        valid_1's l1: 1.32038
[70000]         training's l1: 0.740348        valid_1's l1: 1.31702
[71000]         training's l1: 0.732458        valid_1's l1: 1.31369
```

```
[72000]        training's l1: 0.724734       valid_1's l1: 1.31053
[73000]        training's l1: 0.717047       valid_1's l1: 1.30727
[74000]        training's l1: 0.709549       valid_1's l1: 1.30409
[75000]        training's l1: 0.702233       valid_1's l1: 1.30105
[76000]        training's l1: 0.69503       valid_1's l1: 1.29809
[77000]        training's l1: 0.687908       valid_1's l1: 1.29522
[78000]        training's l1: 0.680868       valid_1's l1: 1.29234
[79000]        training's l1: 0.673916       valid_1's l1: 1.2894
[80000]        training's l1: 0.666885       valid_1's l1: 1.28651
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.666885       valid_1's l1: 1.28651
MAE: 1.286508
RMSE: 3.312443
working fold 1
fold 1
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.0482       valid_1's l1: 2.12459
[2000]        training's l1: 1.87271       valid_1's l1: 1.9718
[3000]        training's l1: 1.80764       valid_1's l1: 1.92407
[4000]        training's l1: 1.76158       valid_1's l1: 1.89175
[5000]        training's l1: 1.72415       valid_1's l1: 1.86709
[6000]        training's l1: 1.691       valid_1's l1: 1.84568
[7000]        training's l1: 1.65896       valid_1's l1: 1.82597
[8000]        training's l1: 1.62804       valid_1's l1: 1.80779
[9000]        training's l1: 1.59866       valid_1's l1: 1.79043
[10000]        training's l1: 1.57001       valid_1's l1: 1.77414
[11000]        training's l1: 1.54374       valid_1's l1: 1.75881
[12000]        training's l1: 1.51868       valid_1's l1: 1.74453
[13000]        training's l1: 1.49446       valid_1's l1: 1.73113
[14000]        training's l1: 1.47057       valid_1's l1: 1.71781
[15000]        training's l1: 1.44825       valid_1's l1: 1.70537
[16000]        training's l1: 1.42656       valid_1's l1: 1.6939
[17000]        training's l1: 1.40529       valid_1's l1: 1.68267
[18000]        training's l1: 1.38412       valid_1's l1: 1.67119
[19000]        training's l1: 1.36399       valid_1's l1: 1.6607
[20000]        training's l1: 1.34463       valid_1's l1: 1.65045
[21000]        training's l1: 1.32586       valid_1's l1: 1.64079
[22000]        training's l1: 1.30746       valid_1's l1: 1.63136
[23000]        training's l1: 1.28927       valid_1's l1: 1.62223
[24000]        training's l1: 1.27149       valid_1's l1: 1.61341
[25000]        training's l1: 1.25364       valid_1's l1: 1.60435
[26000]        training's l1: 1.23641       valid_1's l1: 1.59585
[27000]        training's l1: 1.21984       valid_1's l1: 1.58786
[28000]        training's l1: 1.20352       valid_1's l1: 1.58018
[29000]        training's l1: 1.18757       valid_1's l1: 1.57273
[30000]        training's l1: 1.17151       valid_1's l1: 1.5651
[31000]        training's l1: 1.15598       valid_1's l1: 1.55796
[32000]        training's l1: 1.14085       valid_1's l1: 1.55126
```

```
[33000]         training's l1: 1.12583        valid_1's l1: 1.54436
[34000]         training's l1: 1.11147        valid_1's l1: 1.53797
[35000]         training's l1: 1.09689        valid_1's l1: 1.53141
[36000]         training's l1: 1.08324        valid_1's l1: 1.52553
[37000]         training's l1: 1.06935        valid_1's l1: 1.51925
[38000]         training's l1: 1.05599        valid_1's l1: 1.5133
[39000]         training's l1: 1.04292        valid_1's l1: 1.50742
[40000]         training's l1: 1.03019        valid_1's l1: 1.50188
[41000]         training's l1: 1.01733        valid_1's l1: 1.49633
[42000]         training's l1: 1.00488        valid_1's l1: 1.49085
[43000]         training's l1: 0.992871        valid_1's l1: 1.4858
[44000]         training's l1: 0.980933        valid_1's l1: 1.48074
[45000]         training's l1: 0.969209        valid_1's l1: 1.47594
[46000]         training's l1: 0.957929        valid_1's l1: 1.47114
[47000]         training's l1: 0.946666        valid_1's l1: 1.46627
[48000]         training's l1: 0.935773        valid_1's l1: 1.46159
[49000]         training's l1: 0.924899        valid_1's l1: 1.45707
[50000]         training's l1: 0.914499        valid_1's l1: 1.45294
[51000]         training's l1: 0.904161        valid_1's l1: 1.44868
[52000]         training's l1: 0.893893        valid_1's l1: 1.44416
[53000]         training's l1: 0.884028        valid_1's l1: 1.44019
[54000]         training's l1: 0.874149        valid_1's l1: 1.43608
[55000]         training's l1: 0.864517        valid_1's l1: 1.4321
[56000]         training's l1: 0.854922        valid_1's l1: 1.4283
[57000]         training's l1: 0.845506        valid_1's l1: 1.42459
[58000]         training's l1: 0.836006        valid_1's l1: 1.42061
[59000]         training's l1: 0.826768        valid_1's l1: 1.41707
[60000]         training's l1: 0.817693        valid_1's l1: 1.41362
[61000]         training's l1: 0.808739        valid_1's l1: 1.41006
[62000]         training's l1: 0.799842        valid_1's l1: 1.40667
[63000]         training's l1: 0.791145        valid_1's l1: 1.40333
[64000]         training's l1: 0.782613        valid_1's l1: 1.39998
[65000]         training's l1: 0.77426        valid_1's l1: 1.39677
[66000]         training's l1: 0.766091        valid_1's l1: 1.39351
[67000]         training's l1: 0.75791        valid_1's l1: 1.3903
[68000]         training's l1: 0.749957        valid_1's l1: 1.38732
[69000]         training's l1: 0.742035        valid_1's l1: 1.38423
[70000]         training's l1: 0.734188        valid_1's l1: 1.38107
[71000]         training's l1: 0.726479        valid_1's l1: 1.37812
[72000]         training's l1: 0.718781        valid_1's l1: 1.37504
[73000]         training's l1: 0.711292        valid_1's l1: 1.37208
[74000]         training's l1: 0.704076        valid_1's l1: 1.36943
[75000]         training's l1: 0.696738        valid_1's l1: 1.36665
[76000]         training's l1: 0.689424        valid_1's l1: 1.3637
[77000]         training's l1: 0.682394        valid_1's l1: 1.3609
[78000]         training's l1: 0.675382        valid_1's l1: 1.35814
[79000]         training's l1: 0.668418        valid_1's l1: 1.3554
[80000]         training's l1: 0.661623        valid_1's l1: 1.35268
```

```
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.661623       valid_1's l1: 1.35268
MAE: 1.352682
RMSE: 3.641151
working fold 2
fold 2
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.06166       valid_1's l1: 2.05121
[2000]        training's l1: 1.88595       valid_1's l1: 1.91016
[3000]        training's l1: 1.82232       valid_1's l1: 1.86581
[4000]        training's l1: 1.77403       valid_1's l1: 1.83658
[5000]        training's l1: 1.73561       valid_1's l1: 1.81547
[6000]        training's l1: 1.7024       valid_1's l1: 1.79778
[7000]        training's l1: 1.67072       valid_1's l1: 1.78136
[8000]        training's l1: 1.64       valid_1's l1: 1.76613
[9000]        training's l1: 1.61088       valid_1's l1: 1.75181
[10000]        training's l1: 1.58267       valid_1's l1: 1.73765
[11000]        training's l1: 1.55604       valid_1's l1: 1.72483
[12000]        training's l1: 1.53107       valid_1's l1: 1.71283
[13000]        training's l1: 1.5066       valid_1's l1: 1.70135
[14000]        training's l1: 1.48342       valid_1's l1: 1.69035
[15000]        training's l1: 1.46073       valid_1's l1: 1.67976
[16000]        training's l1: 1.439       valid_1's l1: 1.66986
[17000]        training's l1: 1.418       valid_1's l1: 1.65997
[18000]        training's l1: 1.39719       valid_1's l1: 1.65033
[19000]        training's l1: 1.37692       valid_1's l1: 1.64124
[20000]        training's l1: 1.35711       valid_1's l1: 1.63194
[21000]        training's l1: 1.33791       valid_1's l1: 1.62309
[22000]        training's l1: 1.31887       valid_1's l1: 1.6145
[23000]        training's l1: 1.30079       valid_1's l1: 1.60631
[24000]        training's l1: 1.2825       valid_1's l1: 1.59805
[25000]        training's l1: 1.26536       valid_1's l1: 1.59017
[26000]        training's l1: 1.24819       valid_1's l1: 1.58234
[27000]        training's l1: 1.23134       valid_1's l1: 1.57477
[28000]        training's l1: 1.2149       valid_1's l1: 1.56773
[29000]        training's l1: 1.19892       valid_1's l1: 1.56055
[30000]        training's l1: 1.18303       valid_1's l1: 1.55354
[31000]        training's l1: 1.16808       valid_1's l1: 1.54672
[32000]        training's l1: 1.15294       valid_1's l1: 1.54003
[33000]        training's l1: 1.13855       valid_1's l1: 1.53359
[34000]        training's l1: 1.12436       valid_1's l1: 1.52749
[35000]        training's l1: 1.11006       valid_1's l1: 1.52095
[36000]        training's l1: 1.0963       valid_1's l1: 1.51505
[37000]        training's l1: 1.08257       valid_1's l1: 1.50881
[38000]        training's l1: 1.06921       valid_1's l1: 1.50266
[39000]        training's l1: 1.05633       valid_1's l1: 1.49704
[40000]        training's l1: 1.04338       valid_1's l1: 1.49123
[41000]        training's l1: 1.03056       valid_1's l1: 1.48562
```

```
[42000]         training's l1: 1.01824        valid_1's l1: 1.48021
[43000]         training's l1: 1.00591        valid_1's l1: 1.47484
[44000]         training's l1: 0.993709        valid_1's l1: 1.46943
[45000]         training's l1: 0.982039        valid_1's l1: 1.46433
[46000]         training's l1: 0.970305        valid_1's l1: 1.45926
[47000]         training's l1: 0.958903        valid_1's l1: 1.45442
[48000]         training's l1: 0.947658        valid_1's l1: 1.44957
[49000]         training's l1: 0.936515        valid_1's l1: 1.44469
[50000]         training's l1: 0.925802        valid_1's l1: 1.44033
[51000]         training's l1: 0.915171        valid_1's l1: 1.43586
[52000]         training's l1: 0.904794        valid_1's l1: 1.43168
[53000]         training's l1: 0.894293        valid_1's l1: 1.42729
[54000]         training's l1: 0.884025        valid_1's l1: 1.42298
[55000]         training's l1: 0.874002        valid_1's l1: 1.41875
[56000]         training's l1: 0.863989        valid_1's l1: 1.41474
[57000]         training's l1: 0.85421        valid_1's l1: 1.41058
[58000]         training's l1: 0.844573        valid_1's l1: 1.40647
[59000]         training's l1: 0.835178        valid_1's l1: 1.40265
[60000]         training's l1: 0.826003        valid_1's l1: 1.3987
[61000]         training's l1: 0.816778        valid_1's l1: 1.39487
[62000]         training's l1: 0.807764        valid_1's l1: 1.39133
[63000]         training's l1: 0.798965        valid_1's l1: 1.38773
[64000]         training's l1: 0.790344        valid_1's l1: 1.38439
[65000]         training's l1: 0.781652        valid_1's l1: 1.38101
[66000]         training's l1: 0.773189        valid_1's l1: 1.37767
[67000]         training's l1: 0.764772        valid_1's l1: 1.37425
[68000]         training's l1: 0.756498        valid_1's l1: 1.37099
[69000]         training's l1: 0.748428        valid_1's l1: 1.36773
[70000]         training's l1: 0.74057        valid_1's l1: 1.36448
[71000]         training's l1: 0.732637        valid_1's l1: 1.36117
[72000]         training's l1: 0.725055        valid_1's l1: 1.35813
[73000]         training's l1: 0.717519        valid_1's l1: 1.35497
[74000]         training's l1: 0.710053        valid_1's l1: 1.35203
[75000]         training's l1: 0.702606        valid_1's l1: 1.34908
[76000]         training's l1: 0.695137        valid_1's l1: 1.34595
[77000]         training's l1: 0.688003        valid_1's l1: 1.34309
[78000]         training's l1: 0.680798        valid_1's l1: 1.34011
[79000]         training's l1: 0.673787        valid_1's l1: 1.33728
[80000]         training's l1: 0.666872        valid_1's l1: 1.33451
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.666872        valid_1's l1: 1.33451
MAE: 1.334506
RMSE: 3.545344
working fold 3
fold 3
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.05201        valid_1's l1: 2.09657
[2000]        training's l1: 1.87966        valid_1's l1: 1.94574
```

```
[3000]         training's l1: 1.81466       valid_1's l1: 1.89807
[4000]         training's l1: 1.76639       valid_1's l1: 1.86776
[5000]         training's l1: 1.72794       valid_1's l1: 1.84347
[6000]         training's l1: 1.69505       valid_1's l1: 1.82208
[7000]         training's l1: 1.66389       valid_1's l1: 1.80192
[8000]         training's l1: 1.63415       valid_1's l1: 1.78318
[9000]         training's l1: 1.60552       valid_1's l1: 1.76551
[10000]         training's l1: 1.57794       valid_1's l1: 1.74926
[11000]         training's l1: 1.55128       valid_1's l1: 1.73445
[12000]         training's l1: 1.52621       valid_1's l1: 1.72025
[13000]         training's l1: 1.50249       valid_1's l1: 1.70746
[14000]         training's l1: 1.47881       valid_1's l1: 1.69482
[15000]         training's l1: 1.45619       valid_1's l1: 1.68259
[16000]         training's l1: 1.43425       valid_1's l1: 1.67118
[17000]         training's l1: 1.41298       valid_1's l1: 1.6602
[18000]         training's l1: 1.39247       valid_1's l1: 1.64973
[19000]         training's l1: 1.37212       valid_1's l1: 1.6394
[20000]         training's l1: 1.35221       valid_1's l1: 1.62992
[21000]         training's l1: 1.33279       valid_1's l1: 1.62056
[22000]         training's l1: 1.31395       valid_1's l1: 1.61143
[23000]         training's l1: 1.29528       valid_1's l1: 1.60219
[24000]         training's l1: 1.2773       valid_1's l1: 1.59403
[25000]         training's l1: 1.25958       valid_1's l1: 1.58572
[26000]         training's l1: 1.24215       valid_1's l1: 1.57725
[27000]         training's l1: 1.22511       valid_1's l1: 1.56915
[28000]         training's l1: 1.20867       valid_1's l1: 1.56131
[29000]         training's l1: 1.19247       valid_1's l1: 1.5537
[30000]         training's l1: 1.17659       valid_1's l1: 1.54616
[31000]         training's l1: 1.16128       valid_1's l1: 1.53915
[32000]         training's l1: 1.14605       valid_1's l1: 1.53223
[33000]         training's l1: 1.13091       valid_1's l1: 1.52531
[34000]         training's l1: 1.11646       valid_1's l1: 1.51879
[35000]         training's l1: 1.10229       valid_1's l1: 1.51233
[36000]         training's l1: 1.088       valid_1's l1: 1.50605
[37000]         training's l1: 1.07425       valid_1's l1: 1.5
[38000]         training's l1: 1.06077       valid_1's l1: 1.49401
[39000]         training's l1: 1.04731       valid_1's l1: 1.48799
[40000]         training's l1: 1.03461       valid_1's l1: 1.48237
[41000]         training's l1: 1.02167       valid_1's l1: 1.47685
[42000]         training's l1: 1.00931       valid_1's l1: 1.47149
[43000]         training's l1: 0.997273       valid_1's l1: 1.46647
[44000]         training's l1: 0.985263       valid_1's l1: 1.46138
[45000]         training's l1: 0.973196       valid_1's l1: 1.4563
[46000]         training's l1: 0.961271       valid_1's l1: 1.45135
[47000]         training's l1: 0.950052       valid_1's l1: 1.44657
[48000]         training's l1: 0.938806       valid_1's l1: 1.44189
[49000]         training's l1: 0.928093       valid_1's l1: 1.43739
[50000]         training's l1: 0.917401       valid_1's l1: 1.43291
```

```
[51000]         training's l1: 0.906932        valid_1's l1: 1.42872
[52000]         training's l1: 0.896451        valid_1's l1: 1.42417
[53000]         training's l1: 0.886338        valid_1's l1: 1.42002
[54000]         training's l1: 0.87616        valid_1's l1: 1.41607
[55000]         training's l1: 0.866508        valid_1's l1: 1.41219
[56000]         training's l1: 0.85693        valid_1's l1: 1.40827
[57000]         training's l1: 0.847342        valid_1's l1: 1.40442
[58000]         training's l1: 0.837881        valid_1's l1: 1.40048
[59000]         training's l1: 0.828629        valid_1's l1: 1.39678
[60000]         training's l1: 0.819547        valid_1's l1: 1.39305
[61000]         training's l1: 0.81089        valid_1's l1: 1.38962
[62000]         training's l1: 0.802129        valid_1's l1: 1.38609
[63000]         training's l1: 0.793508        valid_1's l1: 1.38254
[64000]         training's l1: 0.785069        valid_1's l1: 1.37912
[65000]         training's l1: 0.776744        valid_1's l1: 1.37583
[66000]         training's l1: 0.768621        valid_1's l1: 1.37255
[67000]         training's l1: 0.760615        valid_1's l1: 1.36944
[68000]         training's l1: 0.752657        valid_1's l1: 1.36624
[69000]         training's l1: 0.744809        valid_1's l1: 1.36316
[70000]         training's l1: 0.737134        valid_1's l1: 1.36023
[71000]         training's l1: 0.729469        valid_1's l1: 1.35715
[72000]         training's l1: 0.721947        valid_1's l1: 1.35415
[73000]         training's l1: 0.714579        valid_1's l1: 1.35129
[74000]         training's l1: 0.707323        valid_1's l1: 1.34851
[75000]         training's l1: 0.700087        valid_1's l1: 1.34556
[76000]         training's l1: 0.692983        valid_1's l1: 1.34288
[77000]         training's l1: 0.685833        valid_1's l1: 1.34003
[78000]         training's l1: 0.678676        valid_1's l1: 1.33724
[79000]         training's l1: 0.671596        valid_1's l1: 1.33435
[80000]         training's l1: 0.664711        valid_1's l1: 1.33162
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.664711        valid_1's l1: 1.33162
MAE: 1.331624
RMSE: 3.617643
working fold 4
fold 4
Training until validation scores don't improve for 200 rounds.
[1000]         training's l1: 2.05283        valid_1's l1: 2.09467
[2000]         training's l1: 1.87953        valid_1's l1: 1.92807
[3000]         training's l1: 1.81634        valid_1's l1: 1.87442
[4000]         training's l1: 1.77156        valid_1's l1: 1.84144
[5000]         training's l1: 1.73408        valid_1's l1: 1.81775
[6000]         training's l1: 1.70074        valid_1's l1: 1.79734
[7000]         training's l1: 1.66898        valid_1's l1: 1.77817
[8000]         training's l1: 1.63748        valid_1's l1: 1.76006
[9000]         training's l1: 1.60671        valid_1's l1: 1.74331
[10000]         training's l1: 1.57804        valid_1's l1: 1.72772
[11000]         training's l1: 1.55095        valid_1's l1: 1.7135
```

```
[12000]         training's l1: 1.5245       valid_1's l1: 1.69985
[13000]         training's l1: 1.49967       valid_1's l1: 1.6868
[14000]         training's l1: 1.47562       valid_1's l1: 1.67479
[15000]         training's l1: 1.4526       valid_1's l1: 1.66318
[16000]         training's l1: 1.43059       valid_1's l1: 1.65246
[17000]         training's l1: 1.40894       valid_1's l1: 1.64229
[18000]         training's l1: 1.38842       valid_1's l1: 1.6327
[19000]         training's l1: 1.36826       valid_1's l1: 1.62314
[20000]         training's l1: 1.34854       valid_1's l1: 1.6139
[21000]         training's l1: 1.32973       valid_1's l1: 1.60507
[22000]         training's l1: 1.31093       valid_1's l1: 1.59643
[23000]         training's l1: 1.29266       valid_1's l1: 1.58803
[24000]         training's l1: 1.27492       valid_1's l1: 1.58027
[25000]         training's l1: 1.25731       valid_1's l1: 1.57223
[26000]         training's l1: 1.24038       valid_1's l1: 1.5647
[27000]         training's l1: 1.22354       valid_1's l1: 1.55715
[28000]         training's l1: 1.2076       valid_1's l1: 1.55037
[29000]         training's l1: 1.19207       valid_1's l1: 1.54378
[30000]         training's l1: 1.17654       valid_1's l1: 1.5375
[31000]         training's l1: 1.16147       valid_1's l1: 1.53115
[32000]         training's l1: 1.14668       valid_1's l1: 1.52499
[33000]         training's l1: 1.13203       valid_1's l1: 1.5187
[34000]         training's l1: 1.11736       valid_1's l1: 1.51234
[35000]         training's l1: 1.10348       valid_1's l1: 1.50652
[36000]         training's l1: 1.08998       valid_1's l1: 1.50096
[37000]         training's l1: 1.07662       valid_1's l1: 1.49561
[38000]         training's l1: 1.06333       valid_1's l1: 1.49012
[39000]         training's l1: 1.05014       valid_1's l1: 1.48475
[40000]         training's l1: 1.03721       valid_1's l1: 1.47949
[41000]         training's l1: 1.02474       valid_1's l1: 1.47438
[42000]         training's l1: 1.01259       valid_1's l1: 1.46967
[43000]         training's l1: 1.00075       valid_1's l1: 1.46485
[44000]         training's l1: 0.988814       valid_1's l1: 1.46005
[45000]         training's l1: 0.97682       valid_1's l1: 1.45506
[46000]         training's l1: 0.965555       valid_1's l1: 1.4506
[47000]         training's l1: 0.95417       valid_1's l1: 1.44599
[48000]         training's l1: 0.942812       valid_1's l1: 1.44142
[49000]         training's l1: 0.931518       valid_1's l1: 1.43685
[50000]         training's l1: 0.920664       valid_1's l1: 1.43238
[51000]         training's l1: 0.910143       valid_1's l1: 1.42828
[52000]         training's l1: 0.899534       valid_1's l1: 1.42399
[53000]         training's l1: 0.889287       valid_1's l1: 1.41967
[54000]         training's l1: 0.879433       valid_1's l1: 1.41561
[55000]         training's l1: 0.869475       valid_1's l1: 1.41189
[56000]         training's l1: 0.859801       valid_1's l1: 1.40803
[57000]         training's l1: 0.850141       valid_1's l1: 1.40418
[58000]         training's l1: 0.84067       valid_1's l1: 1.40049
[59000]         training's l1: 0.831158       valid_1's l1: 1.39673
```

```
[60000]	training's l1: 0.821779	valid_1's l1: 1.39307
[61000]	training's l1: 0.812678	valid_1's l1: 1.38955
[62000]	training's l1: 0.803698	valid_1's l1: 1.38593
[63000]	training's l1: 0.794756	valid_1's l1: 1.38237
[64000]	training's l1: 0.786109	valid_1's l1: 1.37915
[65000]	training's l1: 0.777501	valid_1's l1: 1.37566
[66000]	training's l1: 0.769232	valid_1's l1: 1.37237
[67000]	training's l1: 0.760901	valid_1's l1: 1.36915
[68000]	training's l1: 0.752585	valid_1's l1: 1.36591
[69000]	training's l1: 0.744643	valid_1's l1: 1.36312
[70000]	training's l1: 0.736593	valid_1's l1: 1.35996
[71000]	training's l1: 0.728776	valid_1's l1: 1.35687
[72000]	training's l1: 0.721197	valid_1's l1: 1.3541
[73000]	training's l1: 0.713484	valid_1's l1: 1.35114
[74000]	training's l1: 0.705995	valid_1's l1: 1.34841
[75000]	training's l1: 0.698675	valid_1's l1: 1.34546
[76000]	training's l1: 0.691478	valid_1's l1: 1.34264
[77000]	training's l1: 0.684134	valid_1's l1: 1.33953
[78000]	training's l1: 0.677093	valid_1's l1: 1.33684
[79000]	training's l1: 0.670104	valid_1's l1: 1.3342
[80000]	training's l1: 0.663161	valid_1's l1: 1.33148
Did not meet early stopping. Best iteration is:
[80000]	training's l1: 0.663161	valid_1's l1: 1.33148
MAE: 1.331479
RMSE: 3.670256
working fold 5
fold 5
Training until validation scores don't improve for 200 rounds.
[1000]	training's l1: 2.05468	valid_1's l1: 2.08885
[2000]	training's l1: 1.87939	valid_1's l1: 1.93676
[3000]	training's l1: 1.8135	valid_1's l1: 1.89084
[4000]	training's l1: 1.76692	valid_1's l1: 1.86306
[5000]	training's l1: 1.72891	valid_1's l1: 1.84017
[6000]	training's l1: 1.69643	valid_1's l1: 1.82066
[7000]	training's l1: 1.66434	valid_1's l1: 1.80101
[8000]	training's l1: 1.63352	valid_1's l1: 1.78335
[9000]	training's l1: 1.60375	valid_1's l1: 1.76683
[10000]	training's l1: 1.57519	valid_1's l1: 1.75096
[11000]	training's l1: 1.54794	valid_1's l1: 1.73669
[12000]	training's l1: 1.52212	valid_1's l1: 1.72311
[13000]	training's l1: 1.49692	valid_1's l1: 1.71029
[14000]	training's l1: 1.47332	valid_1's l1: 1.69851
[15000]	training's l1: 1.4506	valid_1's l1: 1.68741
[16000]	training's l1: 1.42857	valid_1's l1: 1.67674
[17000]	training's l1: 1.40719	valid_1's l1: 1.66655
[18000]	training's l1: 1.38613	valid_1's l1: 1.65626
[19000]	training's l1: 1.36556	valid_1's l1: 1.6465
[20000]	training's l1: 1.34575	valid_1's l1: 1.63719
```

```
[21000]        training's l1: 1.32604      valid_1's l1: 1.62789
[22000]        training's l1: 1.3072      valid_1's l1: 1.61925
[23000]        training's l1: 1.28895      valid_1's l1: 1.61053
[24000]        training's l1: 1.27069      valid_1's l1: 1.60228
[25000]        training's l1: 1.25347      valid_1's l1: 1.59432
[26000]        training's l1: 1.23607      valid_1's l1: 1.58619
[27000]        training's l1: 1.21899      valid_1's l1: 1.57865
[28000]        training's l1: 1.20236      valid_1's l1: 1.57105
[29000]        training's l1: 1.18617      valid_1's l1: 1.56402
[30000]        training's l1: 1.17041      valid_1's l1: 1.55683
[31000]        training's l1: 1.15499      valid_1's l1: 1.54987
[32000]        training's l1: 1.13962      valid_1's l1: 1.54299
[33000]        training's l1: 1.12487      valid_1's l1: 1.53648
[34000]        training's l1: 1.11023      valid_1's l1: 1.52998
[35000]        training's l1: 1.09628      valid_1's l1: 1.52394
[36000]        training's l1: 1.08236      valid_1's l1: 1.51774
[37000]        training's l1: 1.06886      valid_1's l1: 1.51172
[38000]        training's l1: 1.05553      valid_1's l1: 1.50578
[39000]        training's l1: 1.04248      valid_1's l1: 1.49979
[40000]        training's l1: 1.02968      valid_1's l1: 1.4941
[41000]        training's l1: 1.01704      valid_1's l1: 1.48819
[42000]        training's l1: 1.0048      valid_1's l1: 1.48276
[43000]        training's l1: 0.992682      valid_1's l1: 1.47745
[44000]        training's l1: 0.980834      valid_1's l1: 1.47239
[45000]        training's l1: 0.969161      valid_1's l1: 1.46755
[46000]        training's l1: 0.957939      valid_1's l1: 1.4628
[47000]        training's l1: 0.946793      valid_1's l1: 1.45794
[48000]        training's l1: 0.935887      valid_1's l1: 1.45341
[49000]        training's l1: 0.924835      valid_1's l1: 1.44847
[50000]        training's l1: 0.914095      valid_1's l1: 1.4438
[51000]        training's l1: 0.903699      valid_1's l1: 1.43936
[52000]        training's l1: 0.893359      valid_1's l1: 1.43498
[53000]        training's l1: 0.883289      valid_1's l1: 1.43073
[54000]        training's l1: 0.873404      valid_1's l1: 1.42664
[55000]        training's l1: 0.863488      valid_1's l1: 1.42228
[56000]        training's l1: 0.853801      valid_1's l1: 1.41792
[57000]        training's l1: 0.844401      valid_1's l1: 1.41403
[58000]        training's l1: 0.835164      valid_1's l1: 1.41023
[59000]        training's l1: 0.82572      valid_1's l1: 1.40632
[60000]        training's l1: 0.816687      valid_1's l1: 1.40244
[61000]        training's l1: 0.807614      valid_1's l1: 1.39852
[62000]        training's l1: 0.798913      valid_1's l1: 1.39489
[63000]        training's l1: 0.790273      valid_1's l1: 1.39144
[64000]        training's l1: 0.781744      valid_1's l1: 1.38789
[65000]        training's l1: 0.773415      valid_1's l1: 1.38432
[66000]        training's l1: 0.765222      valid_1's l1: 1.38069
[67000]        training's l1: 0.756974      valid_1's l1: 1.37744
[68000]        training's l1: 0.748972      valid_1's l1: 1.37435
```

```
[69000]        training's l1: 0.741047       valid_1's l1: 1.37091
[70000]        training's l1: 0.733351       valid_1's l1: 1.36773
[71000]        training's l1: 0.725507       valid_1's l1: 1.36451
[72000]        training's l1: 0.717895       valid_1's l1: 1.36136
[73000]        training's l1: 0.710513       valid_1's l1: 1.3582
[74000]        training's l1: 0.703123       valid_1's l1: 1.35518
[75000]        training's l1: 0.695876       valid_1's l1: 1.35211
[76000]        training's l1: 0.688611       valid_1's l1: 1.34916
[77000]        training's l1: 0.681461       valid_1's l1: 1.34603
[78000]        training's l1: 0.674479       valid_1's l1: 1.34312
[79000]        training's l1: 0.667694       valid_1's l1: 1.34032
[80000]        training's l1: 0.661062       valid_1's l1: 1.33745
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.661062       valid_1's l1: 1.33745
MAE: 1.337449
RMSE: 3.628906
working fold 6
fold 6
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.05419       valid_1's l1: 2.09199
[2000]        training's l1: 1.87872       valid_1's l1: 1.93505
[3000]        training's l1: 1.81268       valid_1's l1: 1.8846
[4000]        training's l1: 1.7651       valid_1's l1: 1.85505
[5000]        training's l1: 1.72856       valid_1's l1: 1.83364
[6000]        training's l1: 1.69459       valid_1's l1: 1.81355
[7000]        training's l1: 1.66326       valid_1's l1: 1.79484
[8000]        training's l1: 1.63325       valid_1's l1: 1.77762
[9000]        training's l1: 1.60444       valid_1's l1: 1.76152
[10000]        training's l1: 1.57696       valid_1's l1: 1.74585
[11000]        training's l1: 1.5506       valid_1's l1: 1.73128
[12000]        training's l1: 1.52543       valid_1's l1: 1.71677
[13000]        training's l1: 1.50134       valid_1's l1: 1.70316
[14000]        training's l1: 1.47845       valid_1's l1: 1.69002
[15000]        training's l1: 1.45607       valid_1's l1: 1.67746
[16000]        training's l1: 1.43388       valid_1's l1: 1.66494
[17000]        training's l1: 1.41248       valid_1's l1: 1.65288
[18000]        training's l1: 1.3922       valid_1's l1: 1.642
[19000]        training's l1: 1.37208       valid_1's l1: 1.63033
[20000]        training's l1: 1.35262       valid_1's l1: 1.61977
[21000]        training's l1: 1.33319       valid_1's l1: 1.60922
[22000]        training's l1: 1.31453       valid_1's l1: 1.59895
[23000]        training's l1: 1.29584       valid_1's l1: 1.58848
[24000]        training's l1: 1.27779       valid_1's l1: 1.57885
[25000]        training's l1: 1.25965       valid_1's l1: 1.56908
[26000]        training's l1: 1.24247       valid_1's l1: 1.55959
[27000]        training's l1: 1.22594       valid_1's l1: 1.55134
[28000]        training's l1: 1.20961       valid_1's l1: 1.54302
[29000]        training's l1: 1.19353       valid_1's l1: 1.53481
```

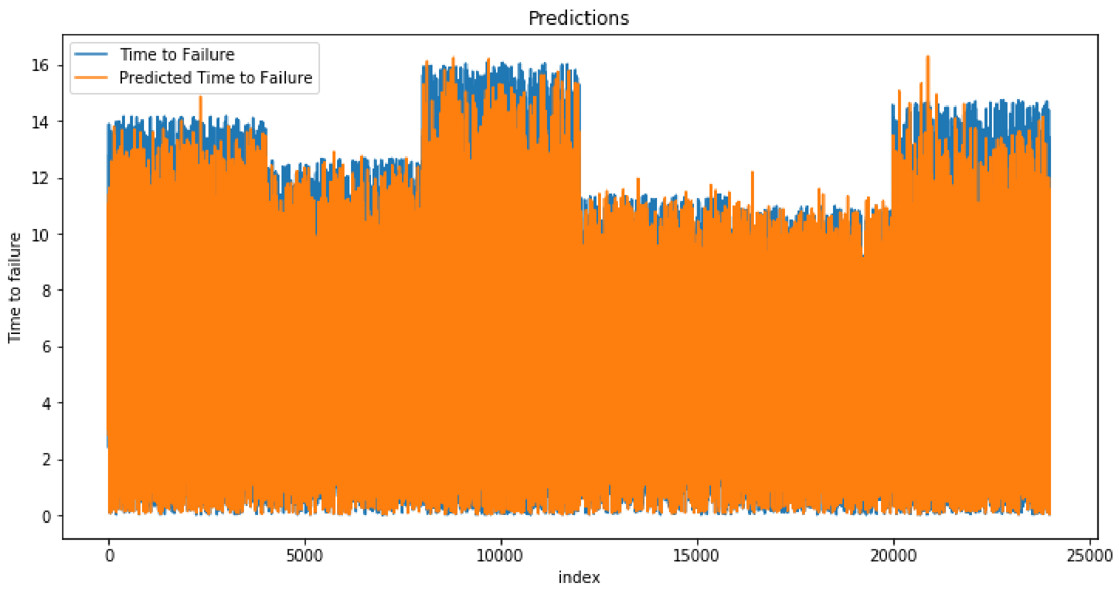
```
[30000]         training's l1: 1.17756        valid_1's l1: 1.52689
[31000]         training's l1: 1.16235        valid_1's l1: 1.51937
[32000]         training's l1: 1.14699        valid_1's l1: 1.51158
[33000]         training's l1: 1.13211        valid_1's l1: 1.50408
[34000]         training's l1: 1.11763        valid_1's l1: 1.49692
[35000]         training's l1: 1.10308        valid_1's l1: 1.48977
[36000]         training's l1: 1.08912        valid_1's l1: 1.48321
[37000]         training's l1: 1.07563        valid_1's l1: 1.4769
[38000]         training's l1: 1.06207        valid_1's l1: 1.47041
[39000]         training's l1: 1.04893        valid_1's l1: 1.46426
[40000]         training's l1: 1.0361        valid_1's l1: 1.45828
[41000]         training's l1: 1.02351        valid_1's l1: 1.45236
[42000]         training's l1: 1.01092        valid_1's l1: 1.44638
[43000]         training's l1: 0.998868        valid_1's l1: 1.44085
[44000]         training's l1: 0.986906        valid_1's l1: 1.43544
[45000]         training's l1: 0.975061        valid_1's l1: 1.43021
[46000]         training's l1: 0.963405        valid_1's l1: 1.42505
[47000]         training's l1: 0.951934        valid_1's l1: 1.41974
[48000]         training's l1: 0.940645        valid_1's l1: 1.41466
[49000]         training's l1: 0.929645        valid_1's l1: 1.40992
[50000]         training's l1: 0.91879        valid_1's l1: 1.4052
[51000]         training's l1: 0.908052        valid_1's l1: 1.40032
[52000]         training's l1: 0.897425        valid_1's l1: 1.39569
[53000]         training's l1: 0.887074        valid_1's l1: 1.39139
[54000]         training's l1: 0.876761        valid_1's l1: 1.38694
[55000]         training's l1: 0.866711        valid_1's l1: 1.38265
[56000]         training's l1: 0.856939        valid_1's l1: 1.37859
[57000]         training's l1: 0.847245        valid_1's l1: 1.37452
[58000]         training's l1: 0.837648        valid_1's l1: 1.37063
[59000]         training's l1: 0.82856        valid_1's l1: 1.36692
[60000]         training's l1: 0.819042        valid_1's l1: 1.36301
[61000]         training's l1: 0.810063        valid_1's l1: 1.35939
[62000]         training's l1: 0.801182        valid_1's l1: 1.35572
[63000]         training's l1: 0.792484        valid_1's l1: 1.35218
[64000]         training's l1: 0.783751        valid_1's l1: 1.34855
[65000]         training's l1: 0.775368        valid_1's l1: 1.3452
[66000]         training's l1: 0.76713        valid_1's l1: 1.34199
[67000]         training's l1: 0.758921        valid_1's l1: 1.33847
[68000]         training's l1: 0.750774        valid_1's l1: 1.3352
[69000]         training's l1: 0.742694        valid_1's l1: 1.33195
[70000]         training's l1: 0.734787        valid_1's l1: 1.32866
[71000]         training's l1: 0.726844        valid_1's l1: 1.32536
[72000]         training's l1: 0.719323        valid_1's l1: 1.3222
[73000]         training's l1: 0.711844        valid_1's l1: 1.31928
[74000]         training's l1: 0.704288        valid_1's l1: 1.31628
[75000]         training's l1: 0.696808        valid_1's l1: 1.3131
[76000]         training's l1: 0.689541        valid_1's l1: 1.31011
[77000]         training's l1: 0.682258        valid_1's l1: 1.30708
```

```
[78000]        training's l1: 0.675206       valid_1's l1: 1.30422
[79000]        training's l1: 0.66815        valid_1's l1: 1.30144
[80000]        training's l1: 0.661315       valid_1's l1: 1.29857
Did not meet early stopping. Best iteration is:
[80000]        training's l1: 0.661315       valid_1's l1: 1.29857
MAE: 1.298571
RMSE: 3.326851
working fold 7
fold 7
Training until validation scores don't improve for 200 rounds.
[1000]        training's l1: 2.05549       valid_1's l1: 2.09004
[2000]        training's l1: 1.88111       valid_1's l1: 1.94927
[3000]        training's l1: 1.81628       valid_1's l1: 1.90262
[4000]        training's l1: 1.76987       valid_1's l1: 1.87145
[5000]        training's l1: 1.72988       valid_1's l1: 1.84389
[6000]        training's l1: 1.69575       valid_1's l1: 1.82236
[7000]        training's l1: 1.66337       valid_1's l1: 1.80337
[8000]        training's l1: 1.63274       valid_1's l1: 1.78517
[9000]        training's l1: 1.60356       valid_1's l1: 1.7676
[10000]        training's l1: 1.57515       valid_1's l1: 1.7515
[11000]        training's l1: 1.54811       valid_1's l1: 1.73634
[12000]        training's l1: 1.52178       valid_1's l1: 1.72151
[13000]        training's l1: 1.49683       valid_1's l1: 1.70732
[14000]        training's l1: 1.47269       valid_1's l1: 1.69385
[15000]        training's l1: 1.44952       valid_1's l1: 1.68107
[16000]        training's l1: 1.42749       valid_1's l1: 1.66918
[17000]        training's l1: 1.40607       valid_1's l1: 1.65764
[18000]        training's l1: 1.38499       valid_1's l1: 1.64645
[19000]        training's l1: 1.36444       valid_1's l1: 1.63559
[20000]        training's l1: 1.34409       valid_1's l1: 1.62484
[21000]        training's l1: 1.32491       valid_1's l1: 1.61502
[22000]        training's l1: 1.30582       valid_1's l1: 1.6055
[23000]        training's l1: 1.28773       valid_1's l1: 1.59651
[24000]        training's l1: 1.27012       valid_1's l1: 1.58789
[25000]        training's l1: 1.25252       valid_1's l1: 1.57924
[26000]        training's l1: 1.23525       valid_1's l1: 1.57053
[27000]        training's l1: 1.21864       valid_1's l1: 1.56256
[28000]        training's l1: 1.20203       valid_1's l1: 1.55425
[29000]        training's l1: 1.18591       valid_1's l1: 1.5467
[30000]        training's l1: 1.17017       valid_1's l1: 1.53899
[31000]        training's l1: 1.15511       valid_1's l1: 1.53229
[32000]        training's l1: 1.13996       valid_1's l1: 1.52547
[33000]        training's l1: 1.12497       valid_1's l1: 1.51876
[34000]        training's l1: 1.11006       valid_1's l1: 1.51188
[35000]        training's l1: 1.09552       valid_1's l1: 1.50491
[36000]        training's l1: 1.08163       valid_1's l1: 1.49845
[37000]        training's l1: 1.06774       valid_1's l1: 1.49204
[38000]        training's l1: 1.05449       valid_1's l1: 1.48625
```

```
[39000]         training's l1: 1.04139       valid_1's l1: 1.4806
[40000]         training's l1: 1.02831       valid_1's l1: 1.47498
[41000]         training's l1: 1.01558       valid_1's l1: 1.46973
[42000]         training's l1: 1.00301       valid_1's l1: 1.46444
[43000]         training's l1: 0.990731       valid_1's l1: 1.45917
[44000]         training's l1: 0.978695       valid_1's l1: 1.45396
[45000]         training's l1: 0.967024       valid_1's l1: 1.44896
[46000]         training's l1: 0.955334       valid_1's l1: 1.44391
[47000]         training's l1: 0.943973       valid_1's l1: 1.4389
[48000]         training's l1: 0.932841       valid_1's l1: 1.43422
[49000]         training's l1: 0.921776       valid_1's l1: 1.42981
[50000]         training's l1: 0.911094       valid_1's l1: 1.42551
[51000]         training's l1: 0.900587       valid_1's l1: 1.42114
[52000]         training's l1: 0.890394       valid_1's l1: 1.41699
[53000]         training's l1: 0.880117       valid_1's l1: 1.41287
[54000]         training's l1: 0.870367       valid_1's l1: 1.40904
[55000]         training's l1: 0.86058       valid_1's l1: 1.40505
[56000]         training's l1: 0.85076       valid_1's l1: 1.40109
[57000]         training's l1: 0.841621       valid_1's l1: 1.39742
[58000]         training's l1: 0.832285       valid_1's l1: 1.39366
[59000]         training's l1: 0.82342       valid_1's l1: 1.39019
[60000]         training's l1: 0.814509       valid_1's l1: 1.38669
[61000]         training's l1: 0.805652       valid_1's l1: 1.38317
[62000]         training's l1: 0.797     valid_1's l1: 1.37973
[63000]         training's l1: 0.788391       valid_1's l1: 1.37638
[64000]         training's l1: 0.779918       valid_1's l1: 1.37316
[65000]         training's l1: 0.771422       valid_1's l1: 1.3699
[66000]         training's l1: 0.7633      valid_1's l1: 1.36669
[67000]         training's l1: 0.755148       valid_1's l1: 1.36356
[68000]         training's l1: 0.747138       valid_1's l1: 1.36046
[69000]         training's l1: 0.739408       valid_1's l1: 1.35734
[70000]         training's l1: 0.73165       valid_1's l1: 1.35434
[71000]         training's l1: 0.724042       valid_1's l1: 1.35148
[72000]         training's l1: 0.716497       valid_1's l1: 1.34858
[73000]         training's l1: 0.70909       valid_1's l1: 1.34594
[74000]         training's l1: 0.701815       valid_1's l1: 1.34321
[75000]         training's l1: 0.694695       valid_1's l1: 1.34049
[76000]         training's l1: 0.687756       valid_1's l1: 1.33774
[77000]         training's l1: 0.680906       valid_1's l1: 1.3352
[78000]         training's l1: 0.674053       valid_1's l1: 1.33281
[79000]         training's l1: 0.667246       valid_1's l1: 1.33039
[80000]         training's l1: 0.660597       valid_1's l1: 1.32817
Did not meet early stopping. Best iteration is:
[80000]         training's l1: 0.660597       valid_1's l1: 1.32817
MAE: 1.328167
RMSE: 3.532193
MAEs [1.286508349172183, 1.3526816090059066, 1.3345057643705267, 1.331624205621076, 1.33147877
MAE mean: 1.325123
```

```
RMSEs [3.3124433391396586, 3.6411511924737865, 3.5453442987192916, 3.617642513462967, 3.670255
RMSE mean: 3.534348
```

```
In [69]: y_predicted=clf7.predict(truncated_train,num_iteration=clf7.best_iteration_)
         plot_op(y_predicted)
```



6 XGBOOST

```
In [3]: def train_model_xgb( params ):
            X=pd.read_csv('scaled_train_X.csv')
            X_testset=pd.read_csv('scaled_test_X.csv')
            scaled_check_X=pd.read_csv('scaled_check_X.csv')

            scaled_train_X=pd.read_csv('scaled_train_X.csv')
            y=pd.read_csv('train_y.csv')

            n_fold = 8
            folds = KFold(n_splits=n_fold, shuffle=True, random_state=42)

            x_value = np.zeros(len(X))
            prediction = np.zeros(len(X_testset))
            prediction_train = np.zeros(len(scaled_train_X))
            prediction_check = np.zeros(len(scaled_check_X))
            scores = []
            feature_importance = pd.DataFrame()
            for fold_n, (trainset_index, valid_set_index) in enumerate(folds.split(X)):
```

```
            print('Fold', fold_n, 'started at', time.ctime())
            X_train_per_fold, X_valid_per_fold = X.iloc[trainset_index], X.iloc[valid_set_i
            y_train_per_fold, y_valid_per_fold = y.iloc[trainset_index], y.iloc[valid_set_i

            train_data = xgb.DMatrix(data=X_train_per_fold, label=y_train_per_fold, feature
            valid_data = xgb.DMatrix(data=X_valid_per_fold, label=y_valid_per_fold, feature

            watchlist = [(train_data, 'train'), (valid_data, 'valid_data')]
            model = xgb.train(dtrain=train_data, num_boost_round=800, evals=watchlist, earl
            y_pred_valid = model.predict(xgb.DMatrix(X_valid_per_fold, feature_names=X.colu
            y_pred = model.predict(xgb.DMatrix(X_testset, feature_names=X.columns), ntree_l

            y_pred_check = model.predict(xgb.DMatrix(scaled_check_X, feature_names=X.column
            y_pred_train = model.predict(xgb.DMatrix(scaled_train_X, feature_names=X.column

            x_value[valid_set_index] = y_pred_valid.reshape(-1,)
            scores.append(mean_absolute_error(y_valid_per_fold, y_pred_valid))

            prediction +=y_pred
            prediction_train+=y_pred_train
            prediction_check+=y_pred_check



        prediction /= n_fold
        prediction_train/= n_fold
        prediction_check/= n_fold

        print('CV mean score: {0:.6f}.'.format(mean_absolute_error(y, x_value)))
        return model,x_value, prediction,prediction_train,prediction_check
```

```
In [4]: xgb_params = {'eta': 0.01,
                      'max_depth': 6,
                      'colsample_bytree': 0.9,
                      'lambda': 0.1,
                      'alpha' : 0.1,
                      'objective': 'reg:gamma',
                      'eval_metric': 'mae',
                      'silent': True, 'nthread':24}
        model, x_value_xgb, prediction_xgb,prediction_train_xgb,prediction_check_xgb = train_mo
```

```
Fold 0 started at Sun May 26 04:16:11 2019
[0]	train-mae:5.29924	valid_data-mae:5.32894
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.51174	valid_data-mae:1.67802
Fold 1 started at Sun May 26 04:18:04 2019
```

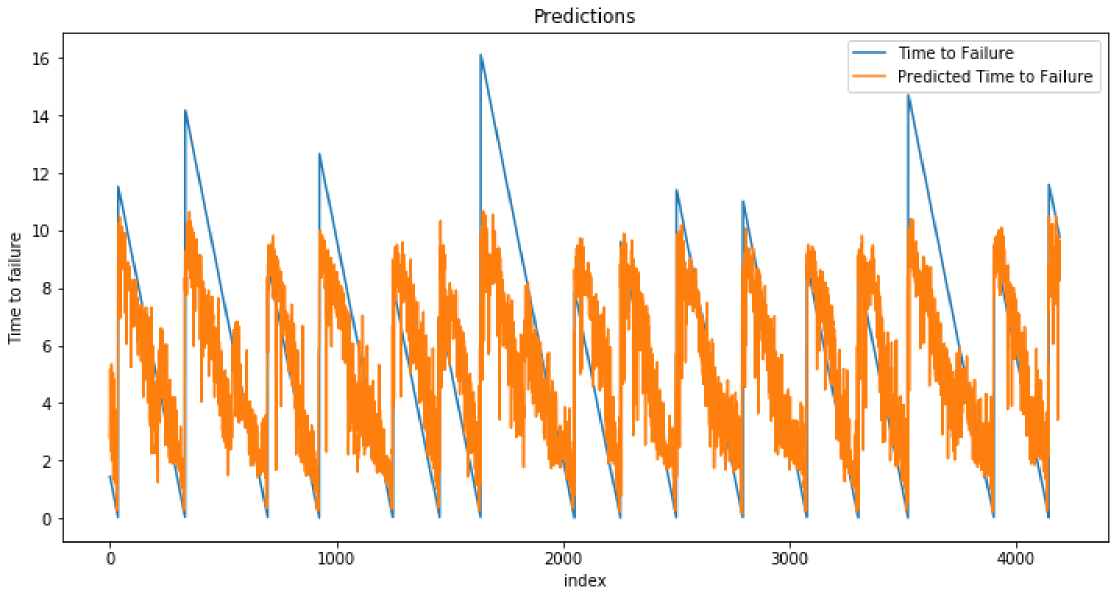
```
[0]	train-mae:5.29436	valid_data-mae:5.36295
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.49884	valid_data-mae:1.72009
Fold 2 started at Sun May 26 04:19:55 2019
[0]	train-mae:5.31318	valid_data-mae:5.23115
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.50613	valid_data-mae:1.67086
Fold 3 started at Sun May 26 04:21:47 2019
[0]	train-mae:5.30434	valid_data-mae:5.29324
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.49652	valid_data-mae:1.71047
Fold 4 started at Sun May 26 04:23:40 2019
[0]	train-mae:5.30728	valid_data-mae:5.27258
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.49666	valid_data-mae:1.68269
Fold 5 started at Sun May 26 04:25:31 2019
[0]	train-mae:5.29138	valid_data-mae:5.38397
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.49961	valid_data-mae:1.70772
Fold 6 started at Sun May 26 04:27:22 2019
[0]	train-mae:5.30489	valid_data-mae:5.28945
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.49795	valid_data-mae:1.66814
Fold 7 started at Sun May 26 04:29:14 2019
[0]	train-mae:5.30887	valid_data-mae:5.26157
Multiple eval metrics have been passed: 'valid_data-mae' will be used for early stopping.

Will train until valid_data-mae hasn't improved in 200 rounds.
[799]	train-mae:1.50718	valid_data-mae:1.69186
CV mean score: 1.691231.
```

```
In [6]: pd.DataFrame(prediction_train_xgb).to_csv("prediction_train_xgb_800.csv", header=None,
        pd.DataFrame(prediction_check_xgb).to_csv("prediction_check_xgb_800.csv", header=None,
```

```
In [5]: submission = pd.read_csv('sample_submission.csv', index_col='seg_id')
```

```
submission['time_to_failure'] = prediction_xgb
submission.to_csv('xgboost800unsavedmodel.csv')
```

```
In [12]: plot_op(prediction_check_xgb)
```



we can see that the predictions are generalising well and not overfitting .

6.1 Stacking

```
In [84]: train_pred_lgb=pd.read_csv('predictions_train_lgb.csv',header=None)
         train_pred_xgb=pd.read_csv('prediction_train_xgb_800.csv',header=None)

         test_pred_lgb=pd.read_csv('submission_lgb_with_gamma.csv')
         test_pred_xgb=pd.read_csv('xgboost800unsavedmodel.csv')
```

```
In [85]: train_pred_lgb=train_pred_lgb.drop(train_pred_lgb.columns[0], axis=1)
```

```
In [87]: test_pred_xgb=test_pred_xgb['time_to_failure']
```

```
In [89]: from sklearn.linear_model import LinearRegression
         #train_stack = np.vstack([train_pred_lgb, train_pred_xgb]).transpose()
         #test_stack = np.vstack([test_pred_lgb,test_pred_xgb]).transpose()
         df = pd.concat([train_pred_lgb, train_pred_xgb], axis=1)
         df_test = pd.concat([test_pred_lgb, test_pred_xgb], axis=1)
         train_y = pd.read_csv('train_y.csv')

         model = LinearRegression(fit_intercept=True)
         model.fit(df,train_y)
         stacked_predictions=model.predict(df_test)
```

```
In [90]: stacked_predictions[0:10]

Out[90]: array([[3.57879183],
                [4.89777915],
                [5.31633825],
                [8.67293754],
                [6.13879032],
                [1.86658712],
                [8.55401978],
                [4.58090179],
                [4.18928864],
                [2.20161818]])
```

```
In [92]: #1.379
         submission = pd.read_csv('sample_submission.csv', index_col='seg_id')
         submission['time_to_failure'] = stacked_predictions
         submission.to_csv('stacked_predictionslr.csv')
```

6.2 Conclusion

Objective:To predict the time remaining before laboratory earthquakes occur from real-time seismic data.

1. We are given a dataset with 629145480 rows and 2 columns: acoustic_data, time_to_failure, where time_to_failure is the time remaining for next earthquake.
2. We visualize the train and test data to get the pattern and observer that there is a spike in siesmic data before earthquake occurs and there are a total of 16 earthquakes in train data.
3. We divide the data into 6 slice and take 4000 random samples from each slice and get 24000 training data rows. We use multiprocessing to reduce the time taken to run.
4. We then featurize the data using simple statistical features like mean,std,moving averages etc and also signal processing features like fft, peaks, hjorth parameters.
5. I tried hyperparameter tuning with gridsearchcv and the performance reduced,we can also see by results that CV is reliable, hence i used the default values.
6. We Apply various machine learning models, we use 8 fold cv compare the cross validation result and plot the corresponding feature importances.
7. Since not all features contribute to the model, we use feature selection to get the top features.
8. We use sklearns selectkbest to find the top 300 features and then apply models on it and compare them, the score went down slightly.
9. We try simple stacking of the models with linear regression as model and the score doesnt improve.

```
In [13]: from prettytable import PrettyTable

         x=PrettyTable()

         x.field_names=['Feature Selection','Feature set','Algorithm','CV MAE','TEST MAE']
         x.add_row([" - ",'Feature set1',"XGB",1.69,1.314])
         x.add_row([" - ",'Feature set1',"LGBM",1.218,1.340])
         x.add_row([" selectkbest ",'Feature set1+2',"LGBM", 1.325,1.455])
```

```
x.add_row(["-",'Feature set1+2',"LGBM",1.199,3.51])

print(x)
```

```
+--------------------+----------------+-----------+--------+----------+
| Feature Selection  |  Feature set   | Algorithm | CV MAE | TEST MAE |
+--------------------+----------------+-----------+--------+----------+
|         -          |  Feature set1  |    XGB    |  1.69  |  1.314   |
|         -          |  Feature set1  |    LGBM   | 1.218  |   1.34   |
|    selectkbest     | Feature set1+2 |    LGBM   | 1.325  |  1.455   |
|         -          | Feature set1+2 |    LGBM   | 1.199  |   3.51   |
+--------------------+----------------+-----------+--------+----------+
```

We can see that cv might not be reliable

XGB gives the highest score of 1.314 which is currently at the 27th position at the kaggle public leaderboard.

```
In [ ]:
```