

ansi_regression

April 9, 2018

1 ANSI Application analysis

```
In [1]: import numpy
        import pandas
        import matplotlib.pyplot as plotter
        from scipy.stats import pearsonr, probplot
        from sklearn.metrics import mean_squared_error, mean_absolute_error

In [2]: def view_boxplot(df):
        %matplotlib
        df.boxplot()
        plotter.show()
```

1.1 CPU data

```
In [3]: cpu_df = pandas.read_csv('data/ansi_fake_data/ansi_fake_data_cpu.csv', index_col='Time')

In [4]: #cpu_df.columns

In [5]: #view_boxplot(cpu_df)
```

1.2 Network TX

```
In [6]: txnet_df = pandas.read_csv('data/ansi_fake_data/ansi_fake_data_network_tx.csv', index_col='Time')

In [7]: #txnet_df.columns

In [8]: #view_boxplot(txnet_df)
```

1.3 Network RX

```
In [9]: rxnet_df = pandas.read_csv('data/ansi_fake_data/ansi_fake_data_network_rx.csv', index_col='Time')

In [10]: #rxnet_df.columns

In [11]: rxnet_df = rxnet_df.clip(lower=0, upper=15000)
        #view_boxplot(rxnet_df)
```

1.4 Disk IO data

```
In [12]: disk_df = pandas.read_csv('data/ansi_fake_data/ansi_fake_data_disk_io.csv', index_col=0)

In [13]: #disk_df.columns

In [14]: disk_df = disk_df.clip(lower=0, upper=4000)
#view_boxplot(disk_df)
```

1.5 Context switching

```
In [15]: context_df = pandas.read_csv('data/ansi_fake_data/ansi_fake_data_context.csv', index_col=0)

In [16]: #context_df.columns

In [17]: context_df = context_df.clip(lower=0, upper=5000)
#view_boxplot(context_df)
```

1.6 Separate into proper dataframes for each node

```
In [18]: dframes = [cpu_df, txnet_df, rxnet_df, context_df, disk_df]
node = {}

for i in range(1,5):
    frames = []

    for dframe in dframes:
        columns = list(filter(lambda x: f'bb{i}l' in x, dframe.columns))
        frames.append(dframe[columns])

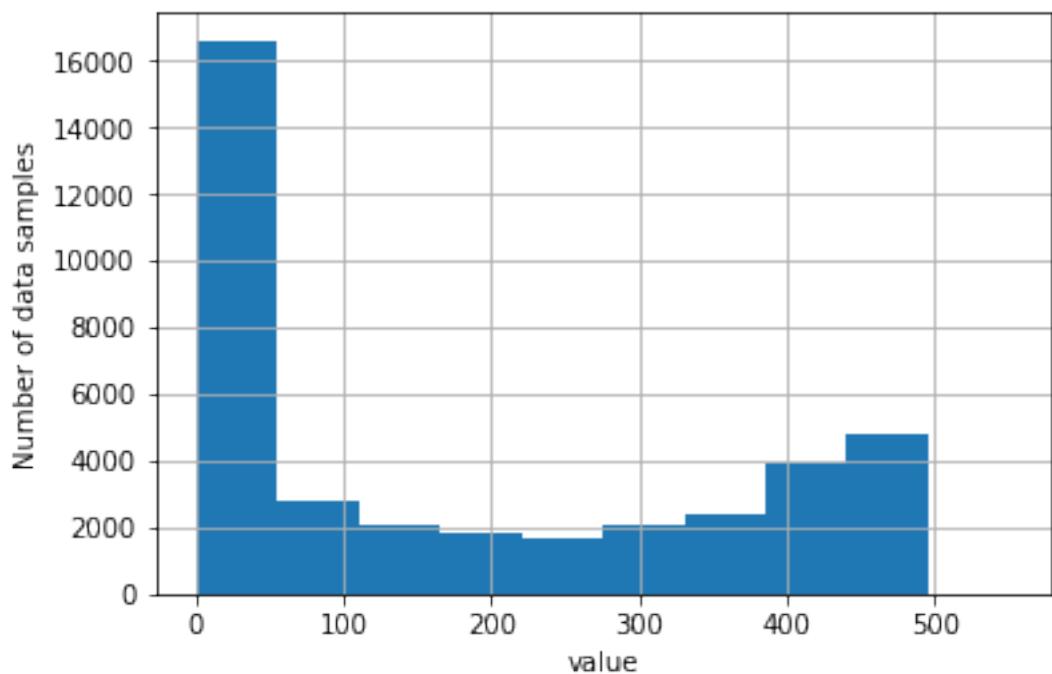
    node[i] = pandas.concat(frames, join='inner', axis=1).fillna(0)[:38200]

In [19]: for i in range(1,5):
    print(node[i].shape)
    print(node[i].columns)
    for column in node[i].columns:
        print(f'Distribution of data for {column}')
        node[i][column].hist()
        plotter.ylabel("Number of data samples")
        plotter.xlabel("value")
        plotter.savefig(f"node{i}_{column}_hist.png")
        plotter.show()
    pass

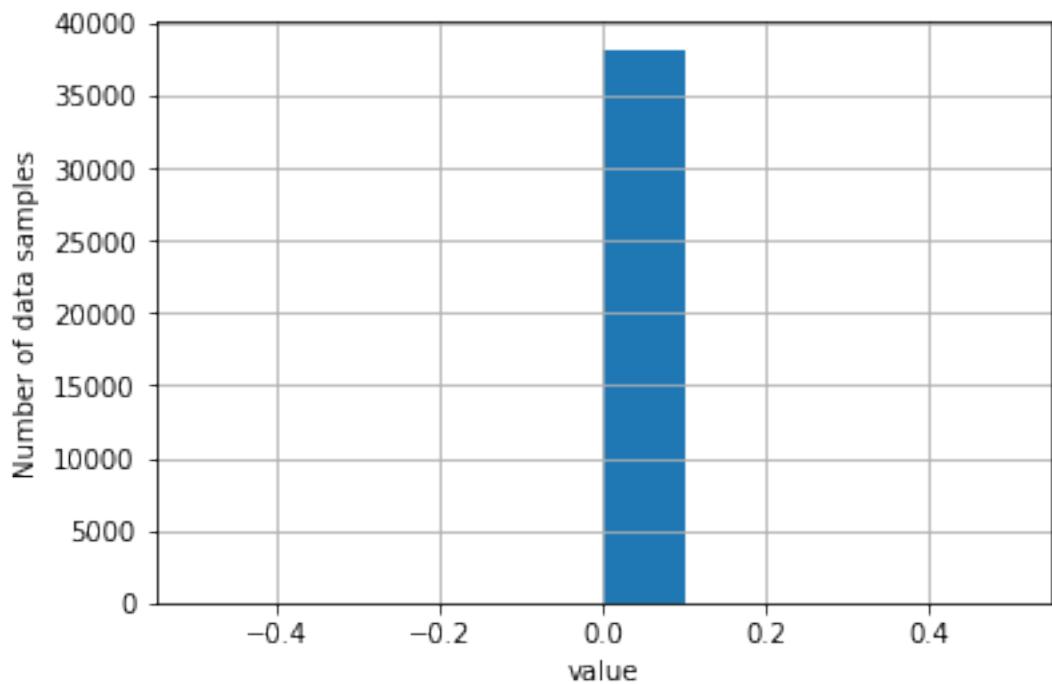
(38200, 29)
Index(['cpu_value host bb1localdomain type_instance idle',
       'cpu_value host bb1localdomain type_instance interrupt',
       'cpu_value host bb1localdomain type_instance nice',
       'cpu_value host bb1localdomain type_instance softirq',
```

```
'cpu_value host bb1localdomain type_instance steal',
'cpu_value host bb1localdomain type_instance system',
'cpu_value host bb1localdomain type_instance user',
'cpu_value host bb1localdomain type_instance wait',
'interface_tx host bb1localdomain instance lo type if_dropped',
'interface_tx host bb1localdomain instance lo type if_errors',
'interface_tx host bb1localdomain instance lo type if_octets',
'interface_tx host bb1localdomain instance lo type if_packets',
'interface_tx host bb1localdomain instance wlan0 type if_dropped',
'interface_tx host bb1localdomain instance wlan0 type if_errors',
'interface_tx host bb1localdomain instance wlan0 type if_octets',
'interface_tx host bb1localdomain instance wlan0 type if_packets',
'interface_rx host bb1localdomain instance lo type if_dropped',
'interface_rx host bb1localdomain instance lo type if_errors',
'interface_rx host bb1localdomain instance lo type if_octets',
'interface_rx host bb1localdomain instance lo type if_packets',
'interface_rx host bb1localdomain instance wlan0 type if_dropped',
'interface_rx host bb1localdomain instance wlan0 type if_errors',
'interface_rx host bb1localdomain instance wlan0 type if_octets',
'interface_rx host bb1localdomain instance wlan0 type if_packets',
'contextswitch_value host bb1localdomain type contextswitch',
'disk_io_time host bb1localdomain instance mmcblk1 type disk_io_time',
'disk_io_time host bb1localdomain instance mmcblk1boot0 type disk_io_time',
'disk_io_time host bb1localdomain instance mmcblk1boot1 type disk_io_time',
'disk_io_time host bb1localdomain instance mmcblk1p1 type disk_io_time'],
dtype='object')
```

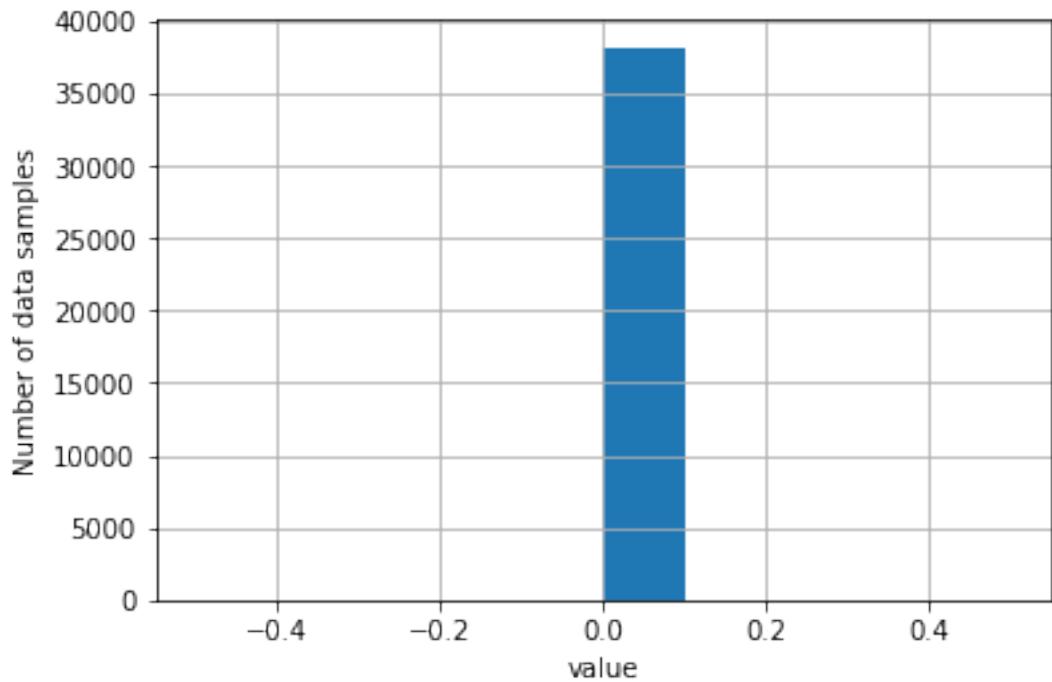
Distribution of data for `cpu_value host bb1localdomain type_instance idle`



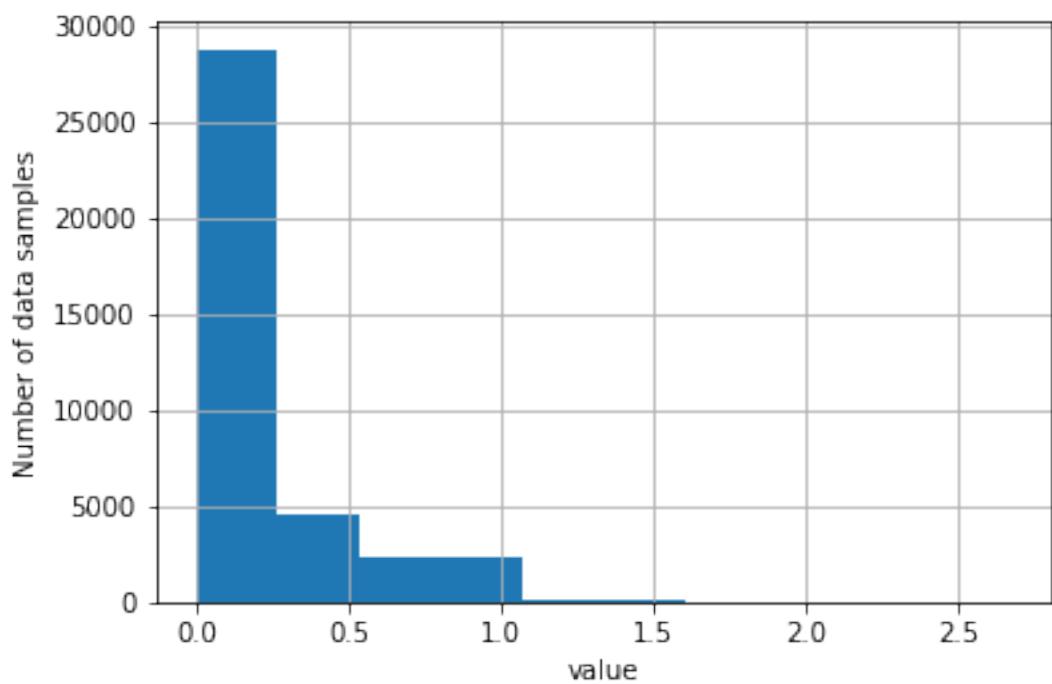
Distribution of data for `cpu_value` host `bb1localdomain` type_`instance` interrupt



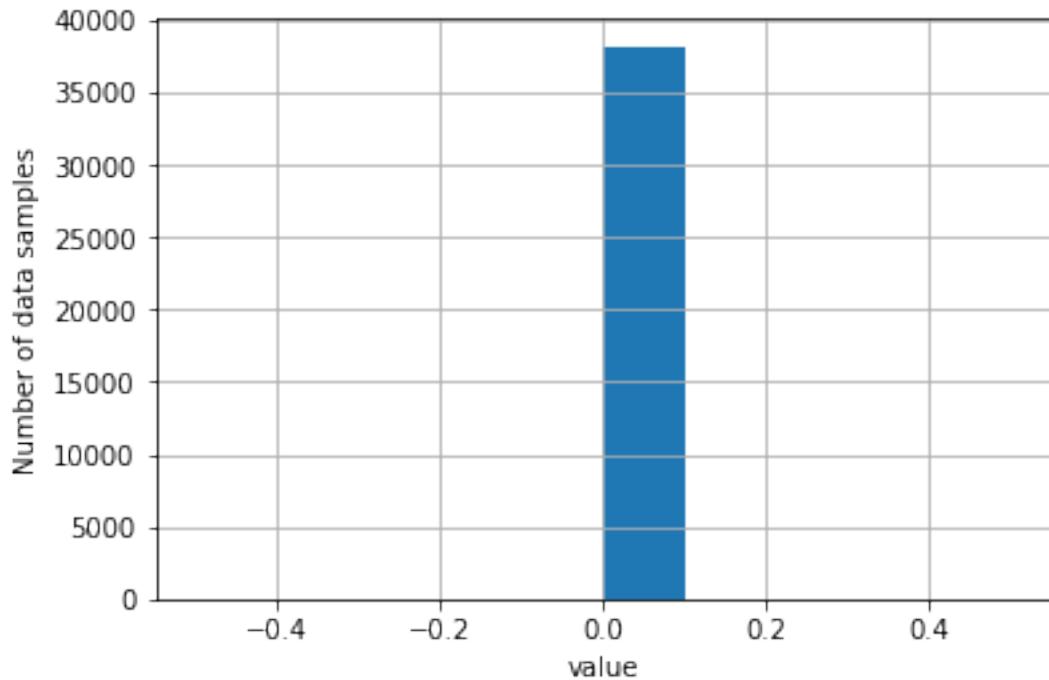
Distribution of data for cpu_value host bb1localdomain type_instance nice



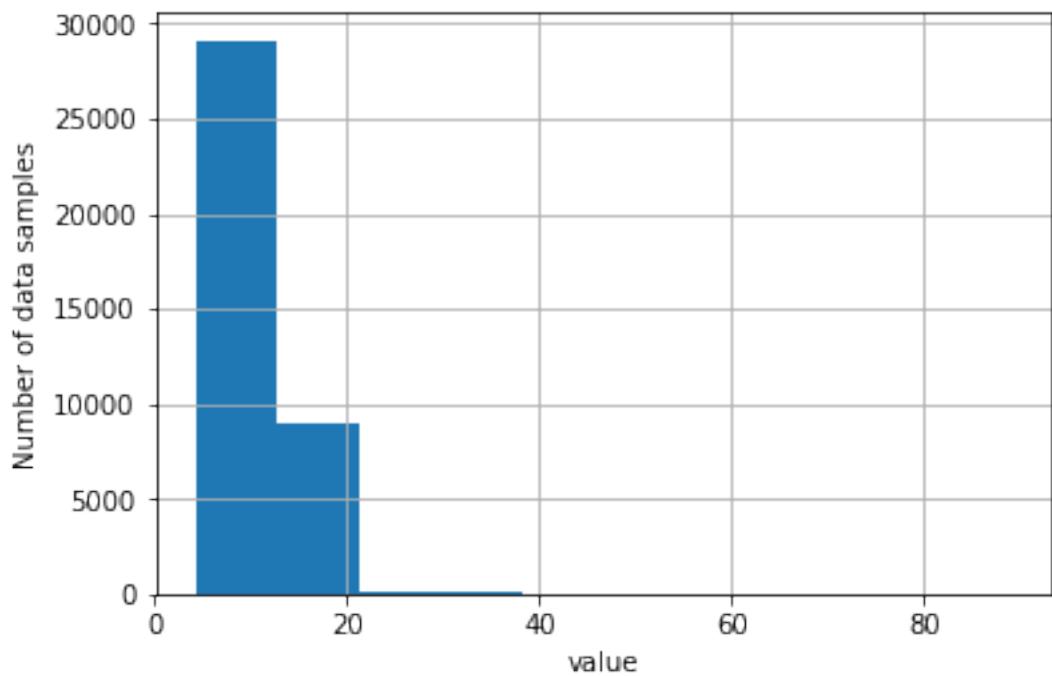
Distribution of data for cpu_value host bb1localdomain type_instance softirq



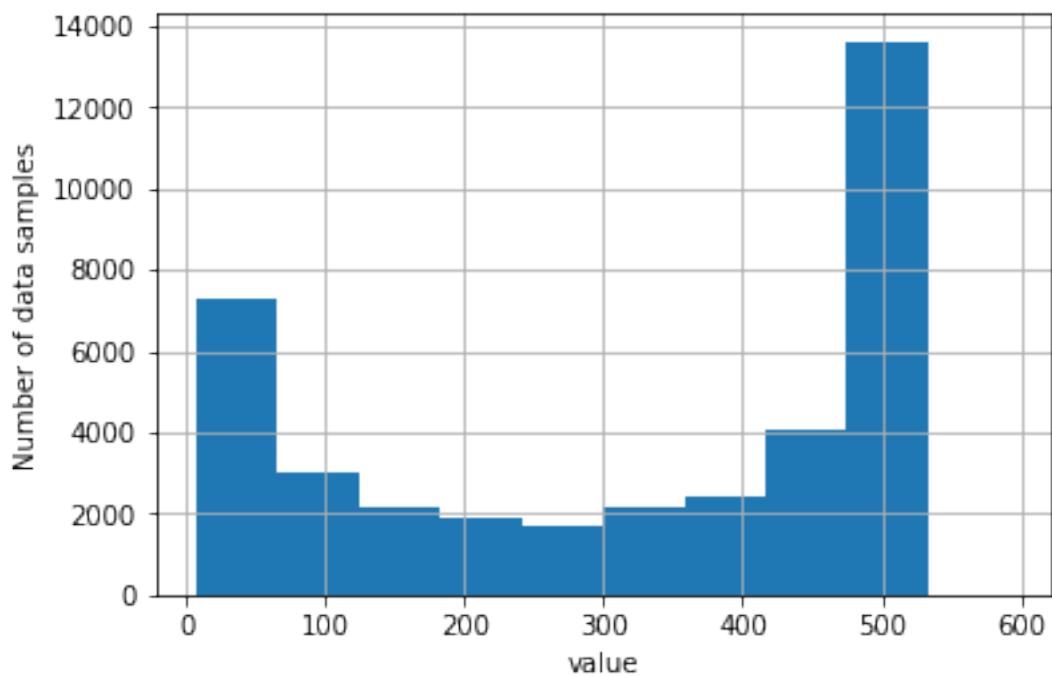
Distribution of data for cpu_value host bb1localdomain type_instance steal



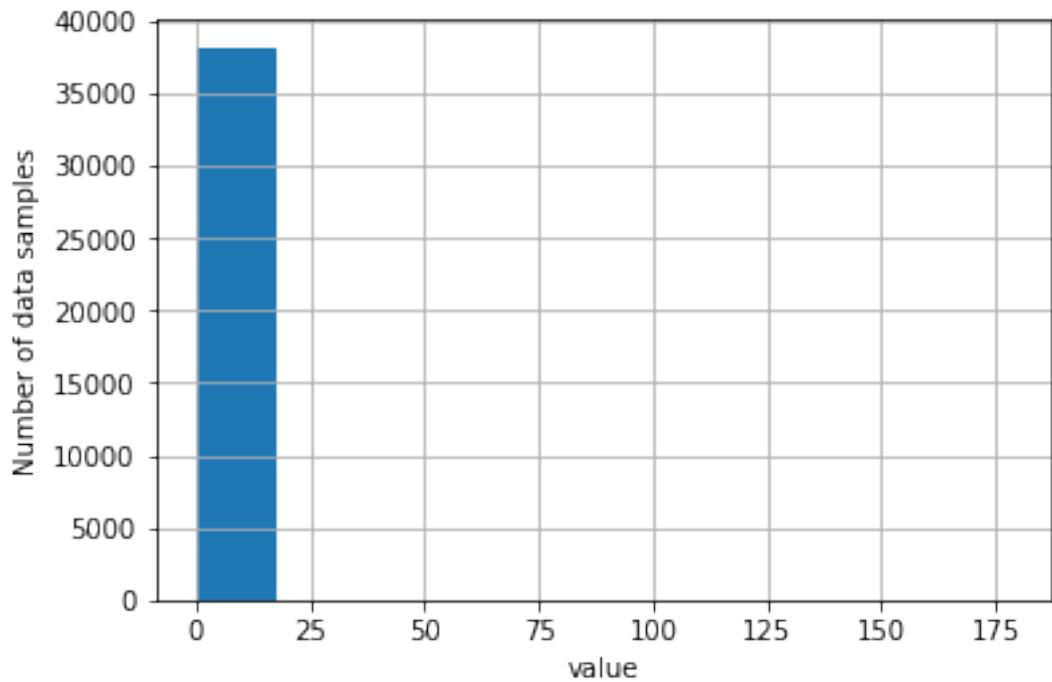
Distribution of data for cpu_value host bb1localdomain type_instance system



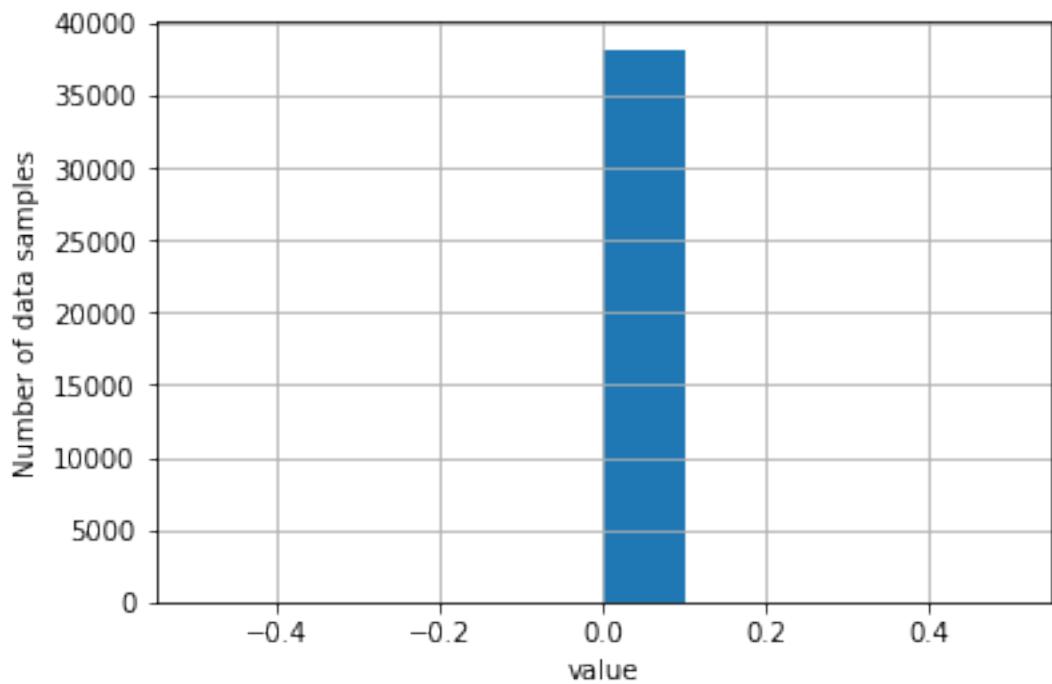
Distribution of data for cpu_value host bb1localdomain type_instance user



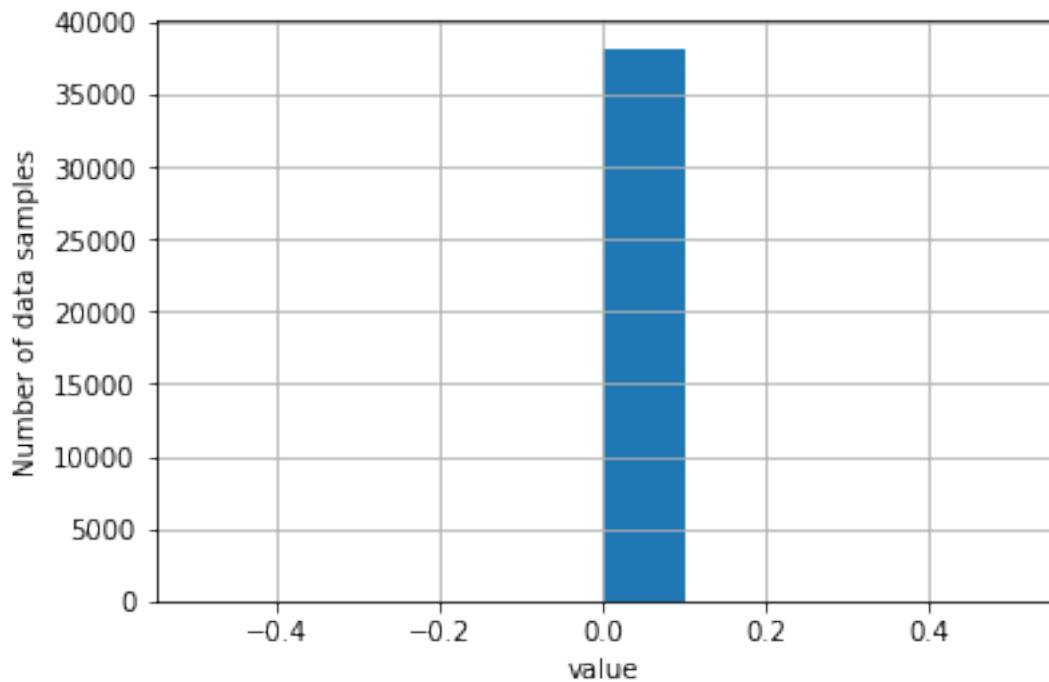
Distribution of data for cpu_value host bb1localdomain type_instance wait



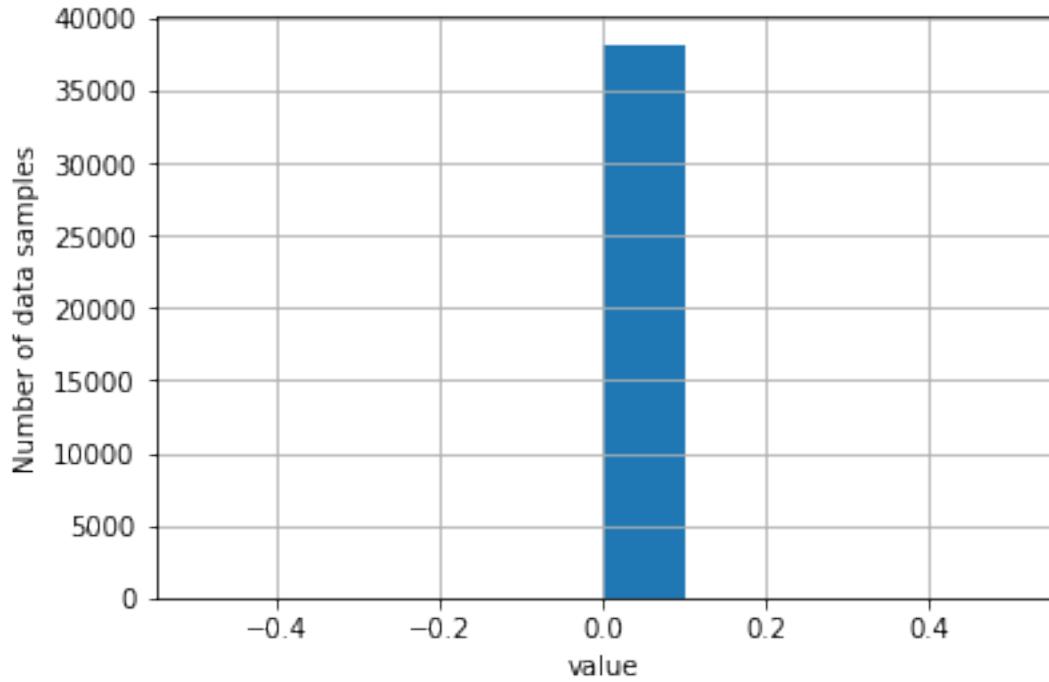
Distribution of data for interface_tx host bb1localdomain instance lo type if_dropped



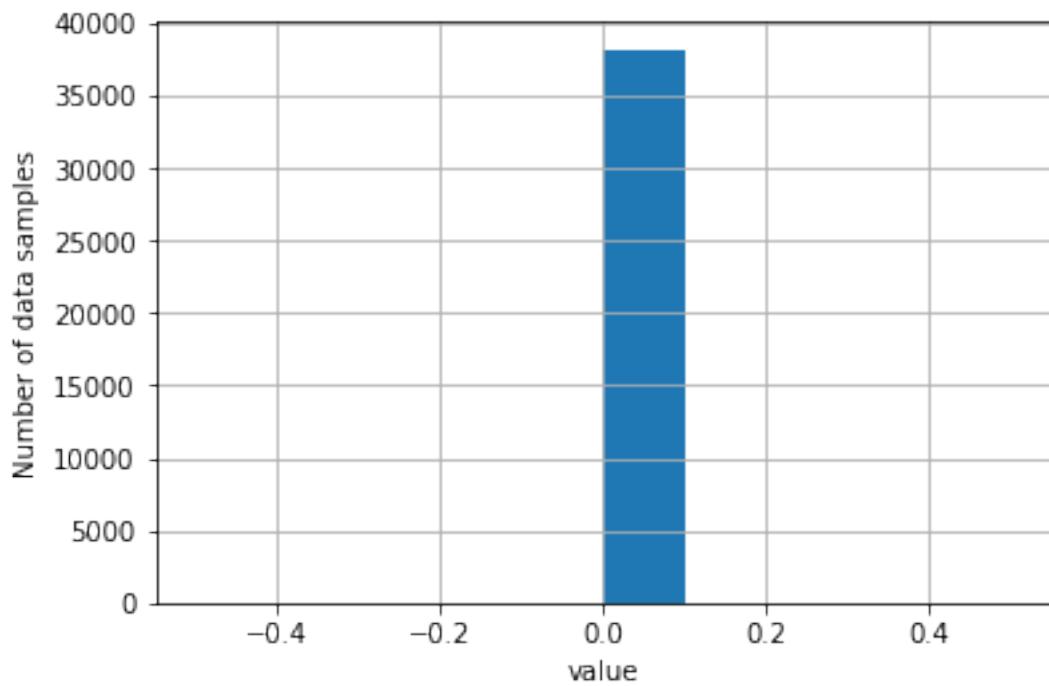
Distribution of data for interface_tx host bb1localdomain instance lo type if_errors



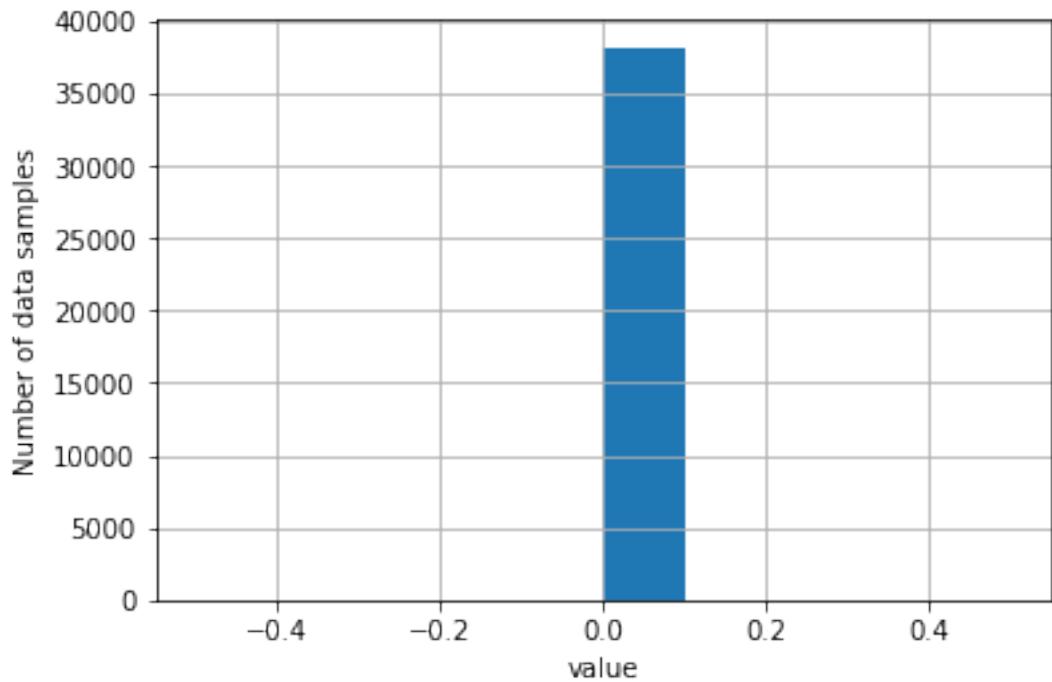
Distribution of data for interface_tx host bb1localdomain instance lo type if_octets



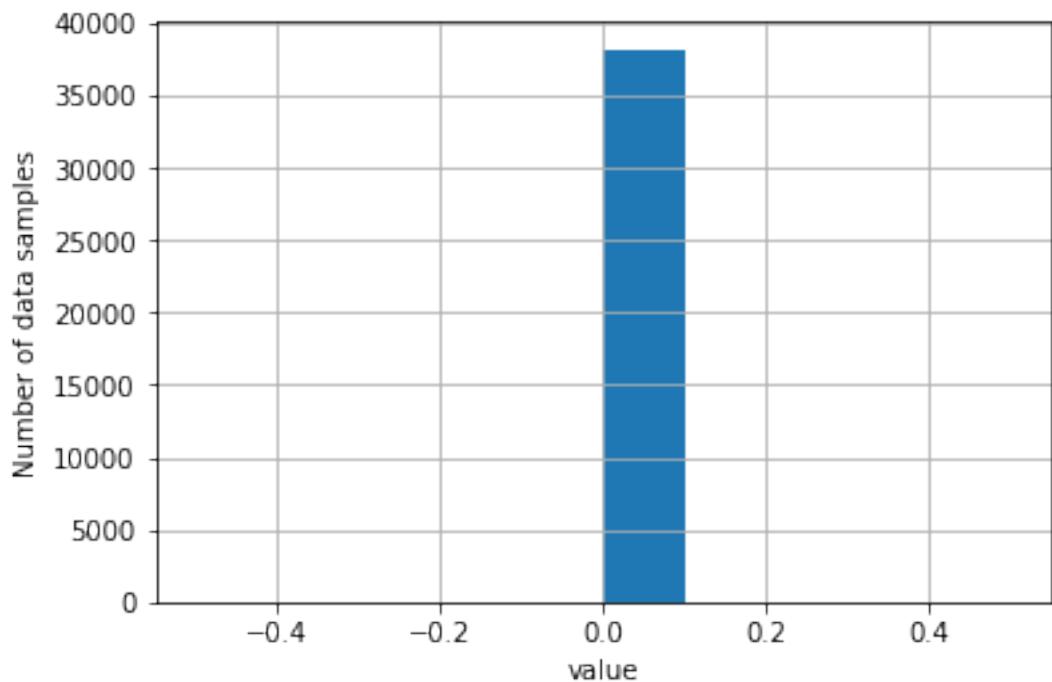
Distribution of data for interface_tx host bb1localdomain instance lo type if_packets



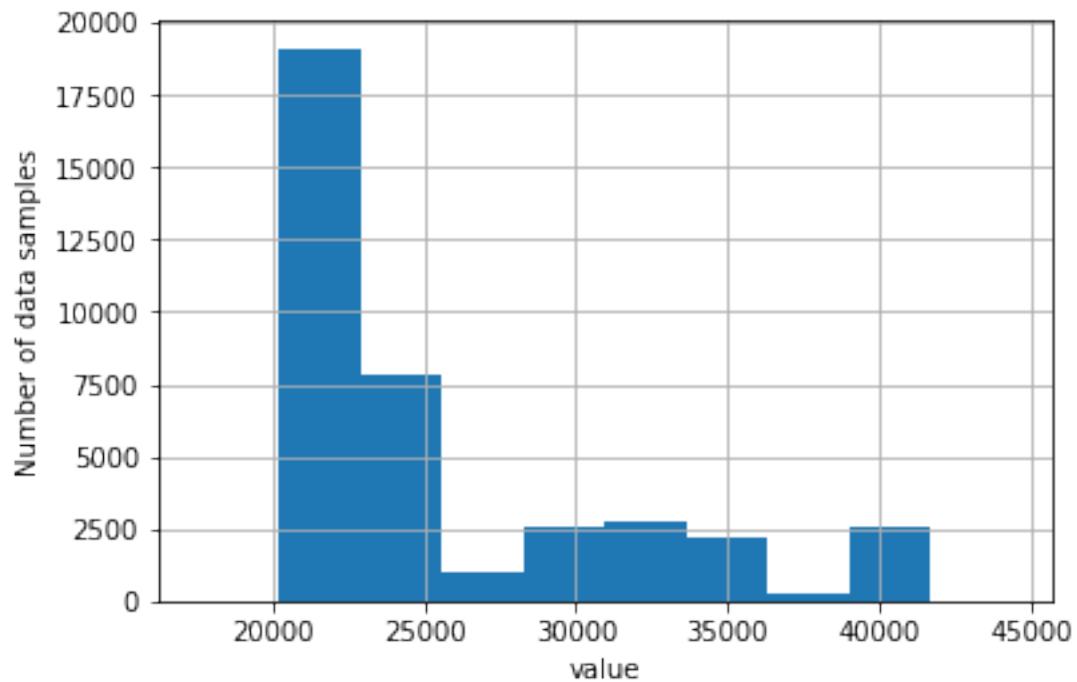
Distribution of data for interface_tx host bbilocaldomain instance wlan0 type if_dropped



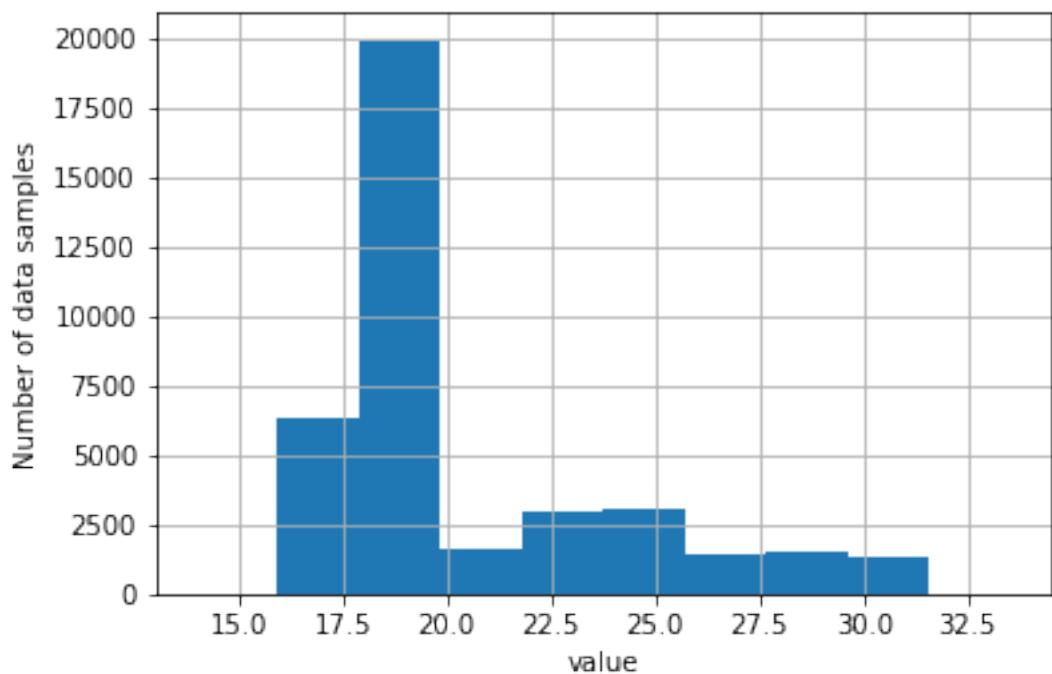
Distribution of data for interface_tx host bbilocaldomain instance wlan0 type if_errors



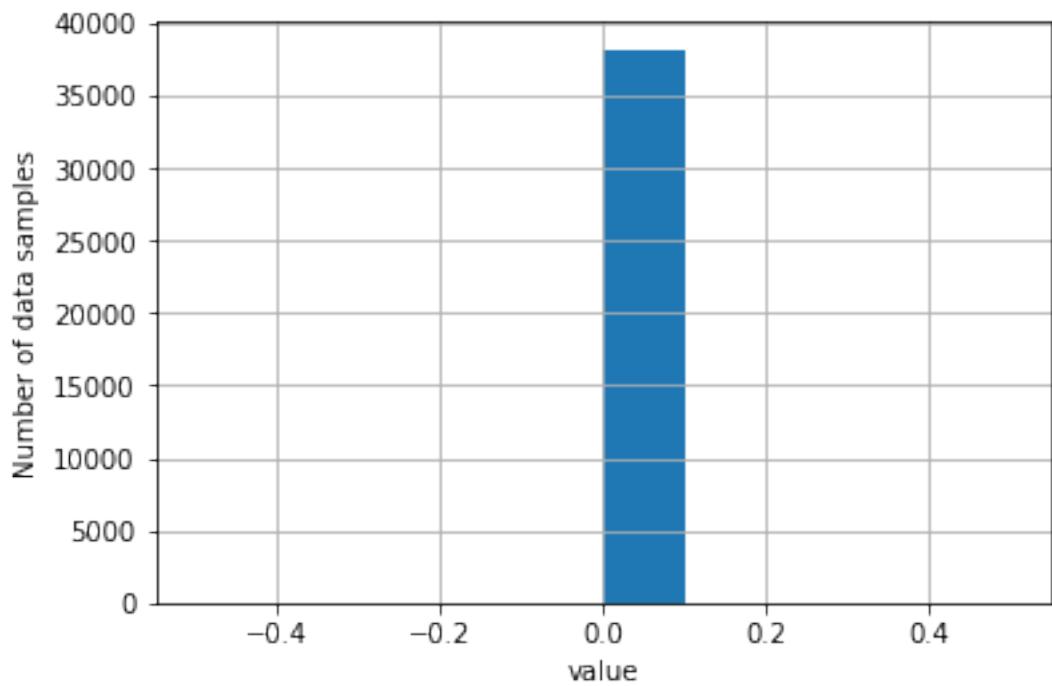
Distribution of data for interface_tx host bb1localdomain instance wlan0 type if_octets



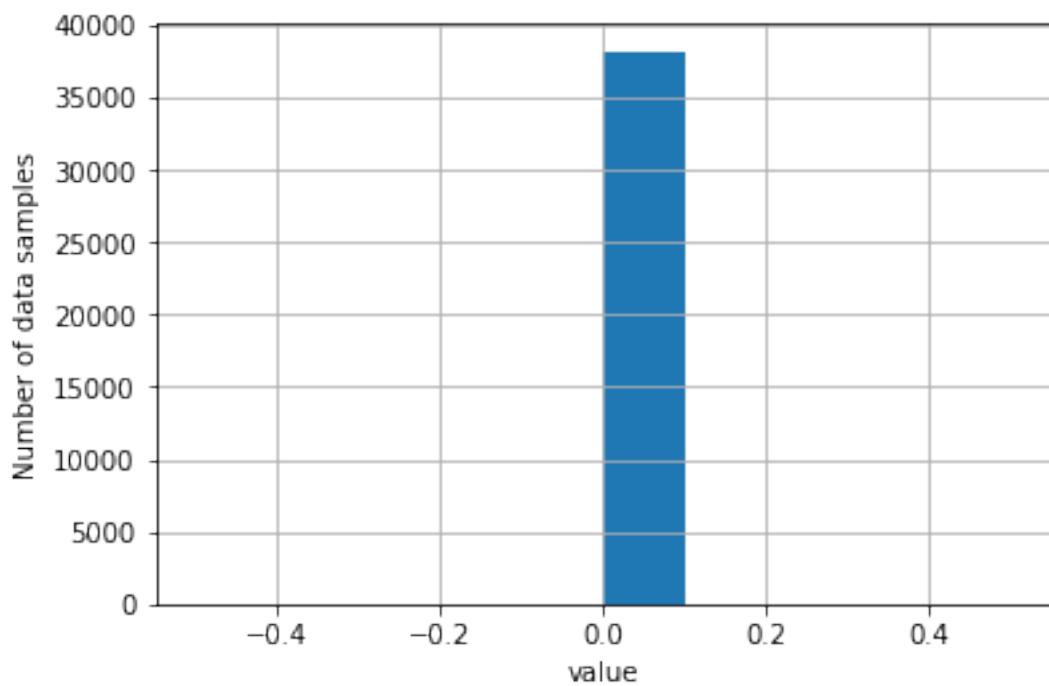
Distribution of data for interface_tx host bb1localdomain instance wlan0 type if_packets



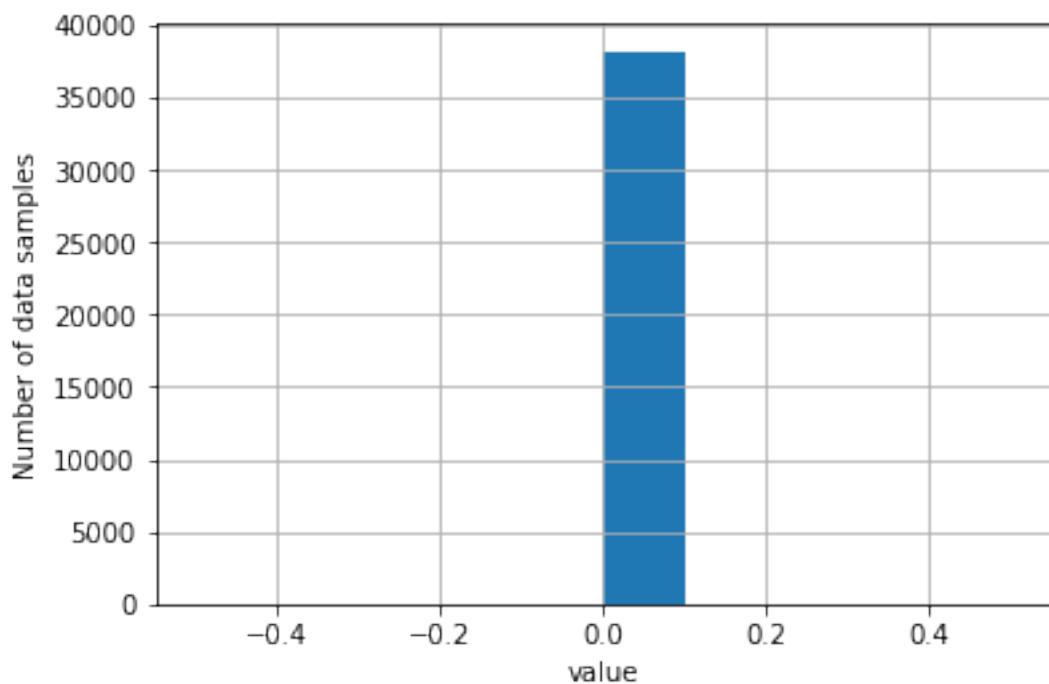
Distribution of data for interface_rx host bb1localdomain instance lo type if_dropped



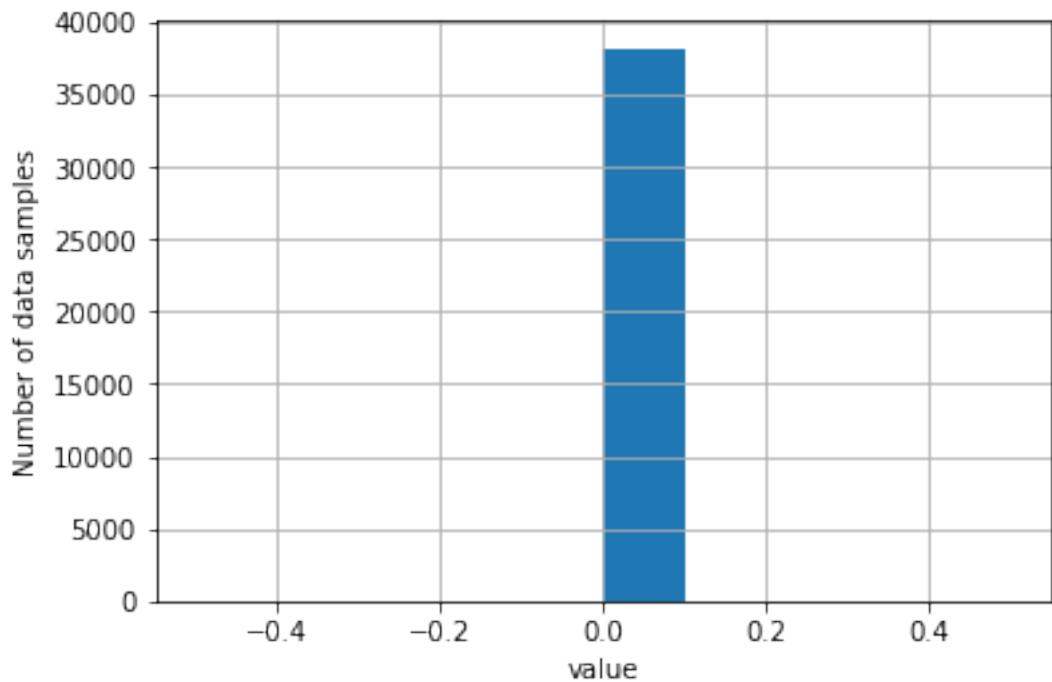
Distribution of data for interface_rx host bbilocaldomain instance lo type if_errors



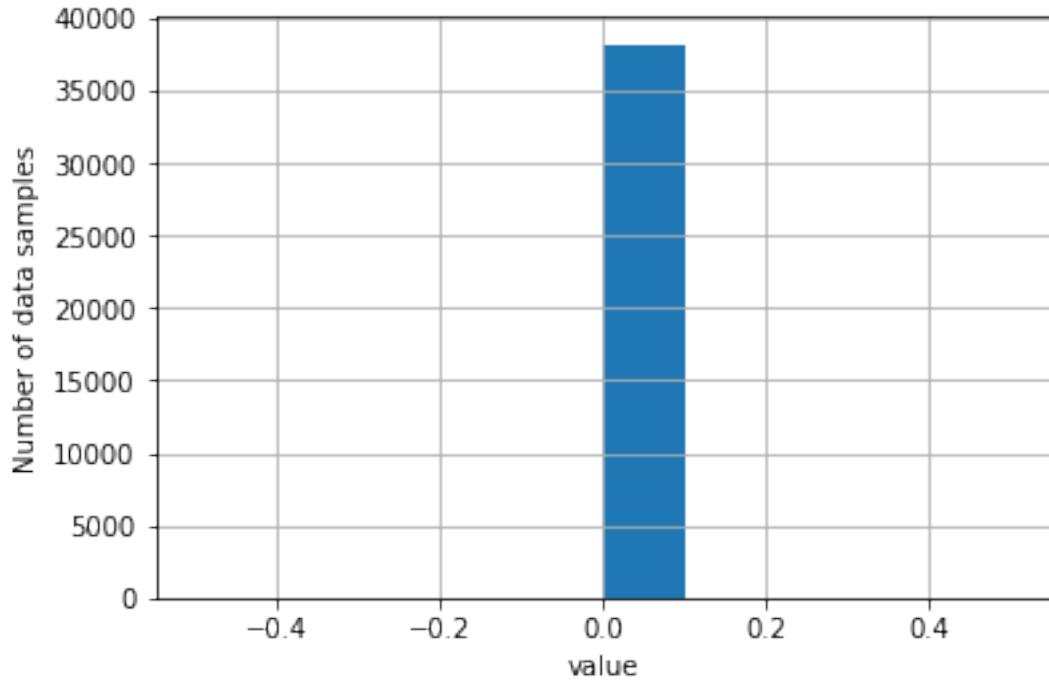
Distribution of data for interface_rx host bbilocaldomain instance lo type if_octets



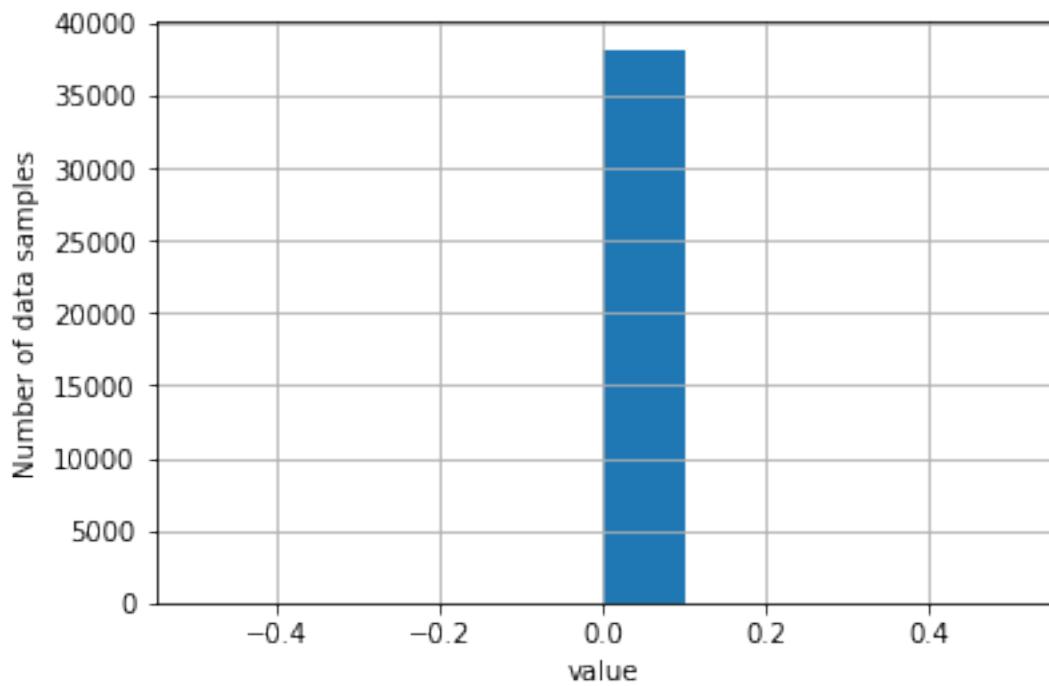
Distribution of data for interface_rx host bb1localdomain instance lo type if_packets



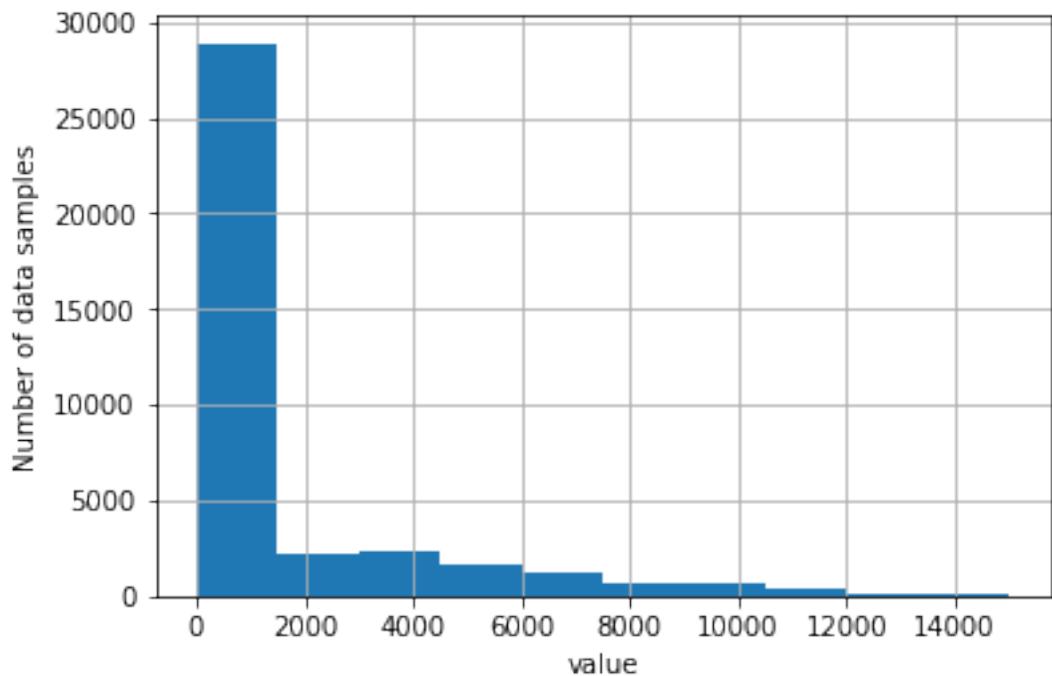
Distribution of data for interface_rx host bb1localdomain instance wlan0 type if_dropped



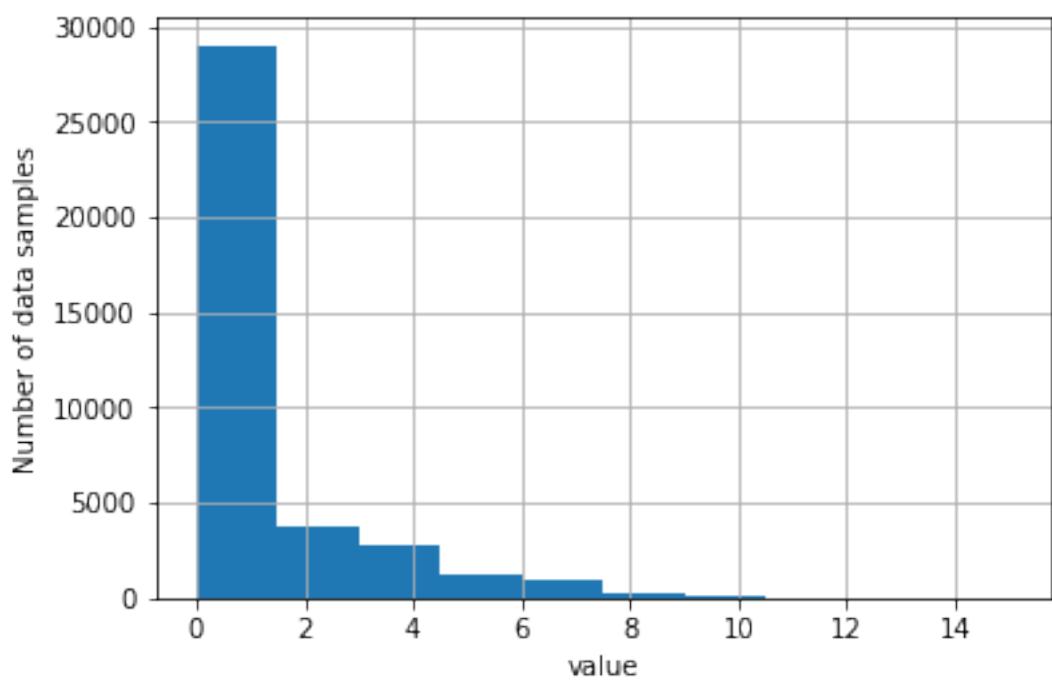
Distribution of data for interface_rx host bb1localdomain instance wlan0 type if_errors



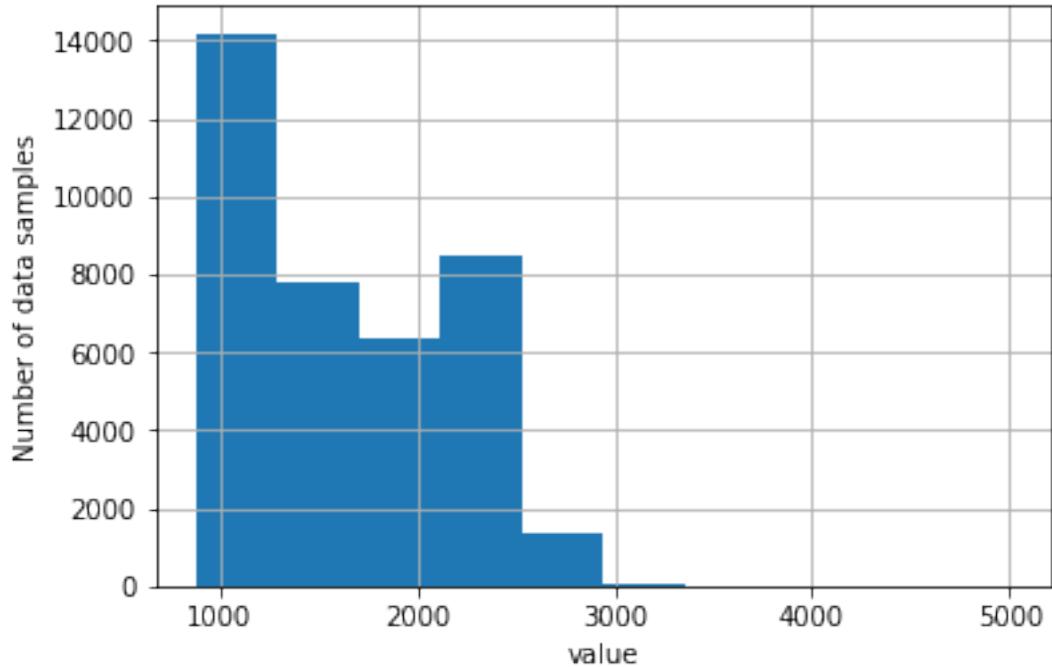
Distribution of data for interface_rx host bb1localdomain instance wlan0 type if_octets



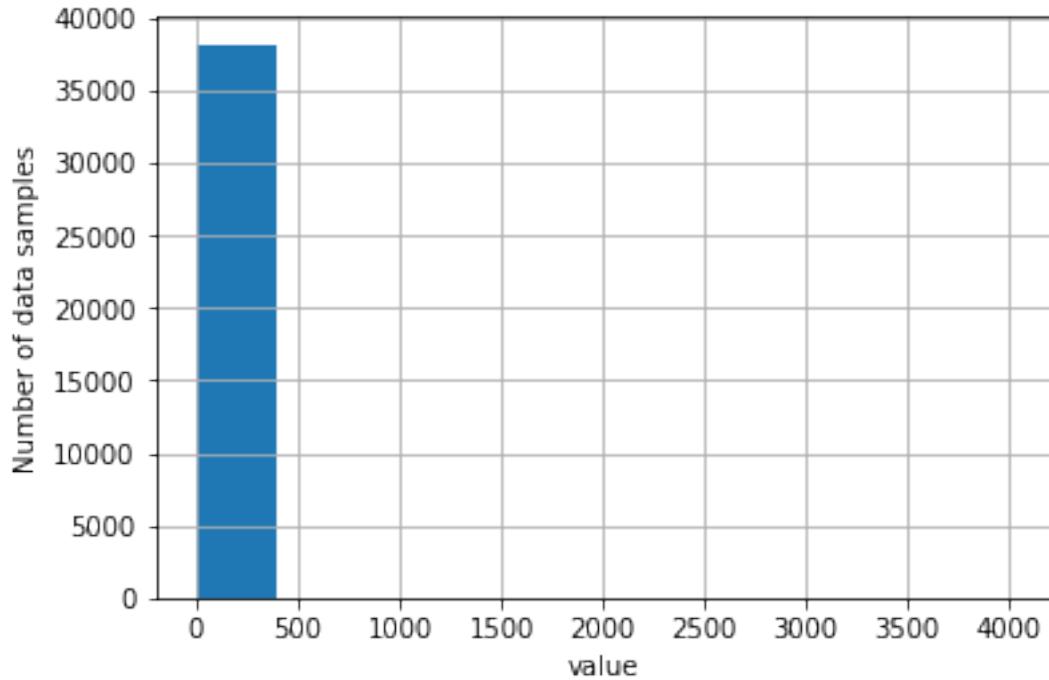
Distribution of data for interface_rx host bb1localdomain instance wlan0 type if_packets



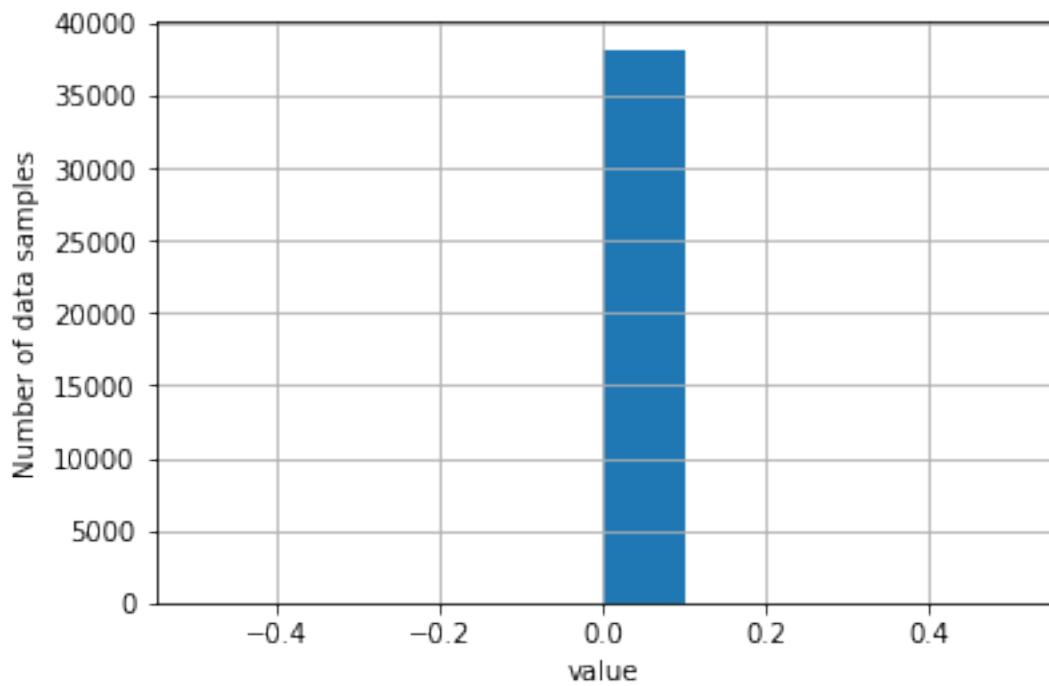
Distribution of data for contextswitch_value host bb1localdomain type contextswitch



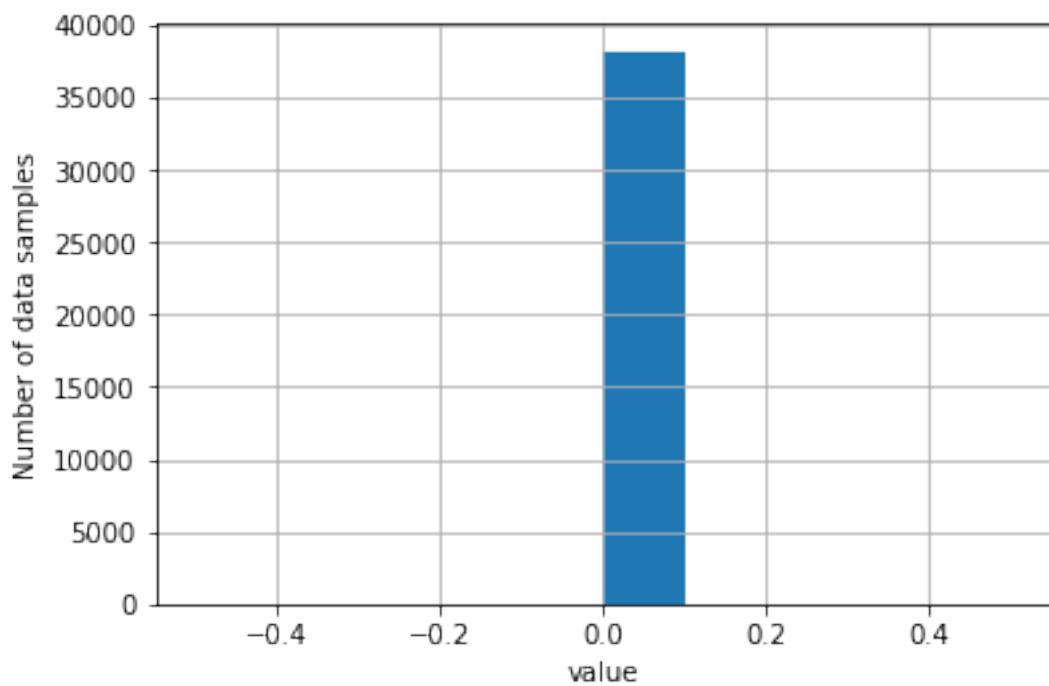
Distribution of data for disk_io_time host bb1localdomain instance mmcblk1 type disk_io_time



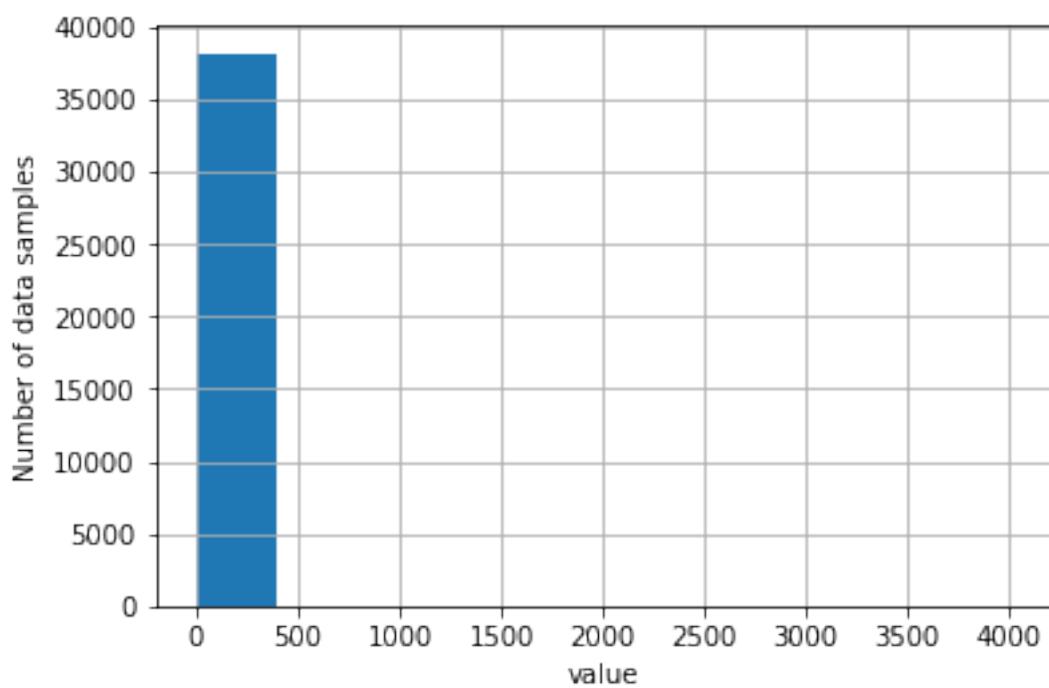
Distribution of data for disk_io_time host bb1localdomain instance mmcblk1boot0 type disk_io_t



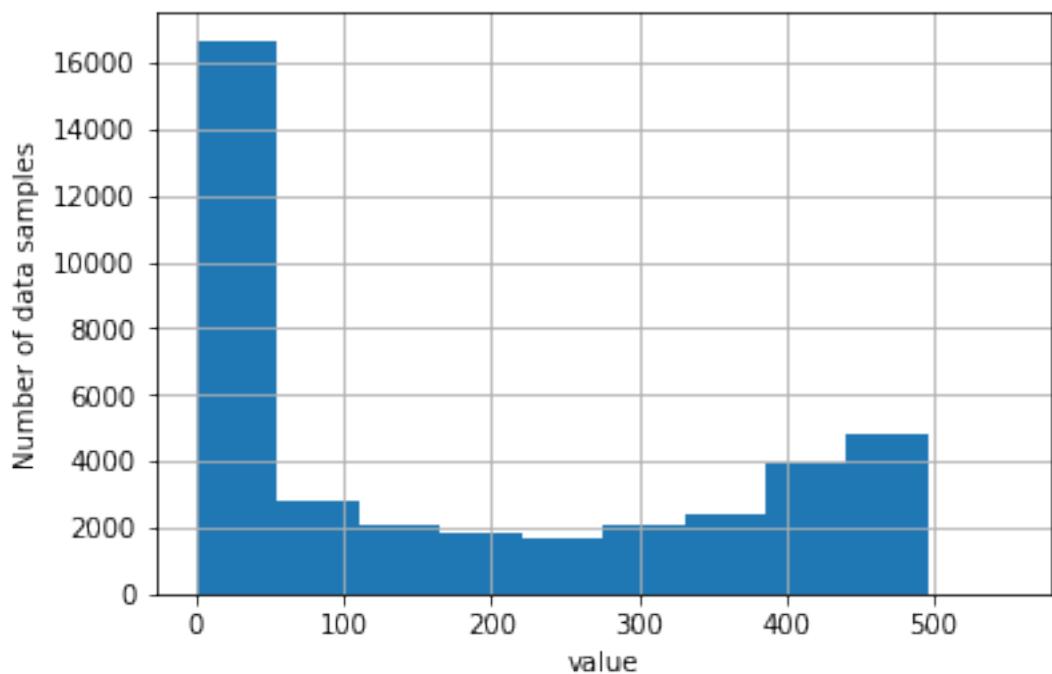
Distribution of data for disk_io_time host bb1localdomain instance mmcblk1boot1 type disk_io_time



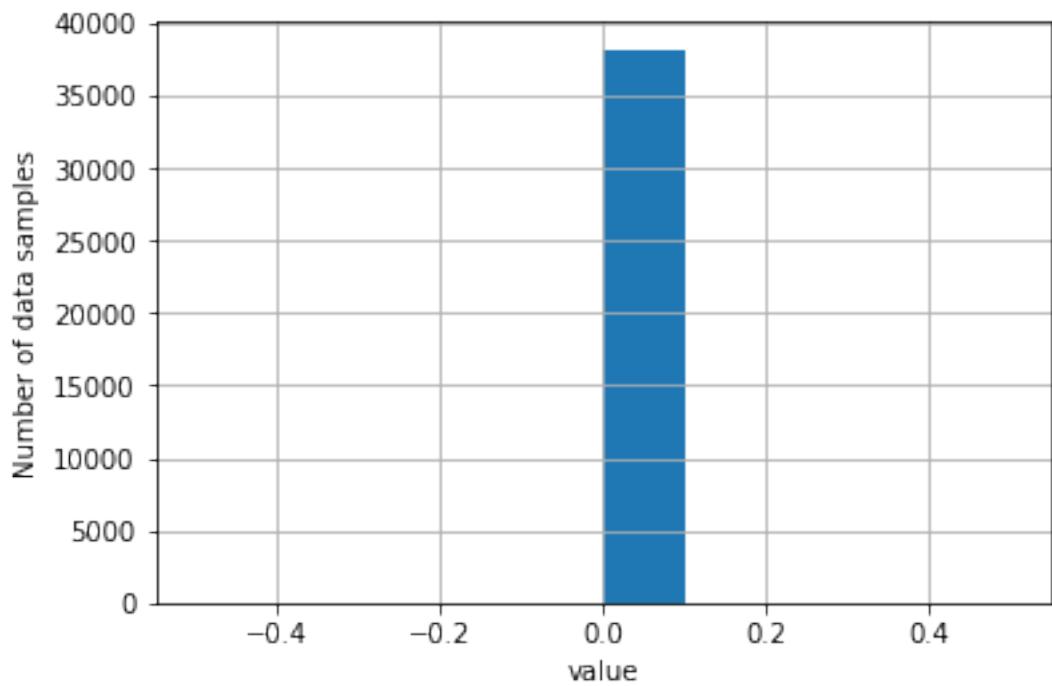
Distribution of data for disk_io_time host bb1localdomain instance mmcblk1p1 type disk_io_time



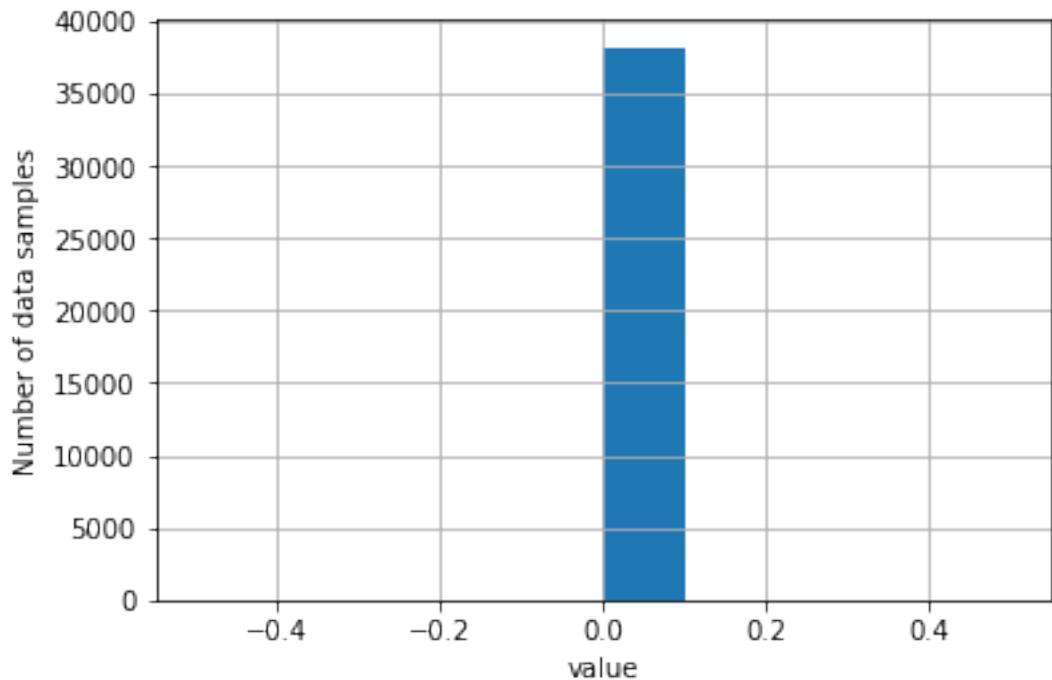
```
(38200, 29)
Index(['cpu_value host bb2localdomain type_instance idle',
       'cpu_value host bb2localdomain type_instance interrupt',
       'cpu_value host bb2localdomain type_instance nice',
       'cpu_value host bb2localdomain type_instance softirq',
       'cpu_value host bb2localdomain type_instance steal',
       'cpu_value host bb2localdomain type_instance system',
       'cpu_value host bb2localdomain type_instance user',
       'cpu_value host bb2localdomain type_instance wait',
       'interface_tx host bb2localdomain instance lo type if_dropped',
       'interface_tx host bb2localdomain instance lo type if_errors',
       'interface_tx host bb2localdomain instance lo type if_octets',
       'interface_tx host bb2localdomain instance lo type if_packets',
       'interface_tx host bb2localdomain instance wlan0 type if_dropped',
       'interface_tx host bb2localdomain instance wlan0 type if_errors',
       'interface_tx host bb2localdomain instance wlan0 type if_octets',
       'interface_tx host bb2localdomain instance wlan0 type if_packets',
       'interface_rx host bb2localdomain instance lo type if_dropped',
       'interface_rx host bb2localdomain instance lo type if_errors',
       'interface_rx host bb2localdomain instance lo type if_octets',
       'interface_rx host bb2localdomain instance lo type if_packets',
       'interface_rx host bb2localdomain instance wlan0 type if_dropped',
       'interface_rx host bb2localdomain instance wlan0 type if_errors',
       'interface_rx host bb2localdomain instance wlan0 type if_octets',
       'interface_rx host bb2localdomain instance wlan0 type if_packets',
       'contextswitch_value host bb2localdomain type contextswitch',
       'disk_io_time host bb2localdomain instance mmcblk1 type disk_io_time',
       'disk_io_time host bb2localdomain instance mmcblk1boot0 type disk_io_time',
       'disk_io_time host bb2localdomain instance mmcblk1boot1 type disk_io_time',
       'disk_io_time host bb2localdomain instance mmcblk1p1 type disk_io_time'],
      dtype='object')
Distribution of data for cpu_value host bb2localdomain type_instance idle
```



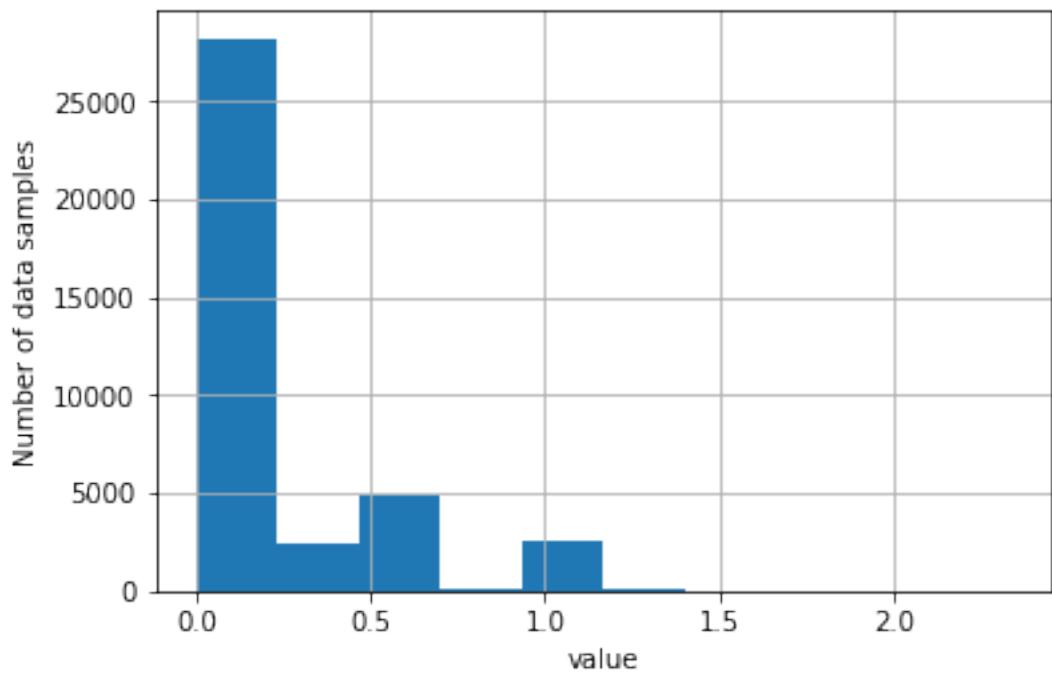
Distribution of data for `cpu_value host bb2localdomain type_instance interrupt`



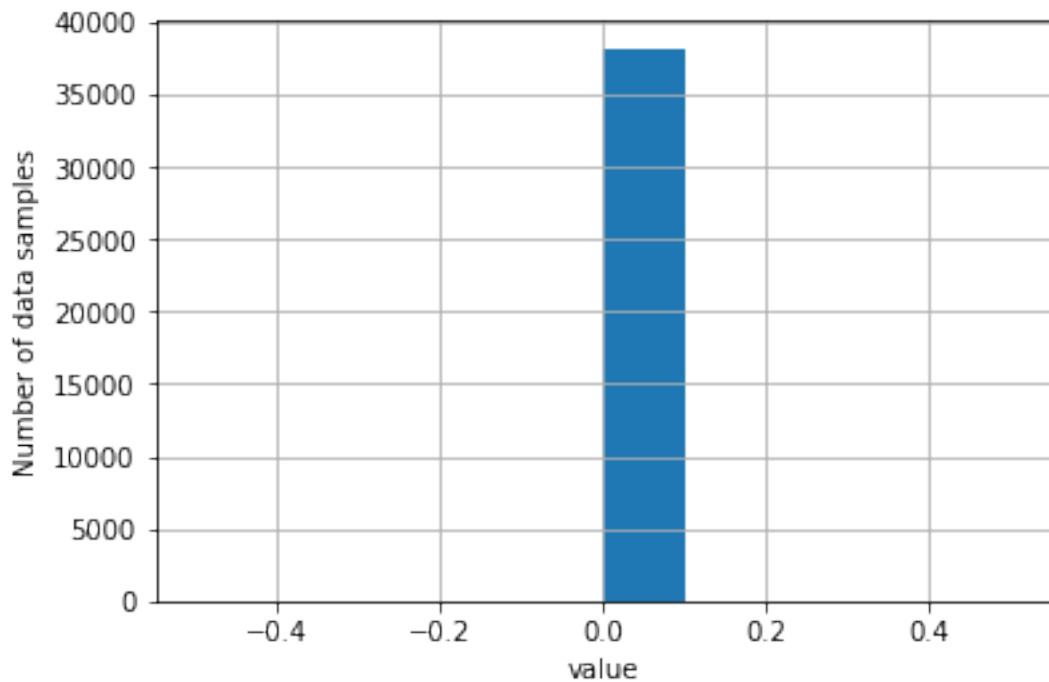
Distribution of data for cpu_value host bb2localdomain type_instance nice



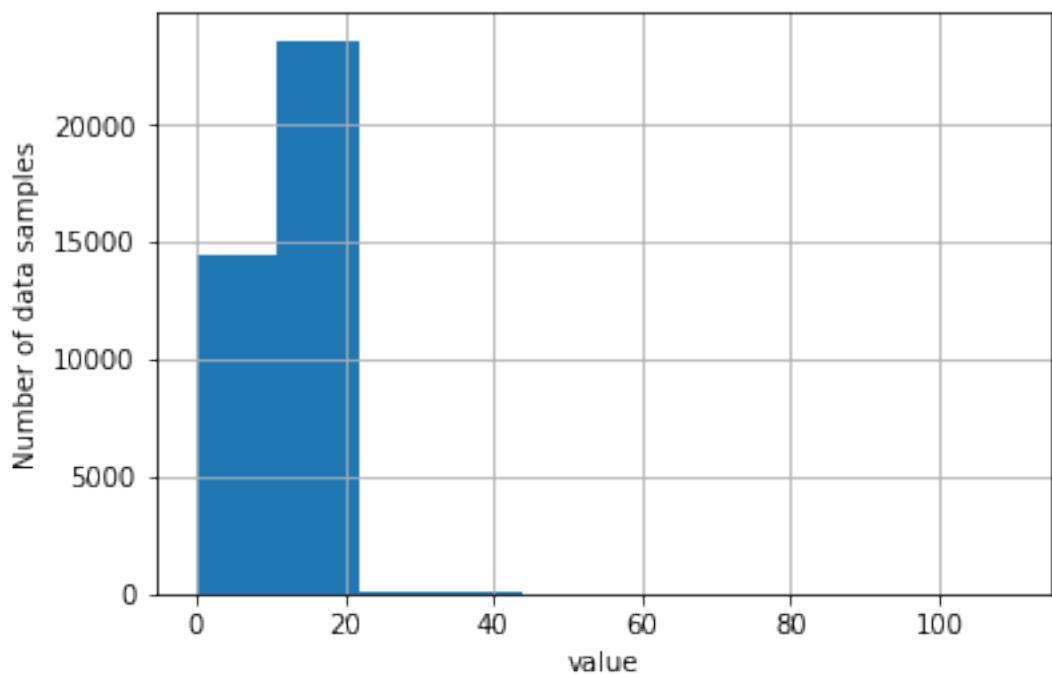
Distribution of data for cpu_value host bb2localdomain type_instance softirq



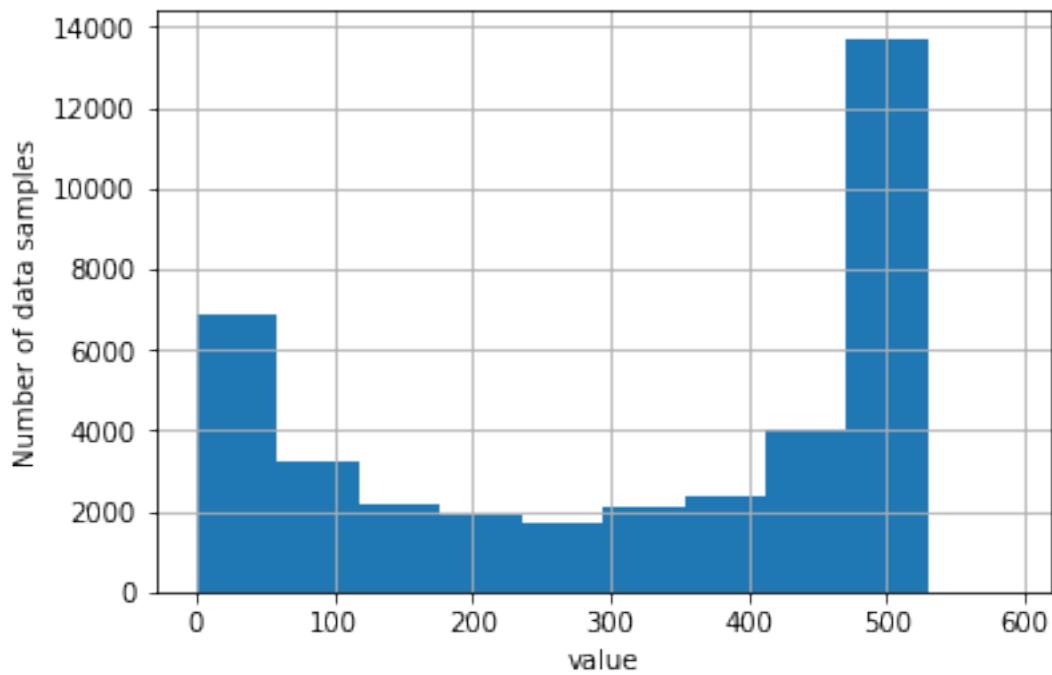
Distribution of data for cpu_value host bb2localdomain type_instance steal



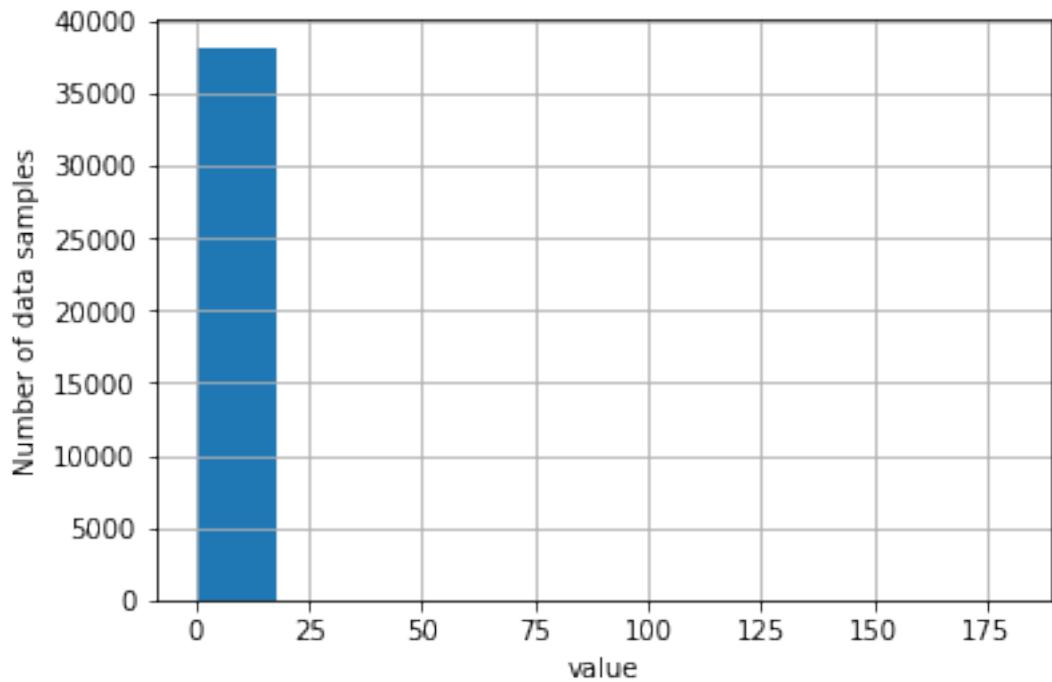
Distribution of data for cpu_value host bb2localdomain type_instance system



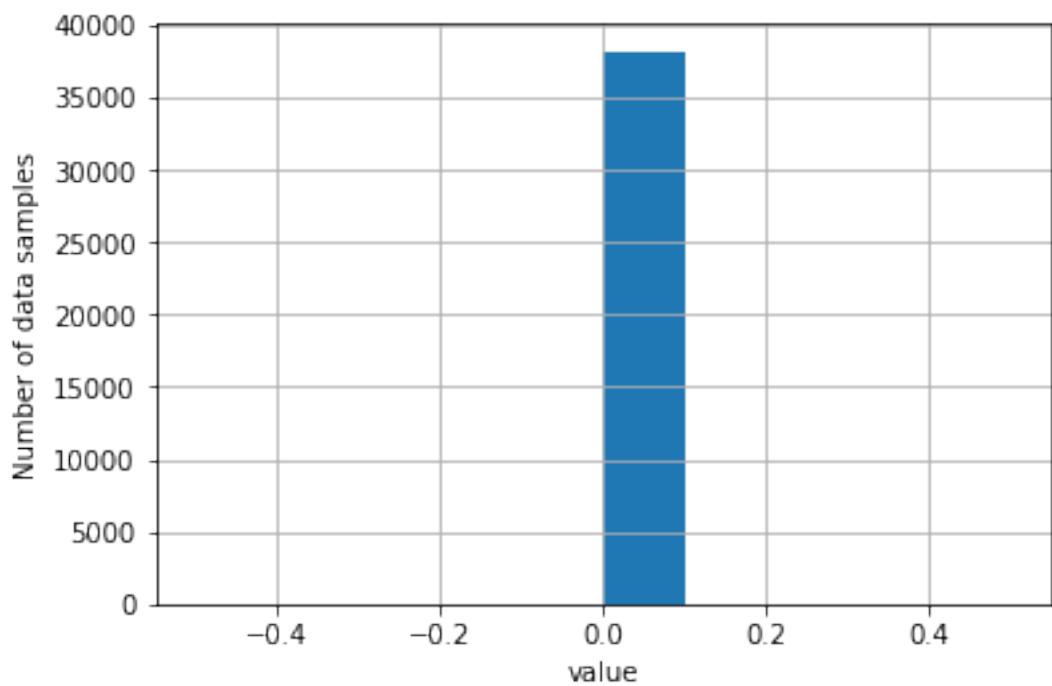
Distribution of data for cpu_value host bb2localdomain type_instance user



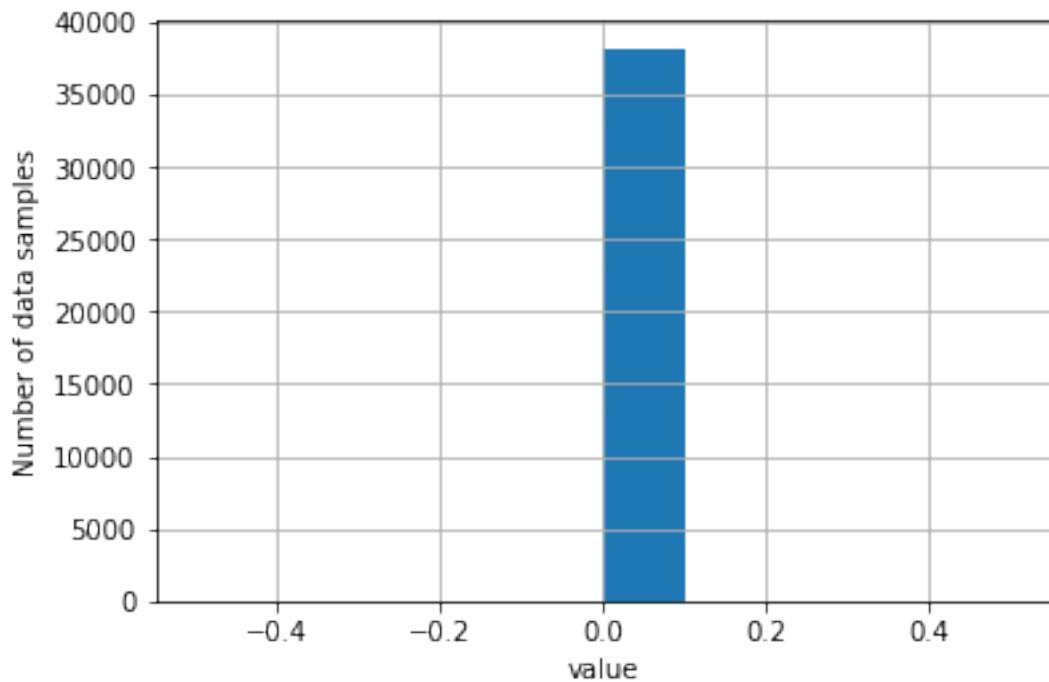
Distribution of data for cpu_value host bb2localdomain type_instance wait



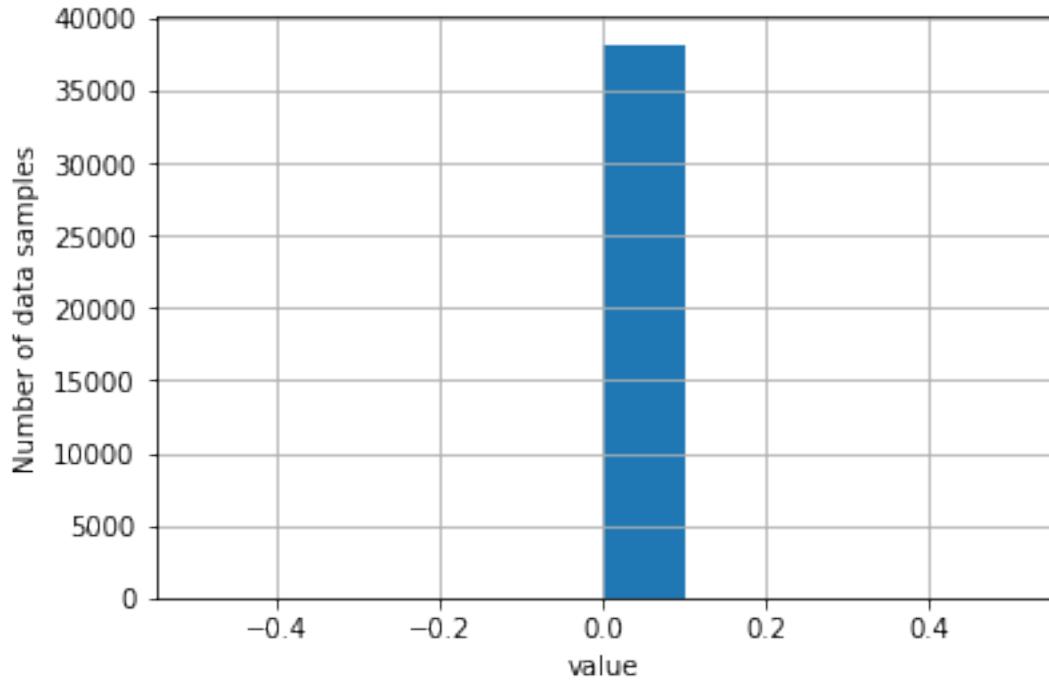
Distribution of data for interface_tx host bb2localdomain instance lo type if_dropped



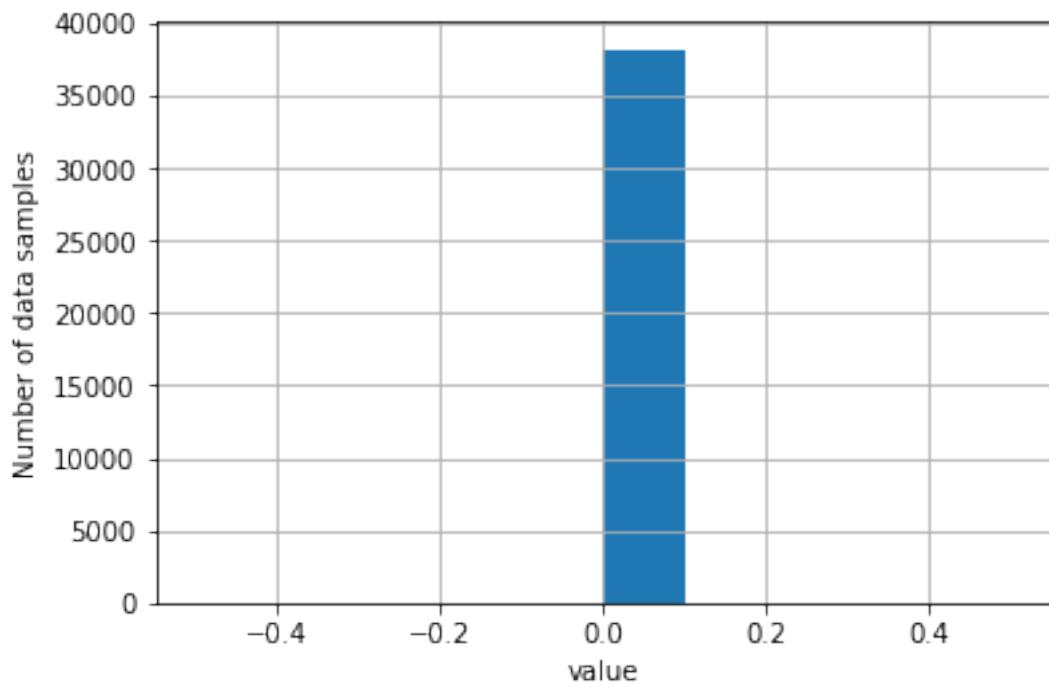
Distribution of data for interface_tx host bb2localdomain instance lo type if_errors



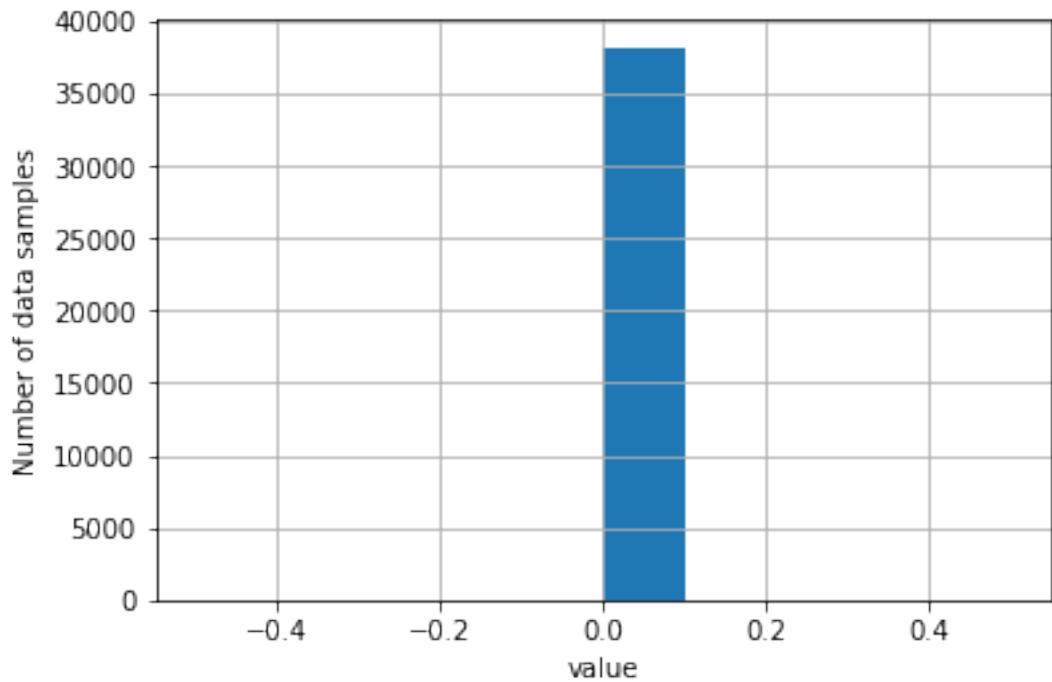
Distribution of data for interface_tx host bb2localdomain instance lo type if_octets



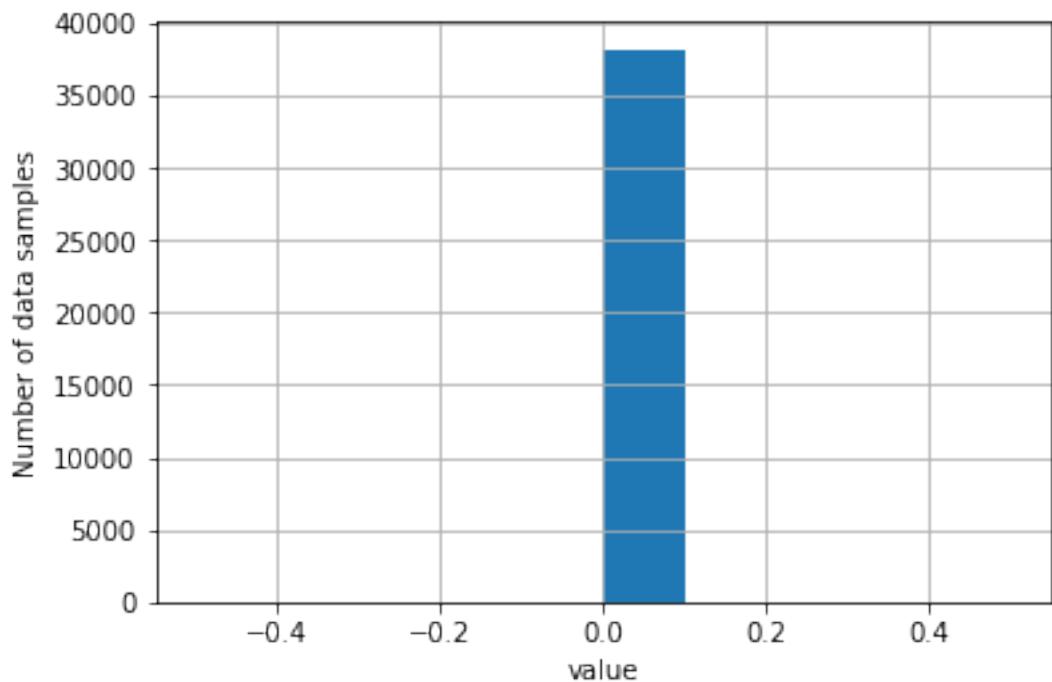
Distribution of data for interface_tx host bb2localdomain instance lo type if_packets



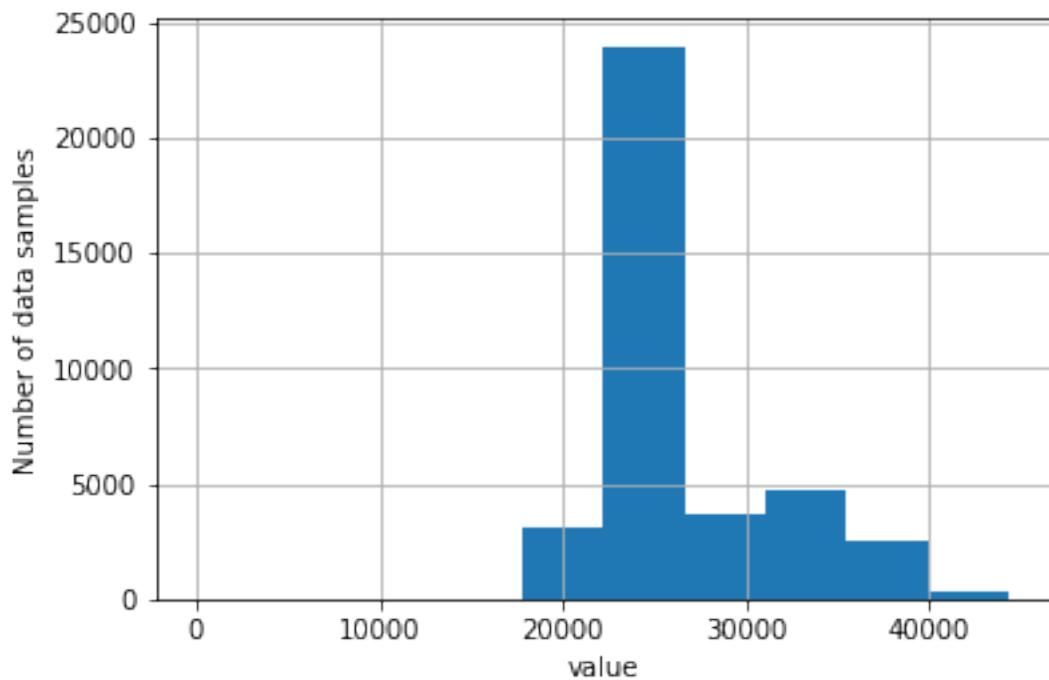
Distribution of data for interface_tx host bb2localdomain instance wlan0 type if_dropped



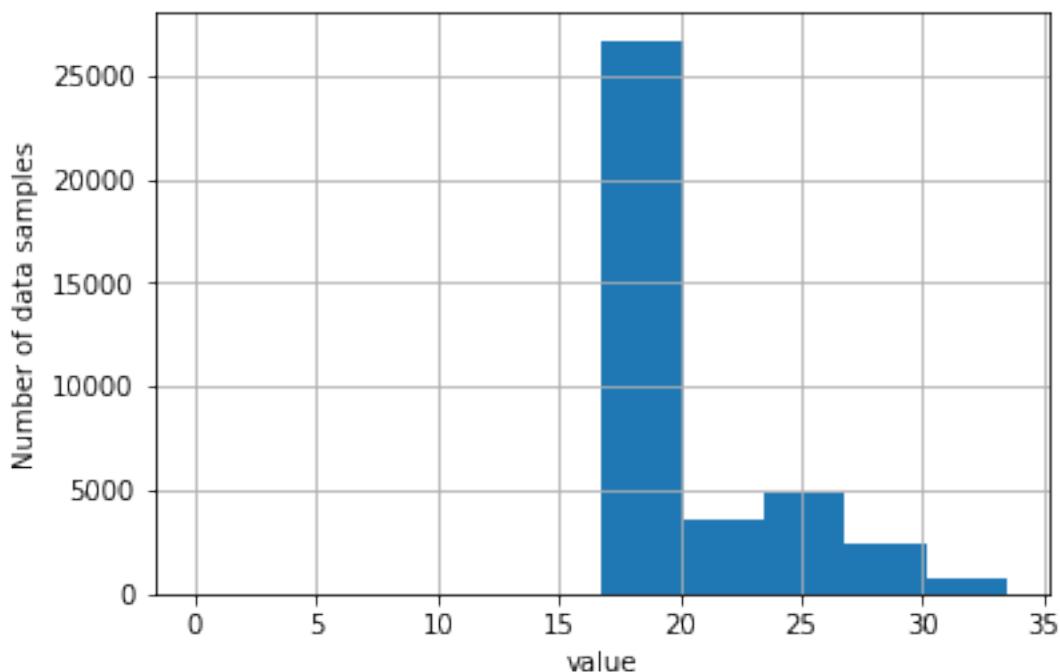
Distribution of data for interface_tx host bb2localdomain instance wlan0 type if_errors



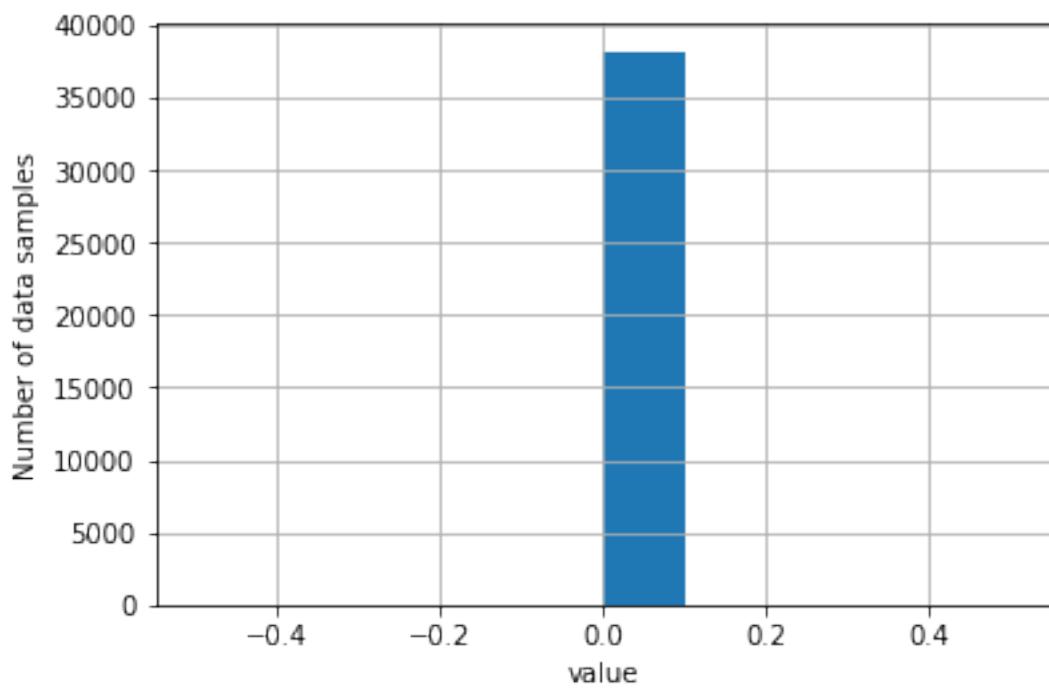
Distribution of data for interface_tx host bb2localdomain instance wlan0 type if_octets



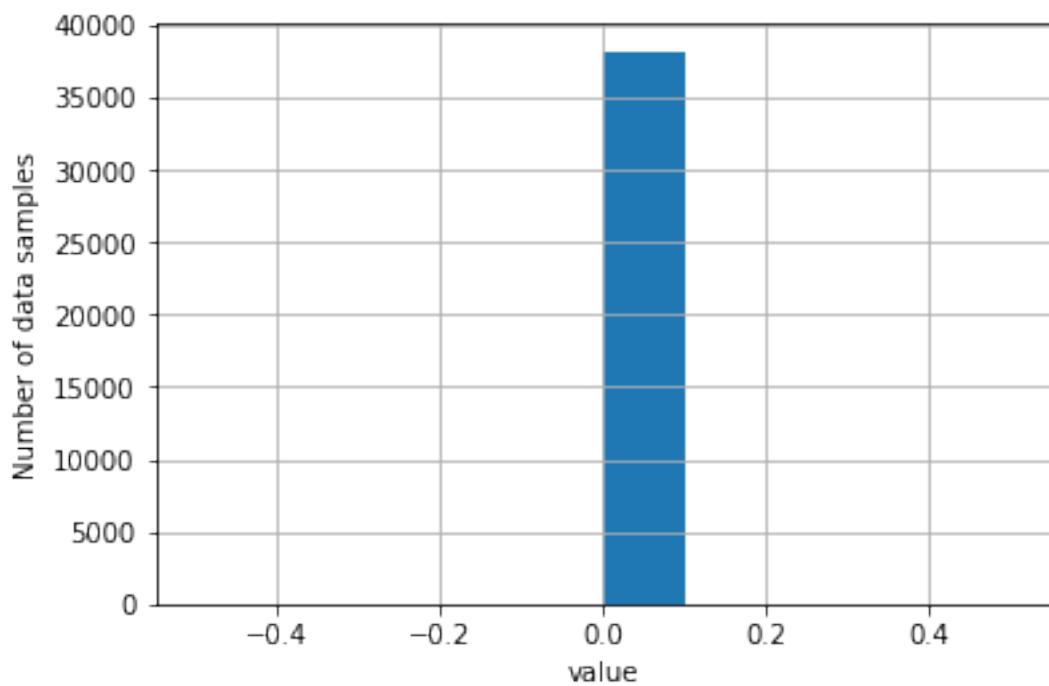
Distribution of data for interface_tx host bb2localdomain instance wlan0 type if_packets



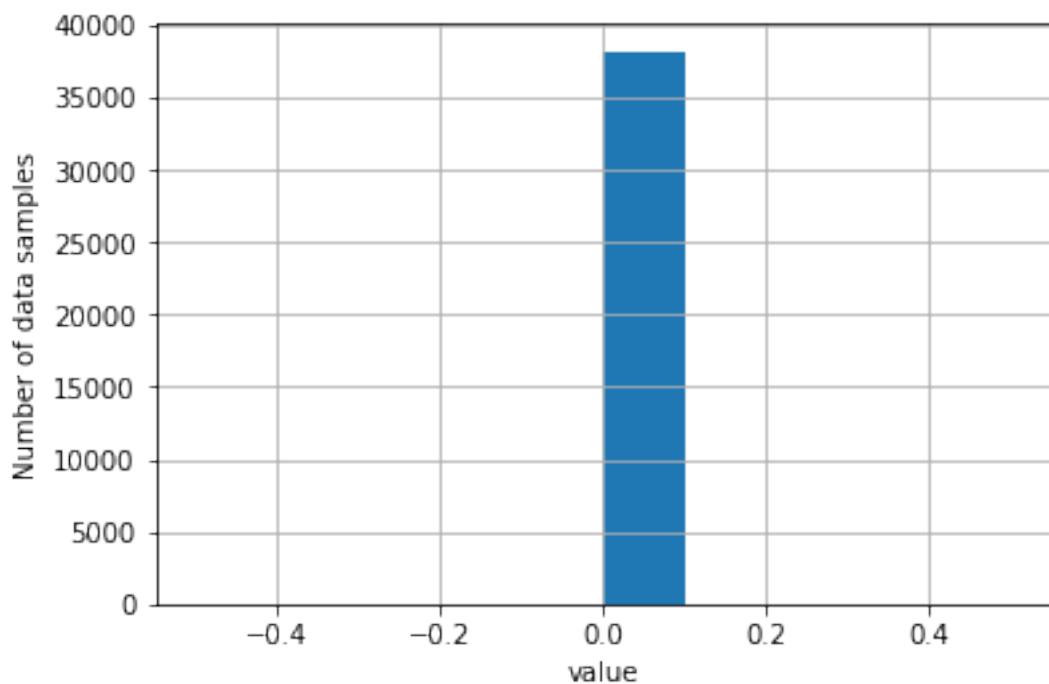
Distribution of data for interface_rx host bb2localdomain instance lo type if_dropped



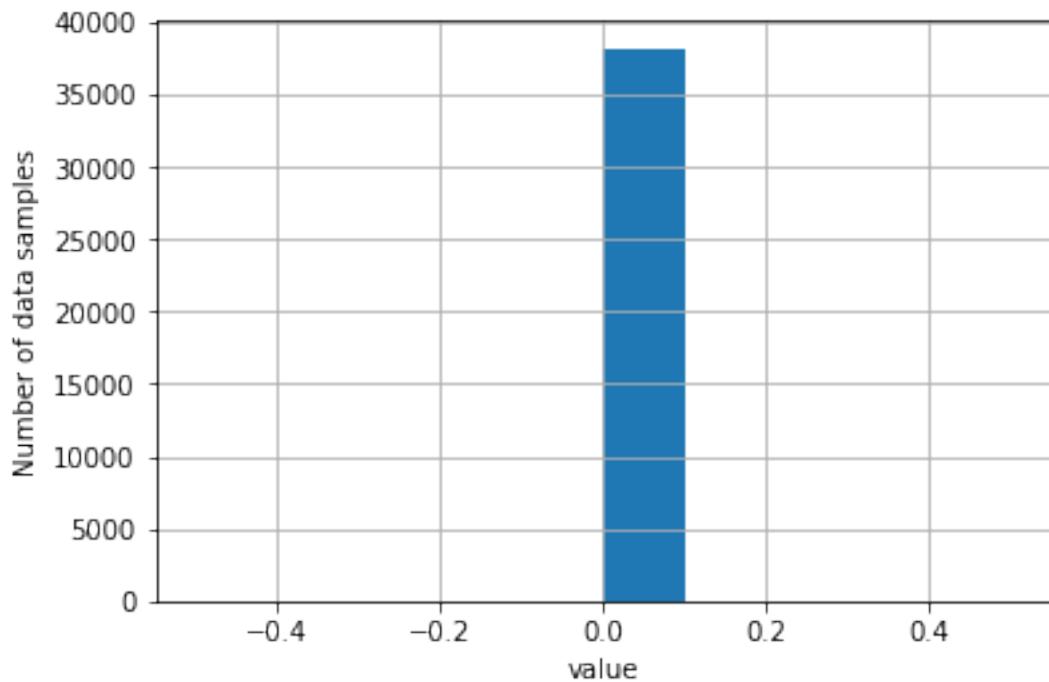
Distribution of data for interface_rx host bb2localdomain instance lo type if_errors



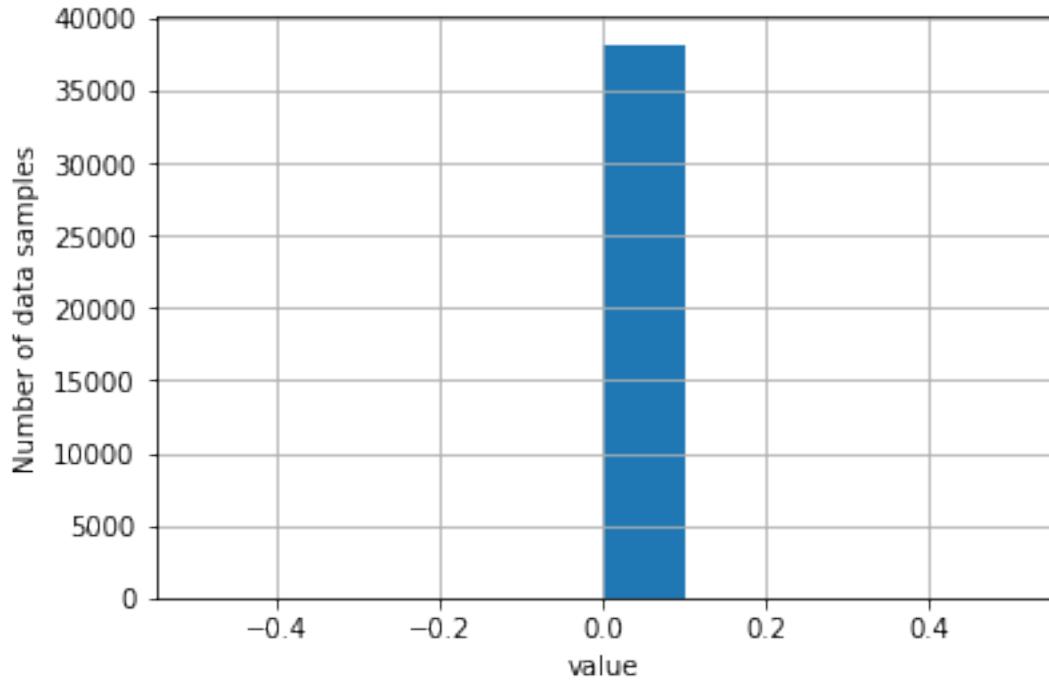
Distribution of data for interface_rx host bb2localdomain instance lo type if_octets



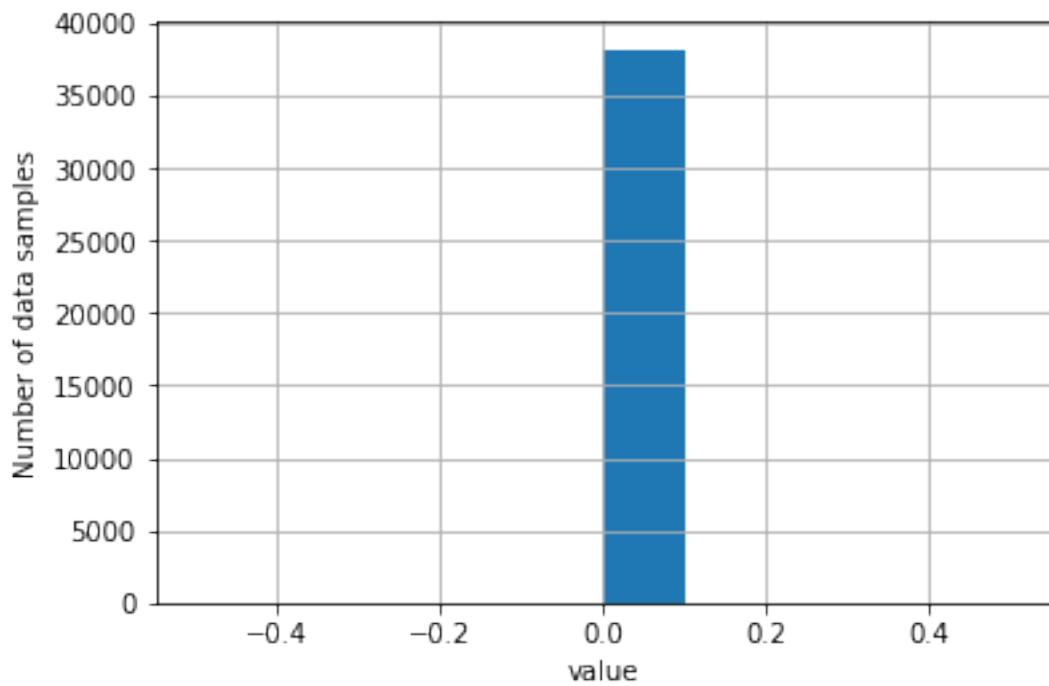
Distribution of data for interface_rx host bb2localdomain instance lo type if_packets



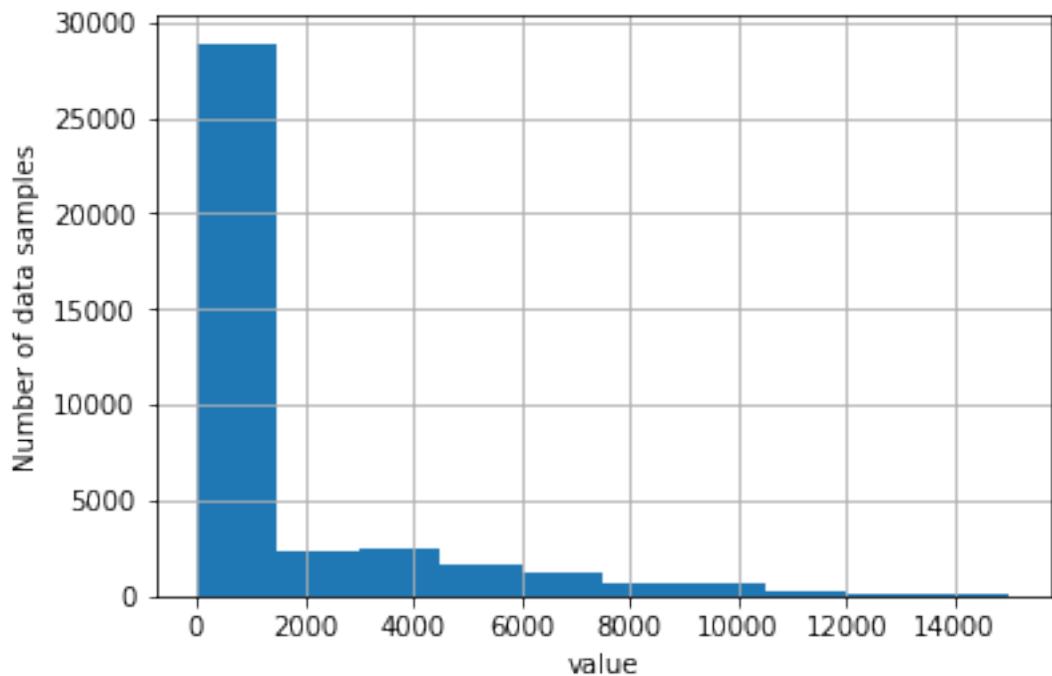
Distribution of data for interface_rx host bb2localdomain instance wlan0 type if_dropped



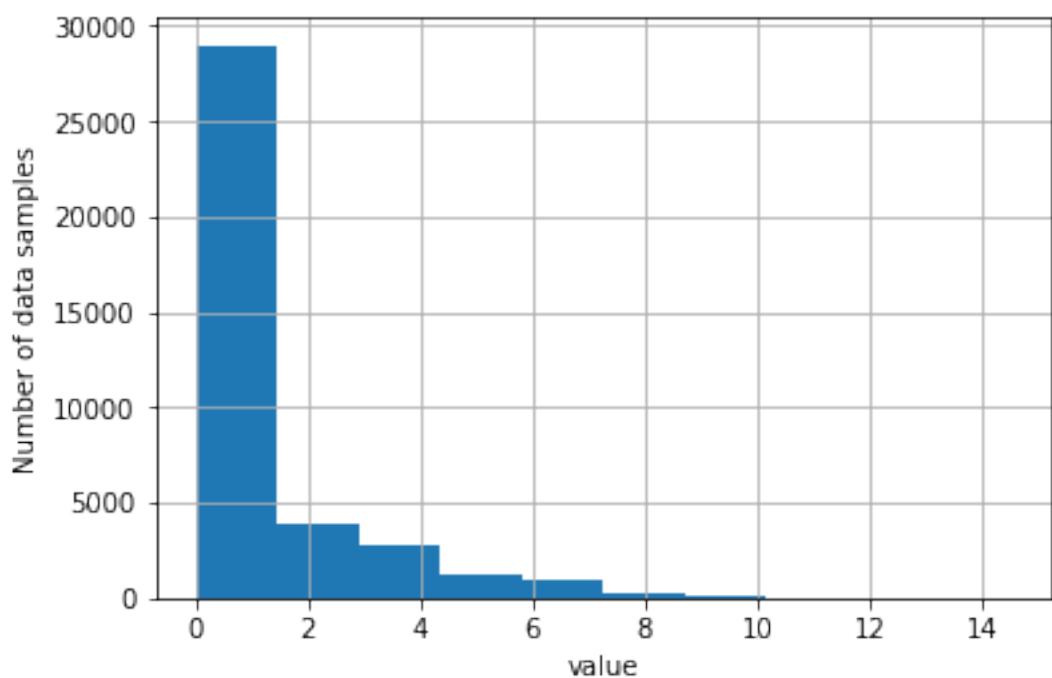
Distribution of data for interface_rx host bb2localdomain instance wlan0 type if_errors



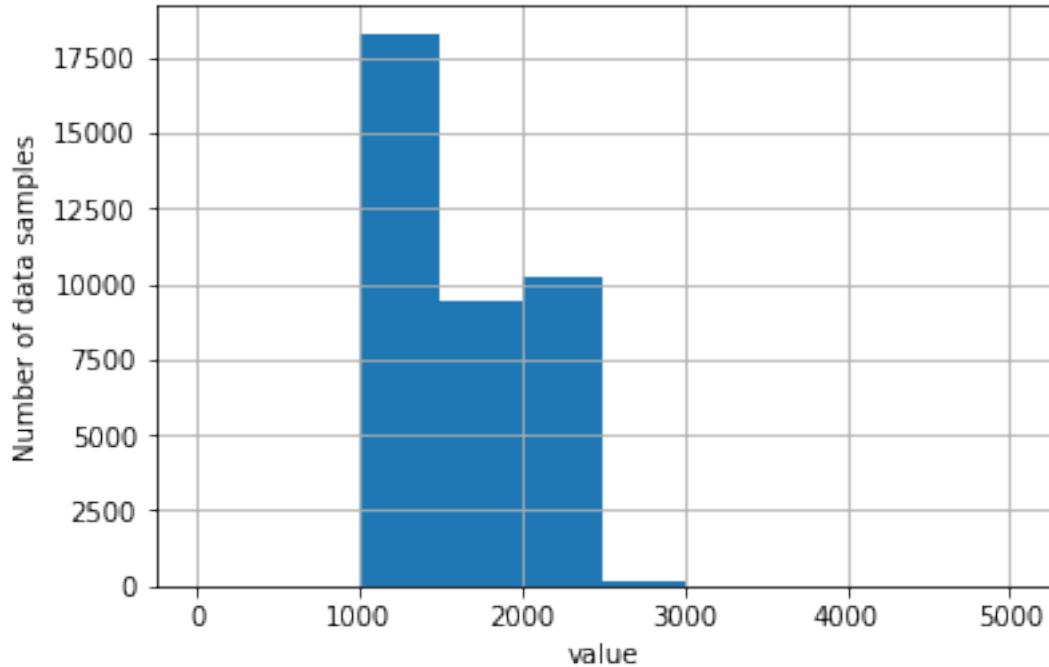
Distribution of data for interface_rx host bb2localdomain instance wlan0 type if_octets



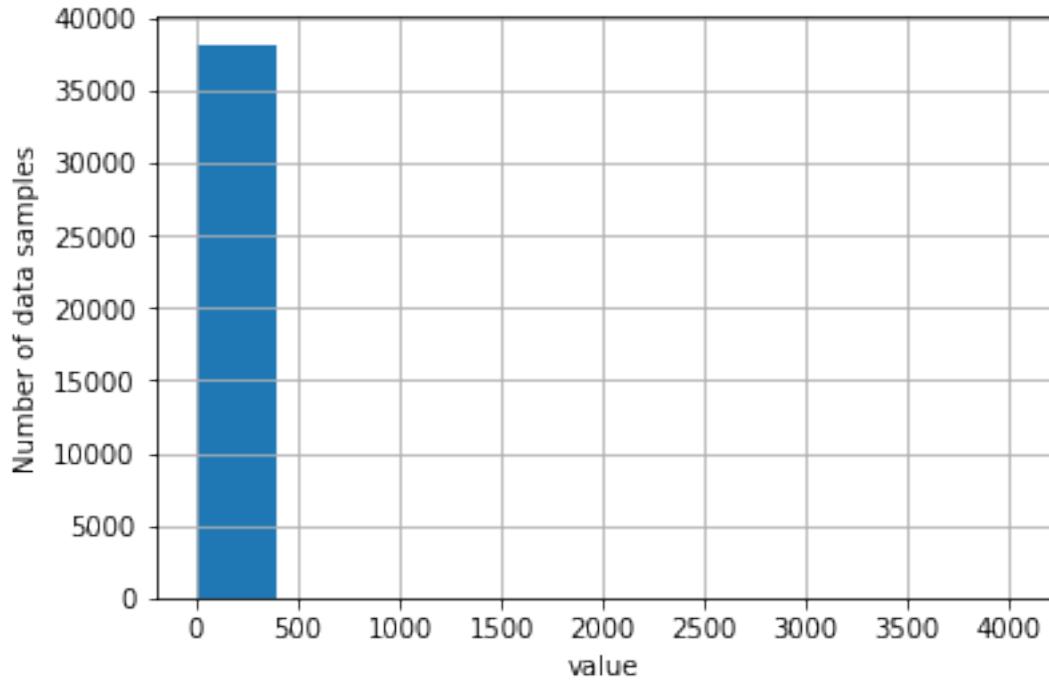
Distribution of data for interface_rx host bb2localdomain instance wlan0 type if_packets



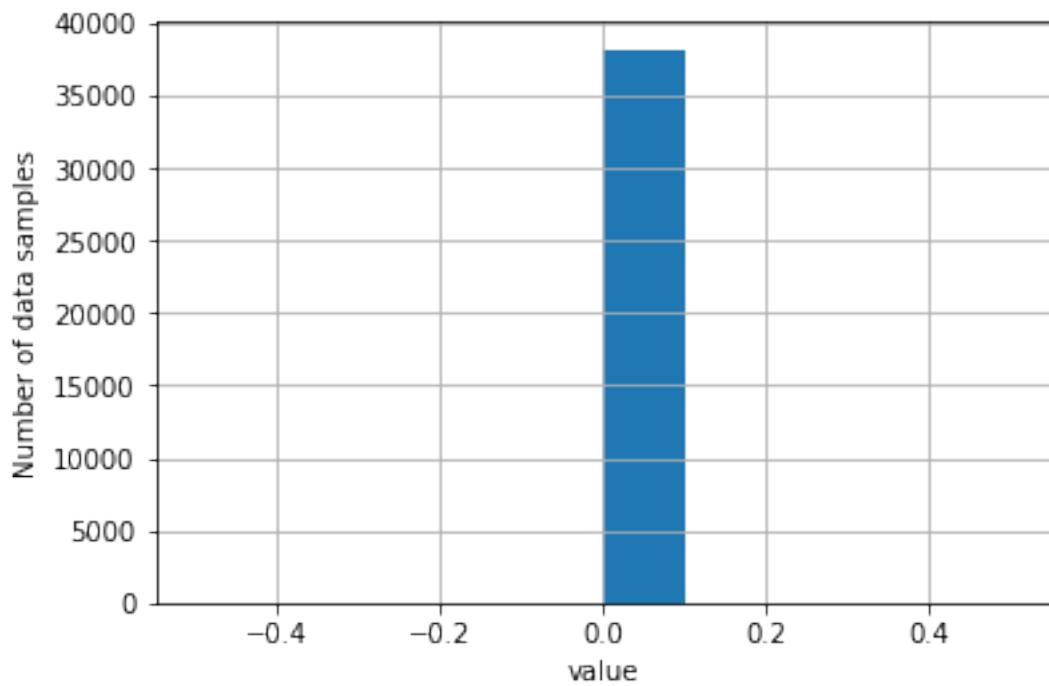
Distribution of data for contextswitch_value host bb2localdomain type contextswitch



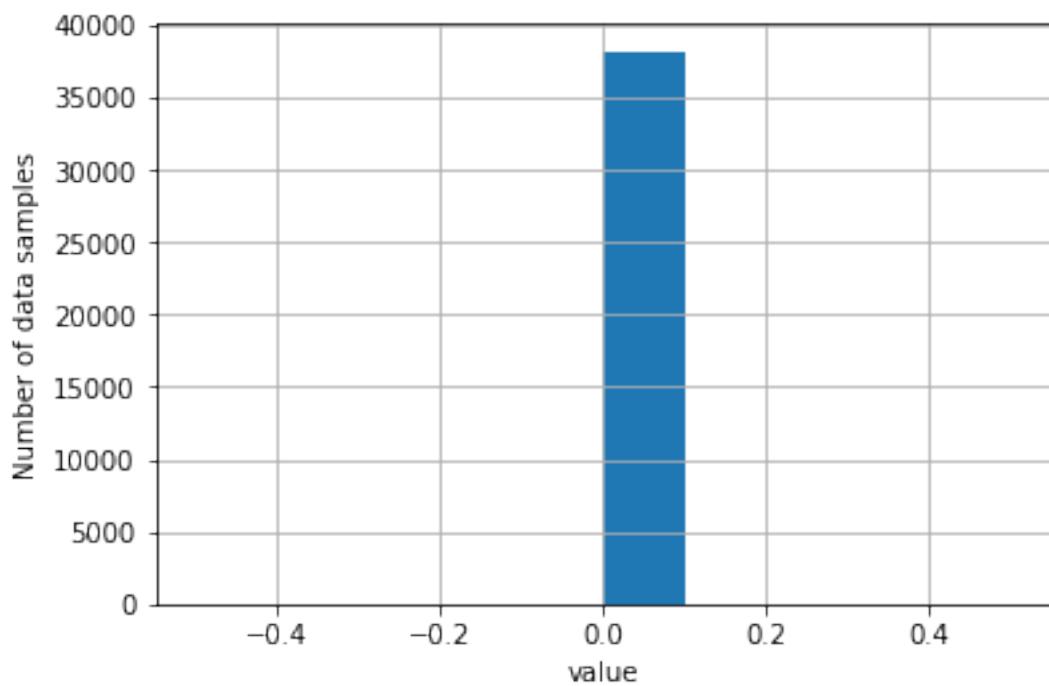
Distribution of data for disk_io_time host bb2localdomain instance mmcblk1 type disk_io_time



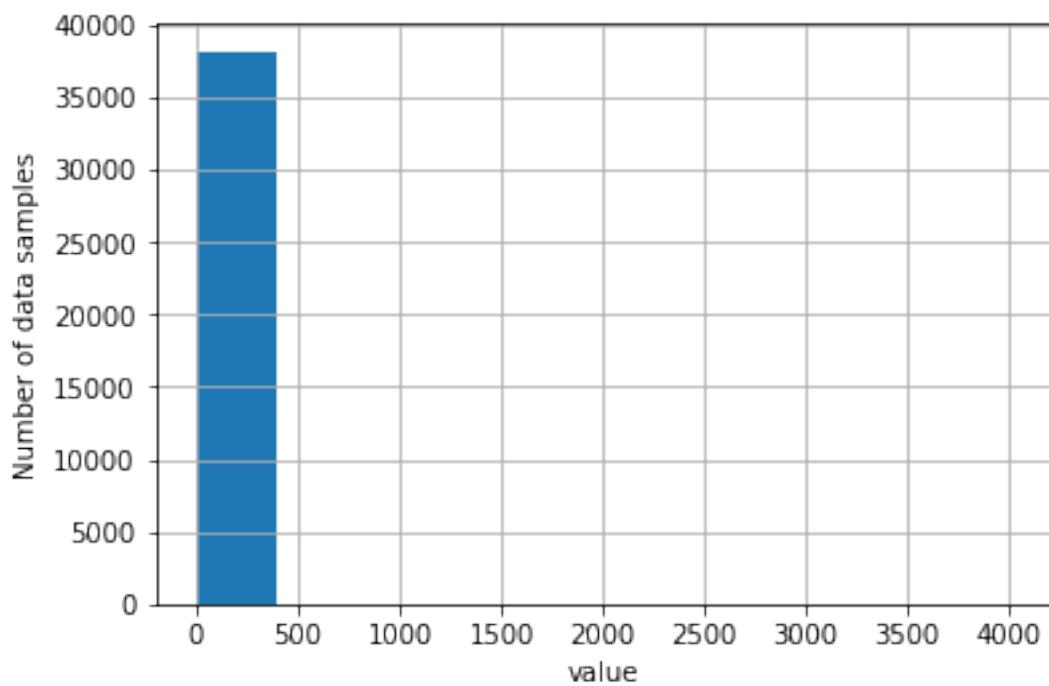
Distribution of data for disk_io_time host bb2localdomain instance mmcblk1boot0 type disk_io_t



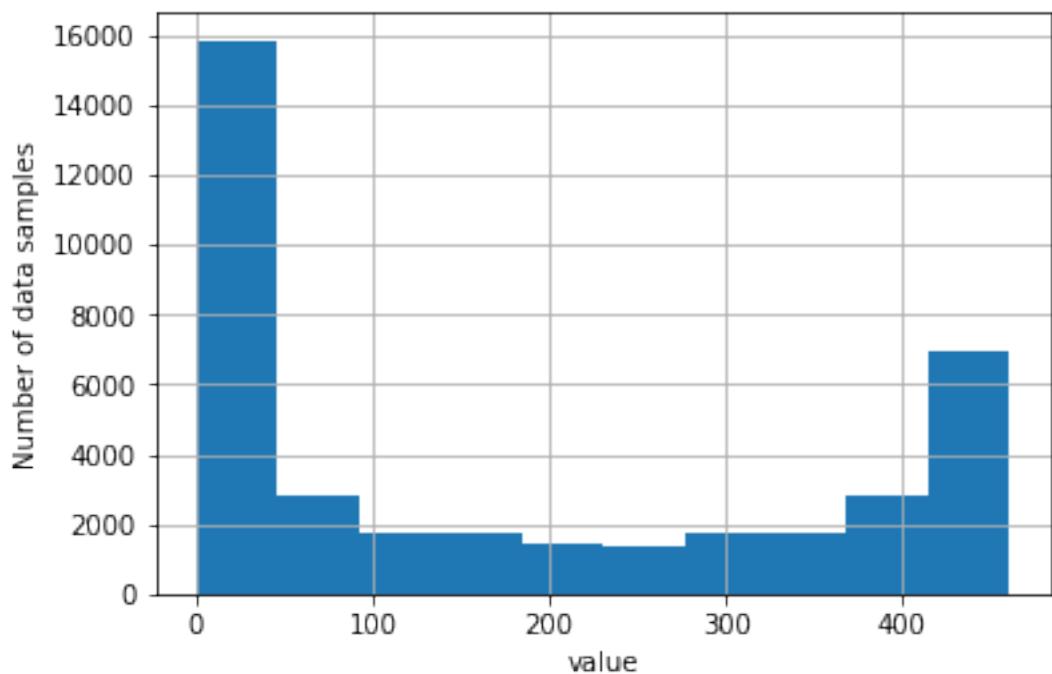
Distribution of data for disk_io_time host bb2localdomain instance mmcblk1boot1 type disk_io_time



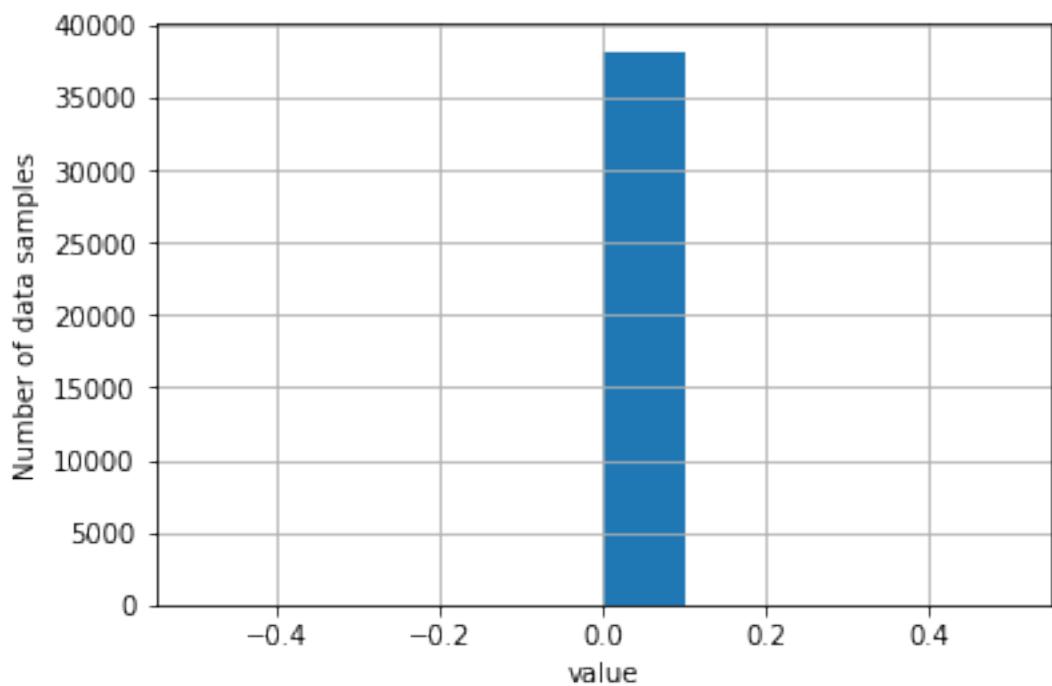
Distribution of data for disk_io_time host bb2localdomain instance mmcblk1p1 type disk_io_time



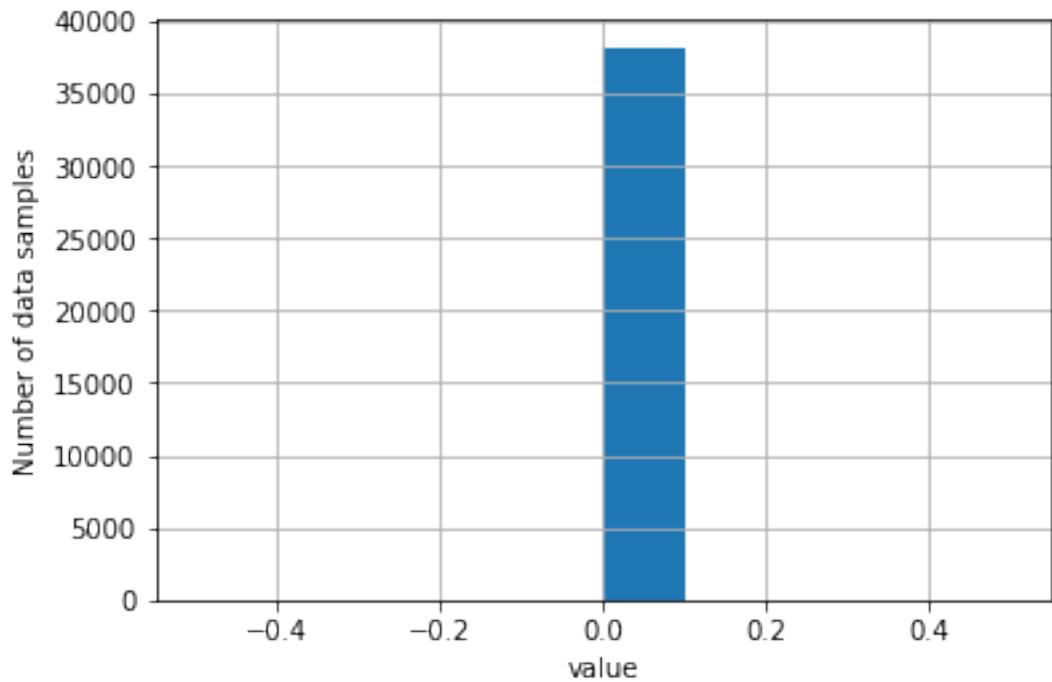
```
(38200, 29)
Index(['cpu_value host bb3localdomain type_instance idle',
       'cpu_value host bb3localdomain type_instance interrupt',
       'cpu_value host bb3localdomain type_instance nice',
       'cpu_value host bb3localdomain type_instance softirq',
       'cpu_value host bb3localdomain type_instance steal',
       'cpu_value host bb3localdomain type_instance system',
       'cpu_value host bb3localdomain type_instance user',
       'cpu_value host bb3localdomain type_instance wait',
       'interface_tx host bb3localdomain instance lo type if_dropped',
       'interface_tx host bb3localdomain instance lo type if_errors',
       'interface_tx host bb3localdomain instance lo type if_octets',
       'interface_tx host bb3localdomain instance lo type if_packets',
       'interface_tx host bb3localdomain instance wlan0 type if_dropped',
       'interface_tx host bb3localdomain instance wlan0 type if_errors',
       'interface_tx host bb3localdomain instance wlan0 type if_octets',
       'interface_tx host bb3localdomain instance wlan0 type if_packets',
       'interface_rx host bb3localdomain instance lo type if_dropped',
       'interface_rx host bb3localdomain instance lo type if_errors',
       'interface_rx host bb3localdomain instance lo type if_octets',
       'interface_rx host bb3localdomain instance lo type if_packets',
       'interface_rx host bb3localdomain instance wlan0 type if_dropped',
       'interface_rx host bb3localdomain instance wlan0 type if_errors',
       'interface_rx host bb3localdomain instance wlan0 type if_octets',
       'interface_rx host bb3localdomain instance wlan0 type if_packets',
       'contextswitch_value host bb3localdomain type contextswitch',
       'disk_io_time host bb3localdomain instance mmcblk1 type disk_io_time',
       'disk_io_time host bb3localdomain instance mmcblk1boot0 type disk_io_time',
       'disk_io_time host bb3localdomain instance mmcblk1boot1 type disk_io_time',
       'disk_io_time host bb3localdomain instance mmcblk1p1 type disk_io_time'],
      dtype='object')
Distribution of data for cpu_value host bb3localdomain type_instance idle
```



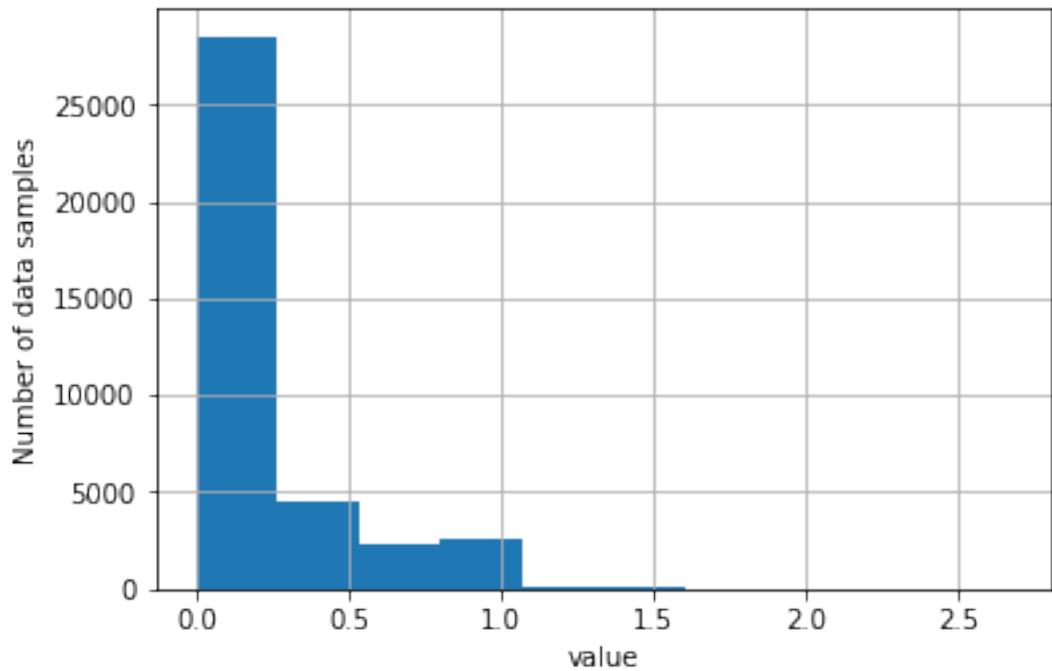
Distribution of data for cpu_value host bb3localdomain type_instance interrupt



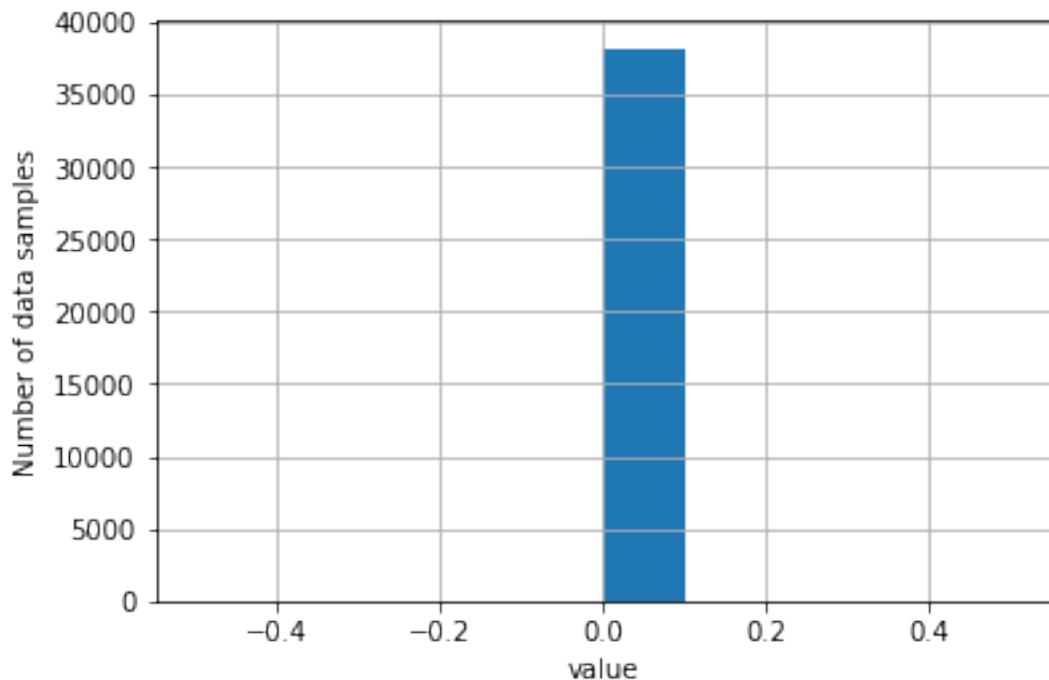
Distribution of data for cpu_value host bb3localdomain type_instance nice



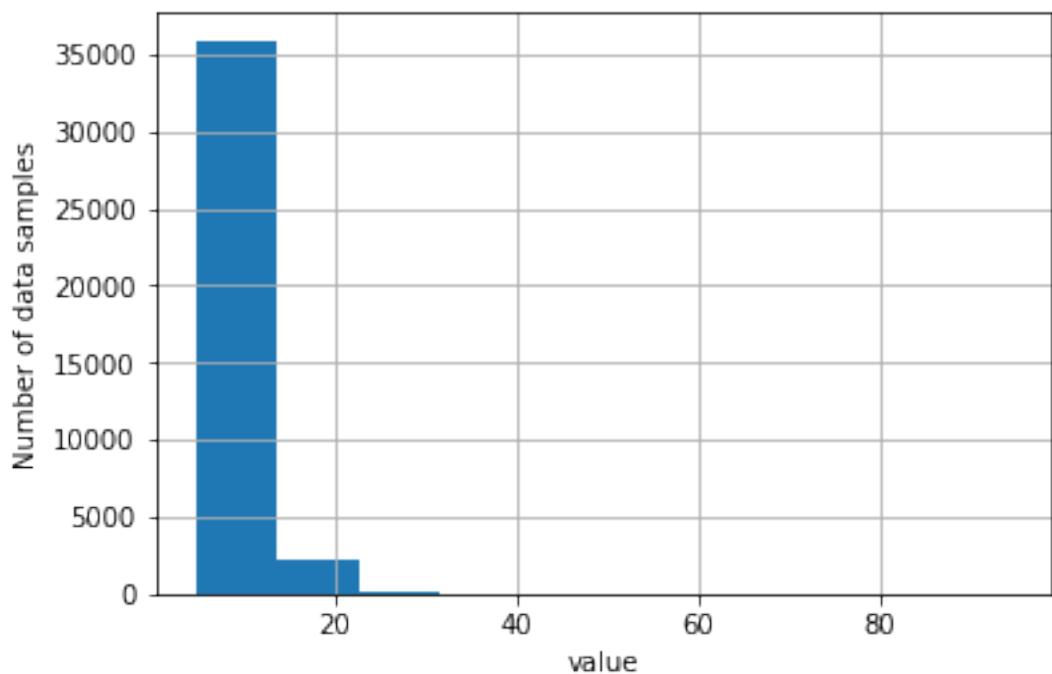
Distribution of data for cpu_value host bb3localdomain type_instance softirq



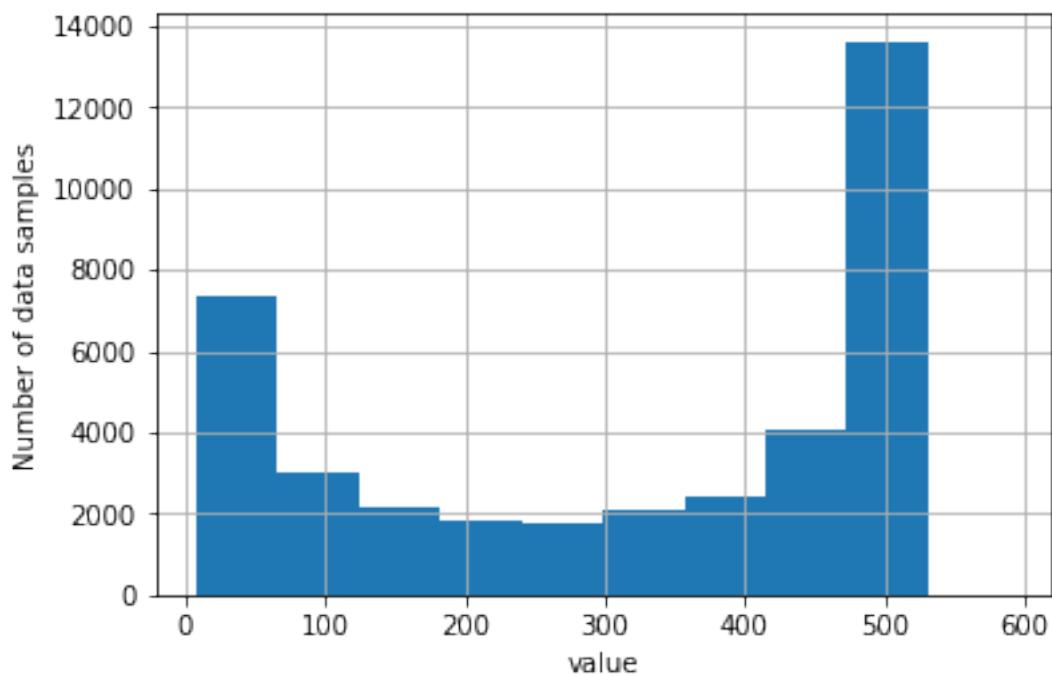
Distribution of data for cpu_value host bb3localdomain type_instance steal



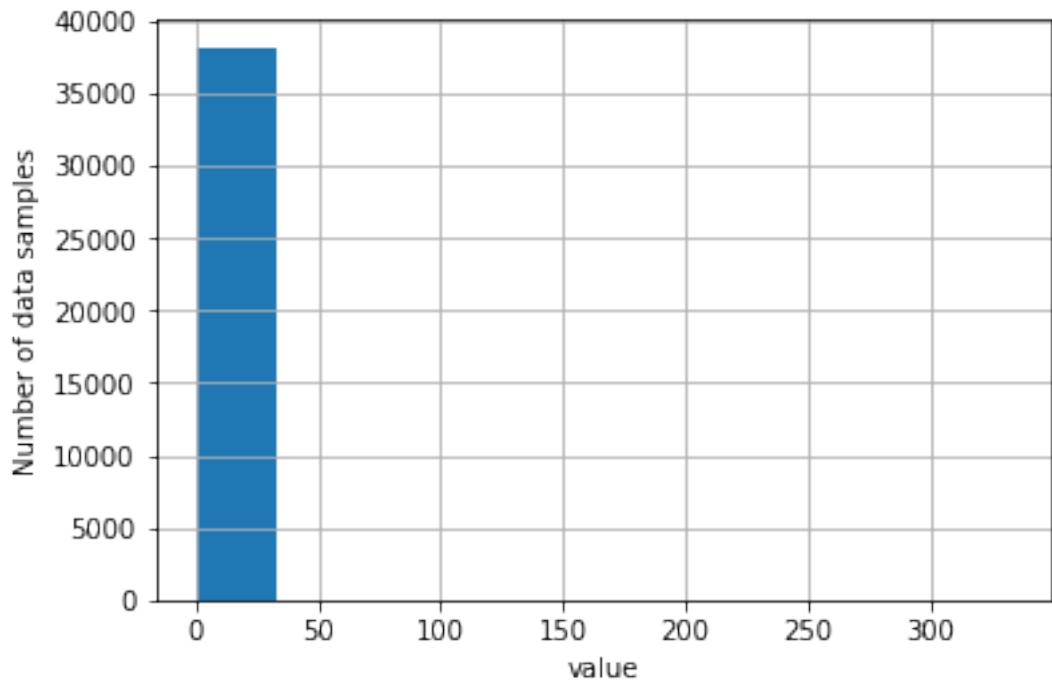
Distribution of data for cpu_value host bb3localdomain type_instance system



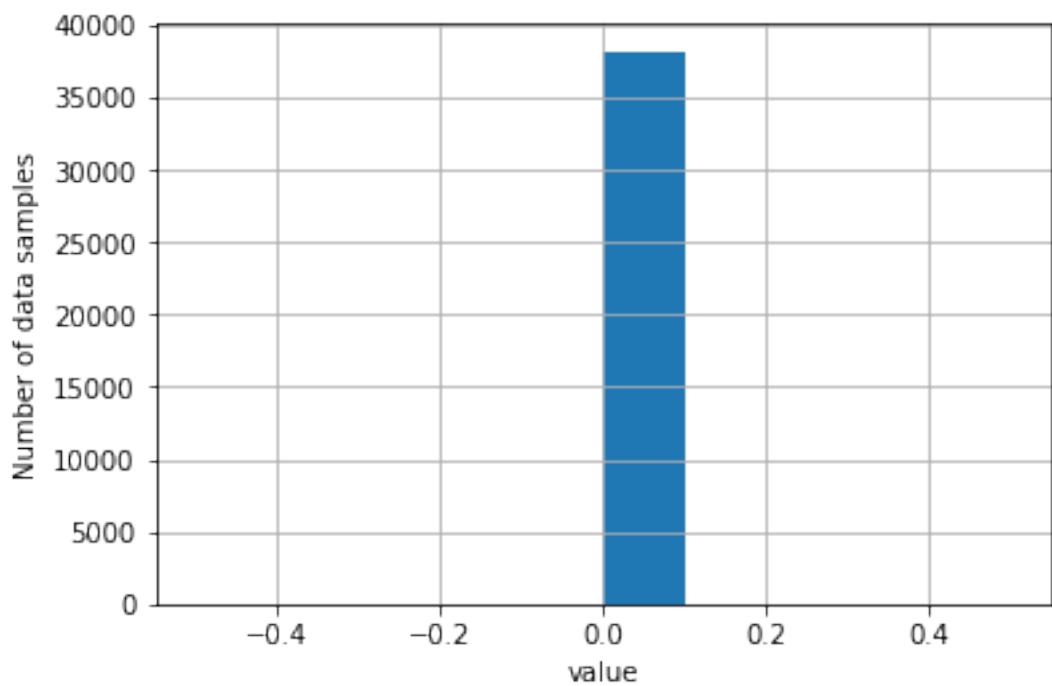
Distribution of data for cpu_value host bb3localdomain type_instance user



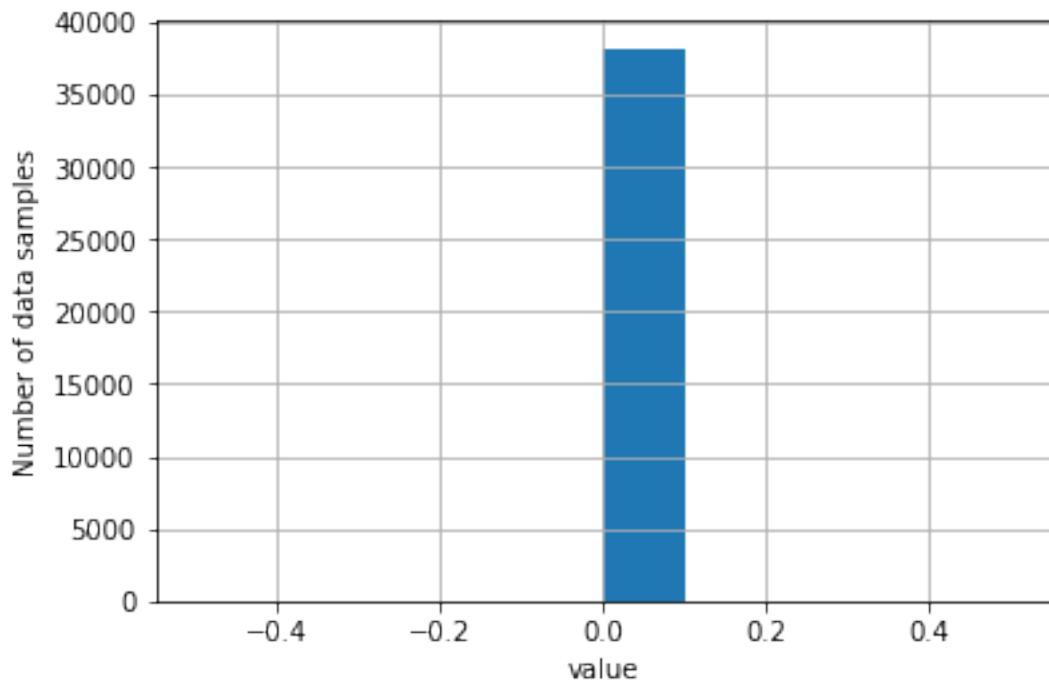
Distribution of data for cpu_value host bb3localdomain type_instance wait



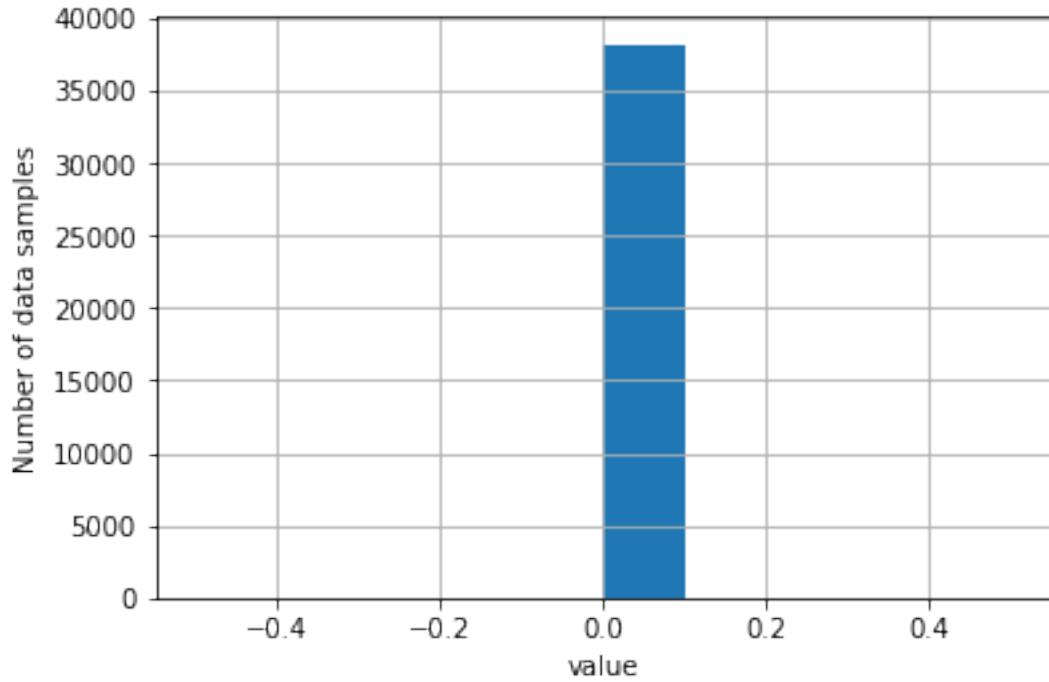
Distribution of data for interface_tx host bb3localdomain instance lo type if_dropped



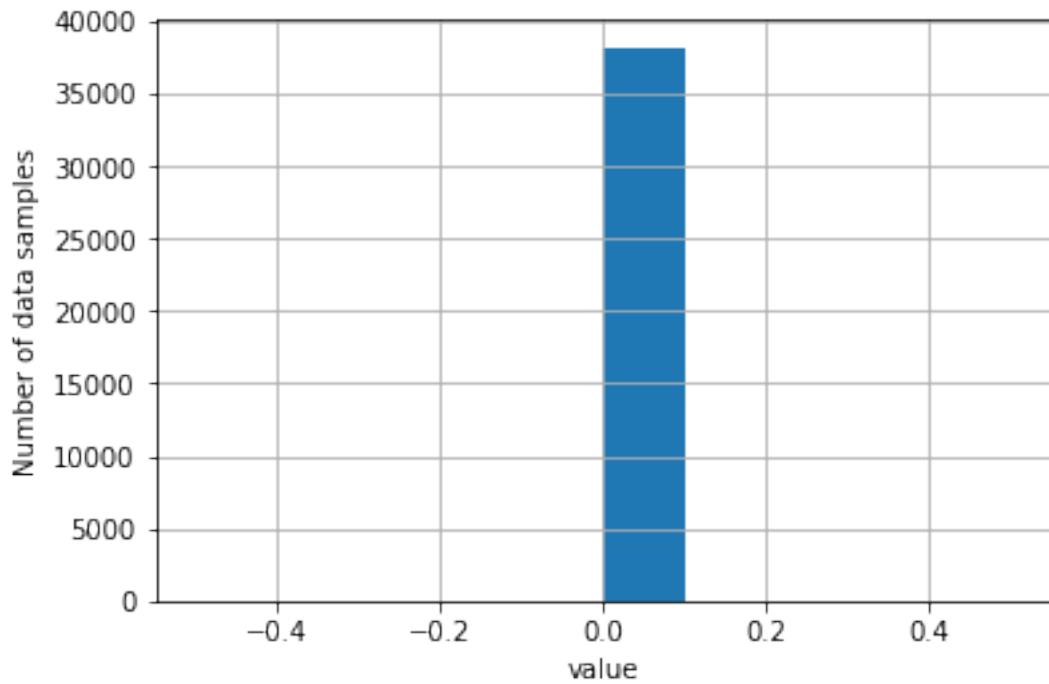
Distribution of data for interface_tx host bb3localdomain instance lo type if_errors



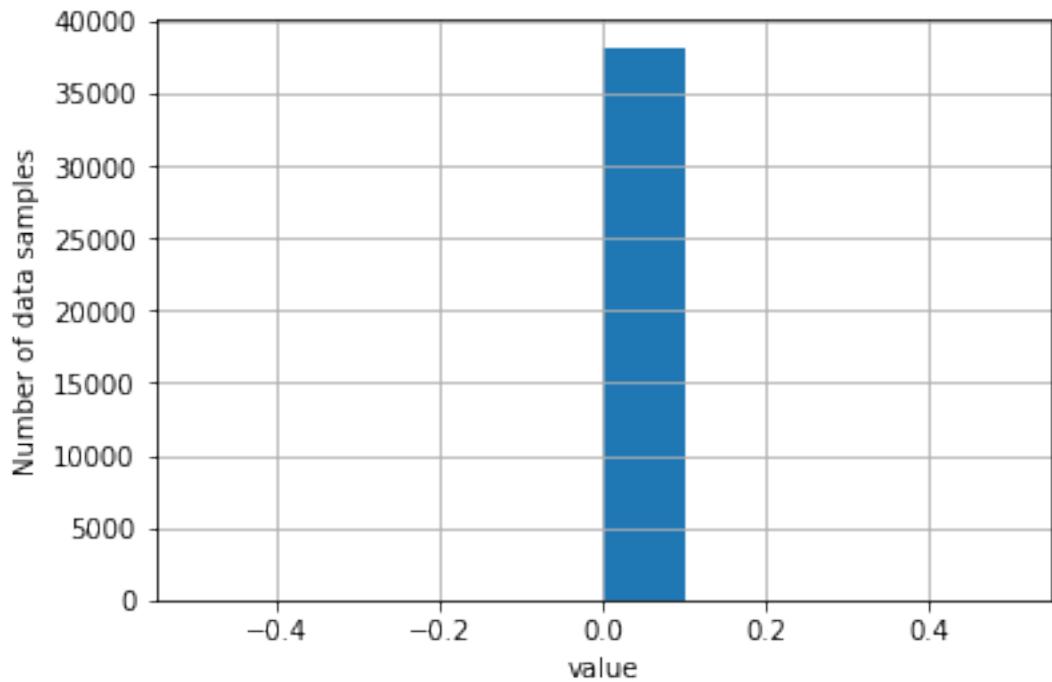
Distribution of data for interface_tx host bb3localdomain instance lo type if_octets



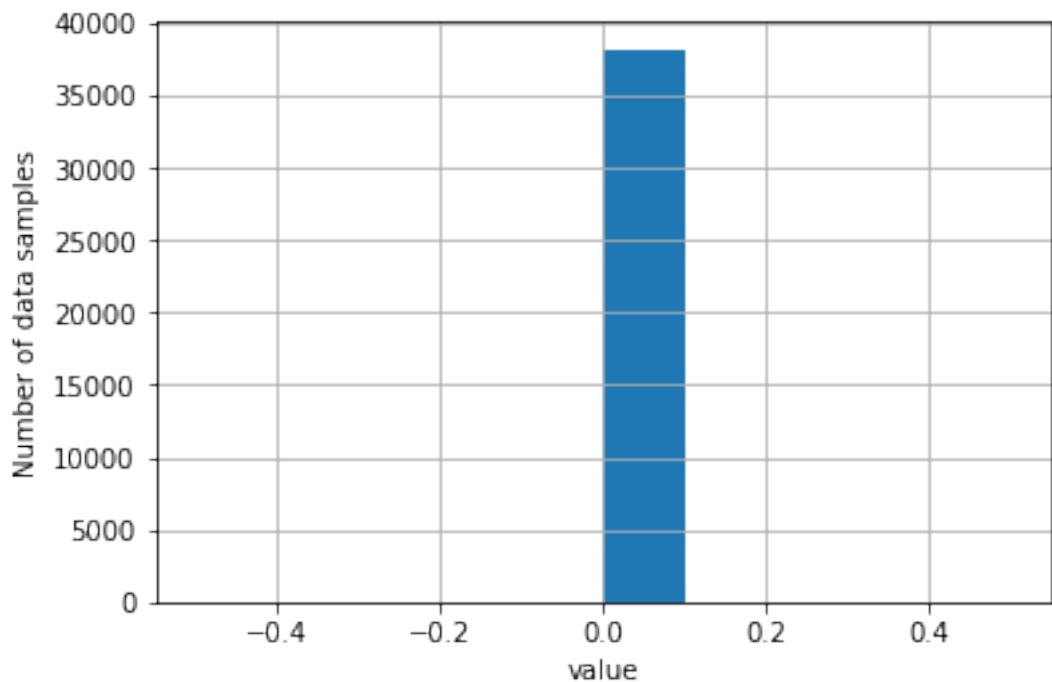
Distribution of data for interface_tx host bb3localdomain instance lo type if_packets



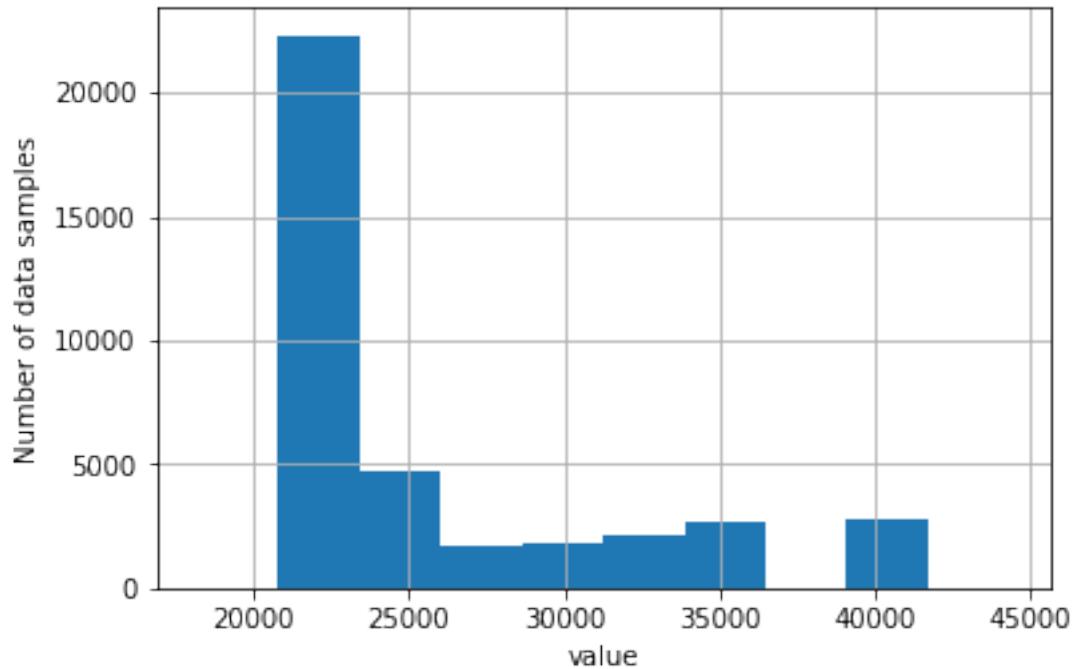
Distribution of data for interface_tx host bb3localdomain instance wlan0 type if_dropped



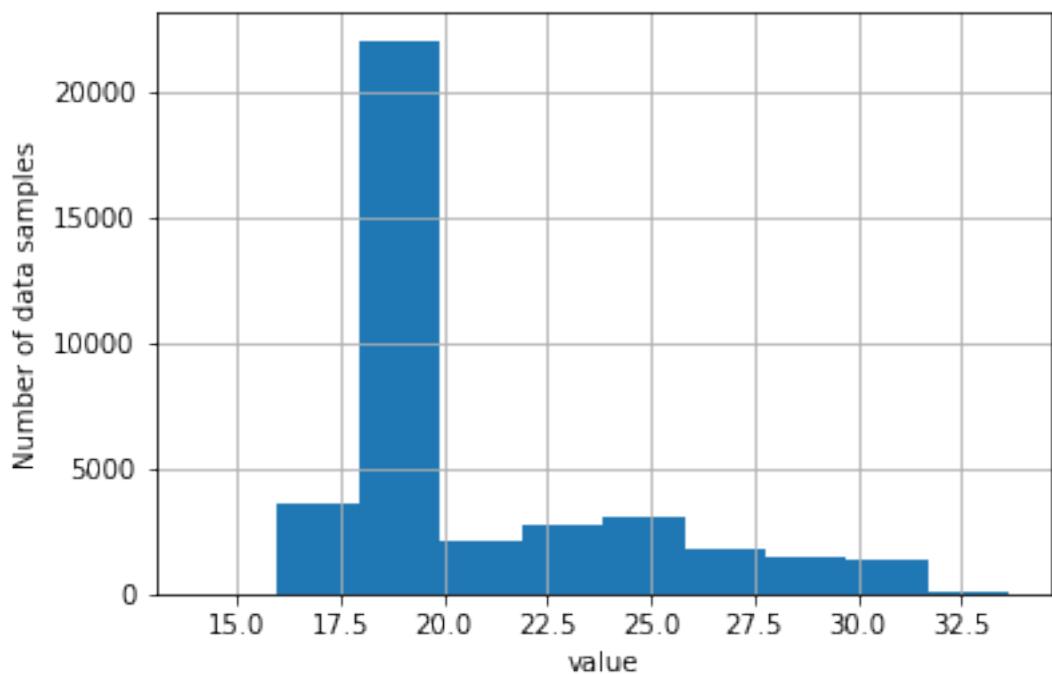
Distribution of data for interface_tx host bb3localdomain instance wlan0 type if_errors



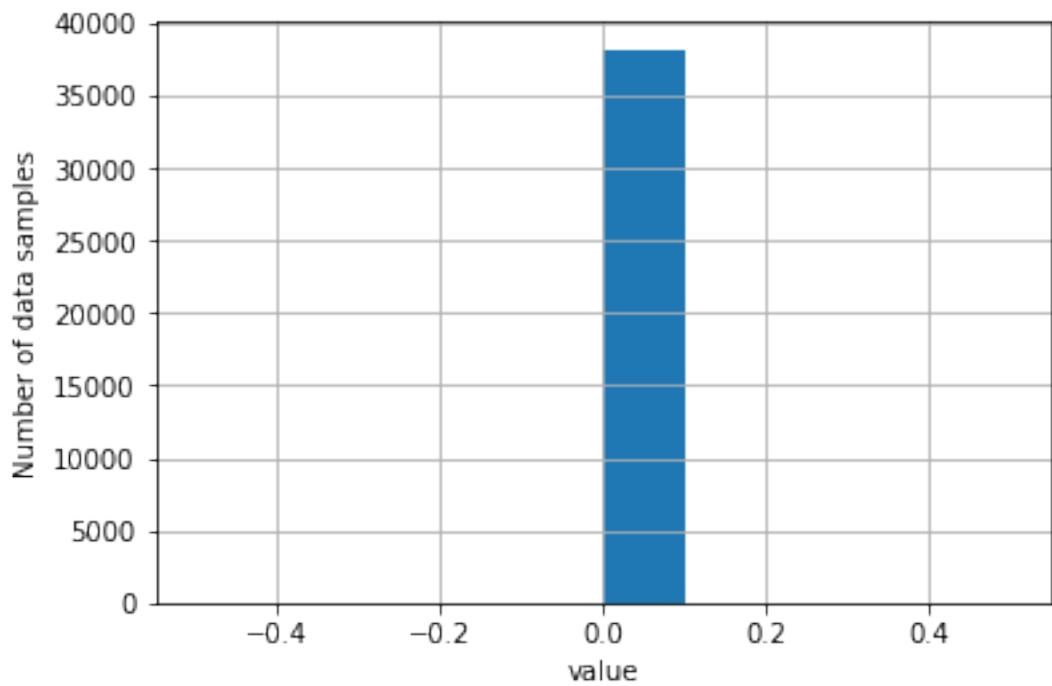
Distribution of data for interface_tx host bb3localdomain instance wlan0 type if_octets



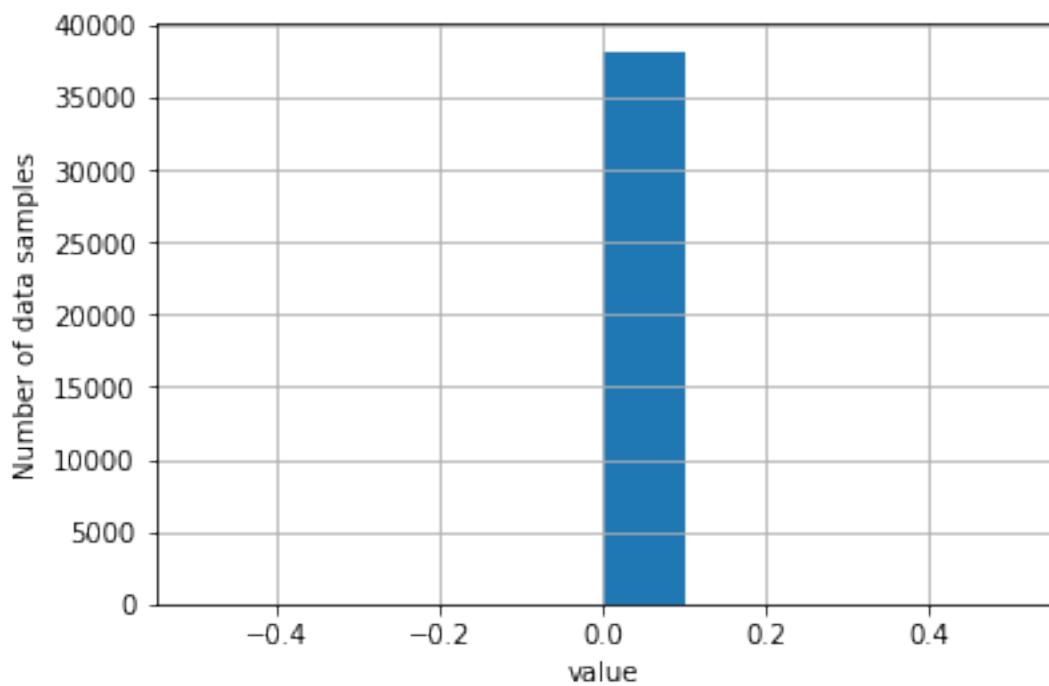
Distribution of data for interface_tx host bb3localdomain instance wlan0 type if_packets



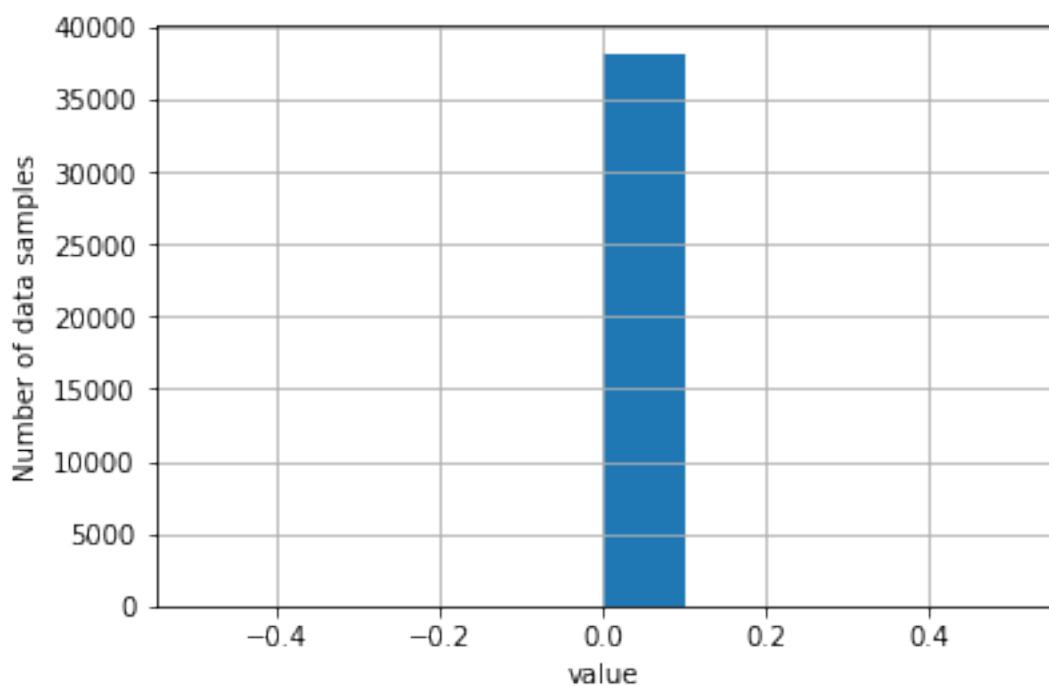
Distribution of data for interface_rx host bb3localdomain instance lo type if_dropped



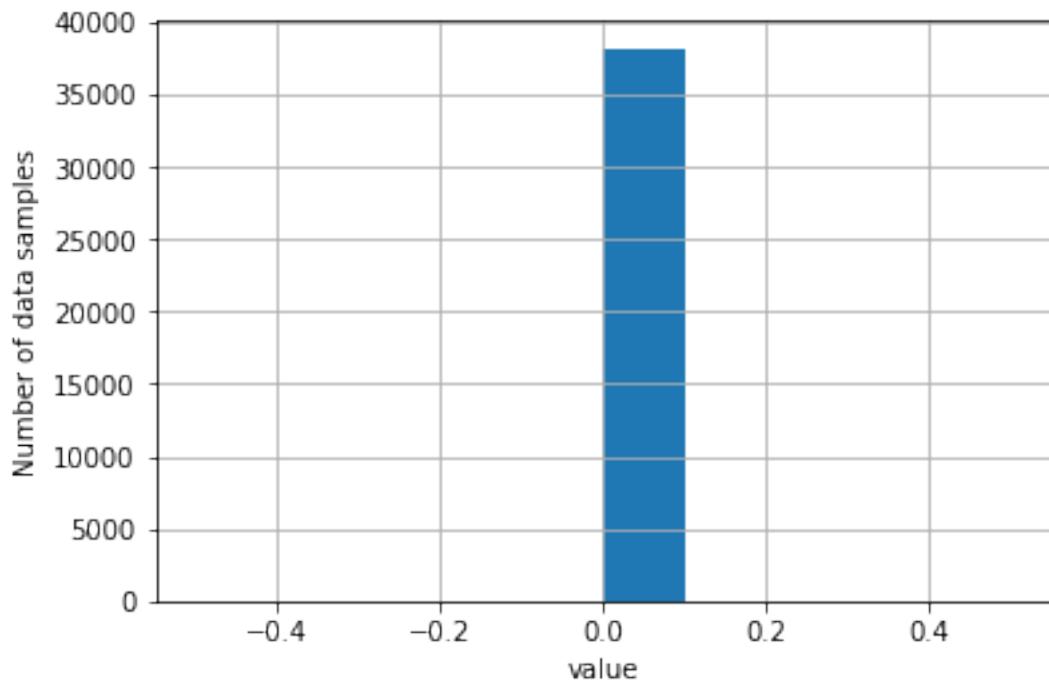
Distribution of data for interface_rx host bb3localdomain instance lo type if_errors



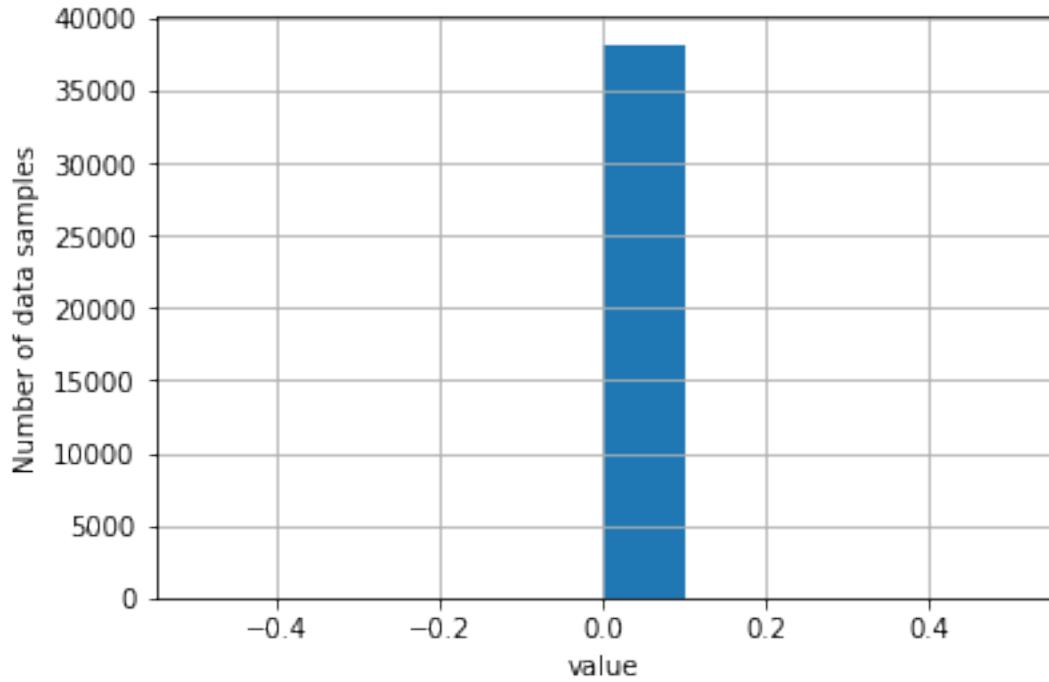
Distribution of data for interface_rx host bb3localdomain instance lo type if_octets



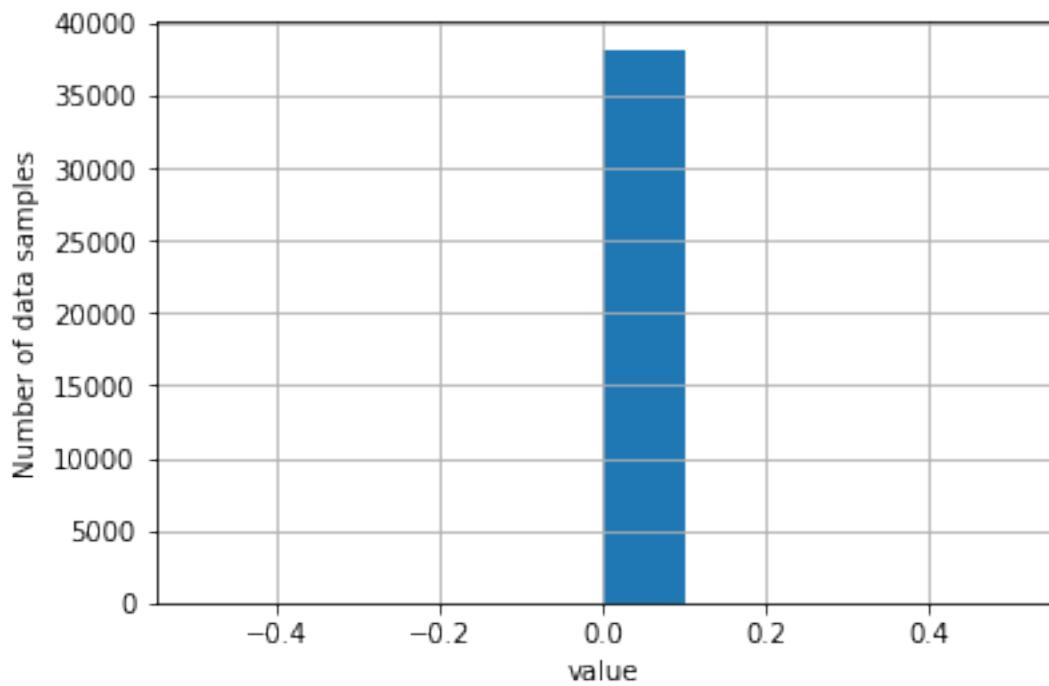
Distribution of data for interface_rx host bb3localdomain instance lo type if_packets



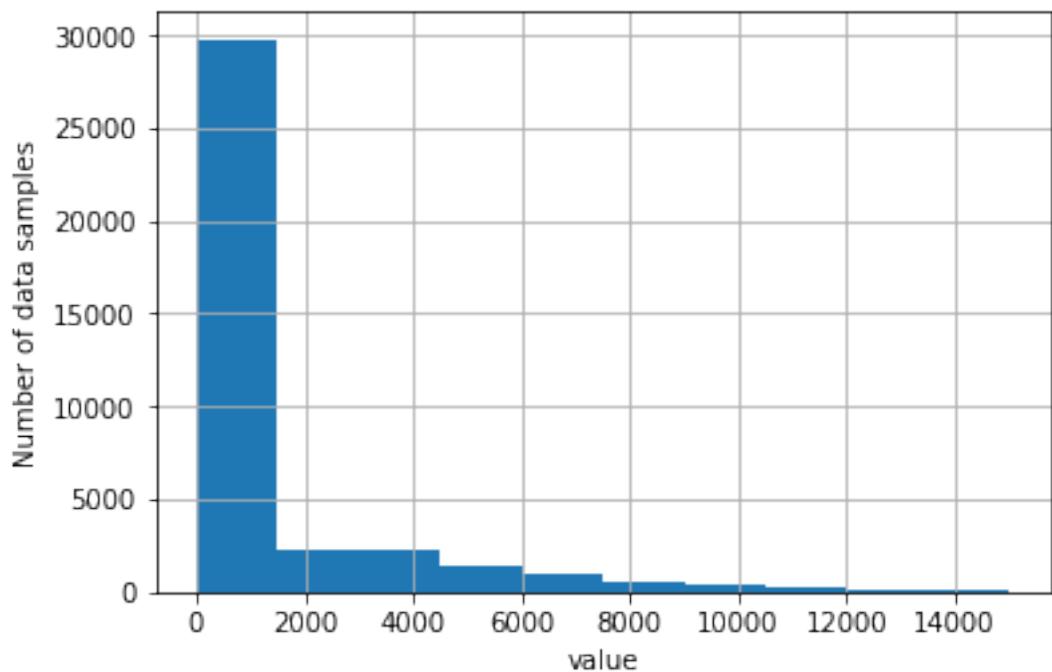
Distribution of data for interface_rx host bb3localdomain instance wlan0 type if_dropped



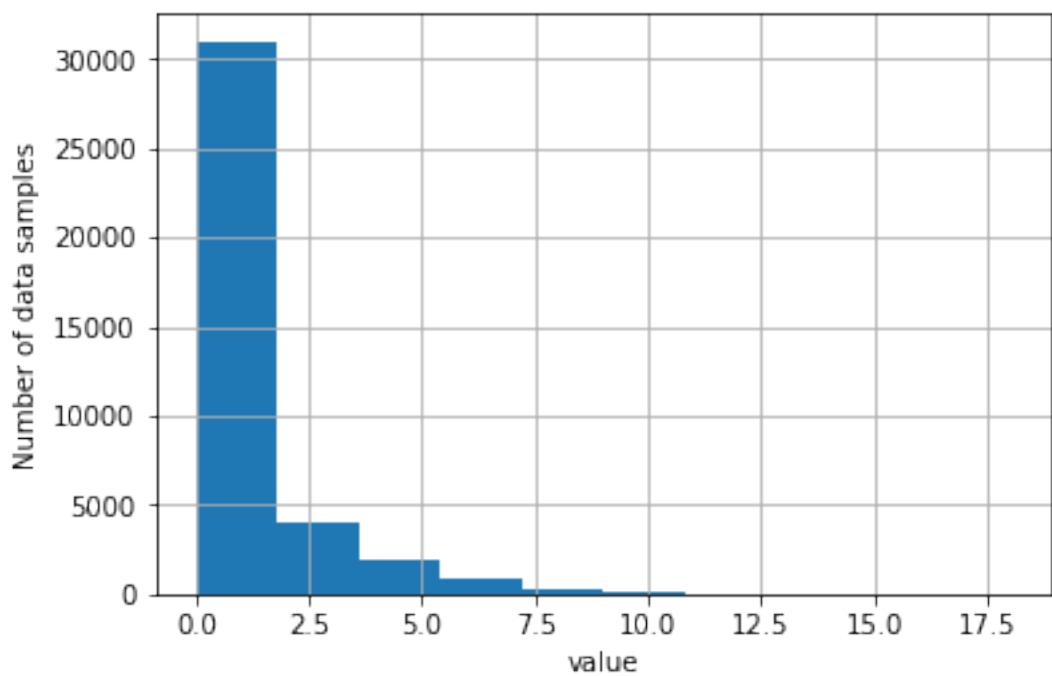
Distribution of data for interface_rx host bb3localdomain instance wlan0 type if_errors



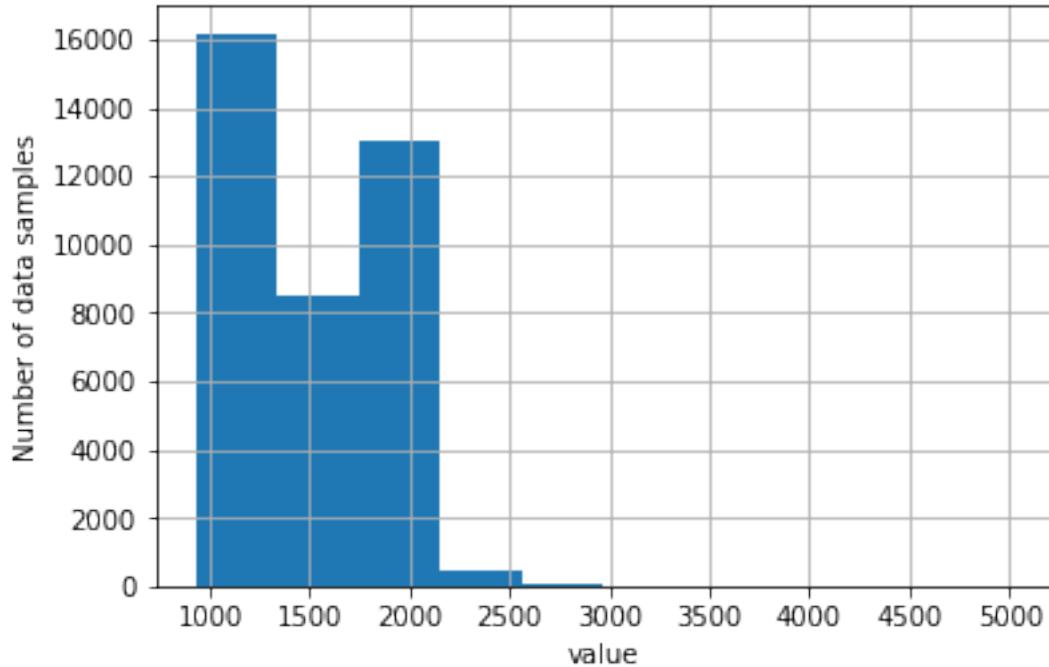
Distribution of data for interface_rx host bb3localdomain instance wlan0 type if_octets



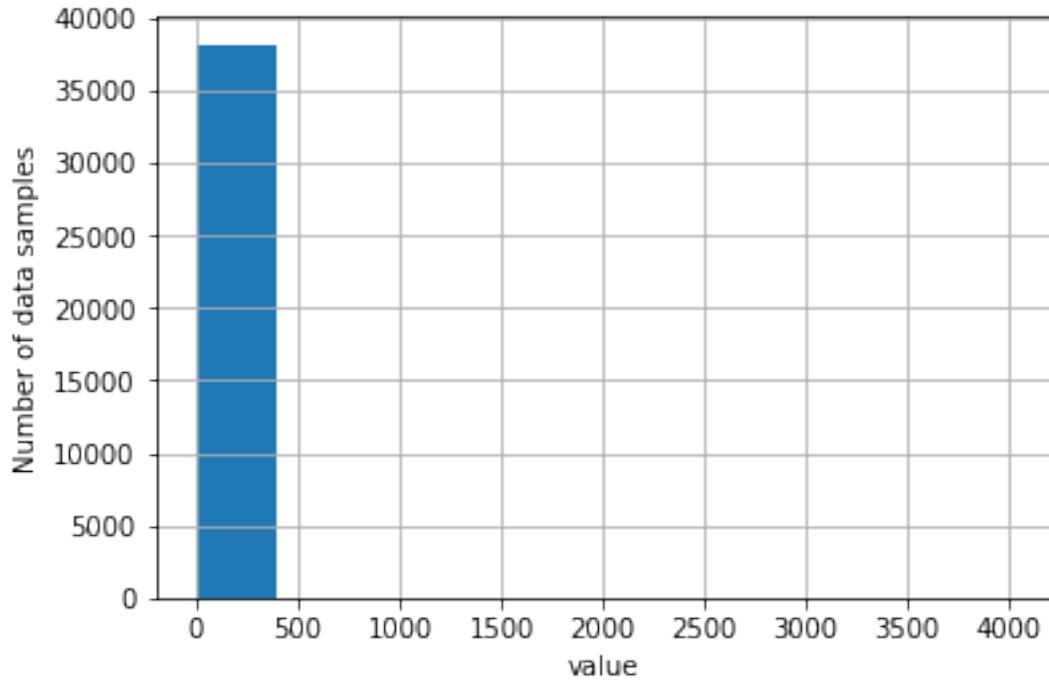
Distribution of data for interface_rx host bb3localdomain instance wlan0 type if_packets



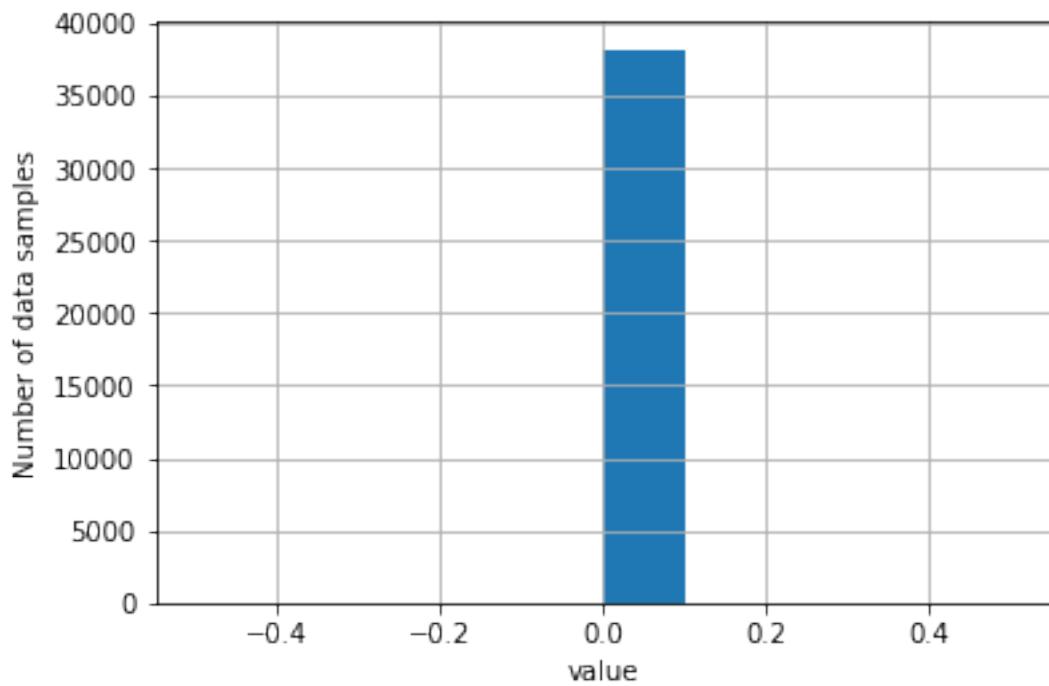
Distribution of data for contextswitch_value host bb3localdomain type contextswitch



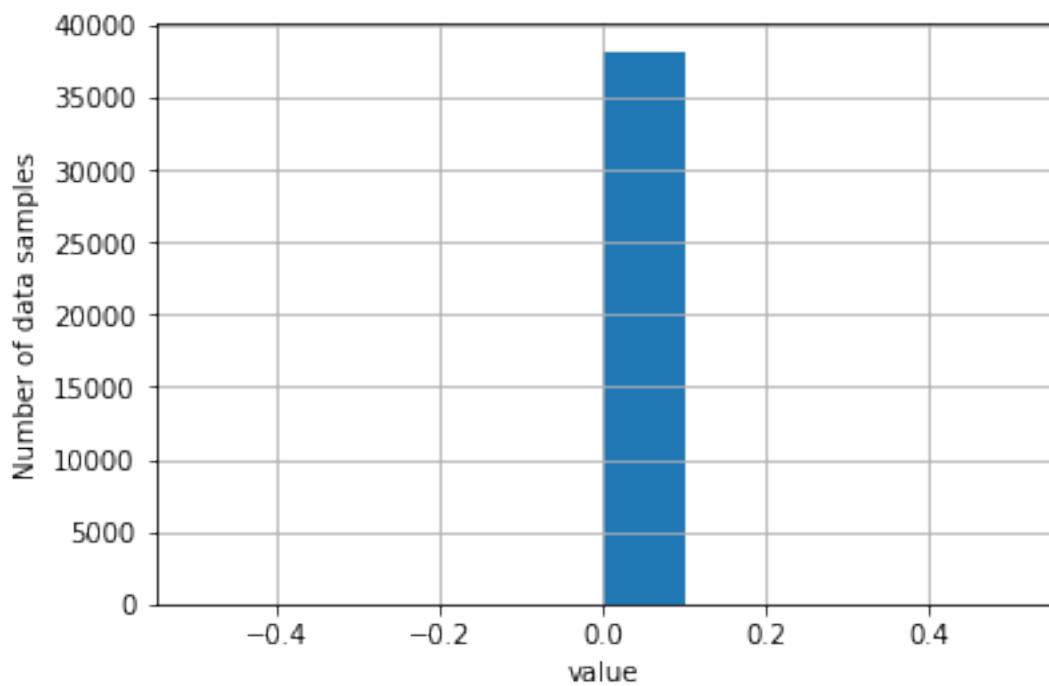
Distribution of data for disk_io_time host bb3localdomain instance mmcblk1 type disk_io_time



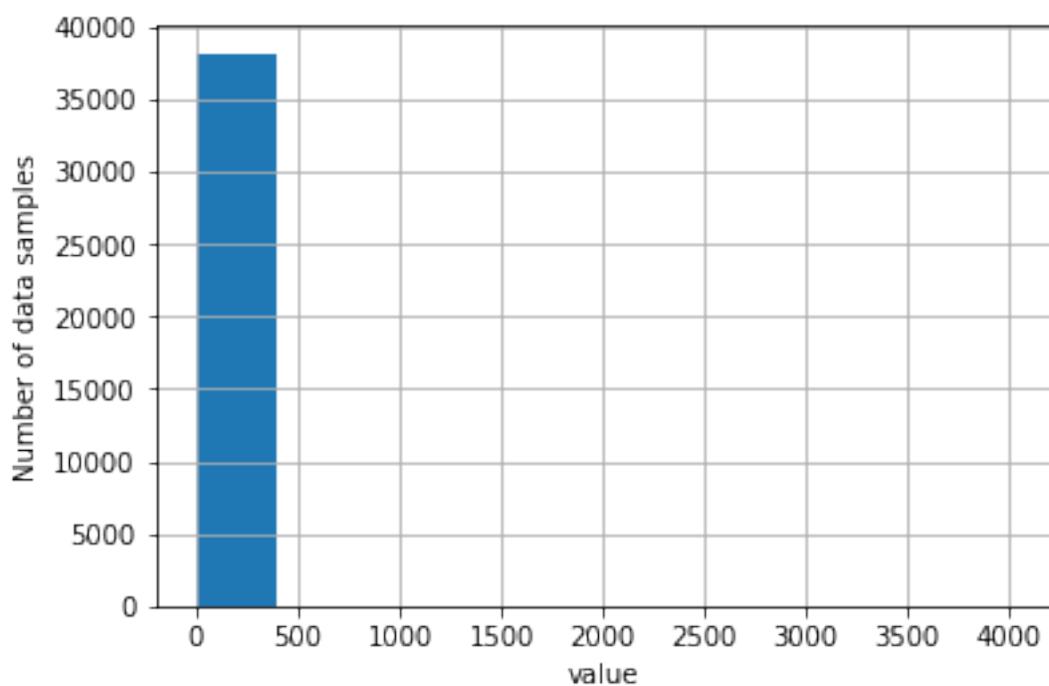
Distribution of data for disk_io_time host bb3localdomain instance mmcblk1boot0 type disk_io_t



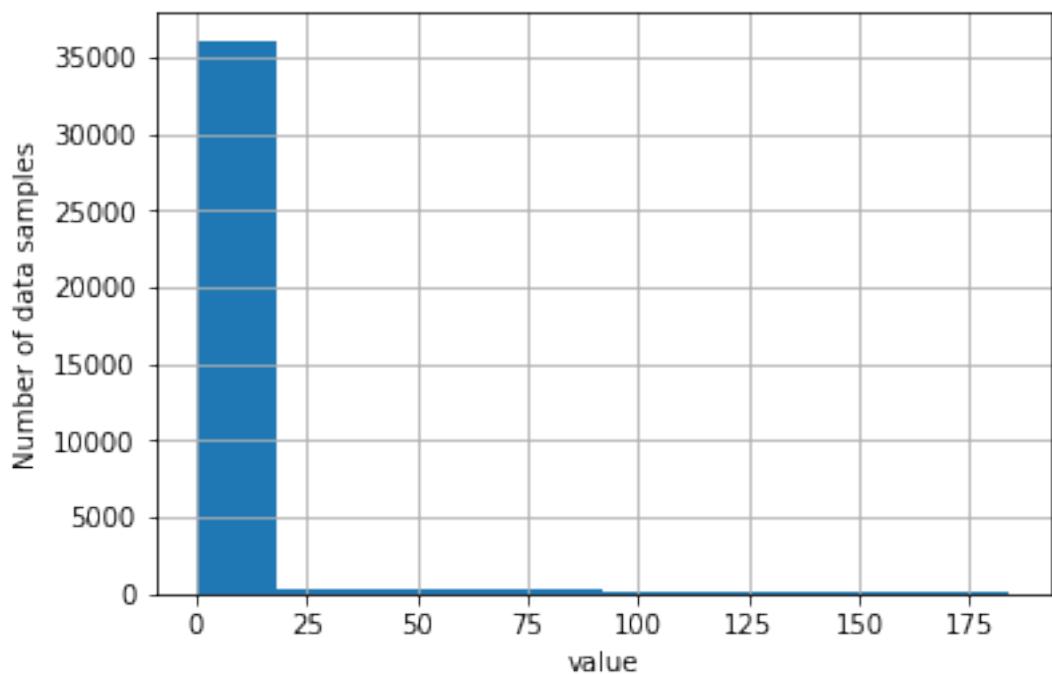
Distribution of data for disk_io_time host bb3localdomain instance mmcblk1boot1 type disk_io_time



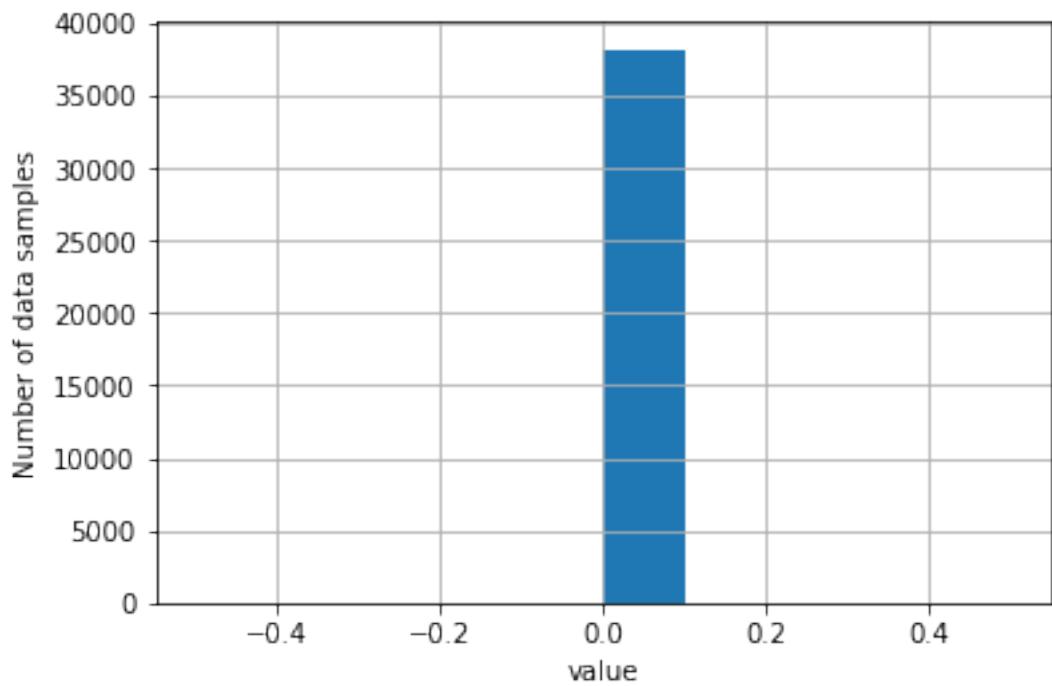
Distribution of data for disk_io_time host bb3localdomain instance mmcblk1p1 type disk_io_time



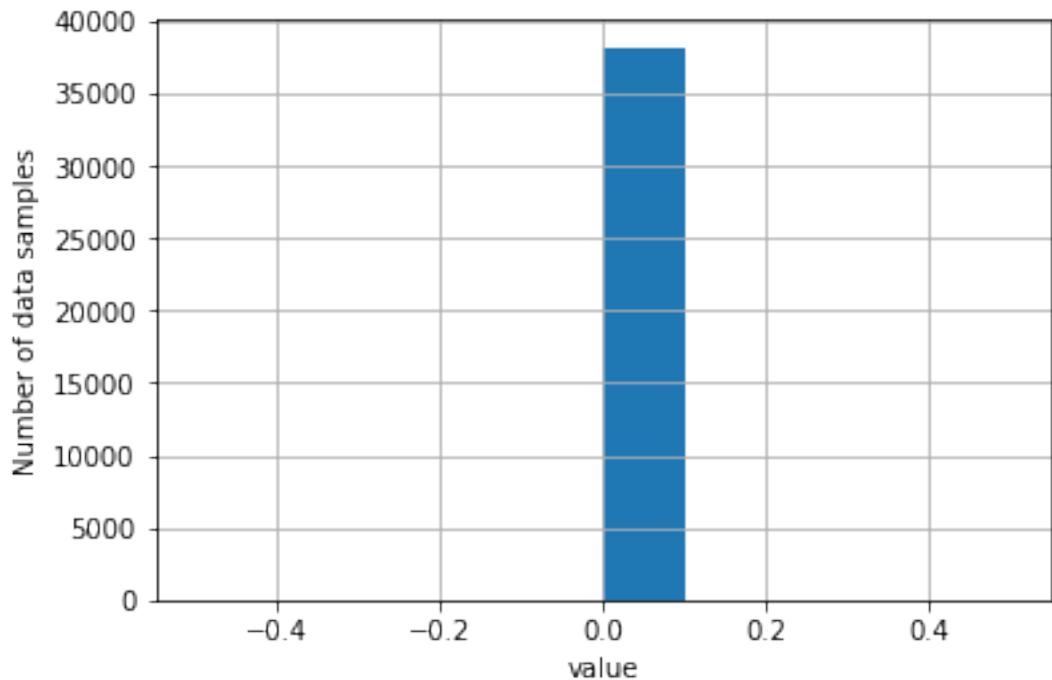
```
(38200, 29)
Index(['cpu_value host bb4localdomain type_instance idle',
       'cpu_value host bb4localdomain type_instance interrupt',
       'cpu_value host bb4localdomain type_instance nice',
       'cpu_value host bb4localdomain type_instance softirq',
       'cpu_value host bb4localdomain type_instance steal',
       'cpu_value host bb4localdomain type_instance system',
       'cpu_value host bb4localdomain type_instance user',
       'cpu_value host bb4localdomain type_instance wait',
       'interface_tx host bb4localdomain instance lo type if_dropped',
       'interface_tx host bb4localdomain instance lo type if_errors',
       'interface_tx host bb4localdomain instance lo type if_octets',
       'interface_tx host bb4localdomain instance lo type if_packets',
       'interface_tx host bb4localdomain instance wlan0 type if_dropped',
       'interface_tx host bb4localdomain instance wlan0 type if_errors',
       'interface_tx host bb4localdomain instance wlan0 type if_octets',
       'interface_tx host bb4localdomain instance wlan0 type if_packets',
       'interface_rx host bb4localdomain instance lo type if_dropped',
       'interface_rx host bb4localdomain instance lo type if_errors',
       'interface_rx host bb4localdomain instance lo type if_octets',
       'interface_rx host bb4localdomain instance lo type if_packets',
       'interface_rx host bb4localdomain instance wlan0 type if_dropped',
       'interface_rx host bb4localdomain instance wlan0 type if_errors',
       'interface_rx host bb4localdomain instance wlan0 type if_octets',
       'interface_rx host bb4localdomain instance wlan0 type if_packets',
       'contextswitch_value host bb4localdomain type contextswitch',
       'disk_io_time host bb4localdomain instance mmcblk1 type disk_io_time',
       'disk_io_time host bb4localdomain instance mmcblk1boot0 type disk_io_time',
       'disk_io_time host bb4localdomain instance mmcblk1boot1 type disk_io_time',
       'disk_io_time host bb4localdomain instance mmcblk1p1 type disk_io_time'],
      dtype='object')
Distribution of data for cpu_value host bb4localdomain type_instance idle
```



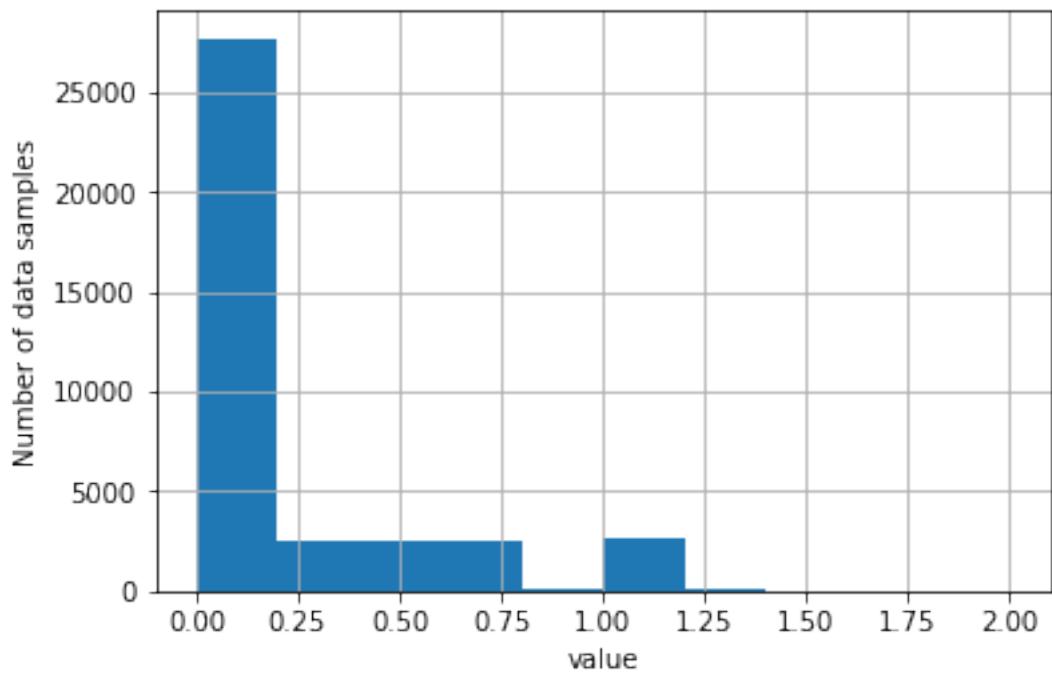
Distribution of data for cpu_value host bb4localdomain type_instance interrupt



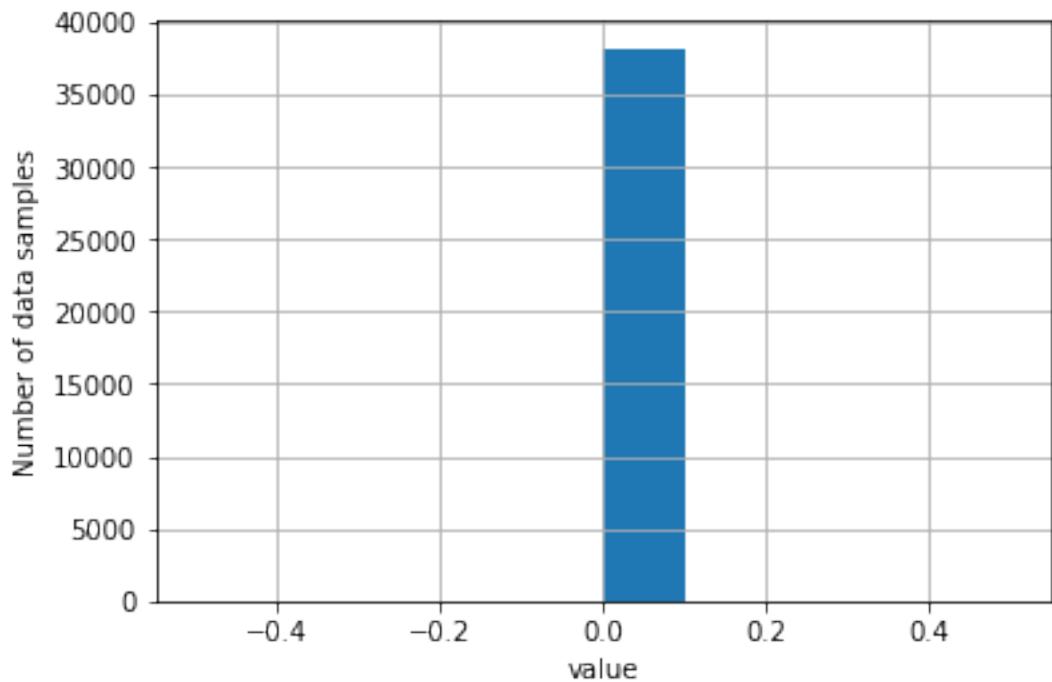
Distribution of data for cpu_value host bb4localdomain type_instance nice



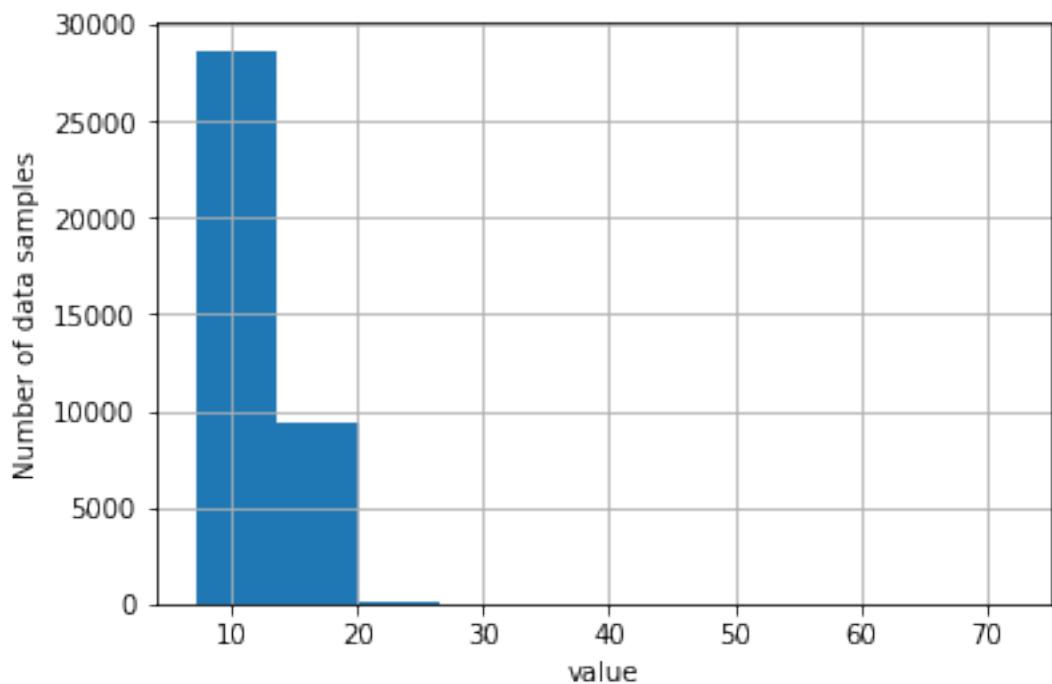
Distribution of data for cpu_value host bb4localdomain type_instance softirq



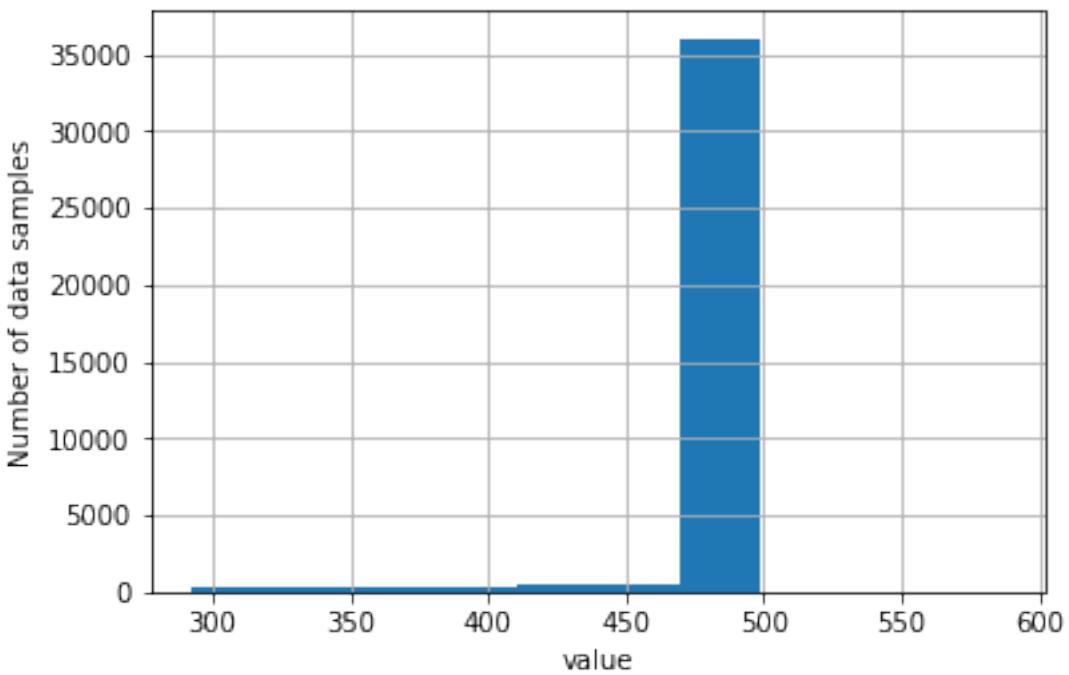
Distribution of data for cpu_value host bb4localdomain type_instance steal



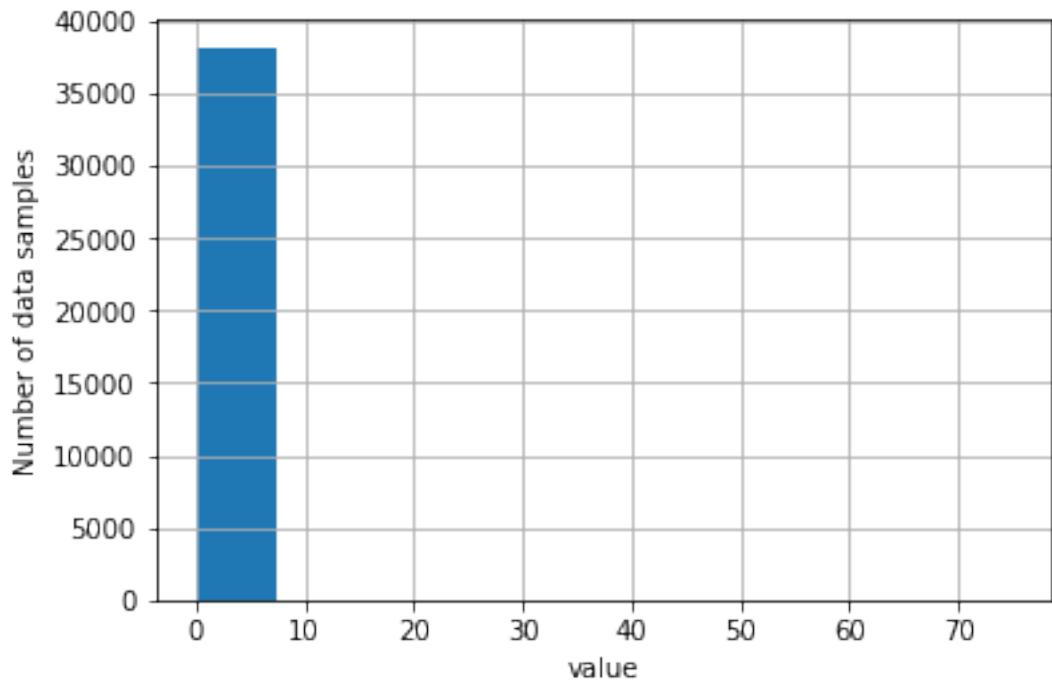
Distribution of data for cpu_value host bb4localdomain type_instance system



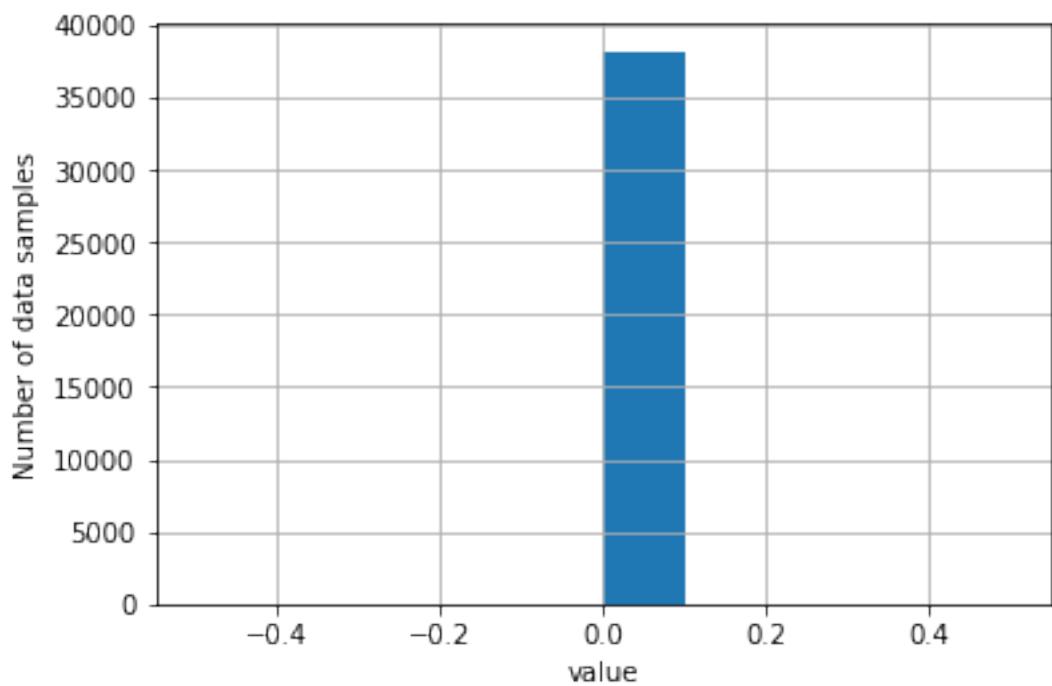
Distribution of data for cpu_value host bb4localdomain type_instance user



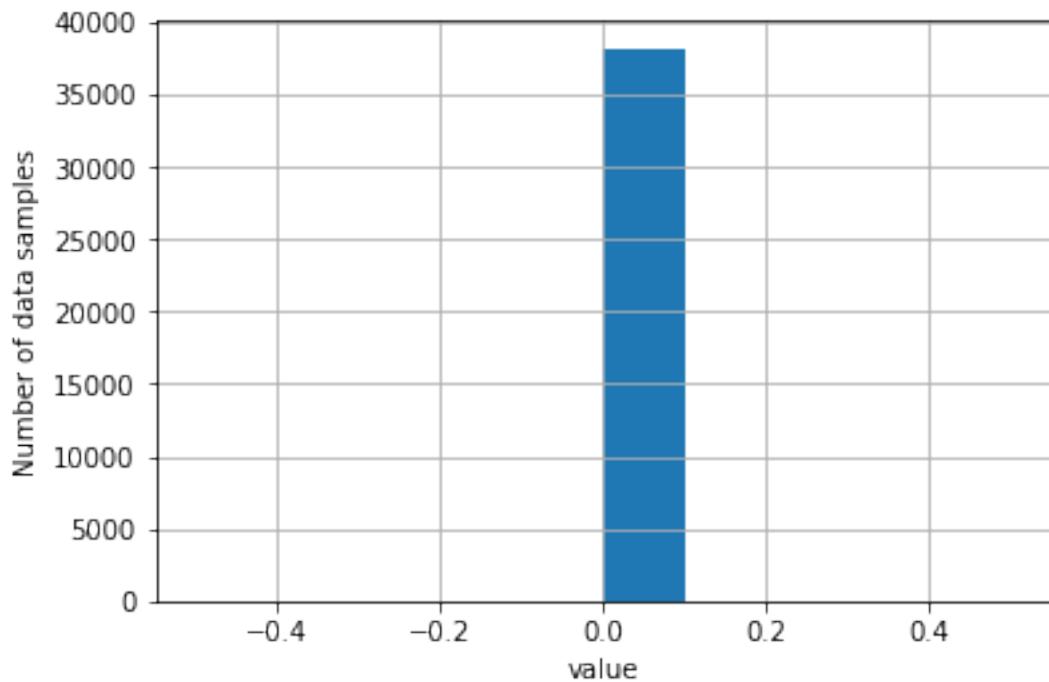
Distribution of data for cpu_value host bb4localdomain type_instance wait



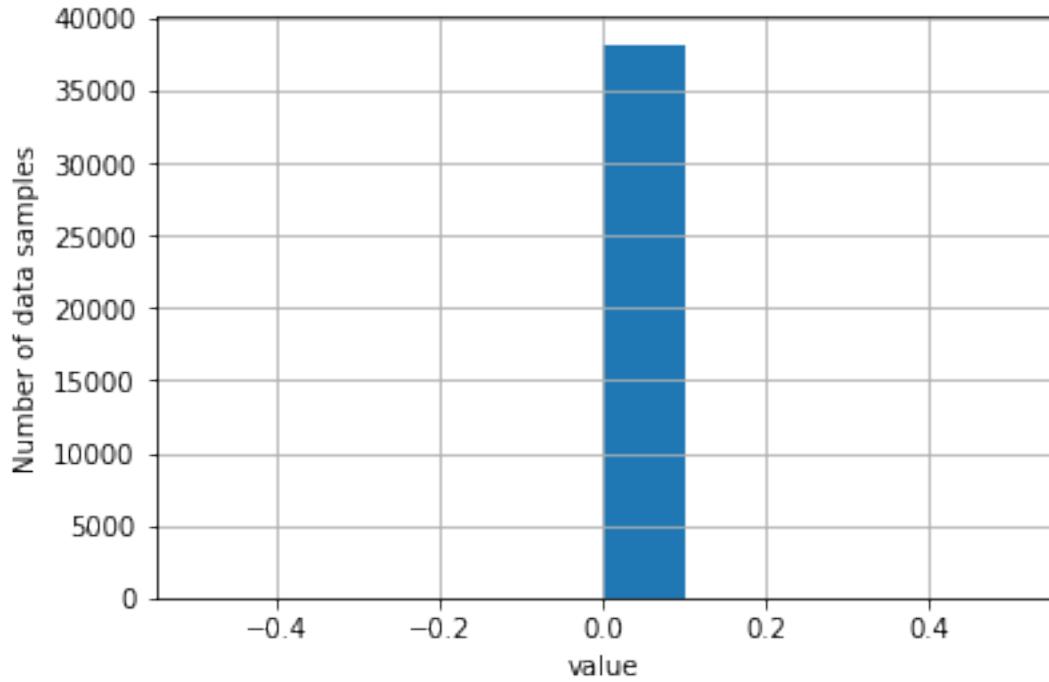
Distribution of data for interface_tx host bb4localdomain instance lo type if_dropped



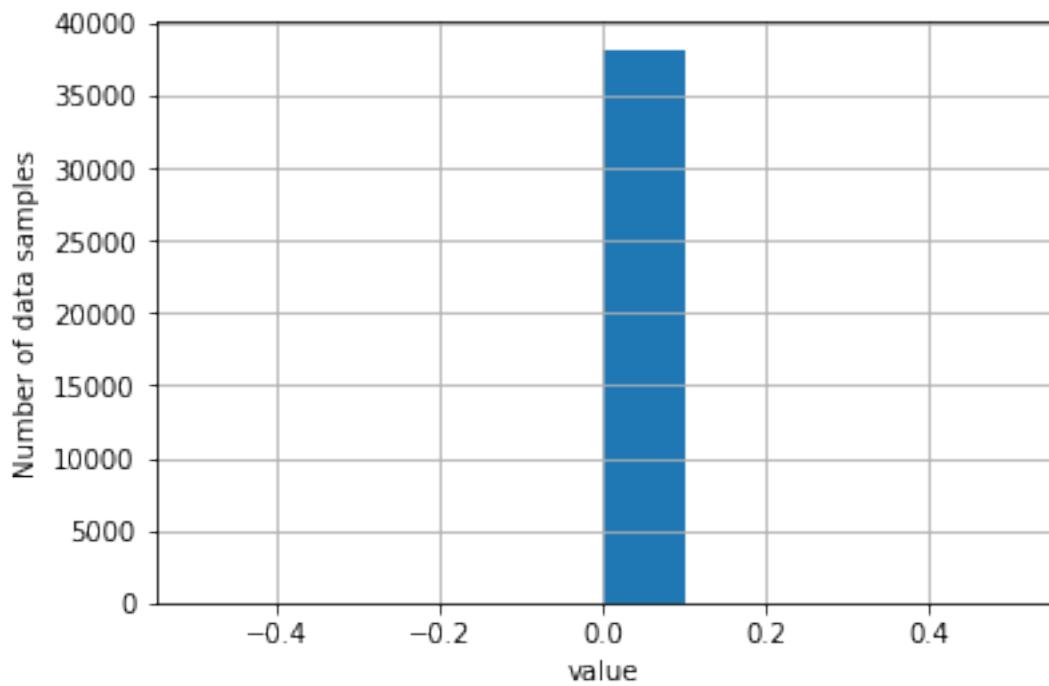
Distribution of data for interface_tx host bb4localdomain instance lo type if_errors



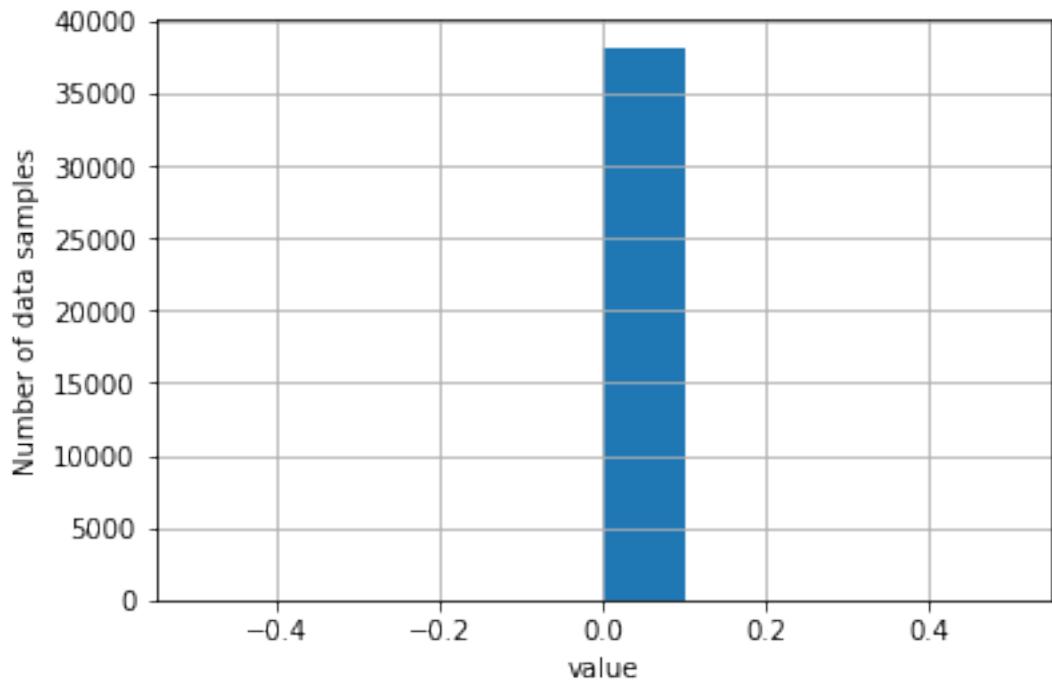
Distribution of data for interface_tx host bb4localdomain instance lo type if_octets



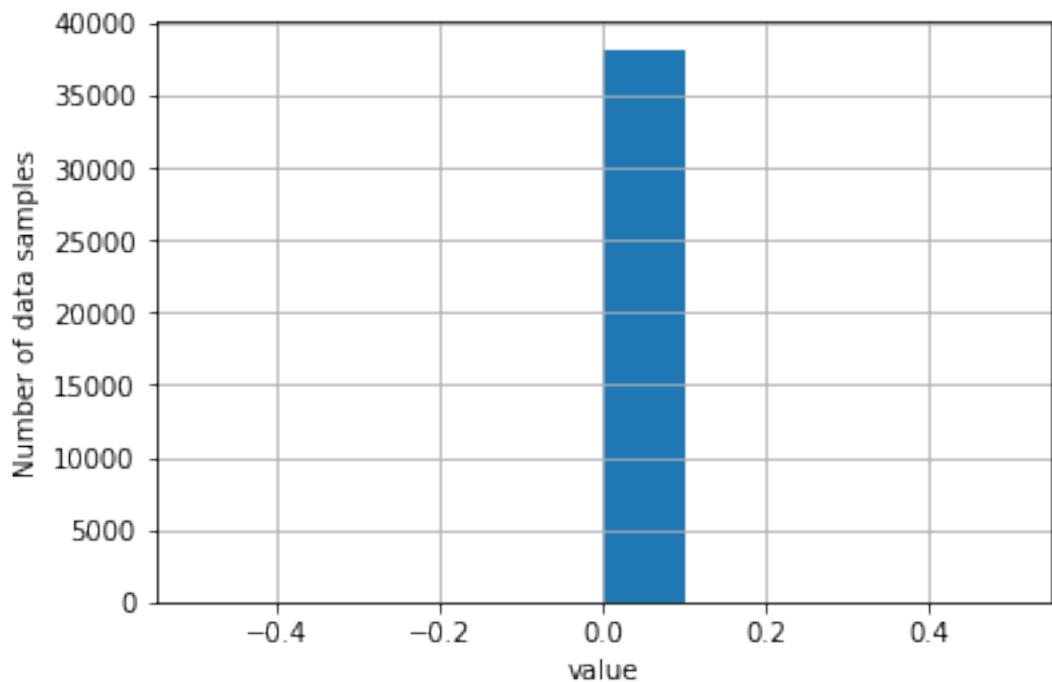
Distribution of data for interface_tx host bb4localdomain instance lo type if_packets



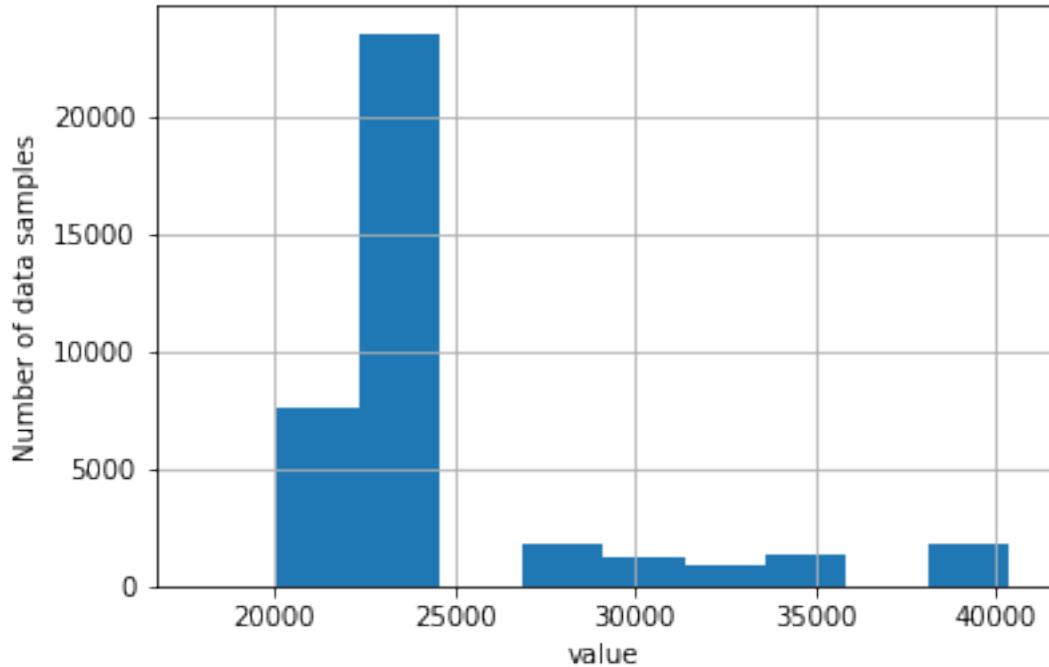
Distribution of data for interface_tx host bb4localdomain instance wlan0 type if_dropped



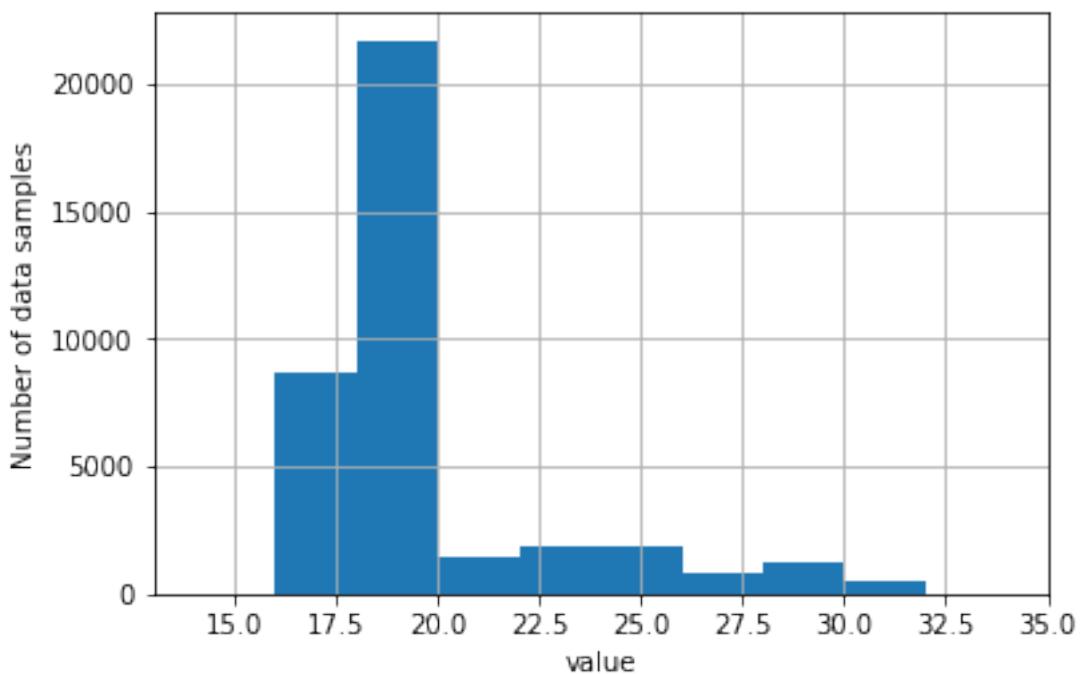
Distribution of data for interface_tx host bb4localdomain instance wlan0 type if_errors



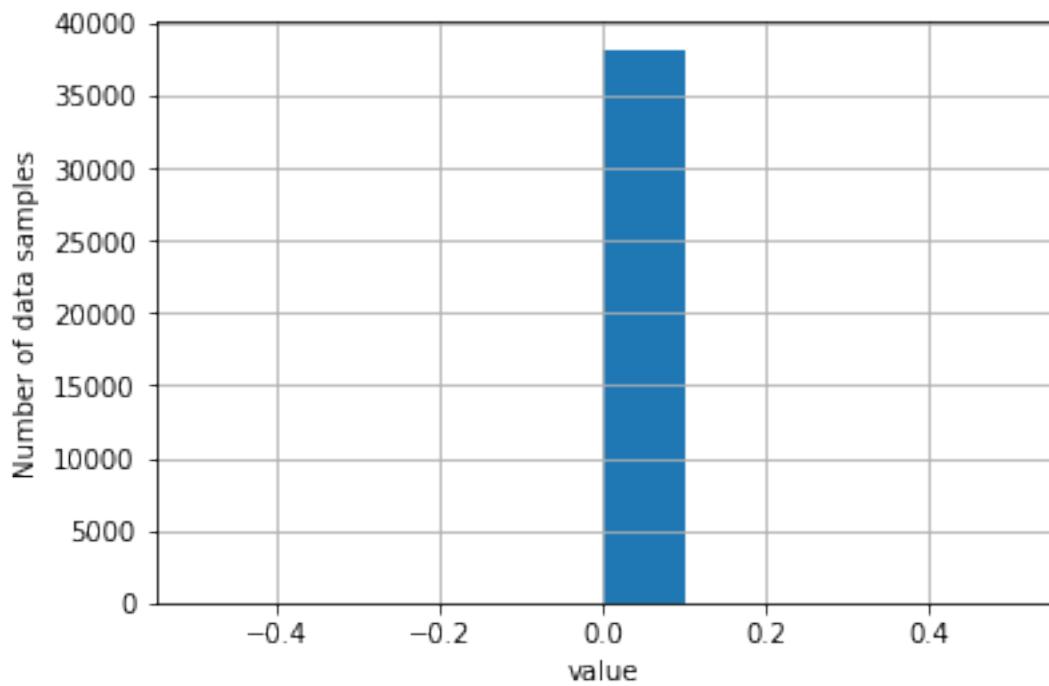
Distribution of data for interface_tx host bb4localdomain instance wlan0 type if_octets



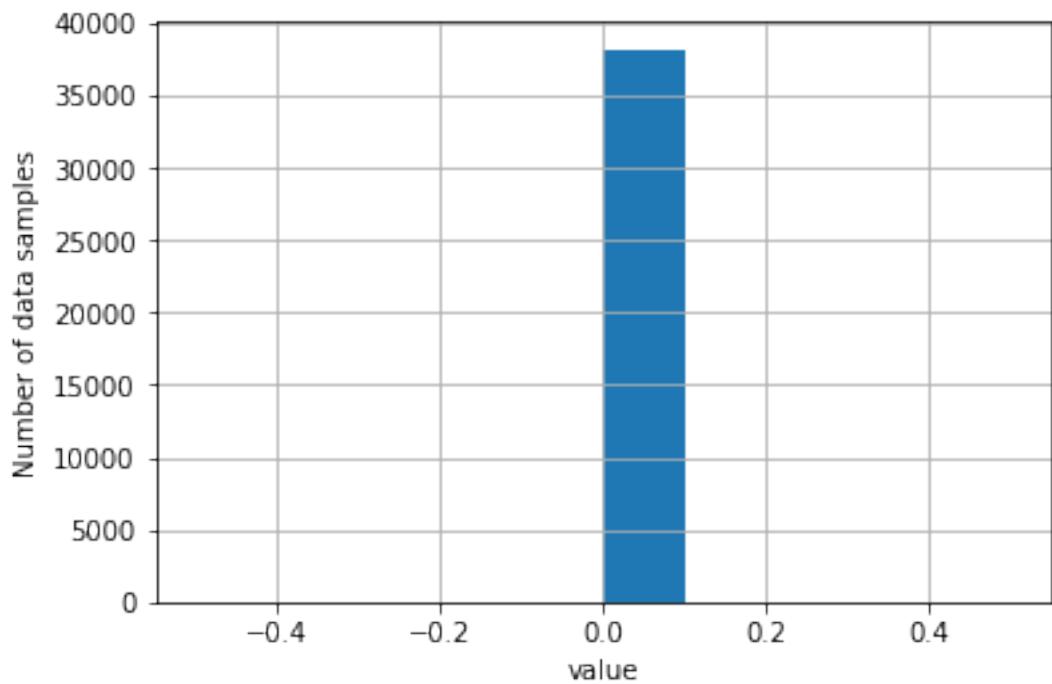
Distribution of data for interface_tx host bb4localdomain instance wlan0 type if_packets



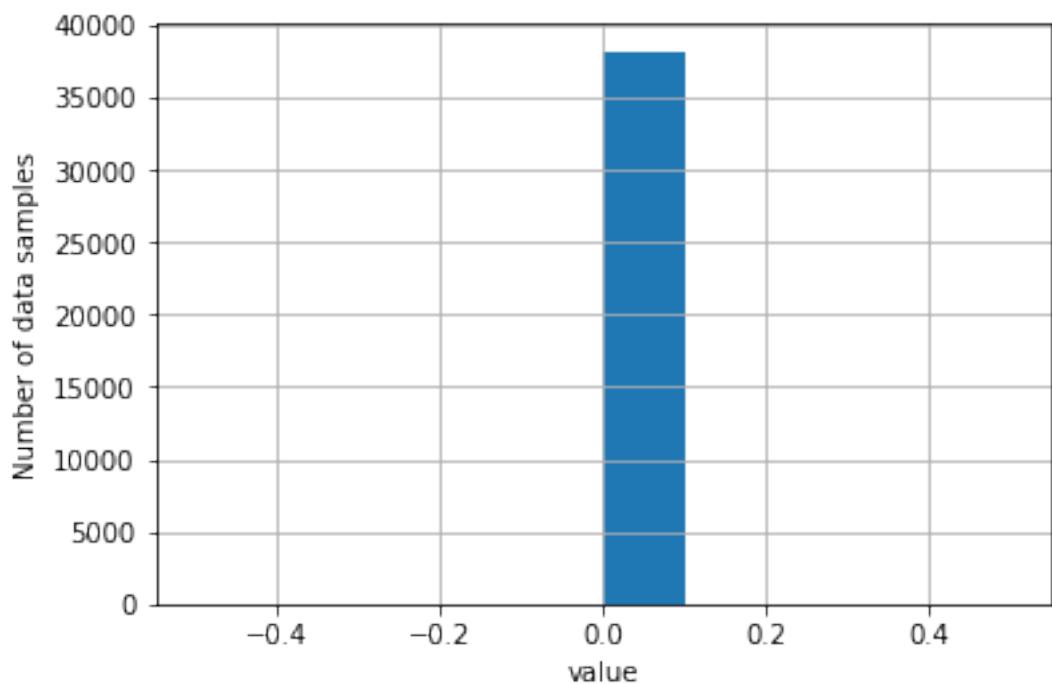
Distribution of data for interface_rx host bb4localdomain instance lo type if_dropped



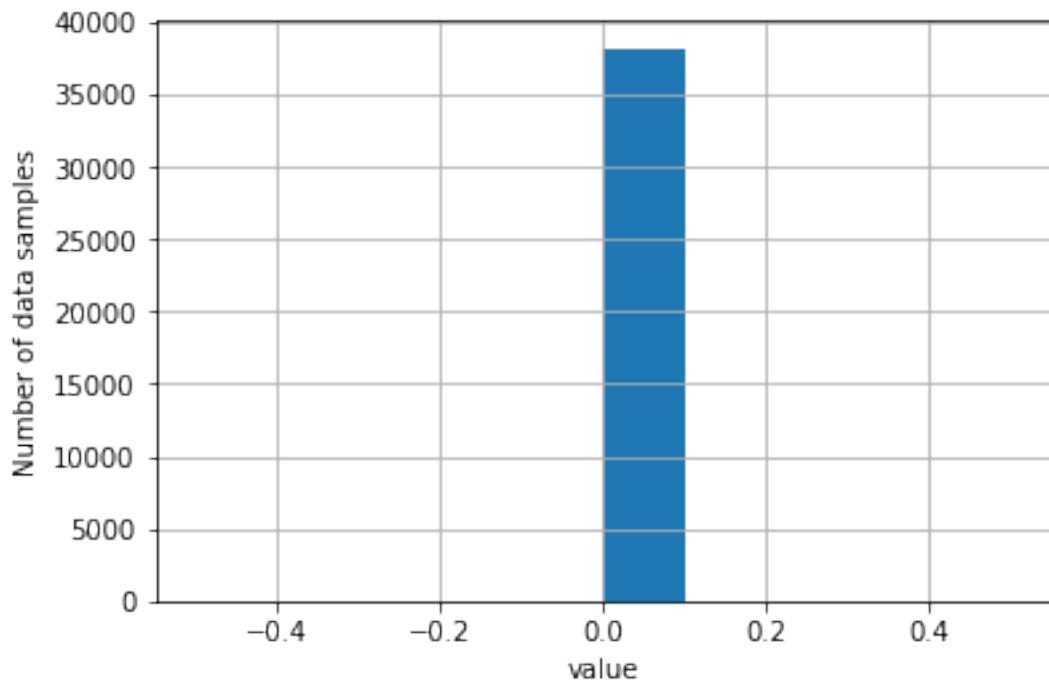
Distribution of data for interface_rx host bb4localdomain instance lo type if_errors



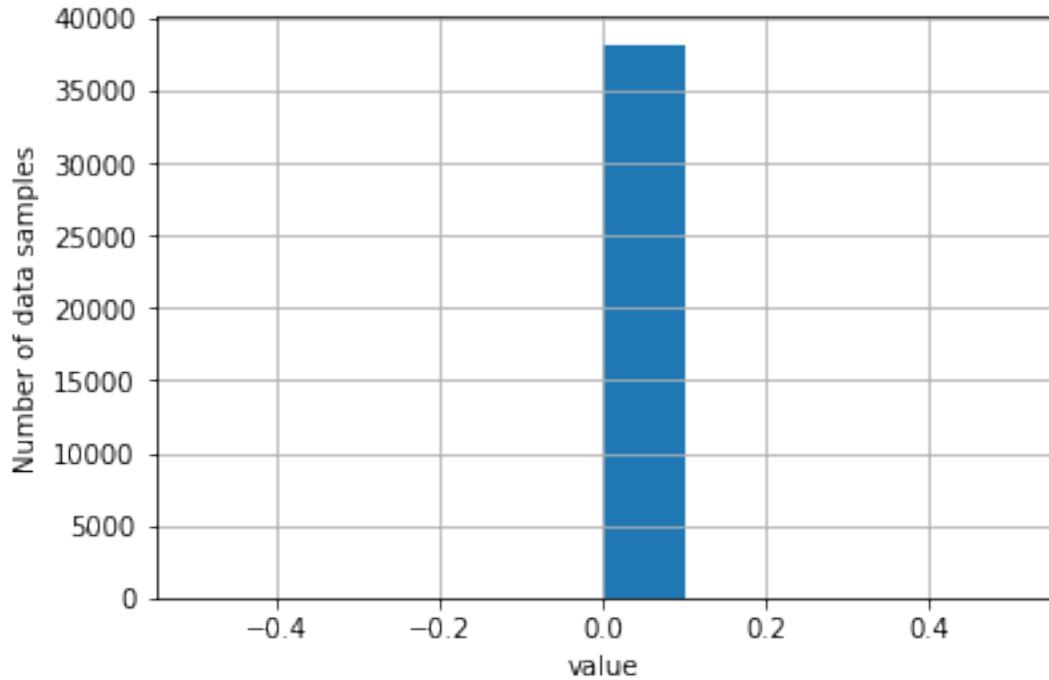
Distribution of data for interface_rx host bb4localdomain instance lo type if_octets



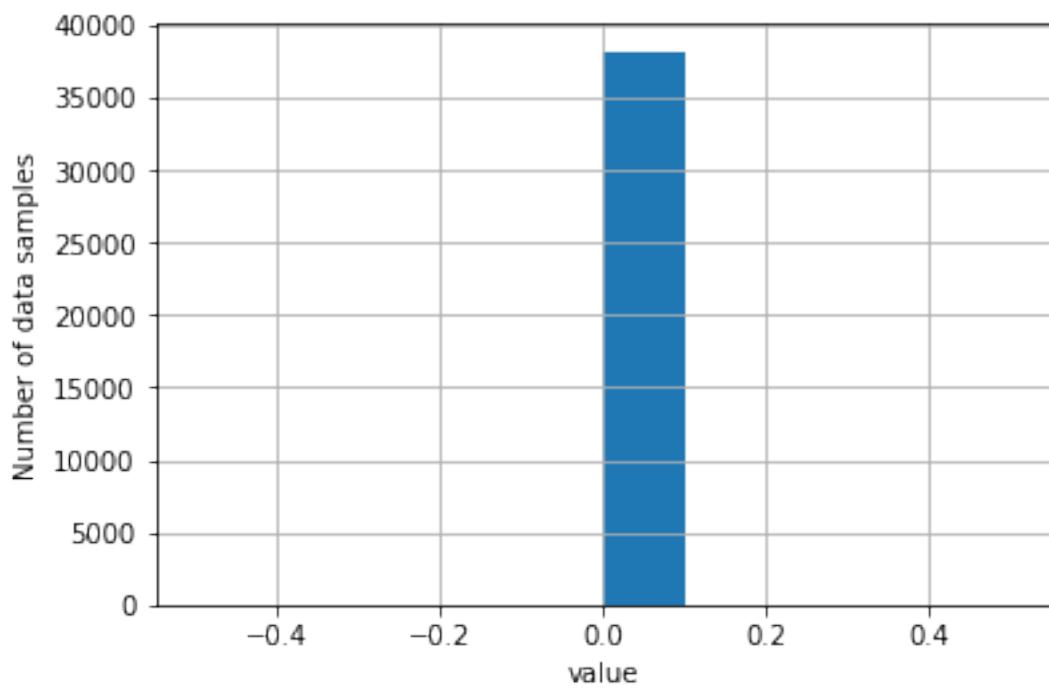
Distribution of data for interface_rx host bb4localdomain instance lo type if_packets



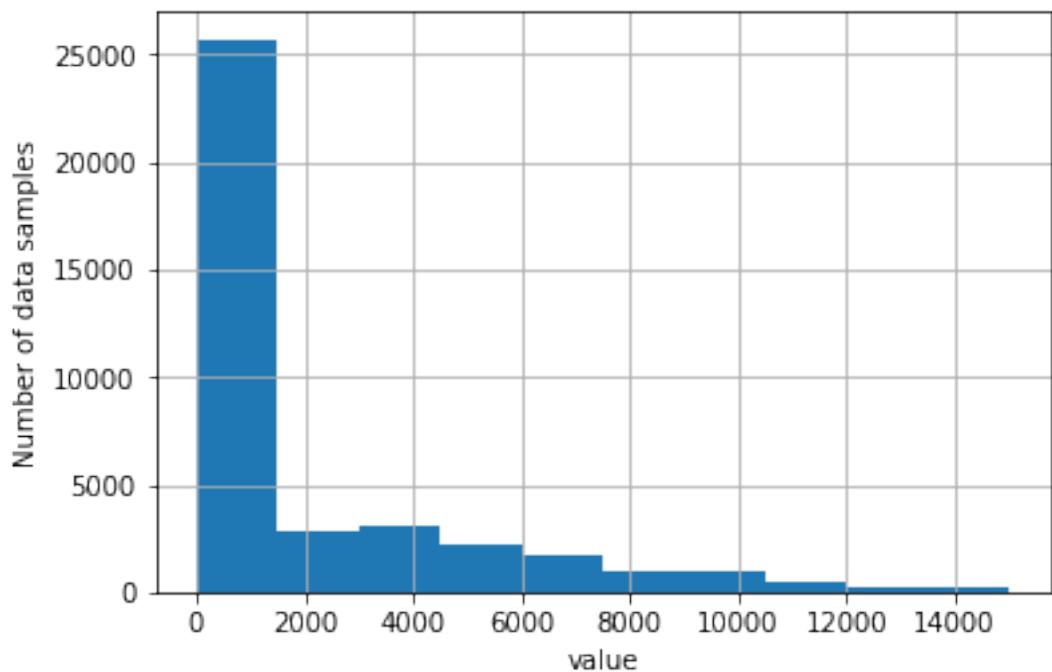
Distribution of data for interface_rx host bb4localdomain instance wlan0 type if_dropped



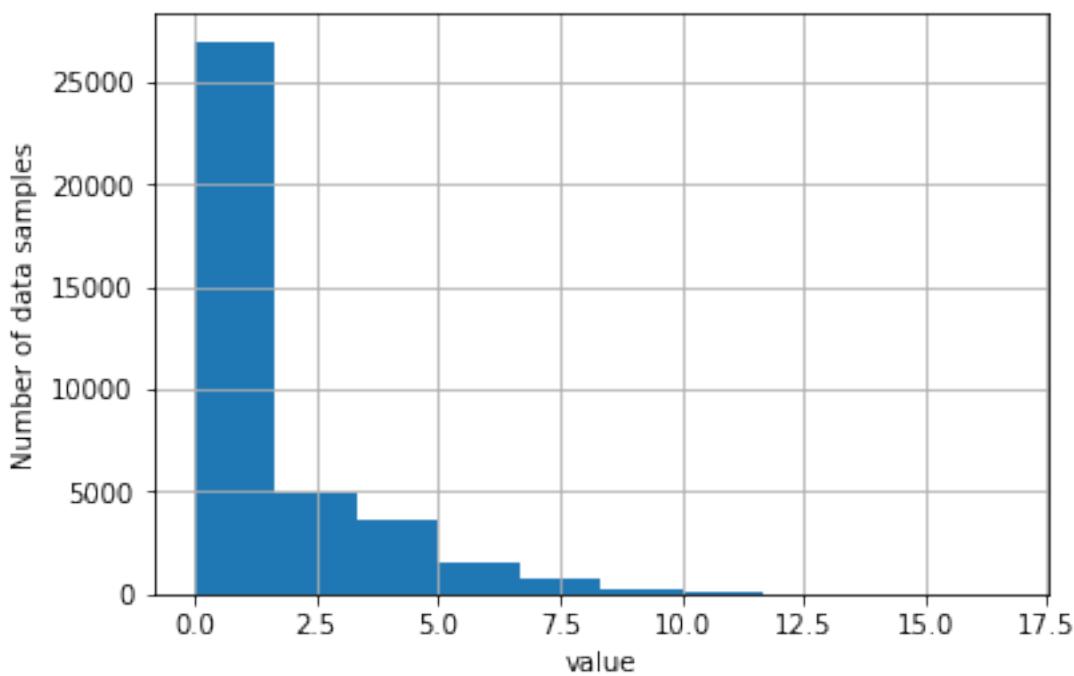
Distribution of data for interface_rx host bb4localdomain instance wlan0 type if_errors



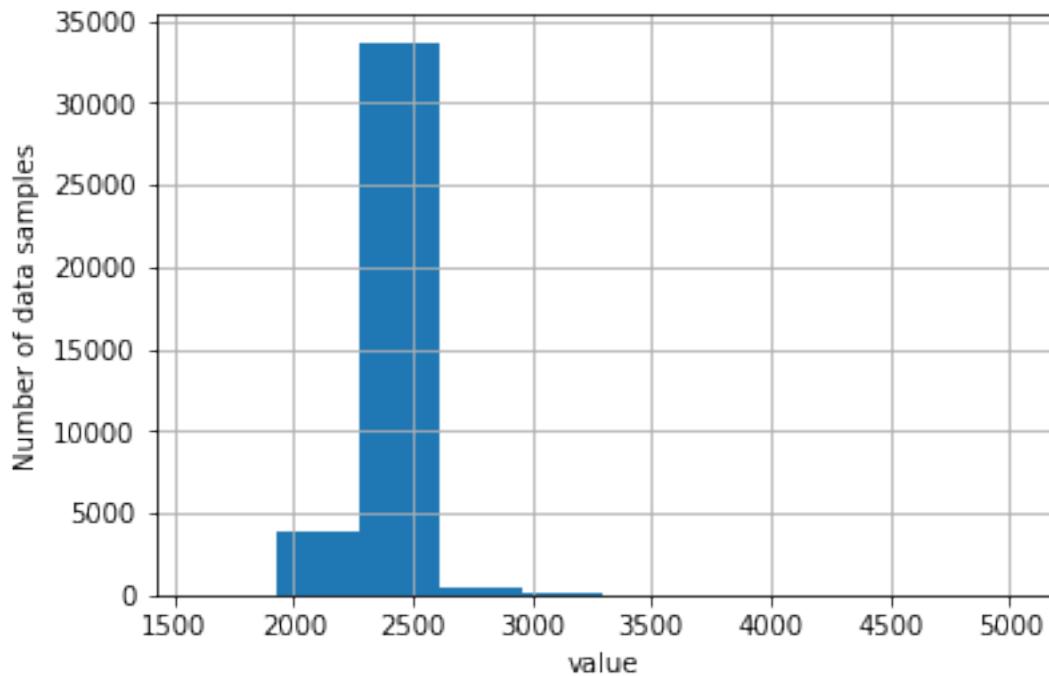
Distribution of data for interface_rx host bb4localdomain instance wlan0 type if_octets



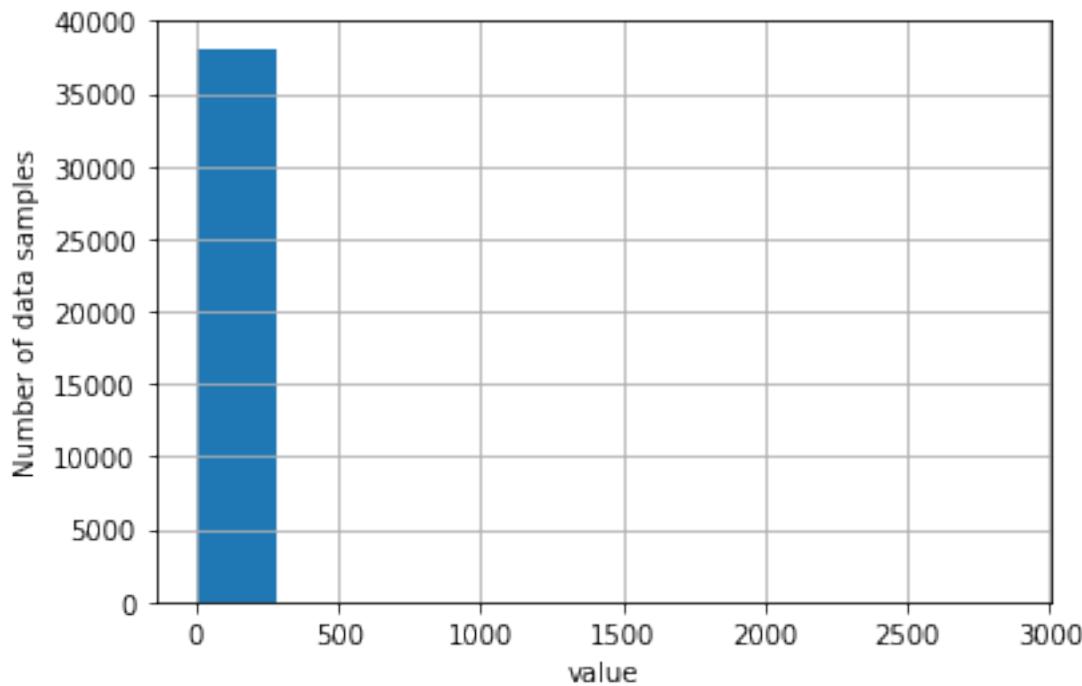
Distribution of data for interface_rx host bb4localdomain instance wlan0 type if_packets



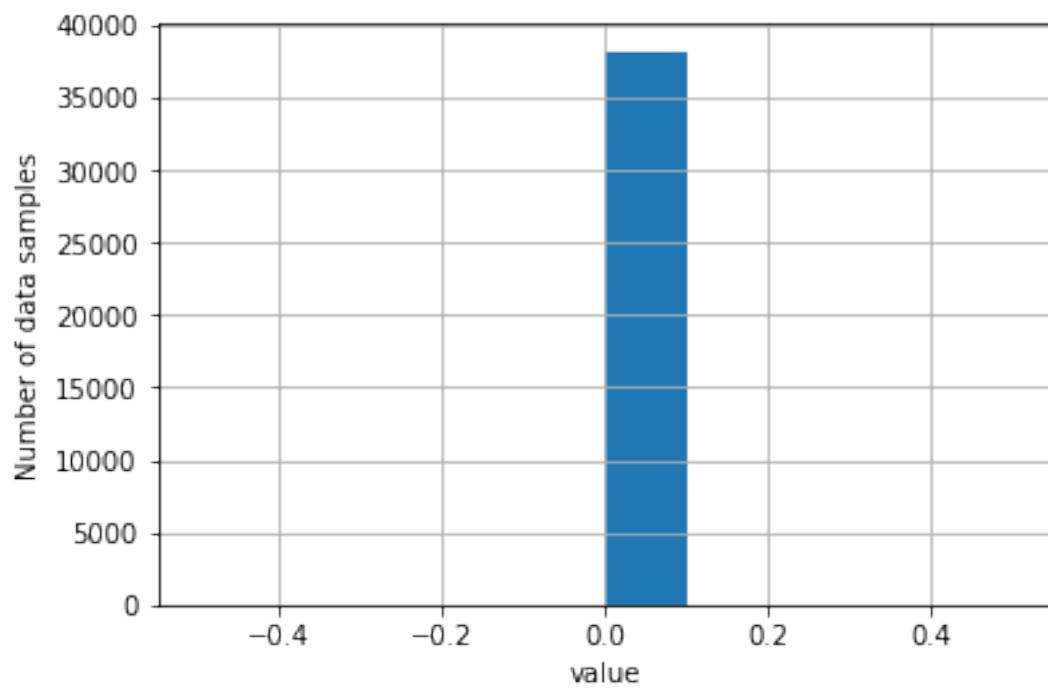
Distribution of data for contextswitch_value host bb4localdomain type contextswitch



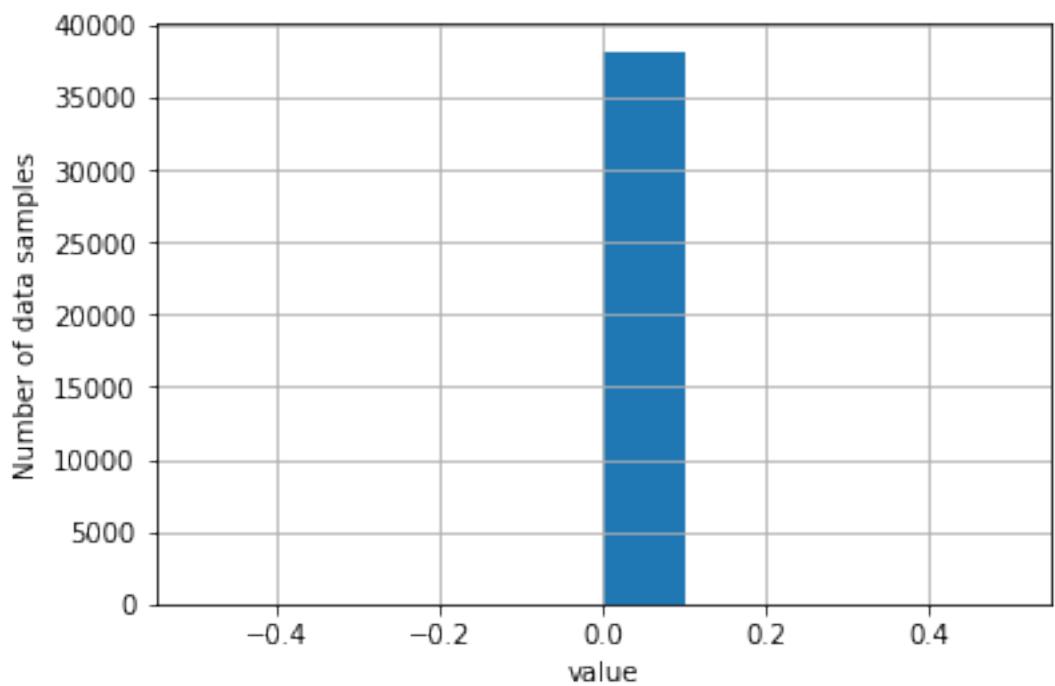
Distribution of data for disk_io_time host bb4localdomain instance mmcblk1 type disk_io_time



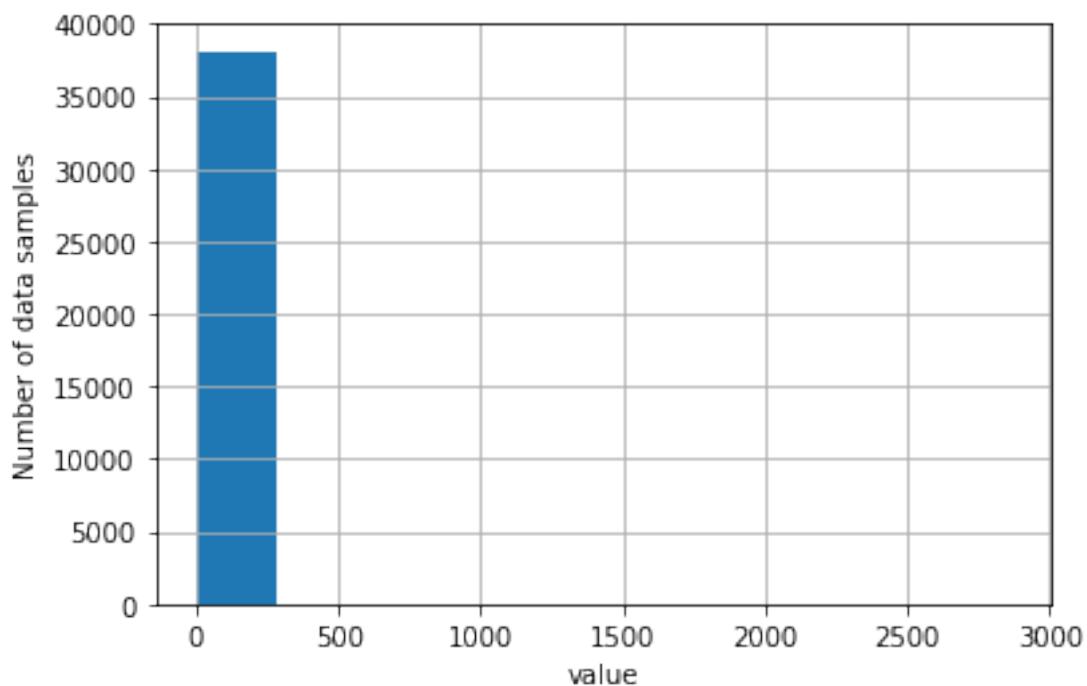
Distribution of data for disk_io_time host bb4localdomain instance mmcblk1boot0 type disk_io_t



Distribution of data for disk_io_time host bb4localdomain instance mmcblk1boot1 type disk_io_time



Distribution of data for disk_io_time host bb4localdomain instance mmcblk1p1 type disk_io_time



1.7 Get data

```
In [20]: data_matrices = []
```

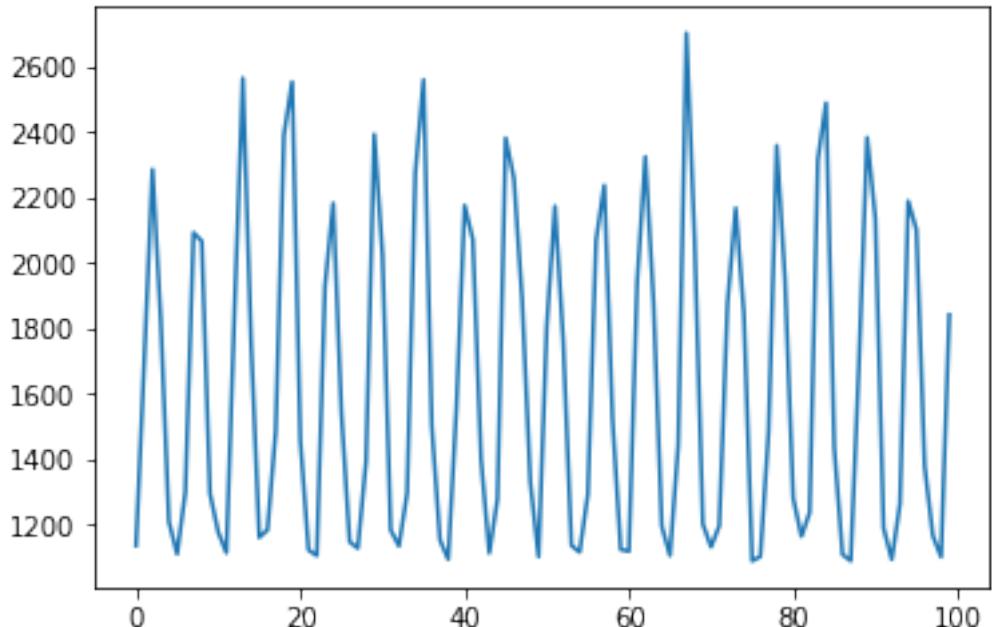
```
for i in range(1,5):
    data_matrices.append(node[i].as_matrix())

data = numpy.vstack(data_matrices)
```

```
In [21]: data.shape
```

```
Out[21]: (152800, 29)
```

```
In [22]: tdata = data[:,24]
plotter.plot(tdata.T[:100])
plotter.show()
print(data.shape)
```



```
(152800, 29)
```

```
In [23]: #data = data[:,24]
```

1.8 Prepare scaler

```
In [24]: from sklearn.preprocessing import MinMaxScaler
         from sklearn.preprocessing import StandardScaler
         from sklearn.preprocessing import RobustScaler
         scaler = MinMaxScaler()

In [25]: scaler.fit(data)
         del data
```

1.9 Correlation measurement

2 Prediction

```
In [26]: for i in range(len(data_matrices)):

    transformed = scaler.transform(data_matrices[i])
    data_matrices[i] = transformed

    X = numpy.stack(data_matrices[:-1], axis=1)
    print(data_matrices[3].shape)
```

(38200, 29)

```
In [27]: print(X.shape)
LEN = X.shape[0]
SPLIT = int(0.85*LEN)

train_X = X[:SPLIT,:,:]
val_X = X[SPLIT:SPLIT+1000,:,:]
test_X = X[SPLIT+1000,:,:]
diff_app_X = data_matrices[3][SPLIT+1000,:,:]
```

(38200, 3, 29)

```
In [28]: X = train_X
X = numpy.transpose(X, (1, 0, 2))
val_X = numpy.transpose(val_X, (1, 0, 2))
test_X = numpy.transpose(test_X, (1, 0, 2))
```

```

ano_test = test_X[0]
norm_test = test_X[2]
diff_app_test = diff_app_X
app_change_test = numpy.vstack([norm_test[:3500,:], diff_app_test[3500:,:]])
net_flood_test = numpy.copy(norm_test)
net_flood_test[3000:,22] = 1.0
net_flood_test[3000:,23] = 1.0

print(f"Validation shape is {val_X.shape}")
print(f"Anomaly test shape is {ano_test.shape}")
print(f"Normal test shape is {norm_test.shape}")
print(f"Different app test shape is {diff_app_test.shape}")
print(f"App change test shape is {app_change_test.shape}")
print(f"Net flood test shape is {net_flood_test.shape}")

Validation shape is (3, 1000, 29)
Anomaly test shape is (4730, 29)
Normal test shape is (4730, 29)
Different app test shape is (4730, 29)
App change test shape is (4730, 29)
Net flood test shape is (4730, 29)

```

```

In [29]: def flat_generator(X, tsteps = 5, ravel=1):
    i = 0

    while True:
        batch_X = X[:,i:i+tsteps,:]
        batch_y = X[:,i+tsteps,:]

        if ravel:
            batch_X = batch_X.reshape((batch_X.shape[0], -1))
        #print(batch_X.shape)
        #print(batch_y.shape)

        yield batch_X, batch_y

        i += 1
        if i > (X.shape[1] - tsteps - 1):
            i = 0
            continue

```

2.1 Flat models

```

In [30]: from keras.models import Model
        from keras.layers import Dense, Input, Dropout, GRU
        from keras.callbacks import EarlyStopping

```

Using TensorFlow backend.

```
In [31]: def train(model, tgen, vgen, name="none"):
    estopper = EarlyStopping(patience=15, min_delta=0.0001)
    history = model.fit_generator(tgen, steps_per_epoch=1000, epochs=10000, callbacks=[estopper])
    plotter.plot(history.history['loss'],label='train')
    plotter.plot(history.history['val_loss'],label='validation')
    plotter.legend()
    plotter.xlim(0,150)
    plotter.xlabel("Epochs")
    plotter.ylabel("Error")
    plotter.savefig(f"{name}_train.png", dpi=1000)
    plotter.show()
    print(f"Training loss for final epoch is {history.history['loss'][-1]}")
    print(f"Validation loss for final epoch is {history.history['val_loss'][-1]")

In [32]: def plot_running_stats(error, name="none", window_size=5, bounds=None, qq=0):
    error = numpy.array(error)
    window = numpy.ones(window_size)/window_size
    running_mean = numpy.convolve(error, window, mode="same")
    running_sigma = pandas.Series(error).rolling(window=window_size, center=True).std()
    difference = 3.0 * running_sigma

    upper = running_mean + difference
    lower = running_mean - difference

    if bounds == None:
        global_mean = numpy.mean(error) * numpy.ones(error.shape[0])
        global_sigma = numpy.std(error) * numpy.ones(error.shape[0])
        bound = (5.0 * global_sigma) + global_mean

    else:
        global_mean = bounds[0]
        bound = bounds[1]

    anomaly = ((error > bound) * error)
    anomaly = numpy.array([float('nan') if x == 0.0 else x for x in anomaly])

    if qq:
        probplot(error,dist="norm", plot=plotter)
        plotter.legend()
        plotter.savefig(f"{name}_qq.png",dpi=1000)
        plotter.show()

    fig = plotter.figure()
    plotter.plot(error, 'g-', alpha=0.4, label="Error", linewidth=0.5)
```

```

plotter.plot(global_mean, 'r-', alpha=0.9, label="Mean", linewidth=0.5)
#plotter.plot(upper,'b-', alpha=0.2, label="Upper Bound", linewidth=0.5)
plotter.plot(bound,'b-', alpha=0.2, label="Bound", linewidth=0.5)
plotter.plot(anomaly,'r.', alpha=0.5, label="Anomaly")
plotter.legend()
plotter.ylim(0,0.2)
plotter.xlabel("time")
plotter.ylabel("Error")
plotter.draw()
fig.savefig(f"{name}_truetestloss.png", dpi=1000)
plotter.show()
fig.clf()
plotter.clf()
plotter.close()
error = numpy.array(error)
print(f"The mean error for {name} is {numpy.mean(error)} for length {error.shape[0]}")

return (global_mean, bound)

```

```

In [33]: def data_test(model, dataset=test_X[0], ravel=1, write=0, name="none", window=5, bounds=[0,1]):
    test_gen = flat_generator(numpy.array([dataset]), window,0)
    error = []
    targets = []
    preds = []
    for i in range(dataset.shape[0]-(window+1)):
        _input,target = next(test_gen)
        targets.append(target.squeeze())
        if ravel:
            _input = _input.ravel()[:,numpy.newaxis].T

        pred = model.predict(_input)
        #print(target.shape)
        #print(pred.shape)
        preds.append(pred.squeeze())
        error.append(mean_absolute_error(y_pred=pred, y_true=target))

    targets = numpy.vstack(targets)
    preds = numpy.vstack(preds)
    return plot_running_stats(error, name=name, window_size=window, bounds=bounds, qq=True)

    #print(error)

```

```

In [34]: def gen_test(model, dataset=test_X[0], ravel=1, write=0, name="none"):
    test_gen = flat_generator(numpy.array([dataset]), Timesteps,0)
    error = []
    targets = []
    preds = []
    for i in range(2000):

```

```

        _input,target = next(test_gen)

        if i != 0:
            #print(_input.shape)
            _input = _input.squeeze()[1:,:]
            #print(_input.shape)
            _input = numpy.append(pred,_input, axis=0)[numpy.newaxis,:,:]
            #print(_input.shape)

        targets.append(target.squeeze())
        if ravel:
            _input = _input.ravel()[:,numpy.newaxis].T

        pred = model.predict(_input)
        #print(target.shape)
        #print(pred.shape)
        preds.append(pred.squeeze())
        error.append(mean_absolute_error(y_pred=pred, y_true=target))

    targets = numpy.vstack(targets)
    preds = numpy.vstack(preds)

    plotter.plot(error, 'g-', alpha=0.5)
    plotter.ylim(0,0.2)
    plotter.xlabel("time")
    plotter.ylabel("Error")
    plotter.savefig(f"{name}_testloss.png")
    plotter.show()
    error = numpy.array(error)
    print(numpy.mean(error))
    plotter.boxplot(error)
    plotter.ylim(0,0.2)
    plotter.xlabel("time")
    plotter.ylabel("Error")
    plotter.savefig(f"{name}_boxplot.png")
    plotter.show()
    if write:
        numpy.savetxt('loss.txt', numpy.array(error))
        true_test(model,dataset,ravel=ravel,name=name)
        #print(error)

```

```

In [35]: def test(model, ravel=1, name="none", window=20):
    print(f"----- Beginning tests for {name} -----")
    print(f"Testing on normal data.")
    bounds = data_test(model, dataset=norm_test, ravel=ravel, name=(name+"_normal_"),
    print(f"Testing on anomaly data.")
    data_test(model, dataset=ano_test, ravel=ravel, name=(name+"_anomaly_"), window=window)
    print(f"Testing on different app data.")

```

```

data_test(model, dataset=diff_app_test, ravel=ravel, name=(name+"_diff_app_"), window=window)
print(f"Testing on App change synthetic data.")
data_test(model, dataset=app_change_test, ravel=ravel, name=(name+"_app_change_"))
print(f"Testing on Net flood synthetic data.")
data_test(model, dataset=net_flood_test, ravel=ravel, name=(name+"_net_flood_"))
print("=="*20)
print("\r\n\r\n")

```

2.1.1 Linear Regression

2 steps

```

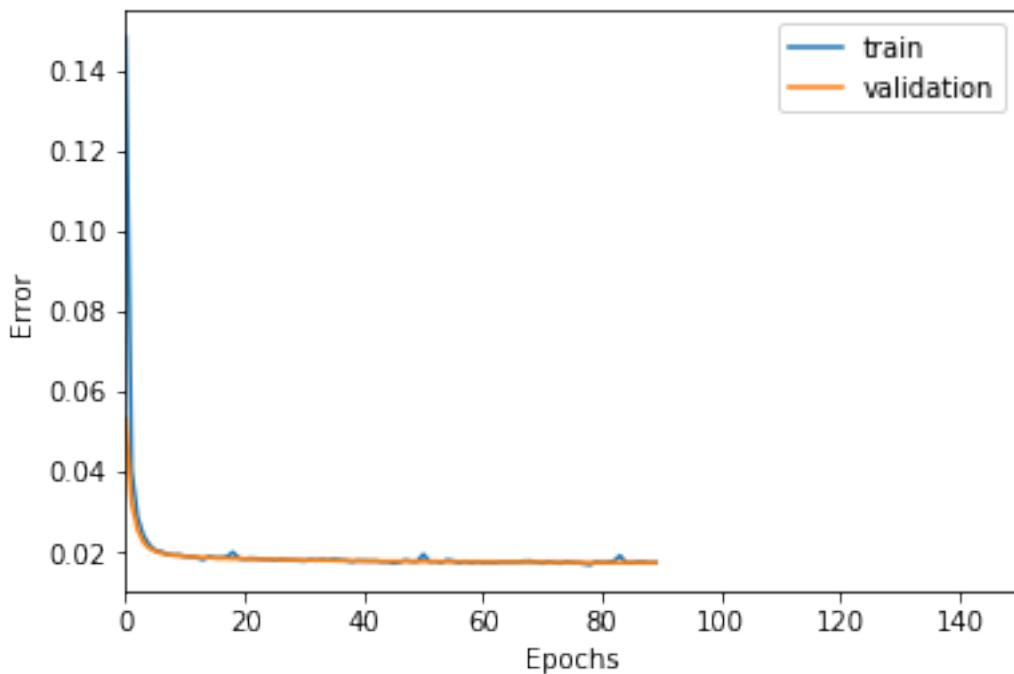
In [36]: TIMESTEPS = 2
        DIM = 29
        tgen = flat_generator(X, TIMESTEPS)
        vgen = flat_generator(val_X, TIMESTEPS)
        name = "lin2"

In [37]: input_layer = Input(shape=(TIMESTEPS*DIM,))
        output = Dense(DIM, activation='sigmoid')(input_layer)

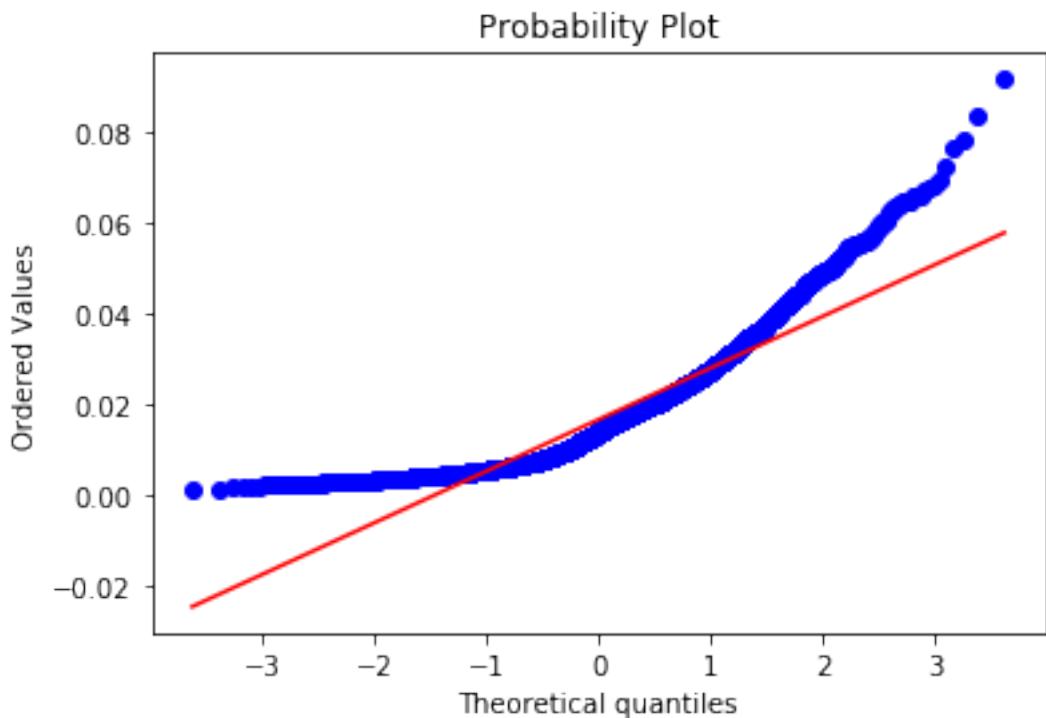
In [38]: model = Model(input_layer, output)
        model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

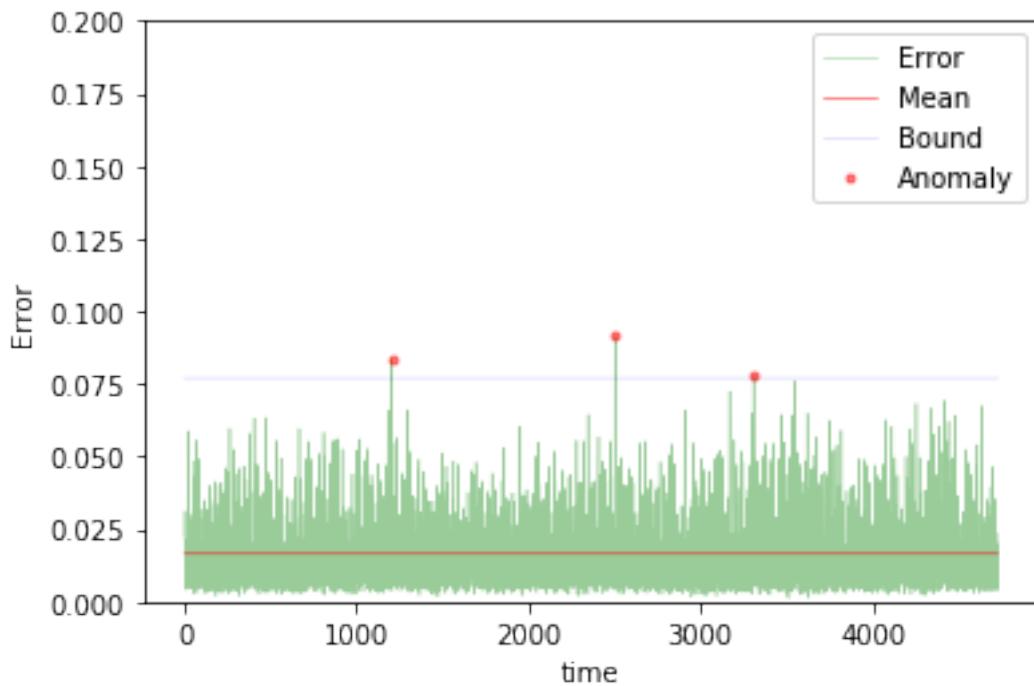
In [39]: train(model, tgen, vgen, name=name)
        test(model, name=name, window=TIMESTEPS)

```

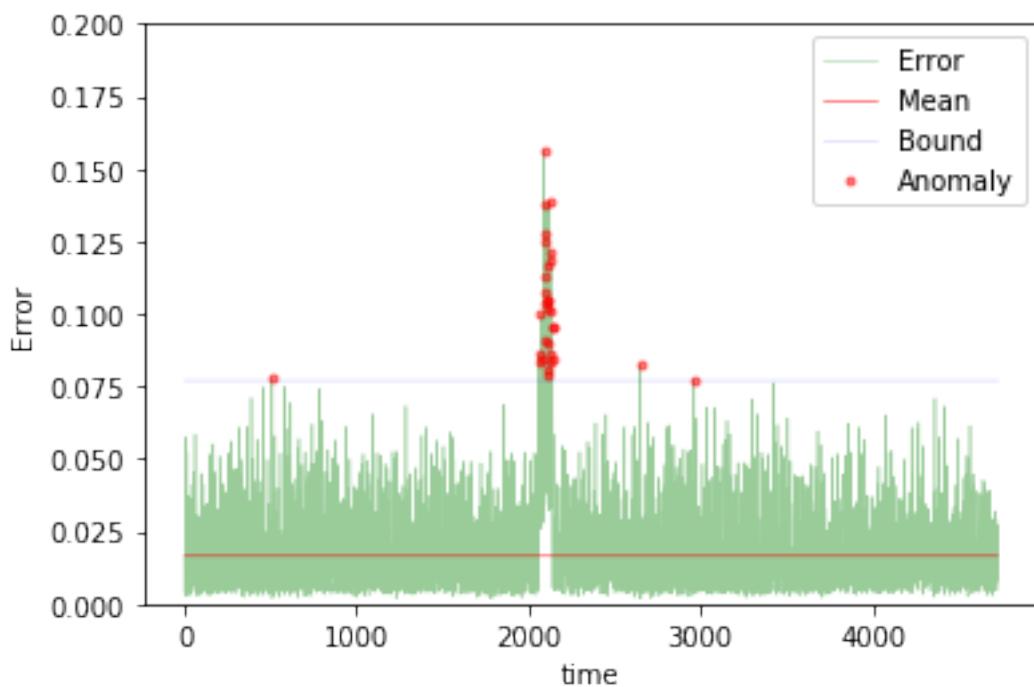


Training loss for final epoch is 0.01738236347469501
Validation loss for final epoch is 0.017289075288921596
----- Beginning tests for lin2 -----
Testing on normal data.

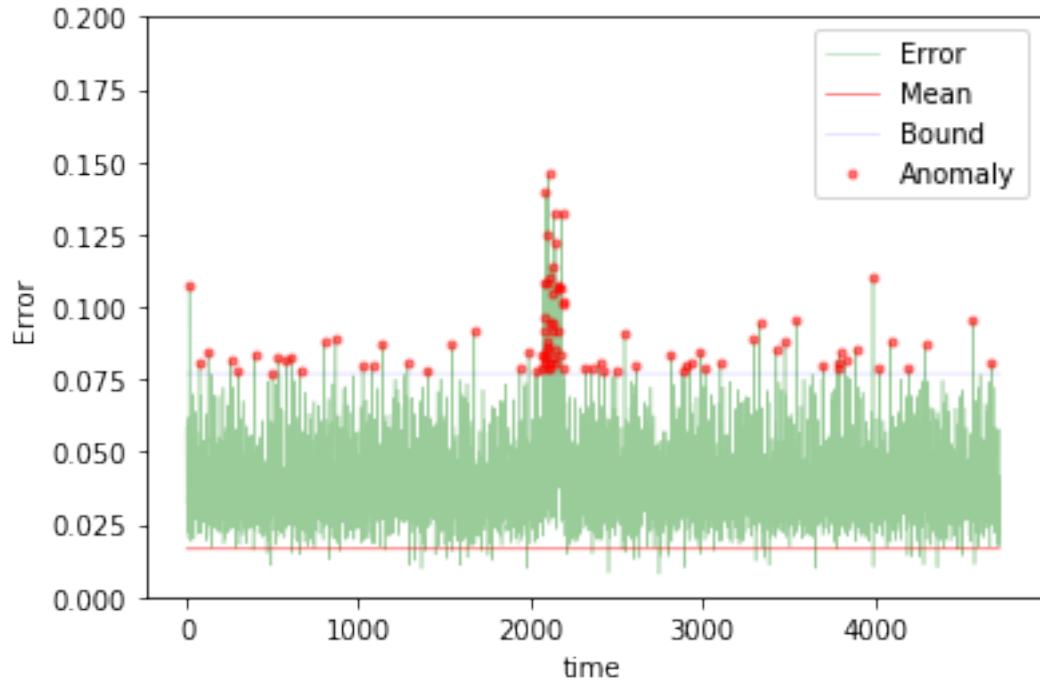




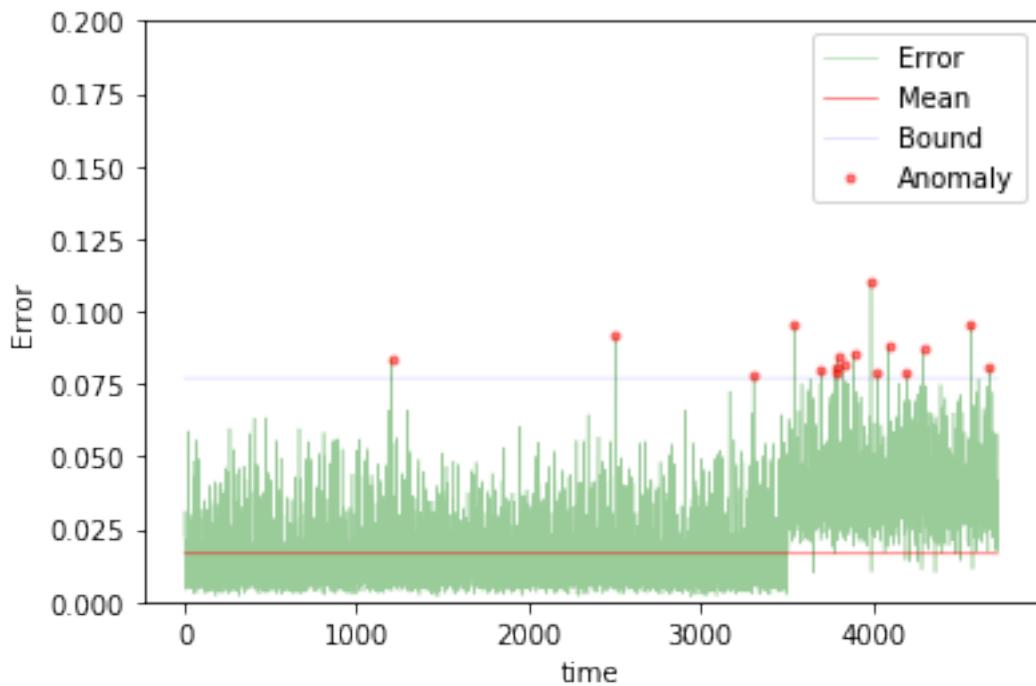
The mean error for lin2_normal_ is 0.01655049127603741 for length 4727
Testing on anomaly data.



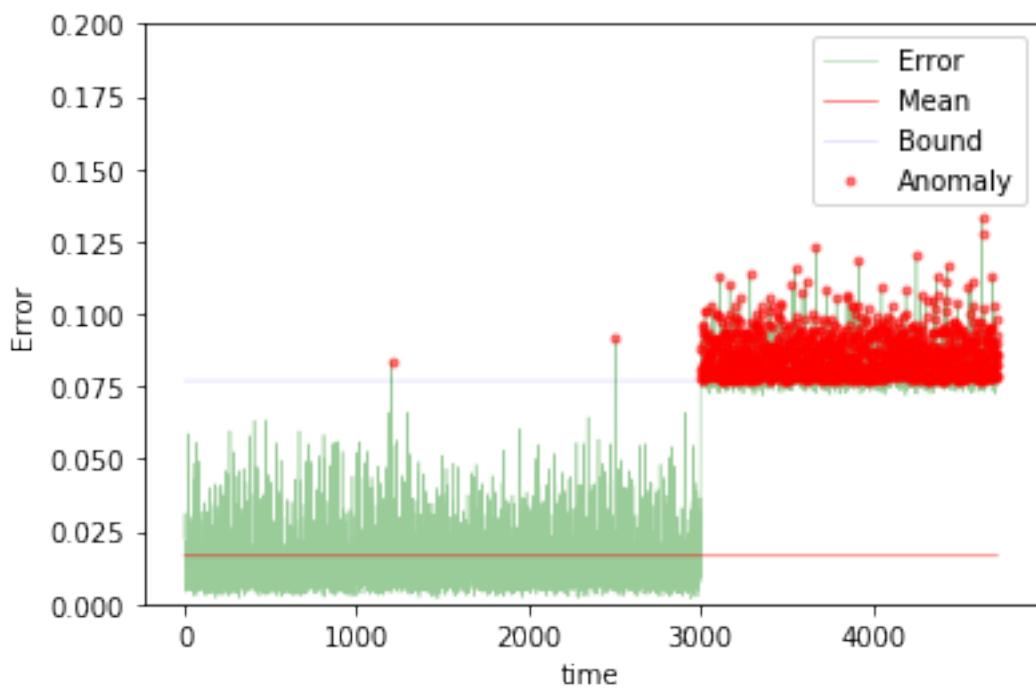
The mean error for lin2_anomaly_ is 0.018543974033581198 for length 4727
Testing on different app data.



The mean error for lin2_diff_app_ is 0.038470196443761914 for length 4727
Testing on App change synthetic data.



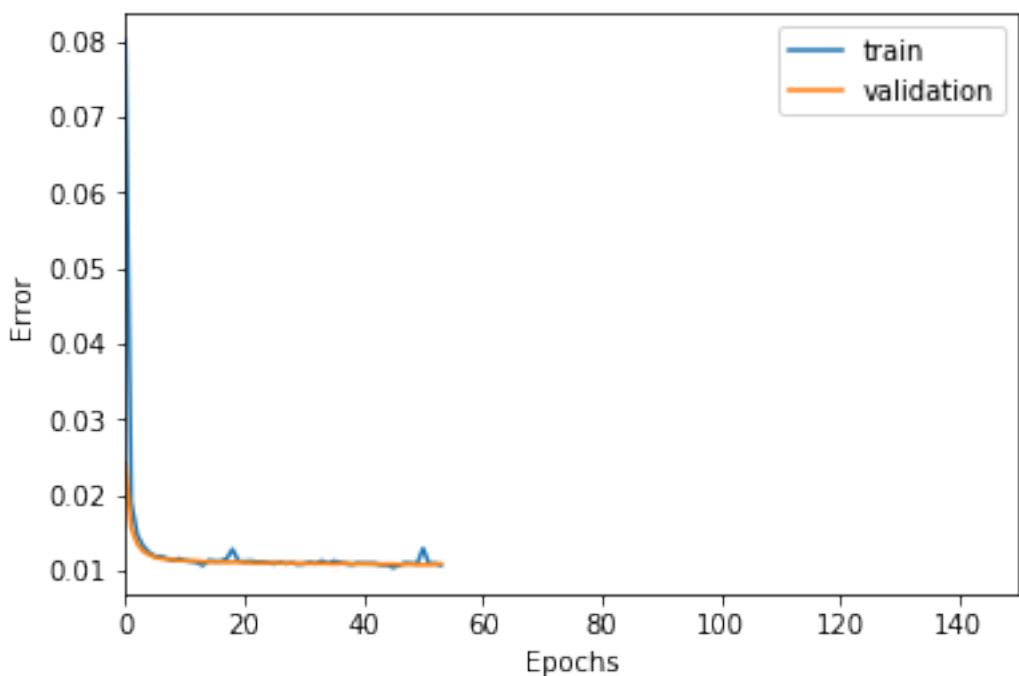
The mean error for lin2_app_change_ is 0.022027103683840286 for length 4727
Testing on Net flood synthetic data.



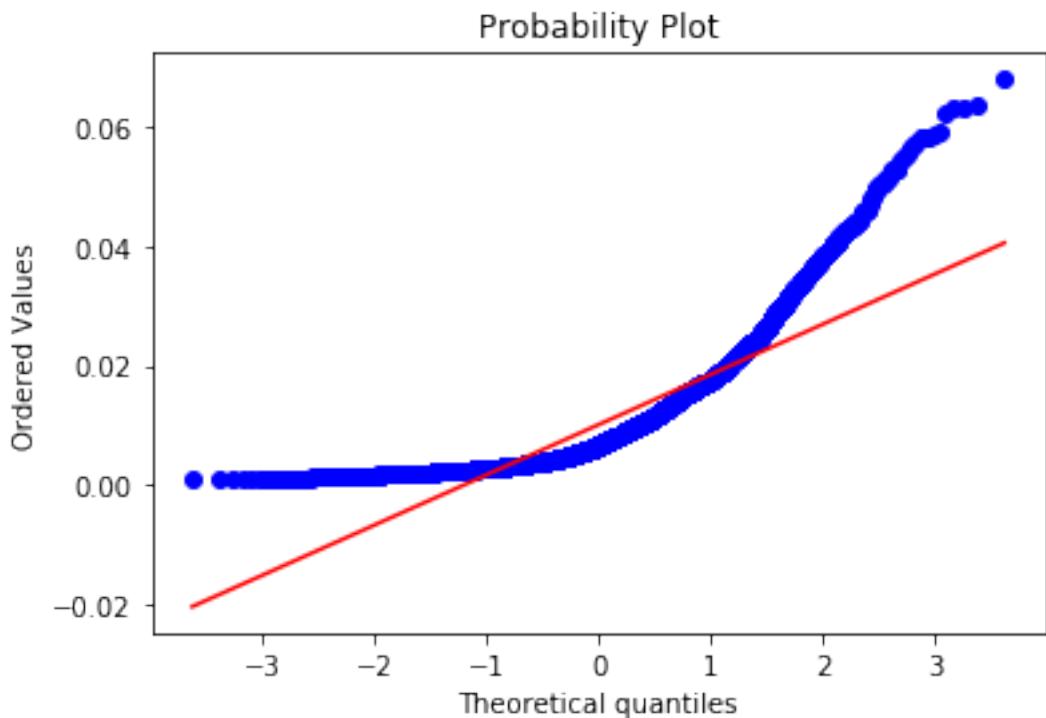
```
The mean error for lin2_net_flood_ is 0.040746253639862444 for length 4727  
=====
```

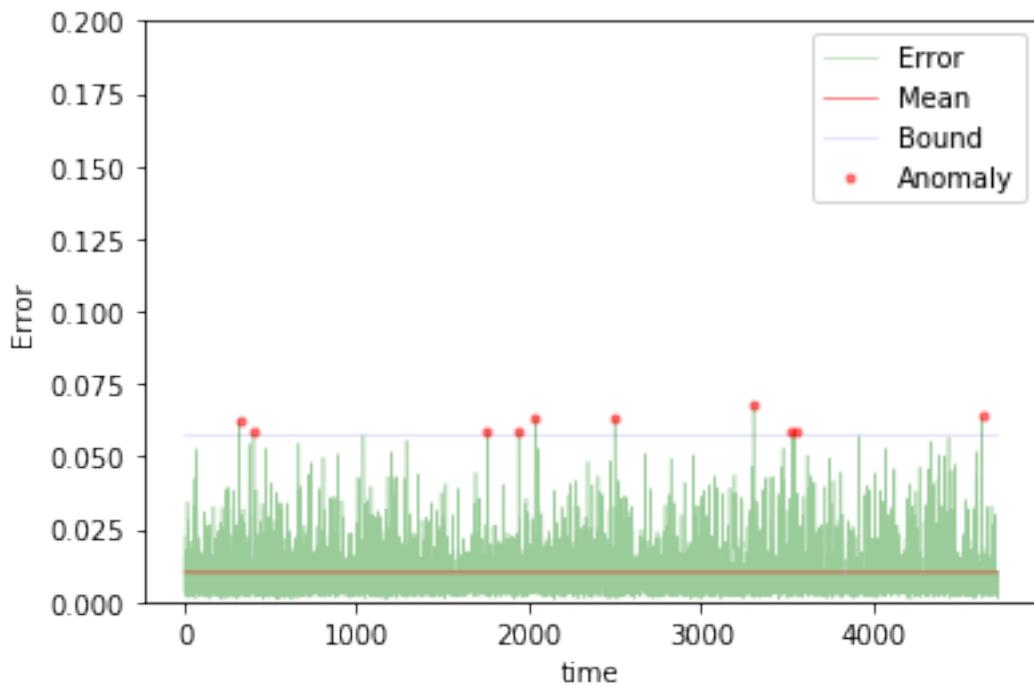
5 steps

```
In [40]: TIMESTEPS = 5  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "lin5"  
  
In [41]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
output = Dense(DIM, activation='sigmoid')(input_layer)  
  
In [42]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [43]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

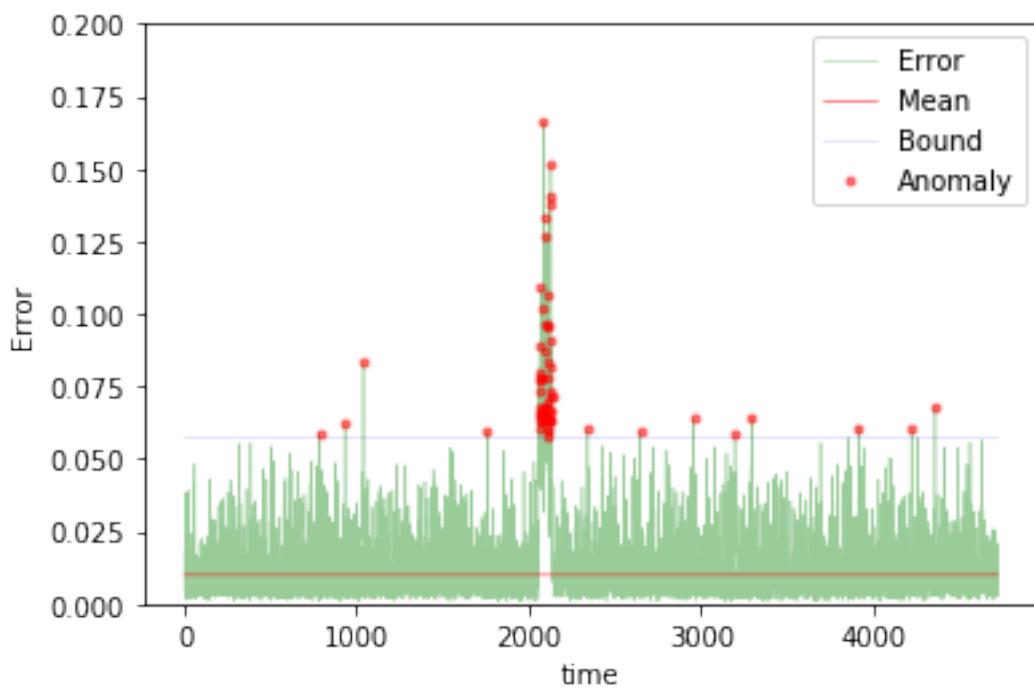


Training loss for final epoch is 0.010696192165021785
Validation loss for final epoch is 0.010877490401617252
----- Beginning tests for lin5 -----
Testing on normal data.

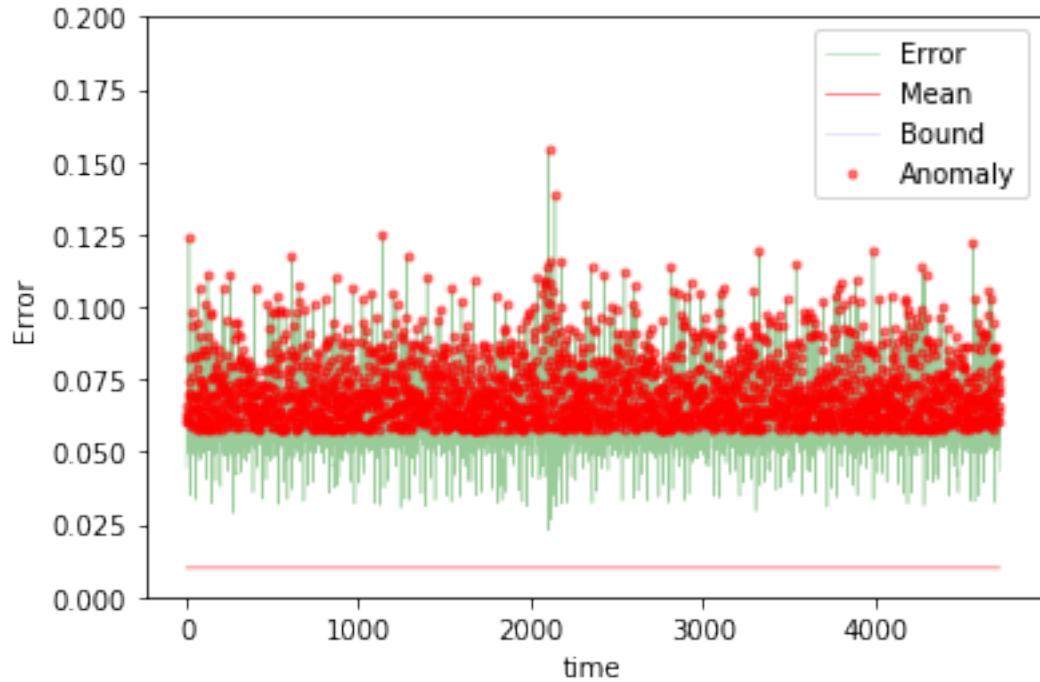




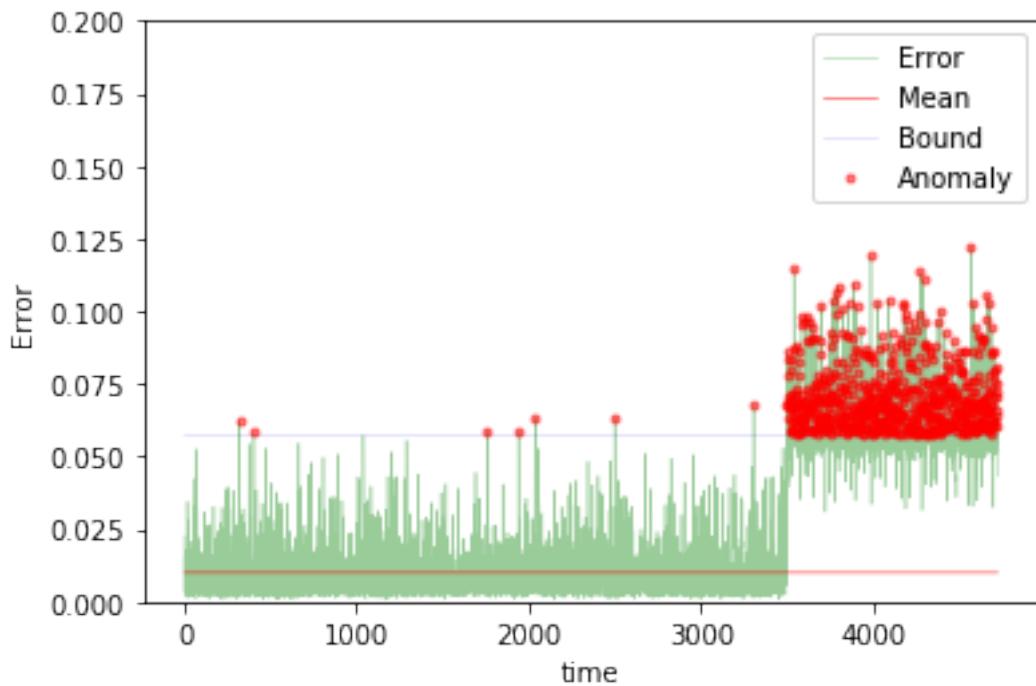
The mean error for lin5_normal_ is 0.010037698833308142 for length 4724
Testing on anomaly data.



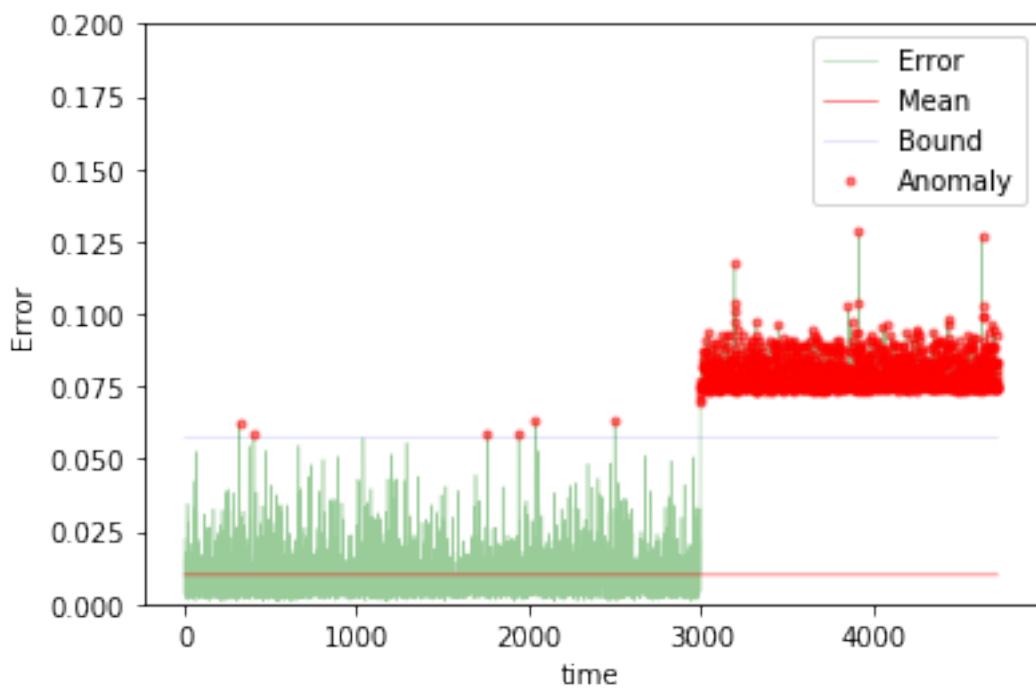
The mean error for lin5_anomaly_ is 0.012254539460783977 for length 4724
Testing on different app data.



The mean error for lin5_diff_app_ is 0.06210815506895952 for length 4724
Testing on App change synthetic data.



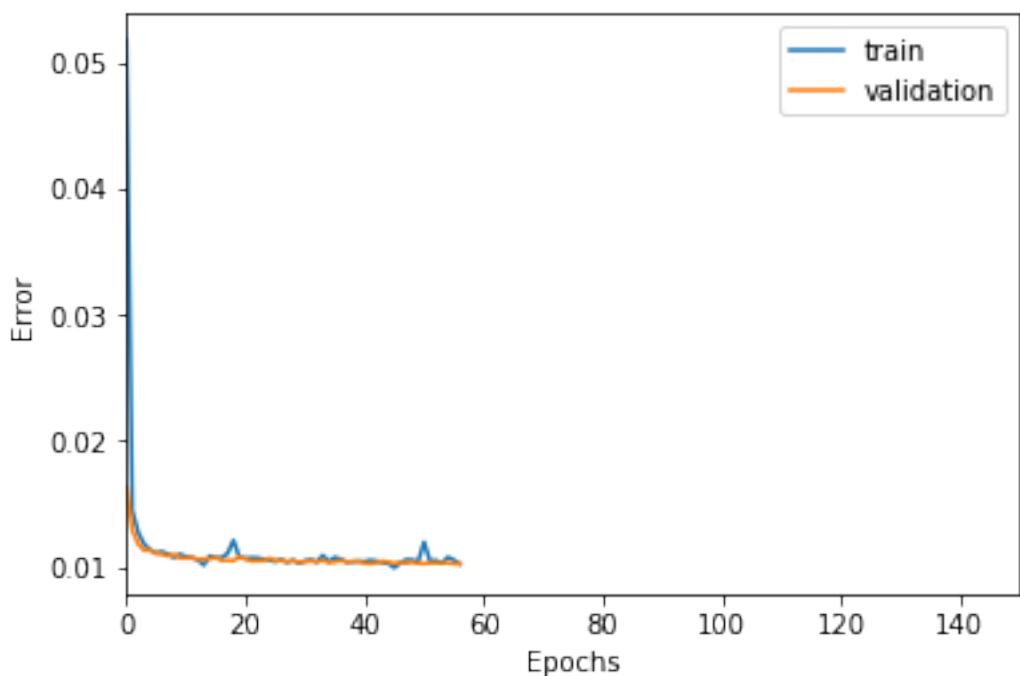
The mean error for lin5_app_change_ is 0.023489377522893762 for length 4724
Testing on Net flood synthetic data.



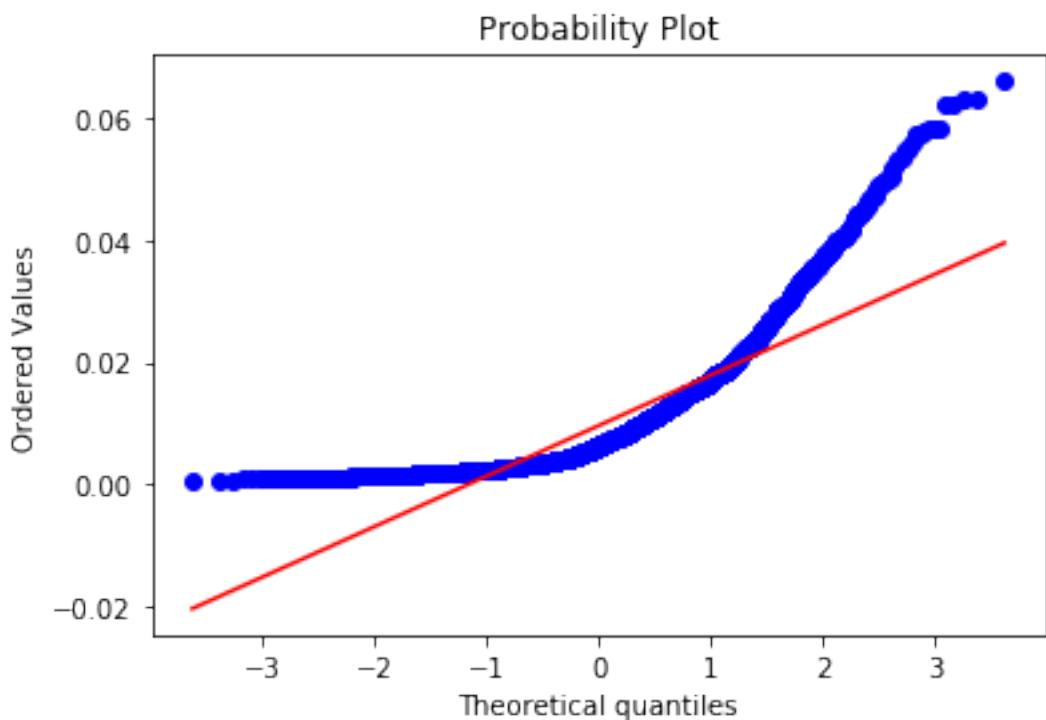
```
The mean error for lin5_net_flood_ is 0.03519256746046171 for length 4724  
=====
```

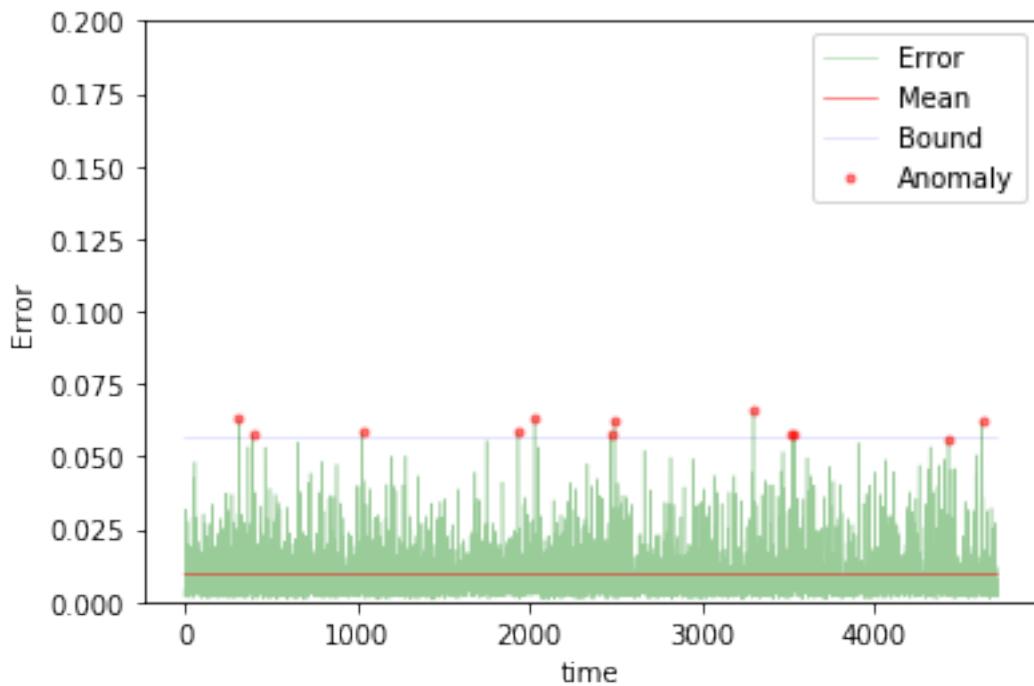
10 steps

```
In [44]: TIMESTEPS = 10  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "lin10"  
  
In [45]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
output = Dense(DIM, activation='sigmoid')(input_layer)  
  
In [46]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [47]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

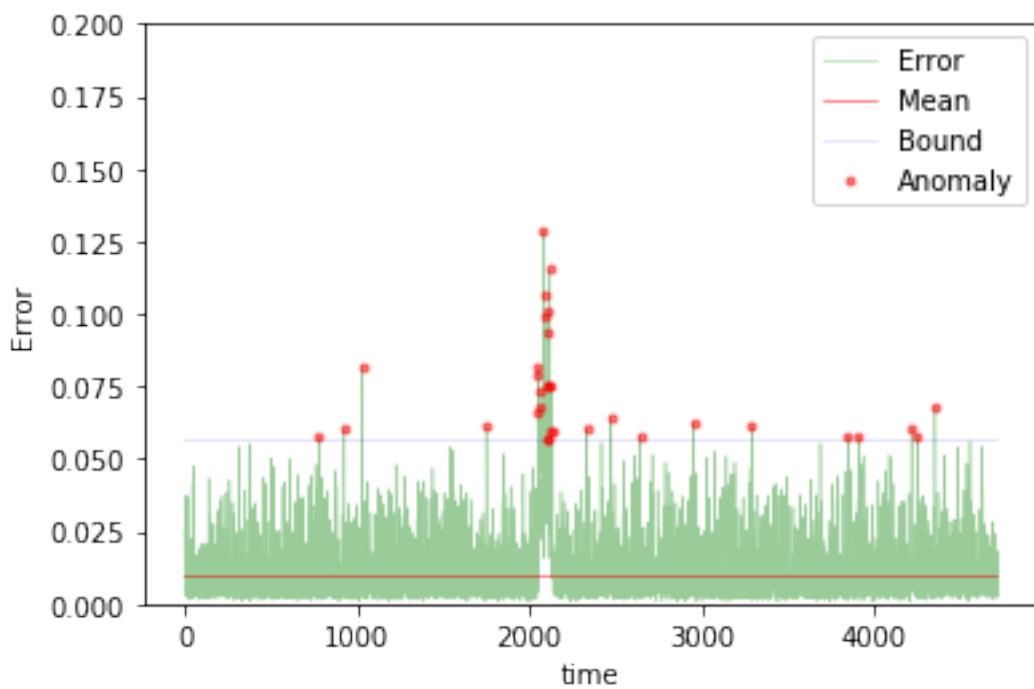


Training loss for final epoch is 0.010256073548807762
Validation loss for final epoch is 0.010299438451649622
----- Beginning tests for lin10 -----
Testing on normal data.

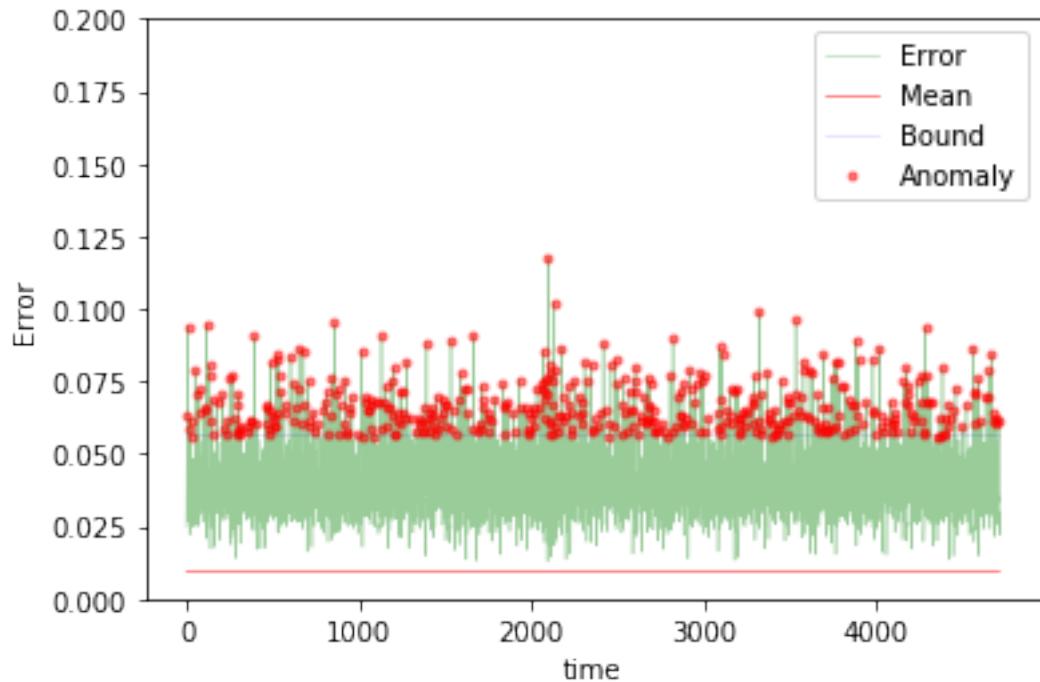




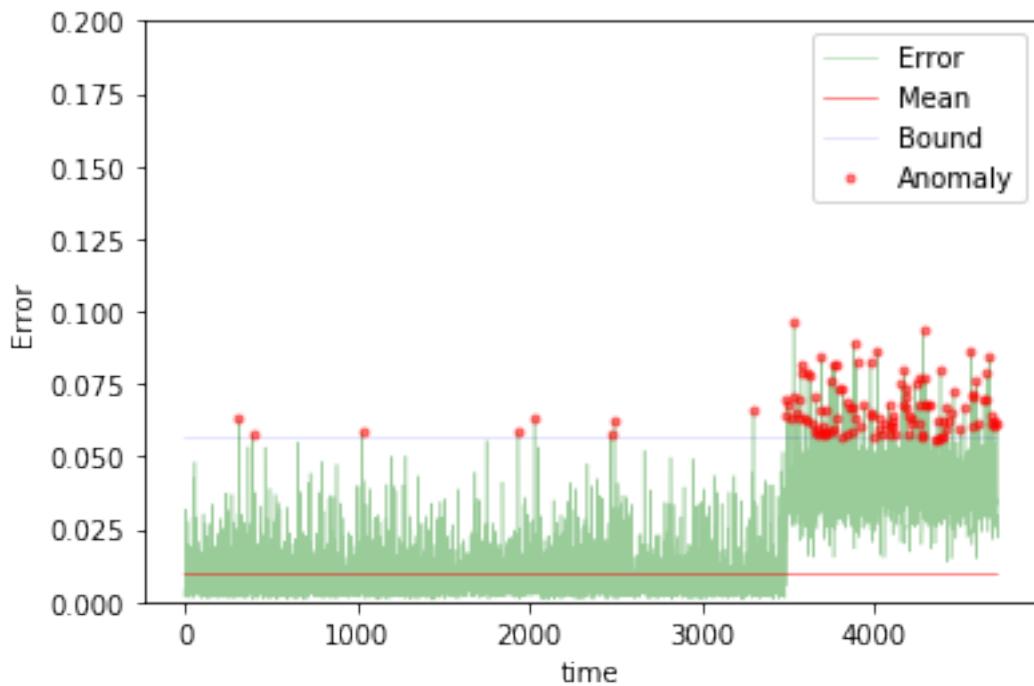
The mean error for lin10_normal_ is 0.009604812940342572 for length 4719
Testing on anomaly data.



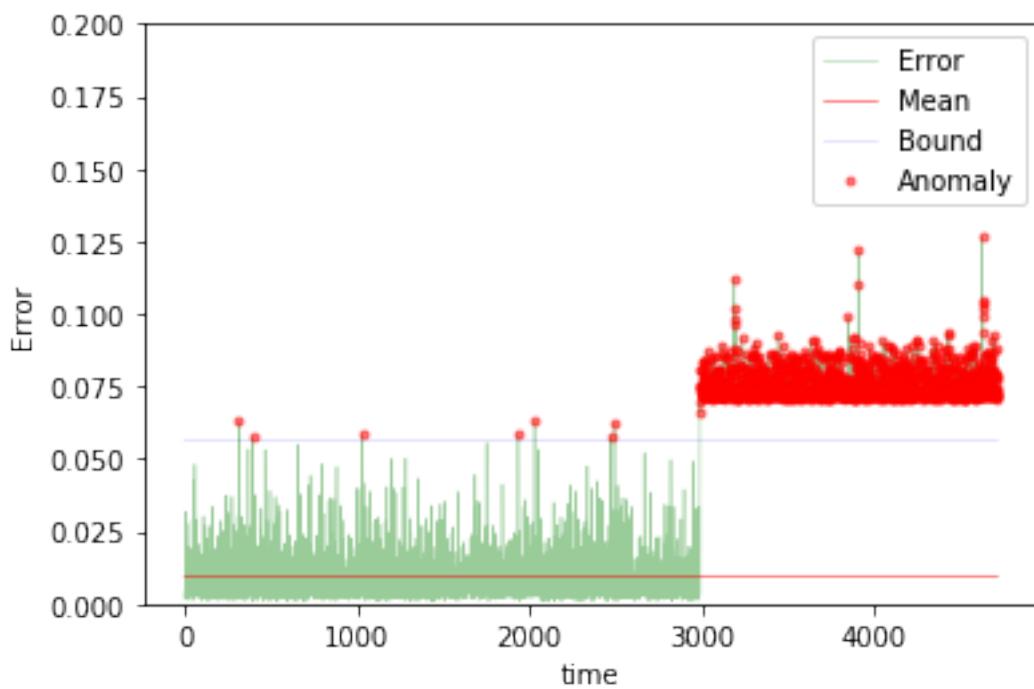
The mean error for lin10_anomaly_ is 0.011336649921305068 for length 4719
Testing on different app data.



The mean error for lin10_diff_app_ is 0.03963713064291286 for length 4719
Testing on App change synthetic data.



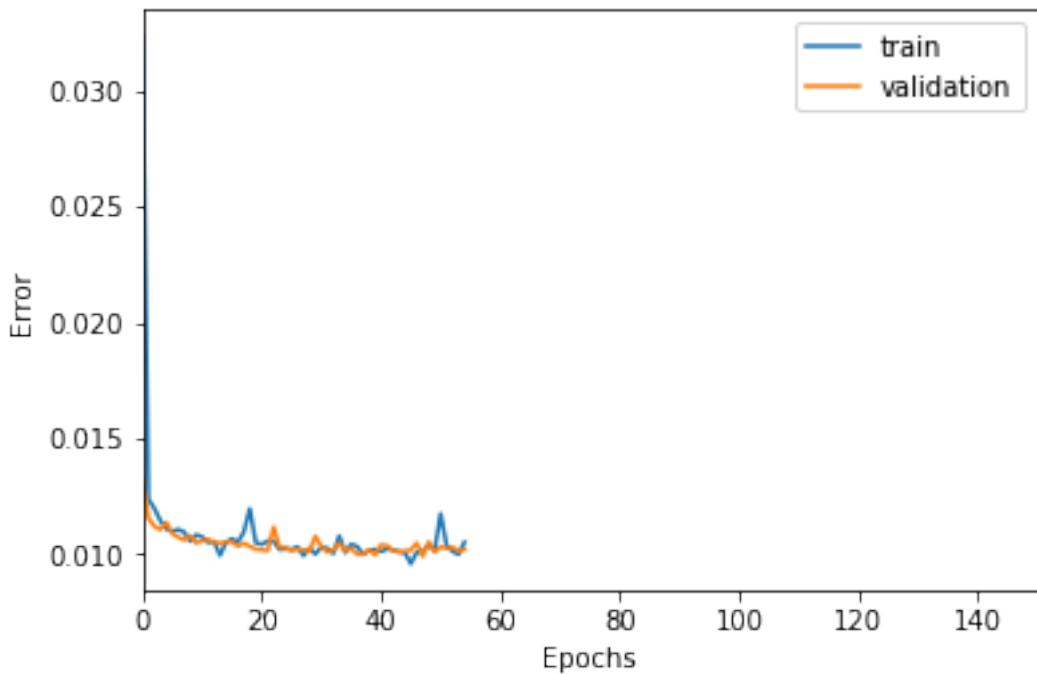
The mean error for lin10_app_change_ is 0.017365355792304173 for length 4719
Testing on Net flood synthetic data.



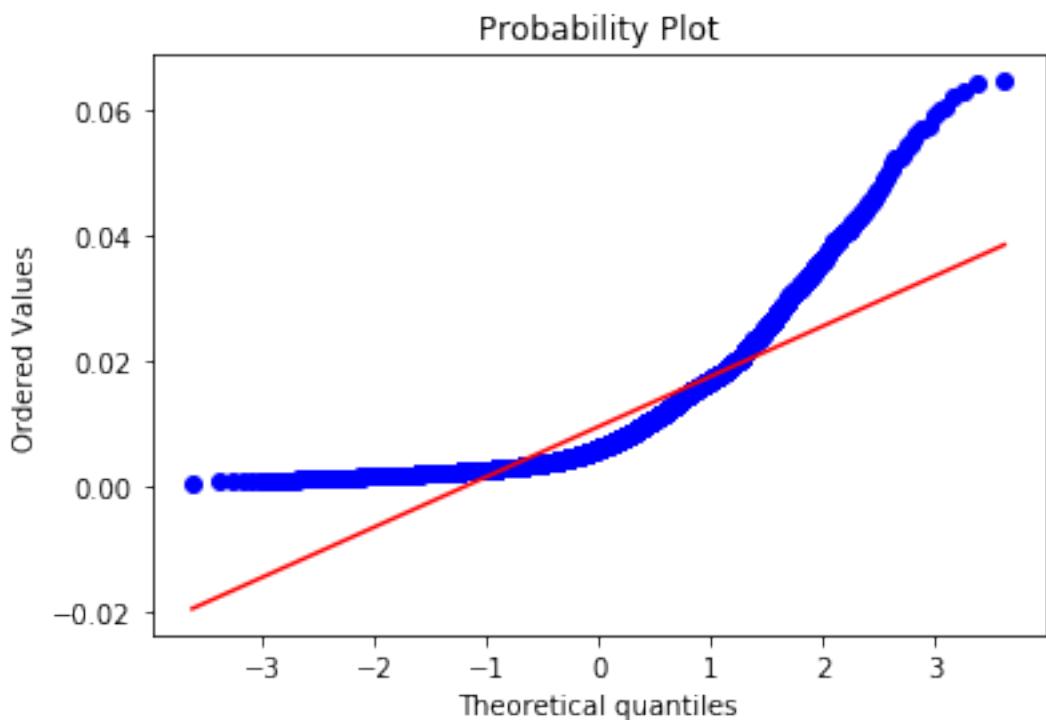
```
The mean error for lin10_net_flood_ is 0.03371274332688824 for length 4719  
=====
```

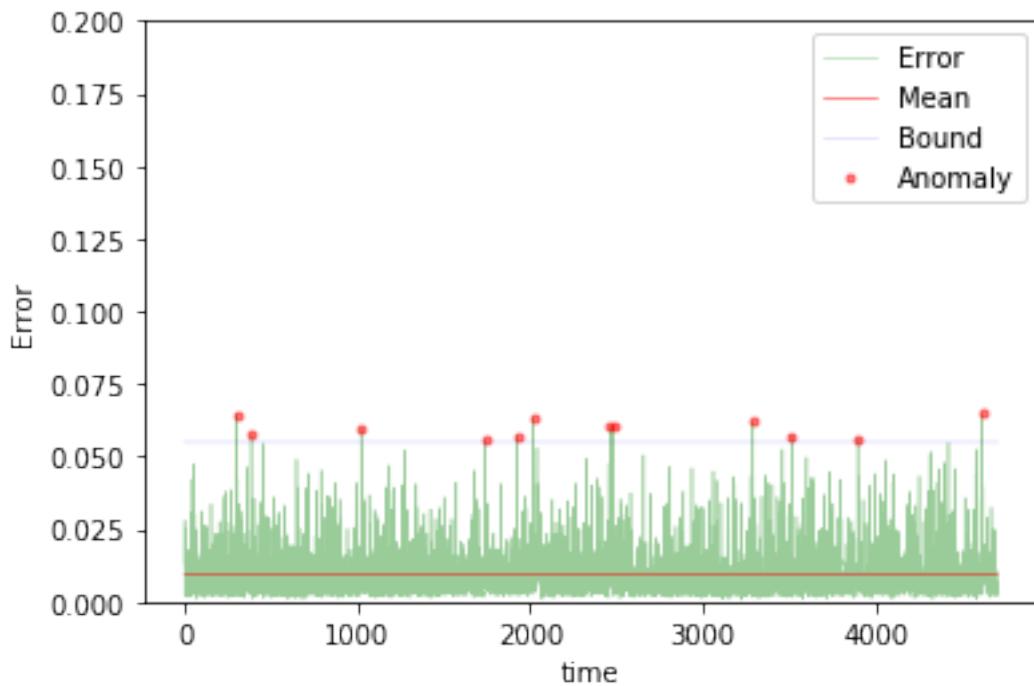
20 steps

```
In [48]: TIMESTEPS = 20  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "lin20"  
  
In [49]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
output = Dense(DIM, activation='sigmoid')(input_layer)  
  
In [50]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [51]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

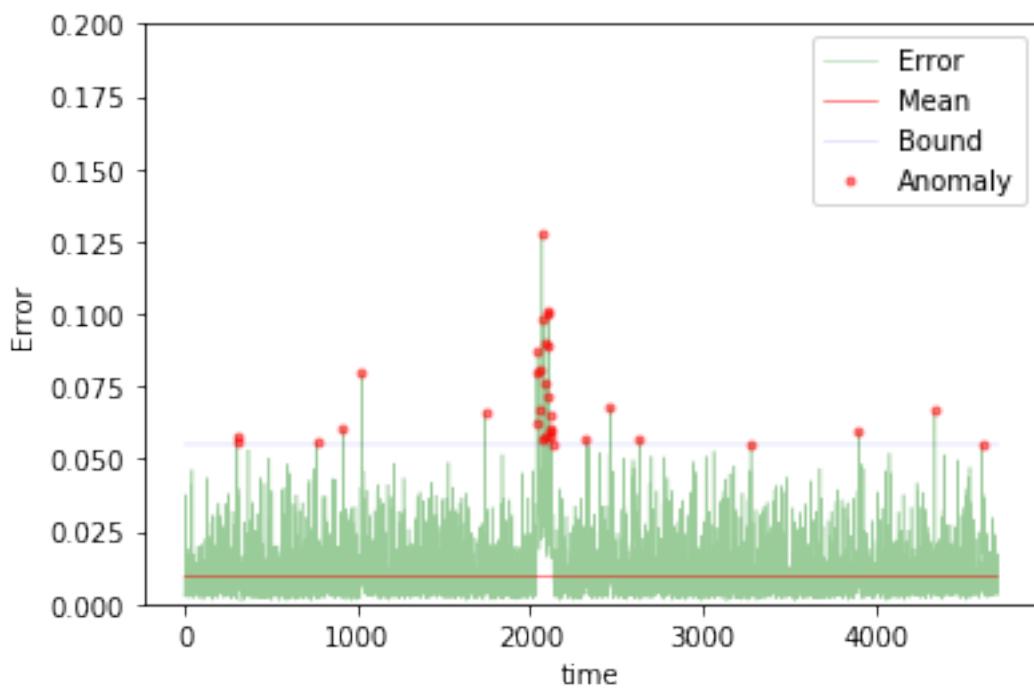


Training loss for final epoch is 0.010524493468925356
Validation loss for final epoch is 0.010219356117071584
----- Beginning tests for lin20 -----
Testing on normal data.

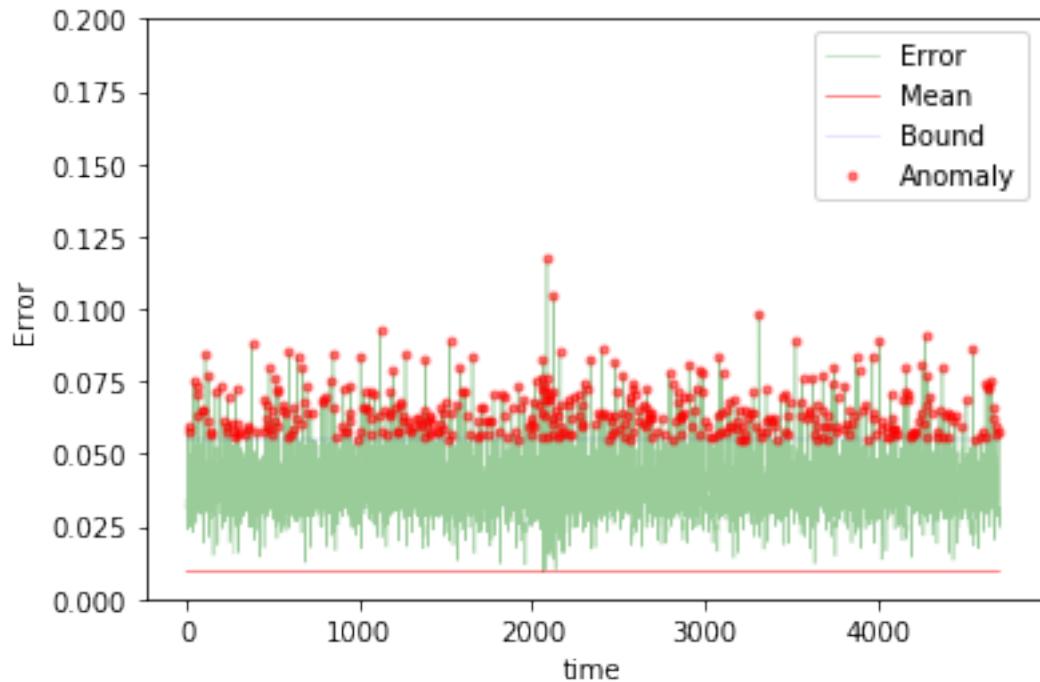




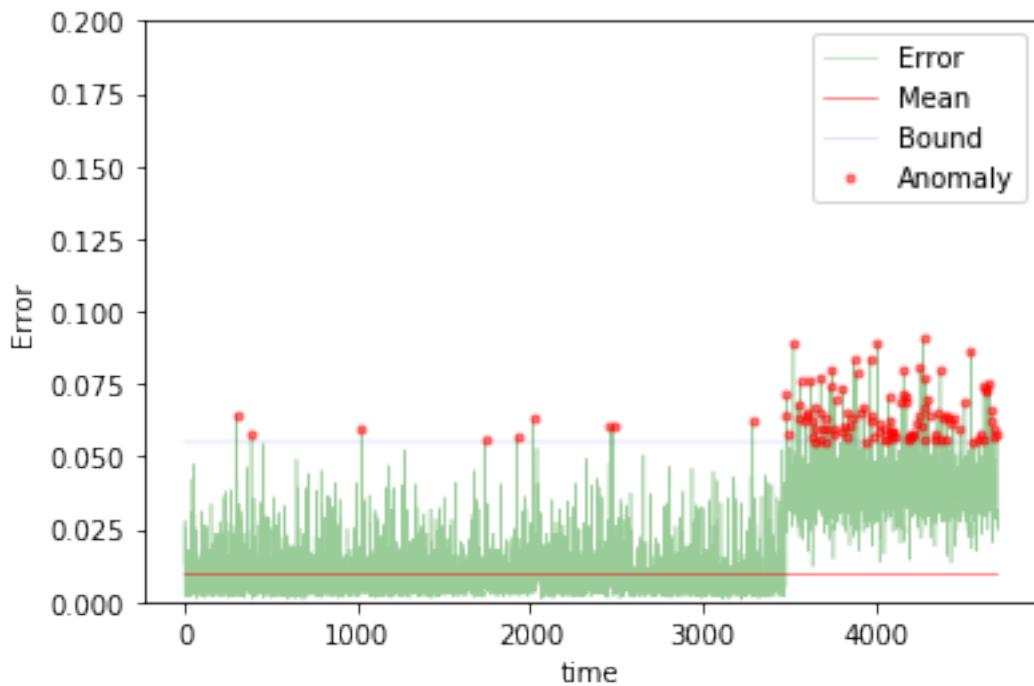
The mean error for lin20_normal_ is 0.009666225217579986 for length 4709
Testing on anomaly data.



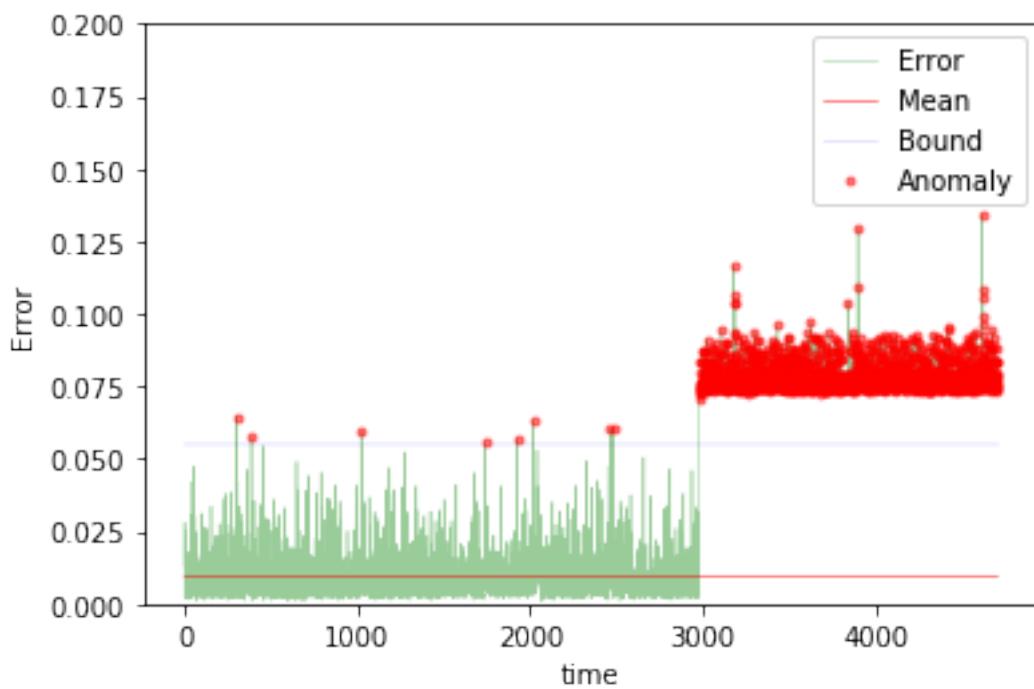
The mean error for lin20_anomaly_ is 0.011099141133381191 for length 4709
Testing on different app data.



The mean error for lin20_diff_app_ is 0.038907357442730404 for length 4709
Testing on App change synthetic data.



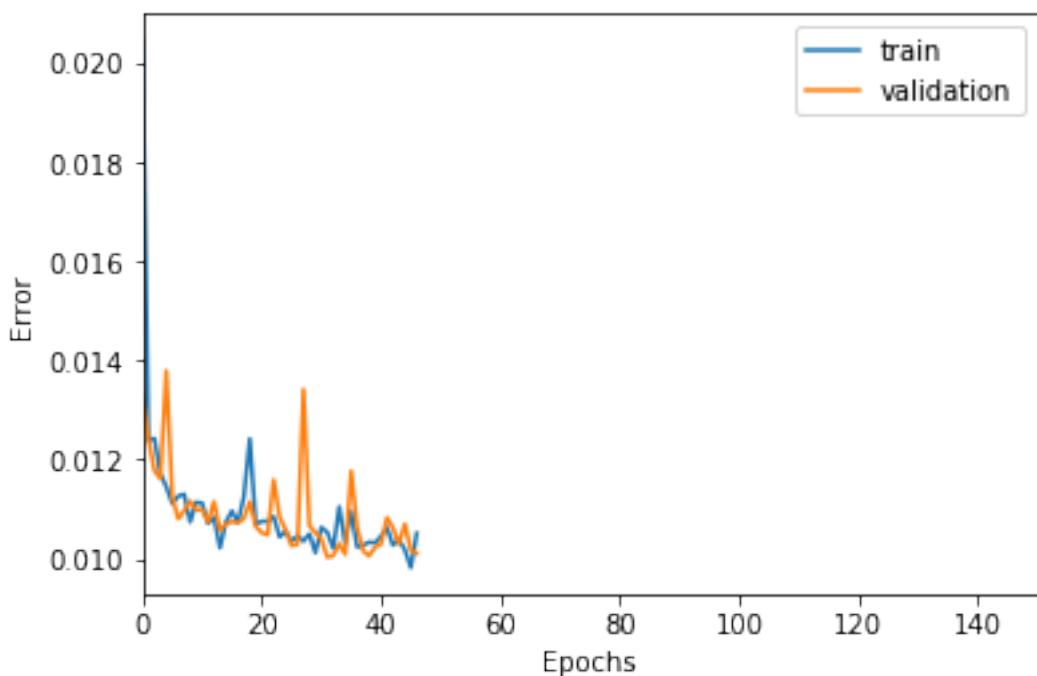
The mean error for lin20_app_change_ is 0.017234608536168736 for length 4709
Testing on Net flood synthetic data.



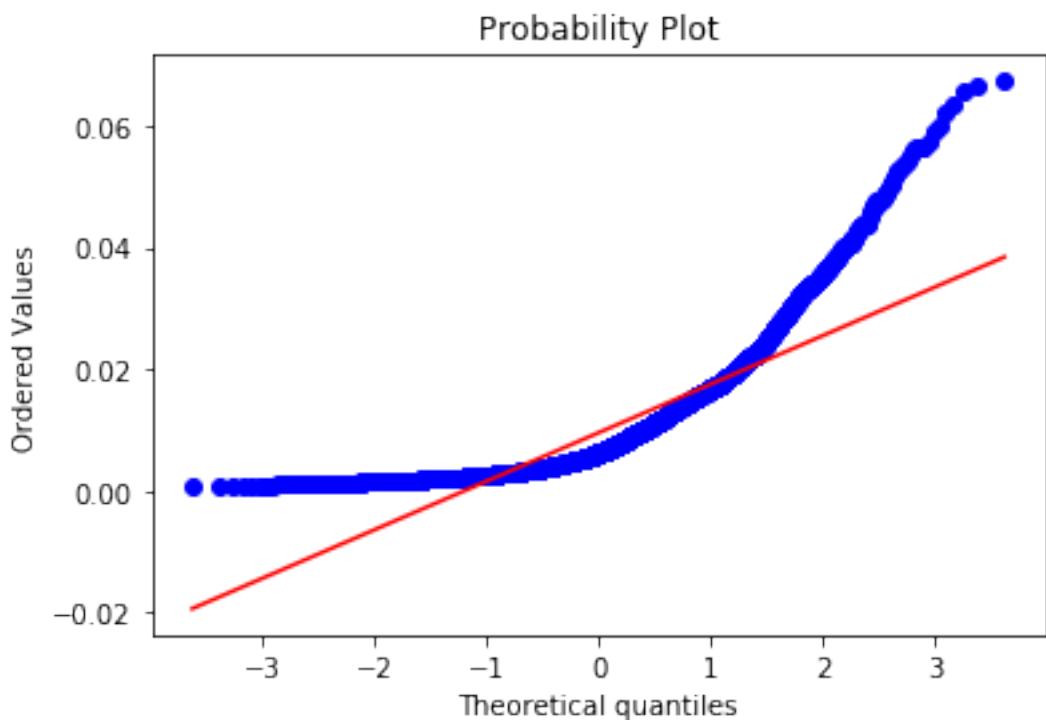
```
The mean error for lin20_net_flood_ is 0.03499923295355831 for length 4709  
=====
```

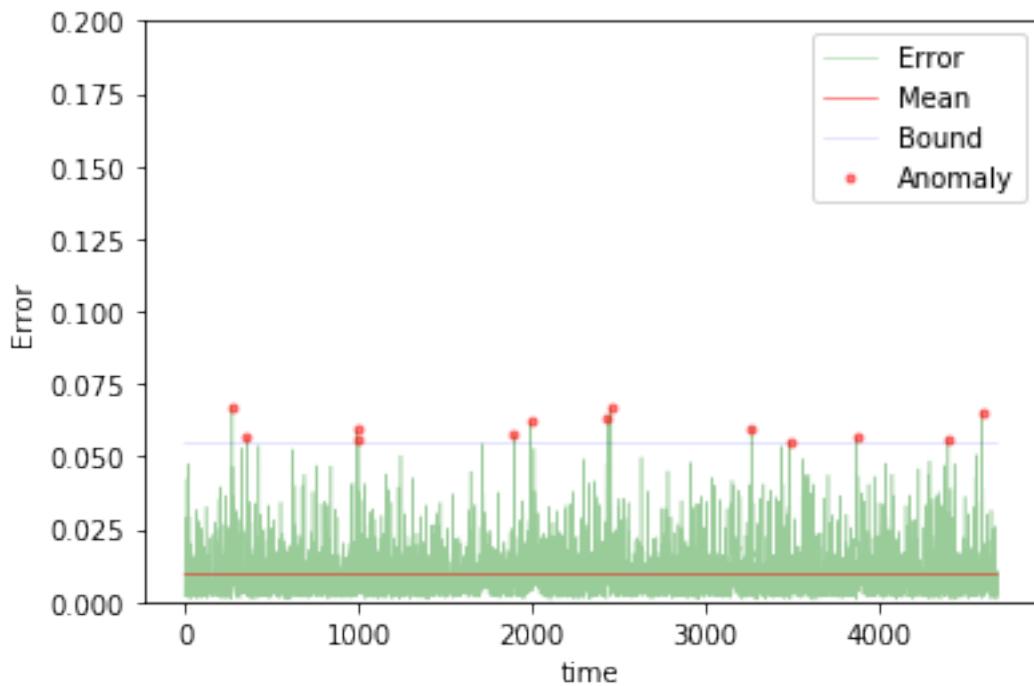
50 steps

```
In [52]: TIMESTEPS = 50  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "lin50"  
  
In [53]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
output = Dense(DIM, activation='sigmoid')(input_layer)  
  
In [54]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [55]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

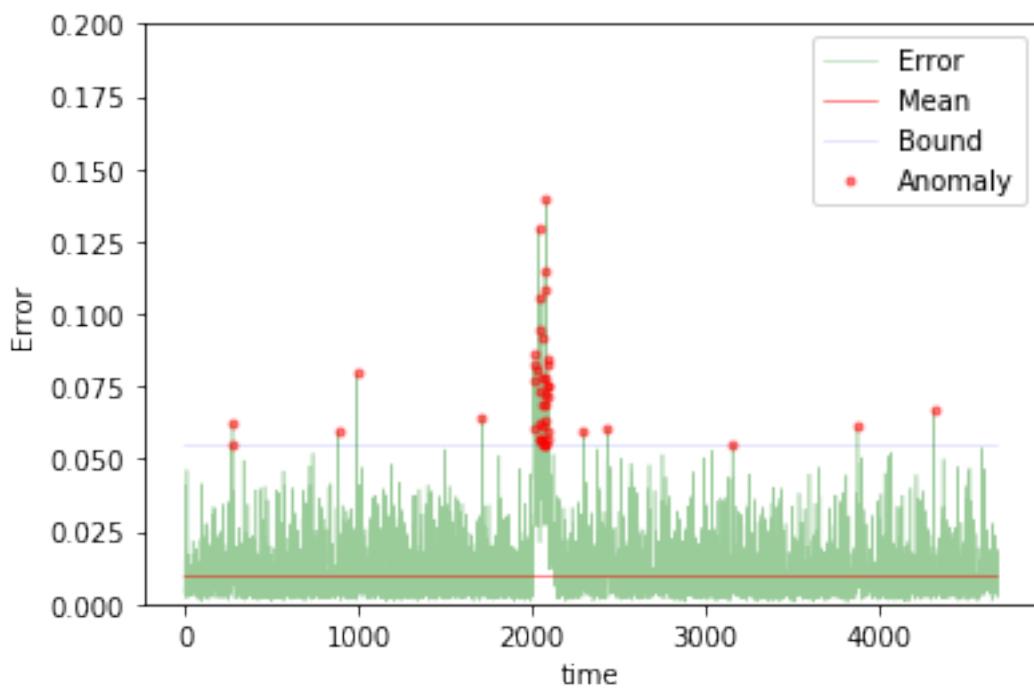


Training loss for final epoch is 0.01051153808424715
Validation loss for final epoch is 0.010109798953169958
----- Beginning tests for lin50 -----
Testing on normal data.

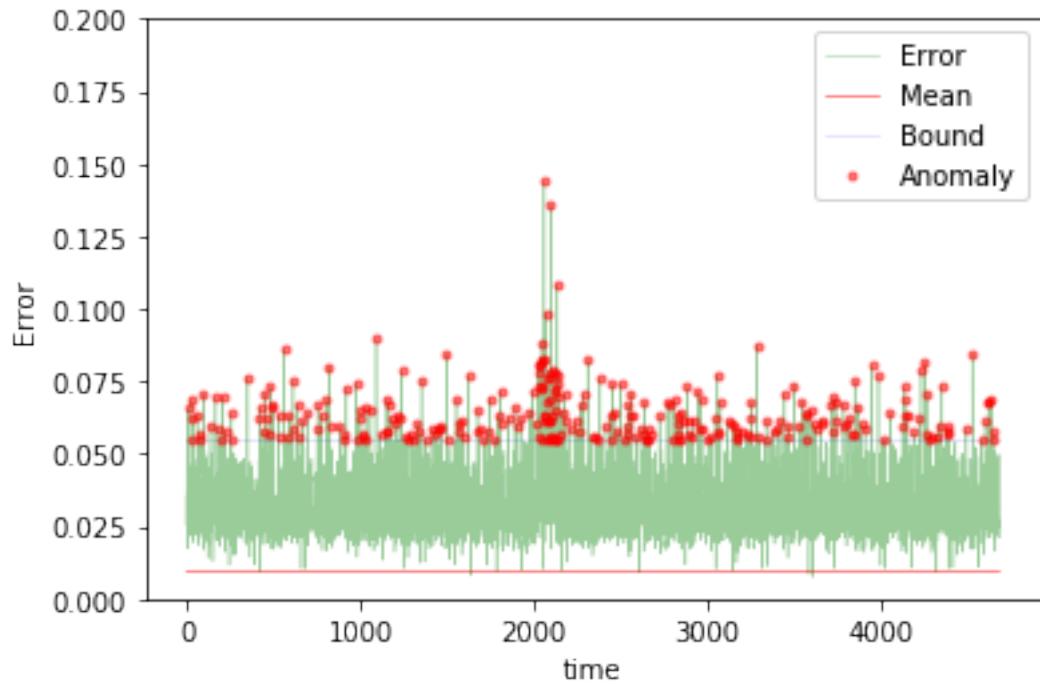




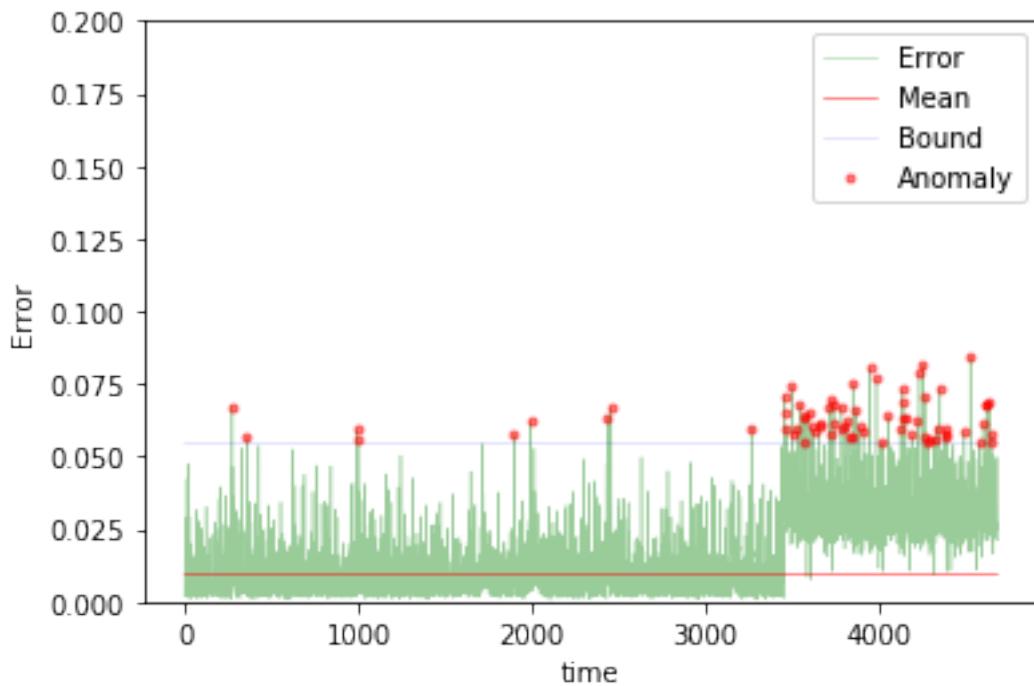
The mean error for lin50_normal_ is 0.009520417047226675 for length 4679
Testing on anomaly data.



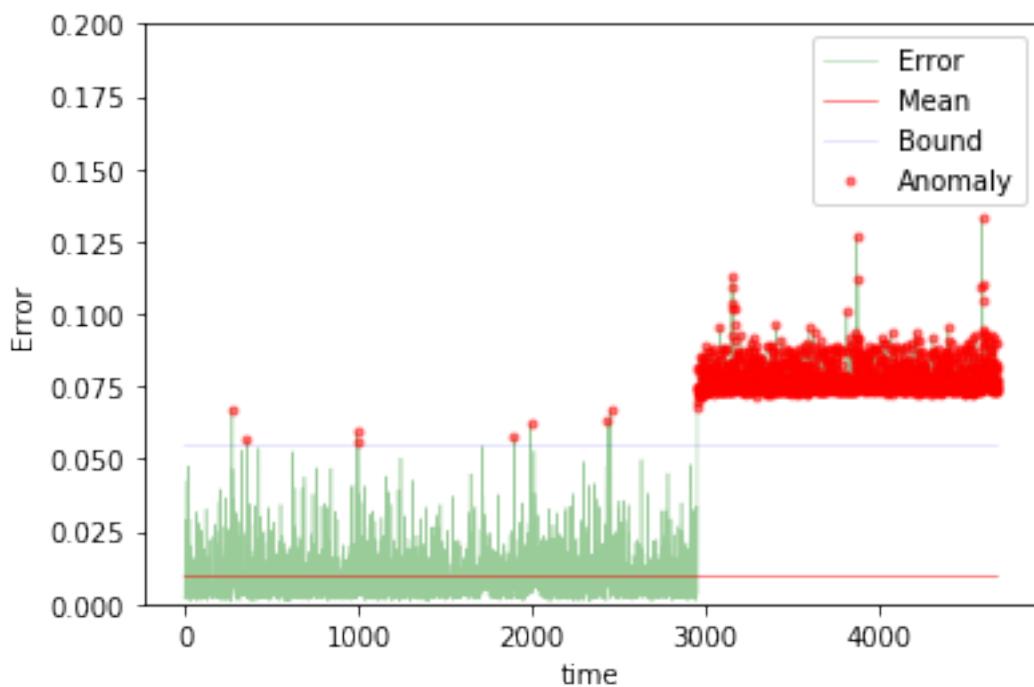
The mean error for lin50_anomaly_ is 0.011300727549279542 for length 4679
Testing on different app data.



The mean error for lin50_diff_app_ is 0.03255132094135868 for length 4679
Testing on App change synthetic data.



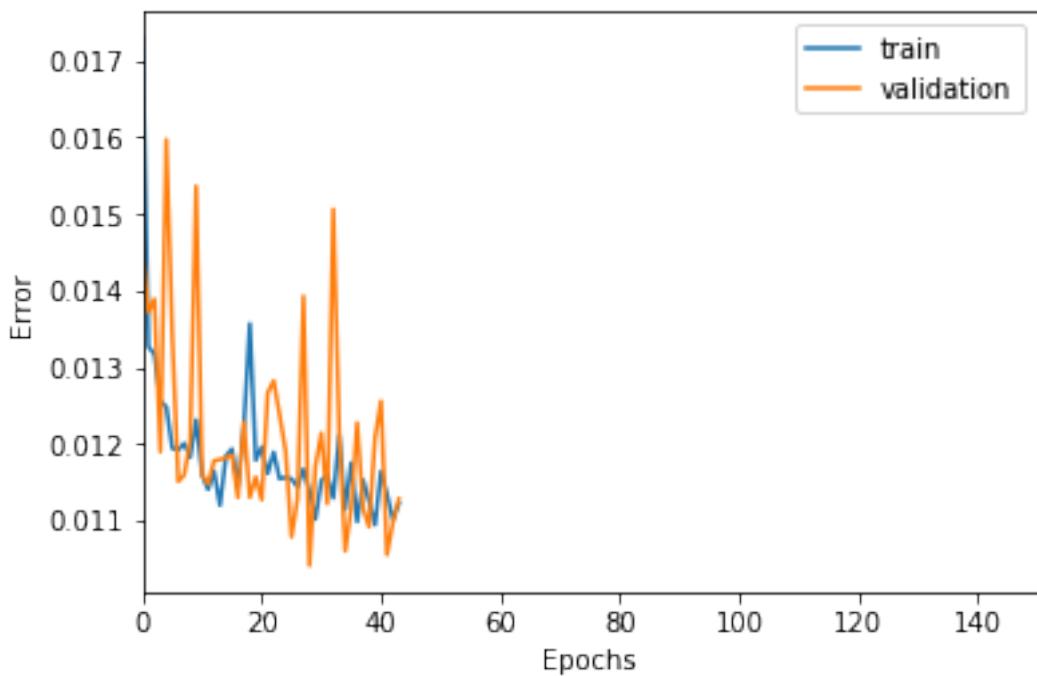
The mean error for lin50_app_change_ is 0.01548097545769297 for length 4679
Testing on Net flood synthetic data.



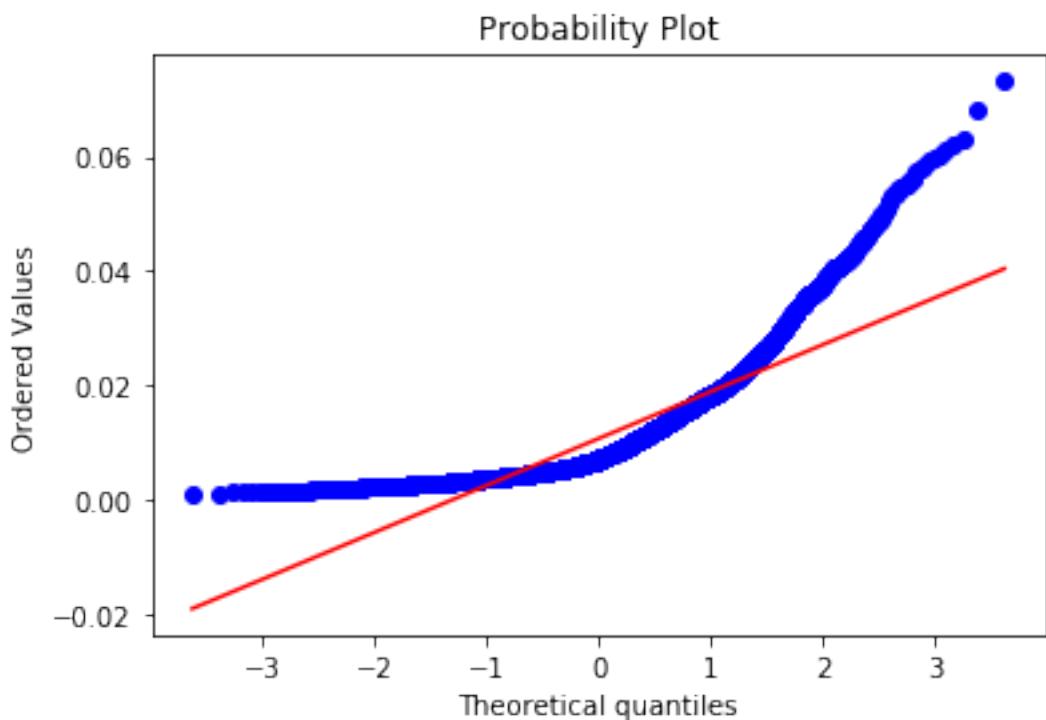
```
The mean error for lin50_net_flood_ is 0.03475463660352724 for length 4679  
=====
```

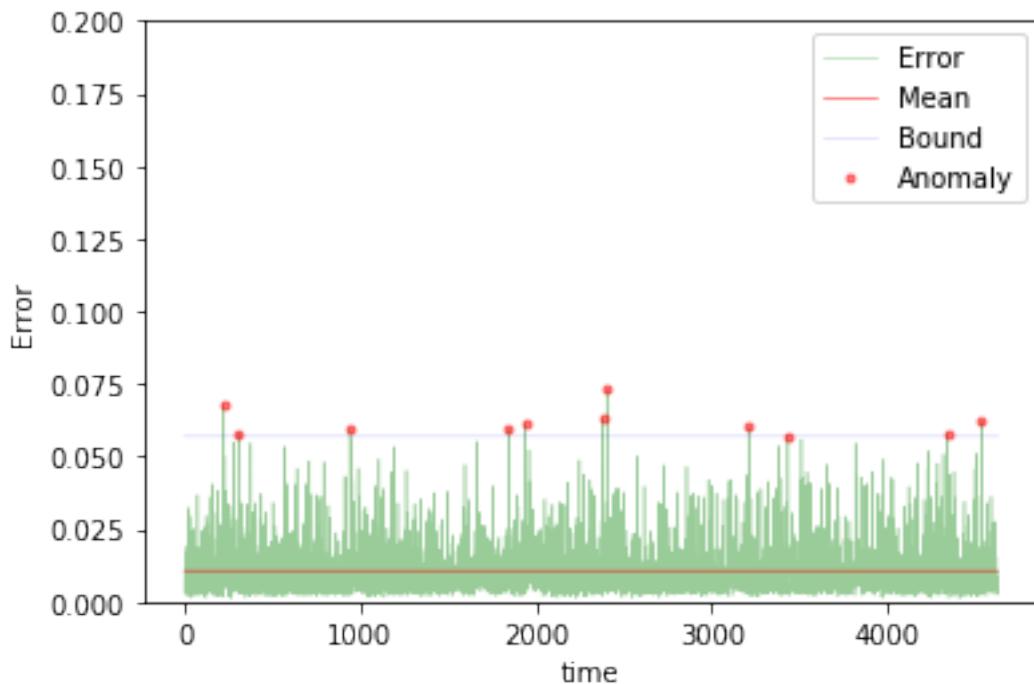
100 steps

```
In [56]: TIMESTEPS = 100  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "lin100"  
  
In [57]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
output = Dense(DIM, activation='sigmoid')(input_layer)  
  
In [58]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [59]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

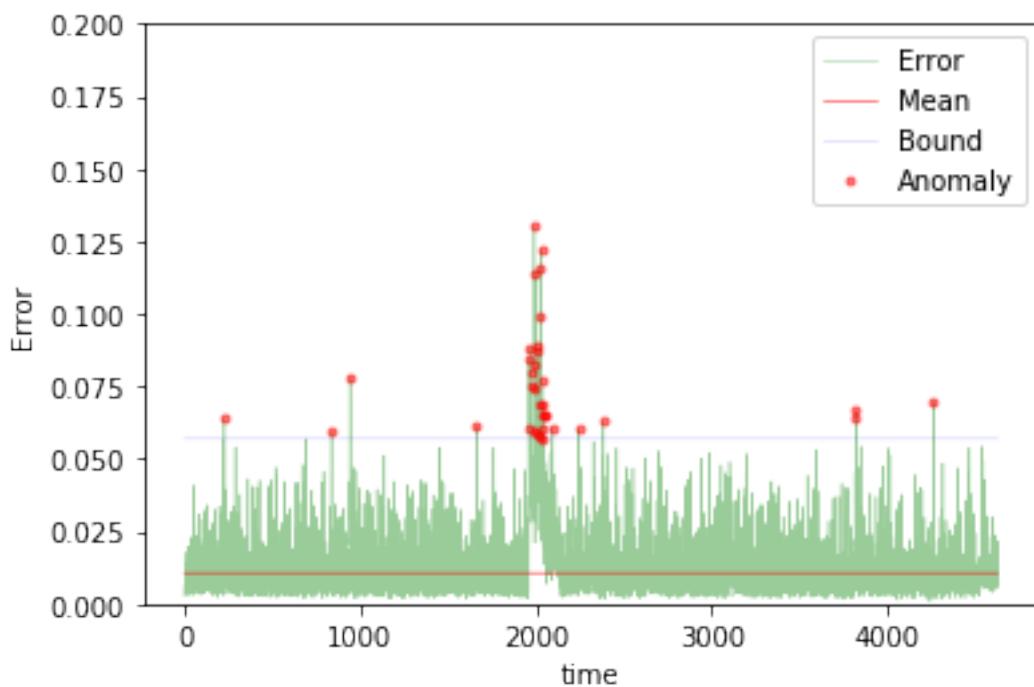


Training loss for final epoch is 0.01122209146944806
Validation loss for final epoch is 0.011278831264004112
----- Beginning tests for lin100 -----
Testing on normal data.

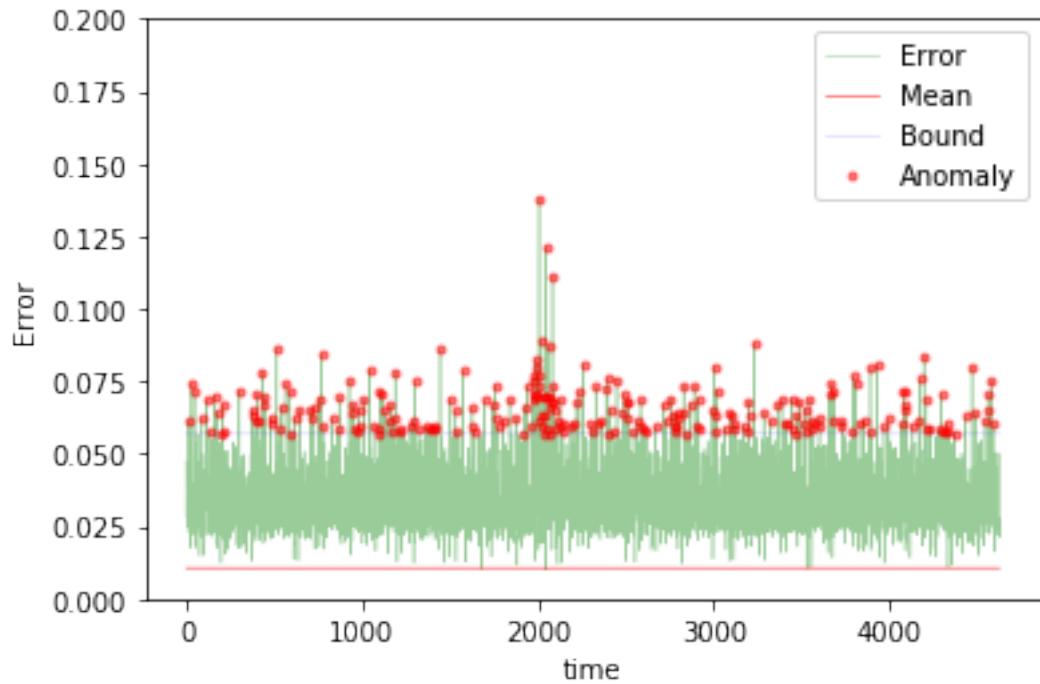




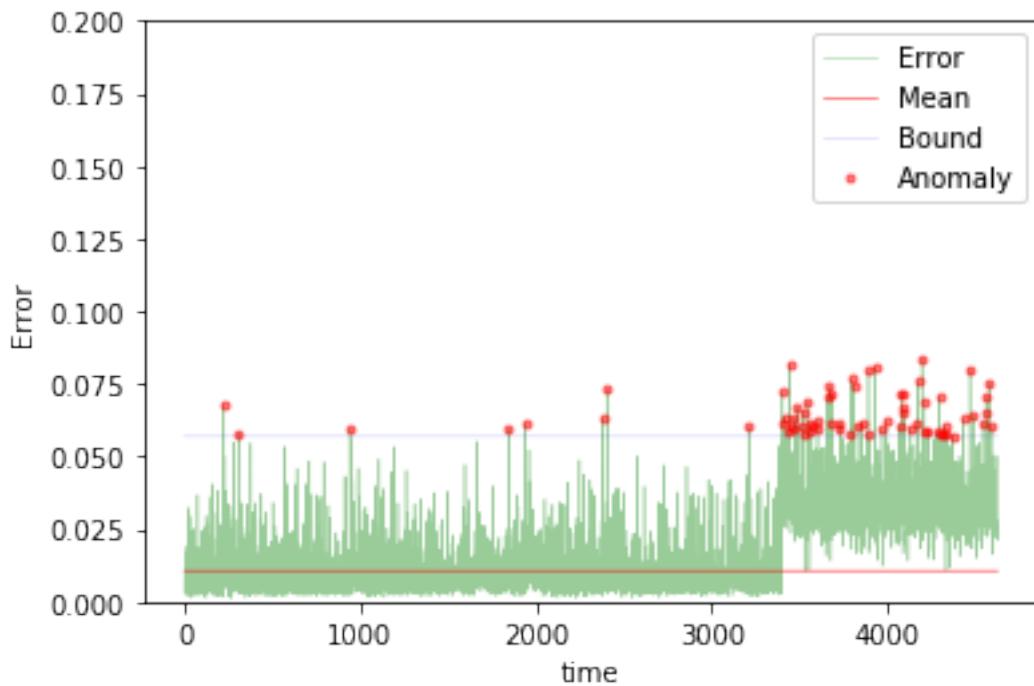
The mean error for lin100_normal_ is 0.010707669924525128 for length 4629
Testing on anomaly data.



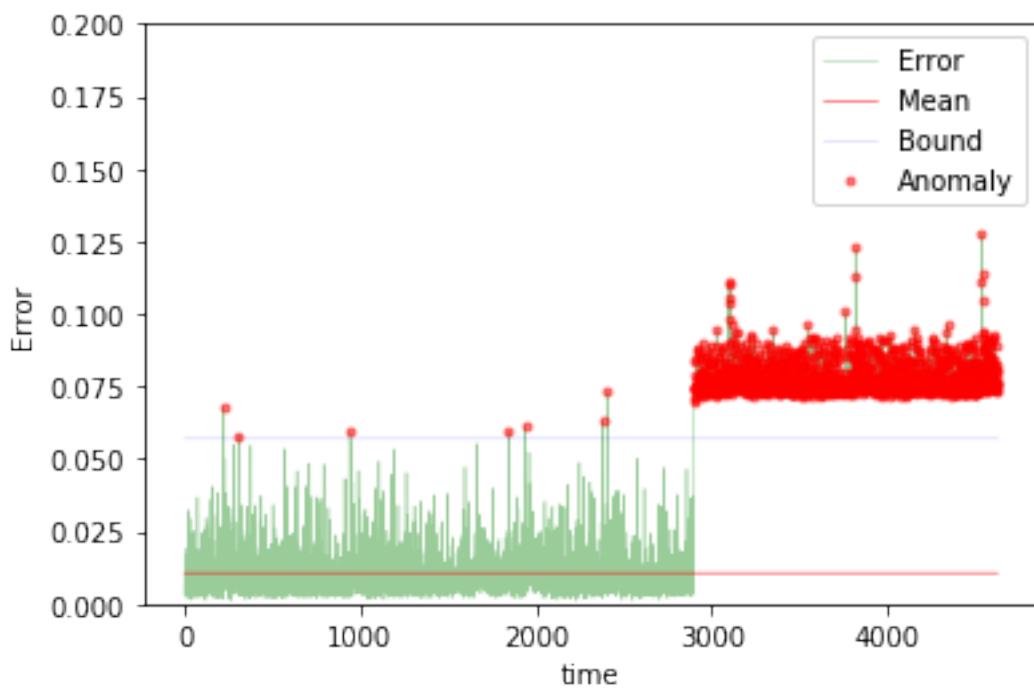
The mean error for lin100_anomaly_ is 0.01260882885807616 for length 4629
Testing on different app data.



The mean error for lin100_diff_app_ is 0.03414769892459869 for length 4629
Testing on App change synthetic data.



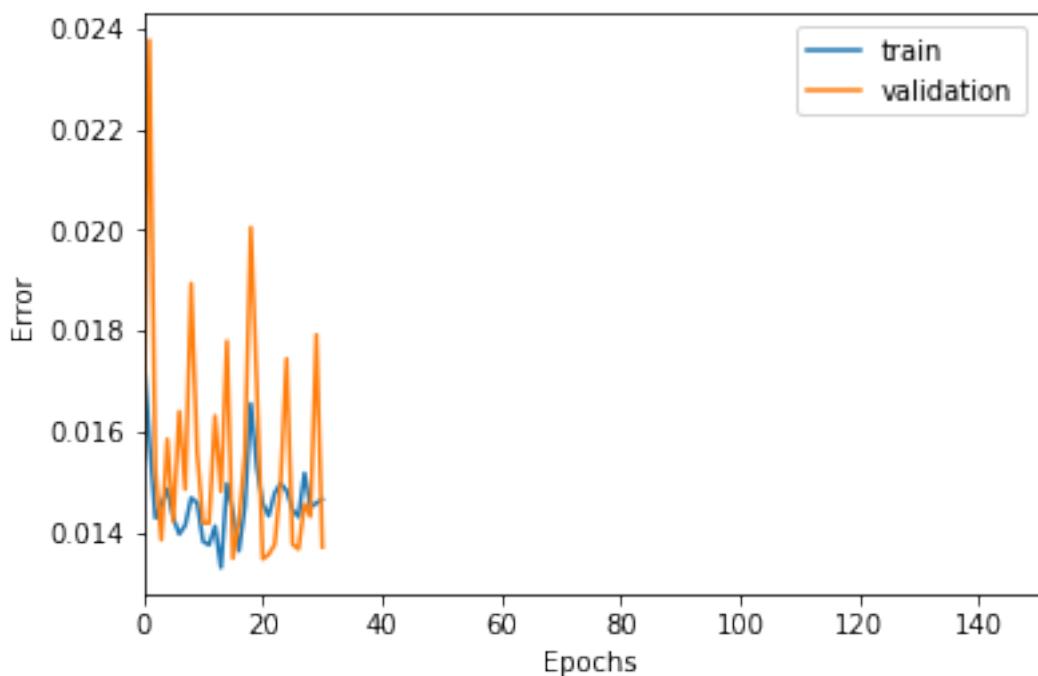
The mean error for lin100_app_change_ is 0.016878917614525413 for length 4629
Testing on Net flood synthetic data.



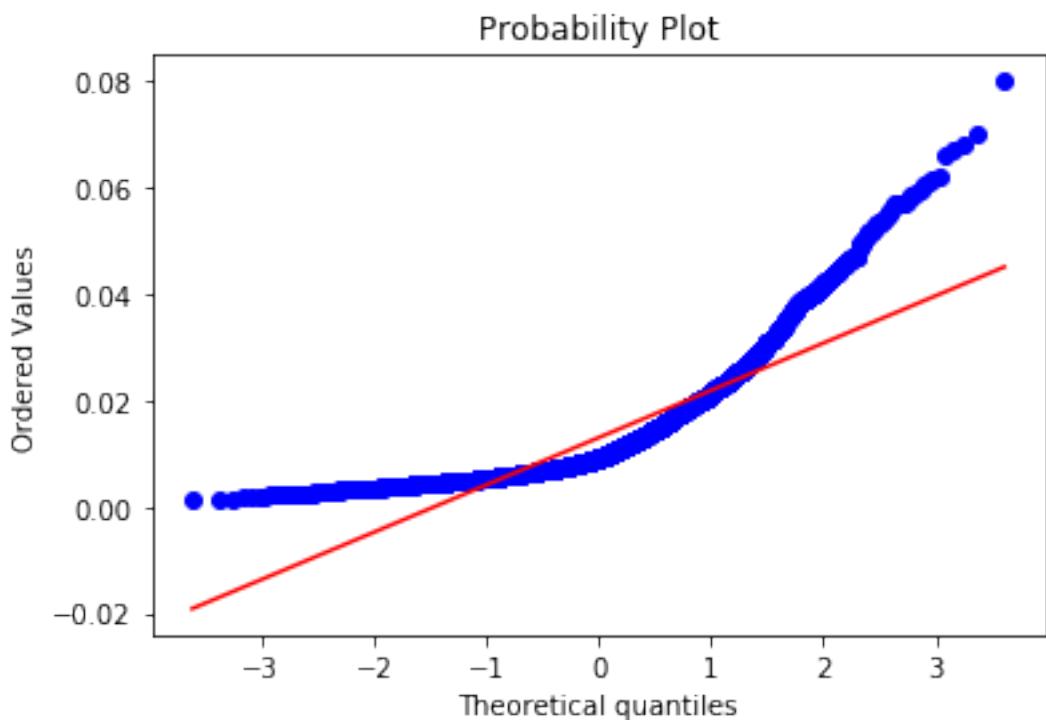
```
The mean error for lin100_net_flood_ is 0.03583395345285629 for length 4629  
=====
```

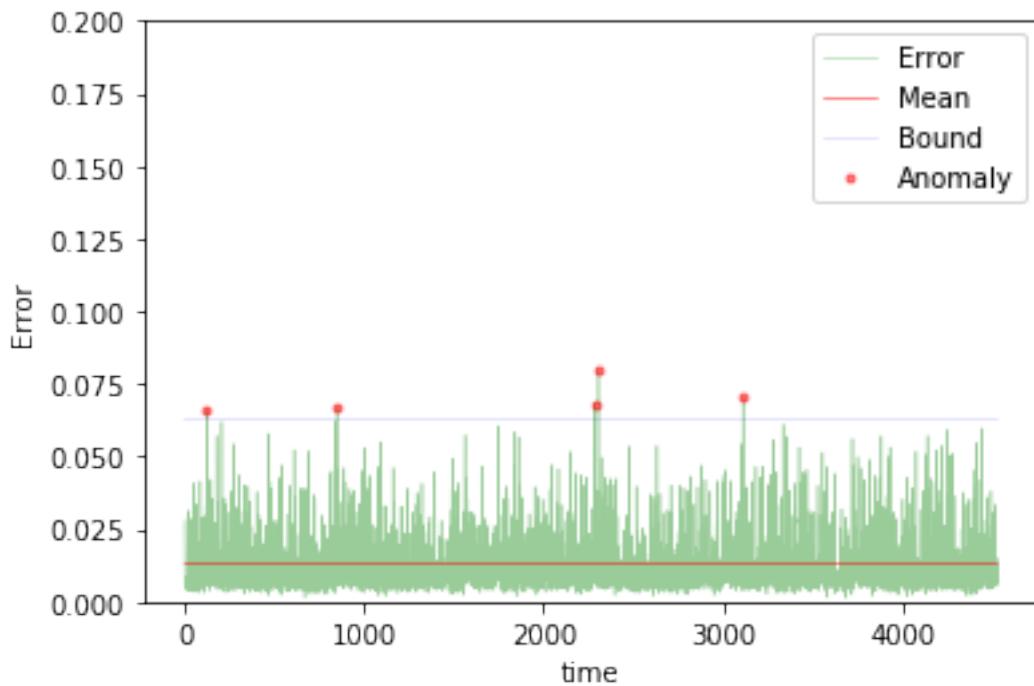
200 steps

```
In [60]: TIMESTEPS = 200  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "lin200"  
  
In [61]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
output = Dense(DIM, activation='sigmoid')(input_layer)  
  
In [62]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [63]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

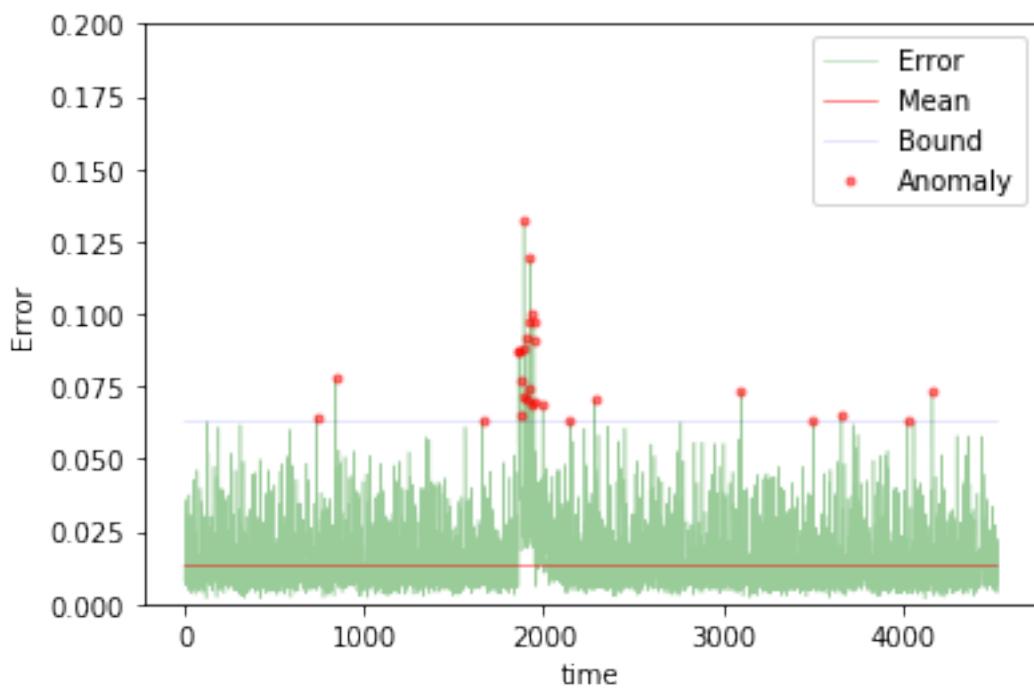


Training loss for final epoch is 0.014651732355356216
Validation loss for final epoch is 0.013700883424375206
----- Beginning tests for lin200 -----
Testing on normal data.

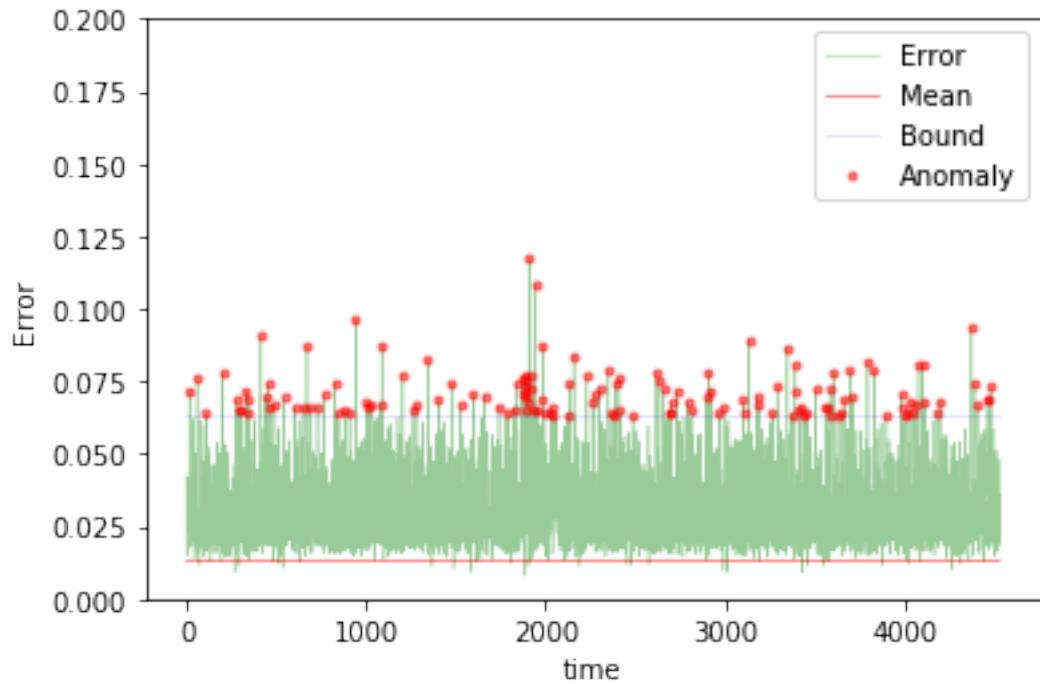




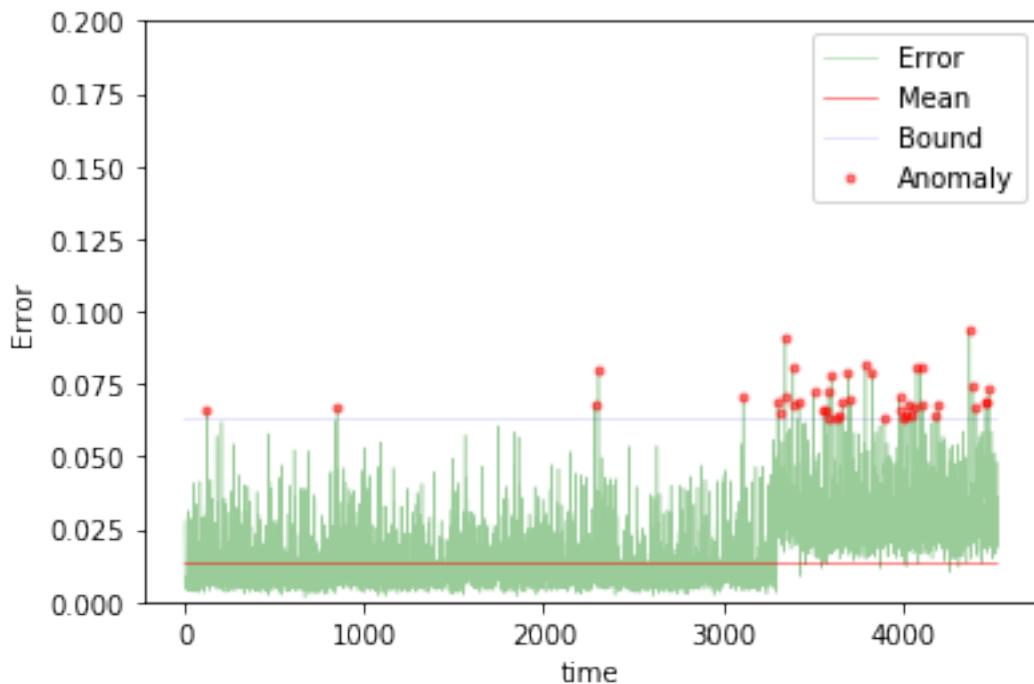
The mean error for lin200_normal_ is 0.012988427271719267 for length 4529
Testing on anomaly data.



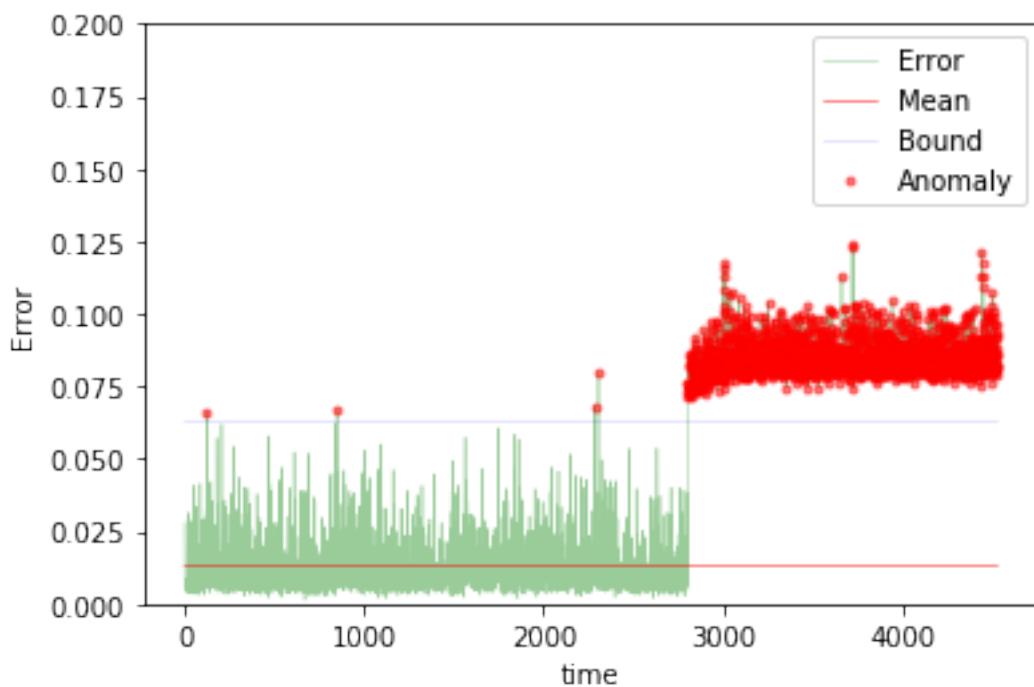
The mean error for lin200_anomaly_ is 0.01540420045229175 for length 4529
Testing on different app data.



The mean error for lin200_diff_app_ is 0.030497862121186662 for length 4529
Testing on App change synthetic data.



The mean error for lin200_app_change_ is 0.017762024479095927 for length 4529
Testing on Net flood synthetic data.

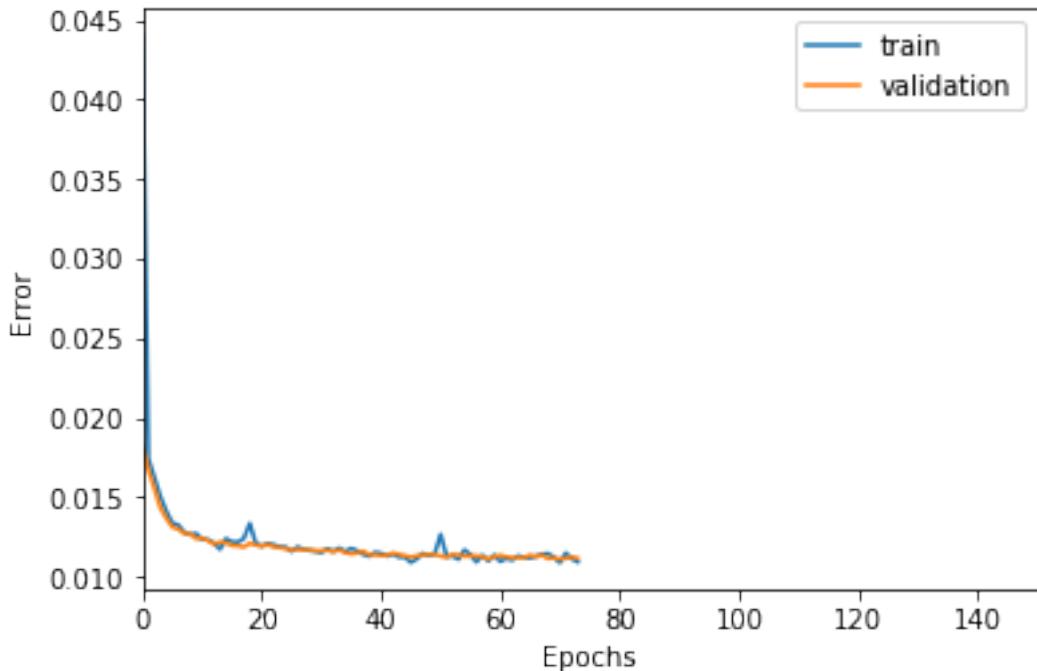


```
The mean error for lin200_net_flood_ is 0.04062379451994406 for length 4529  
=====
```

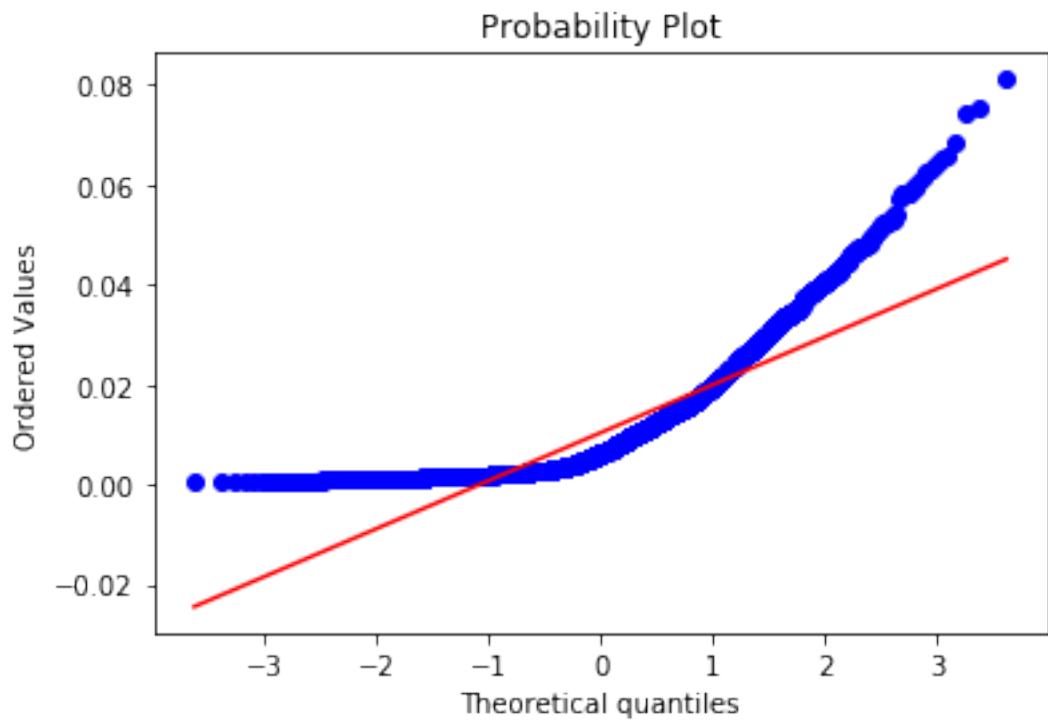
2.1.2 NN with 1 hidden layer

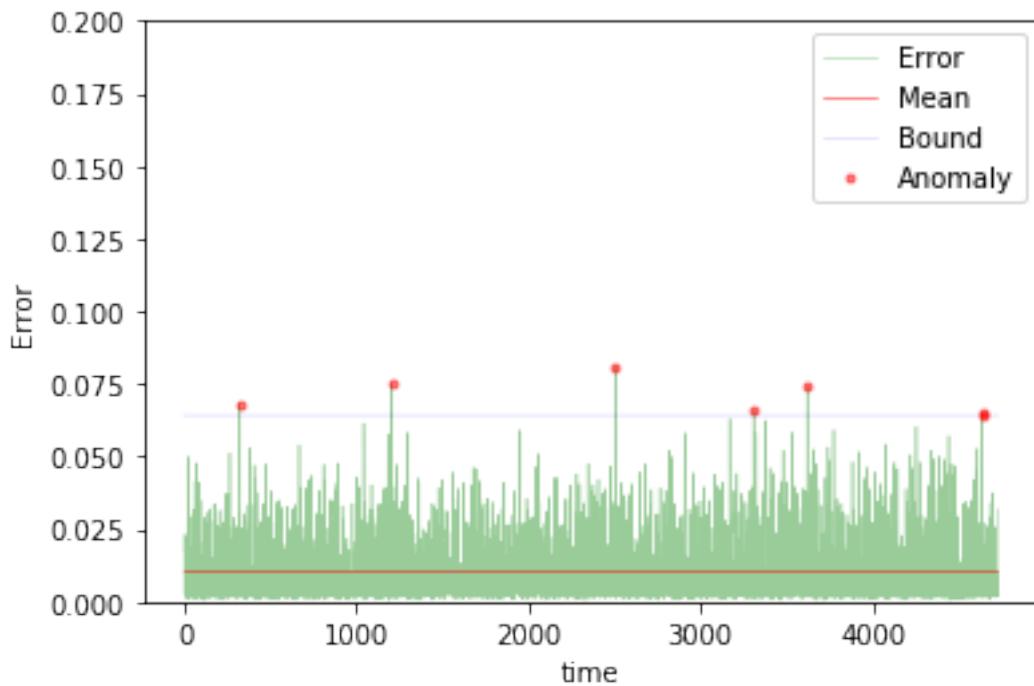
2 steps

```
In [64]: TIMESTEPS = 2  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_2"  
  
In [65]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [66]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [67]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

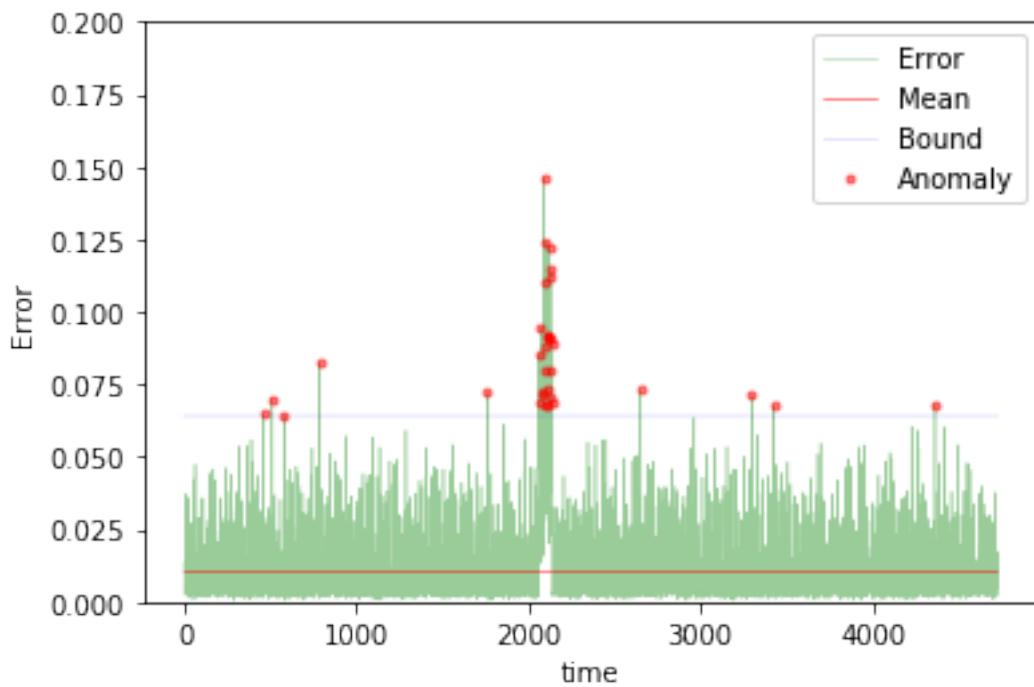


```
Training loss for final epoch is 0.010933455783408135
Validation loss for final epoch is 0.011155421051429585
----- Beginning tests for nn1_2 -----
Testing on normal data.
```

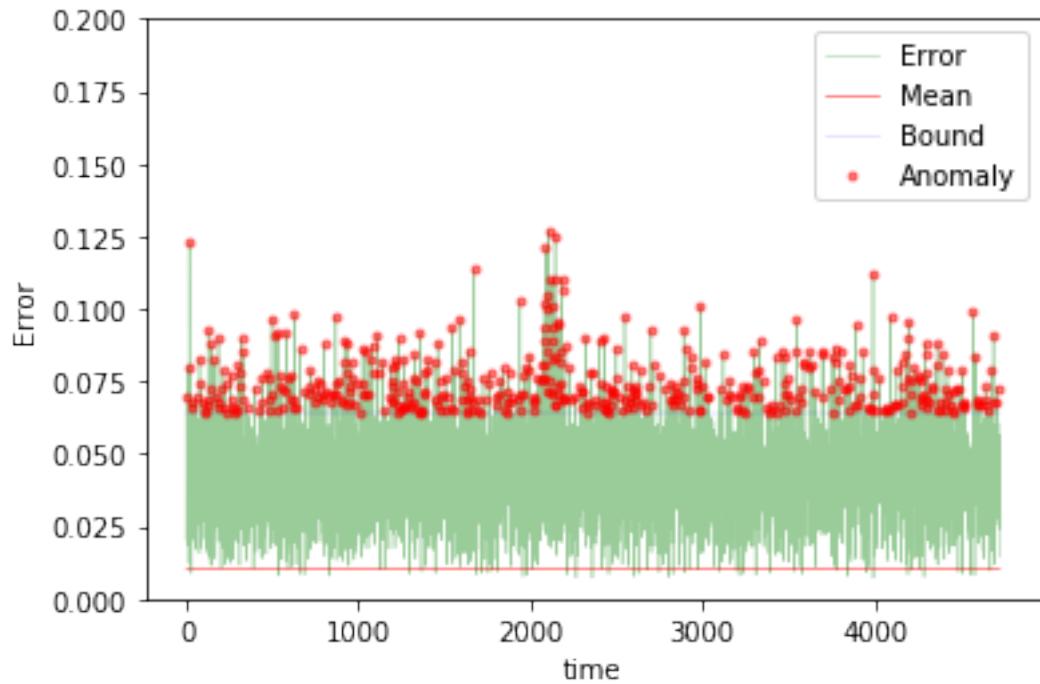




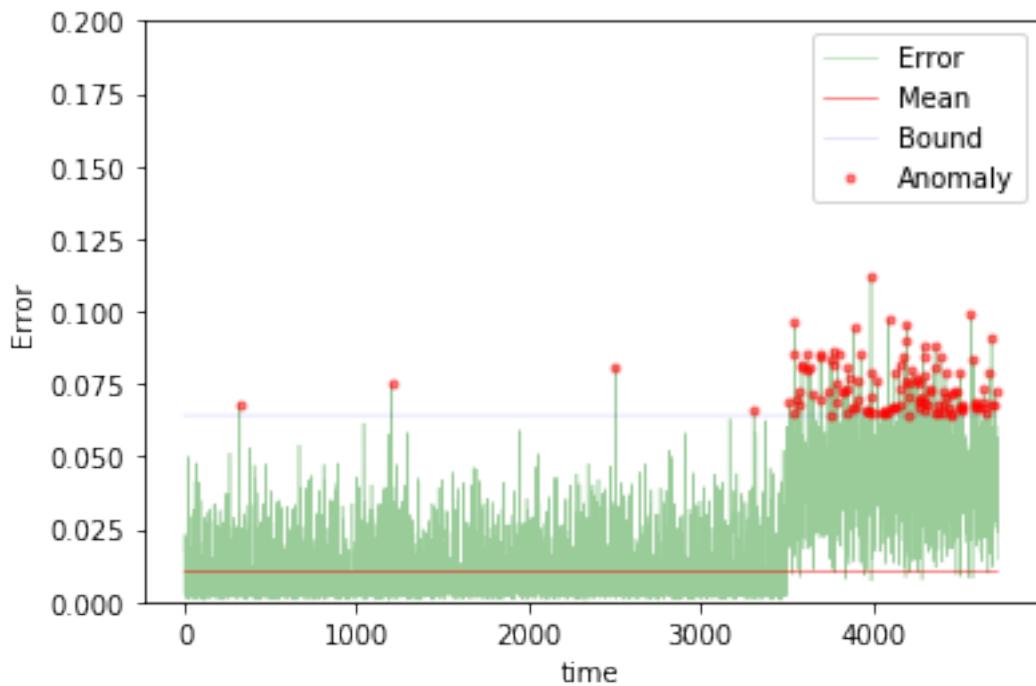
The mean error for nn1_2_normal_ is 0.010325911397999418 for length 4727
Testing on anomaly data.



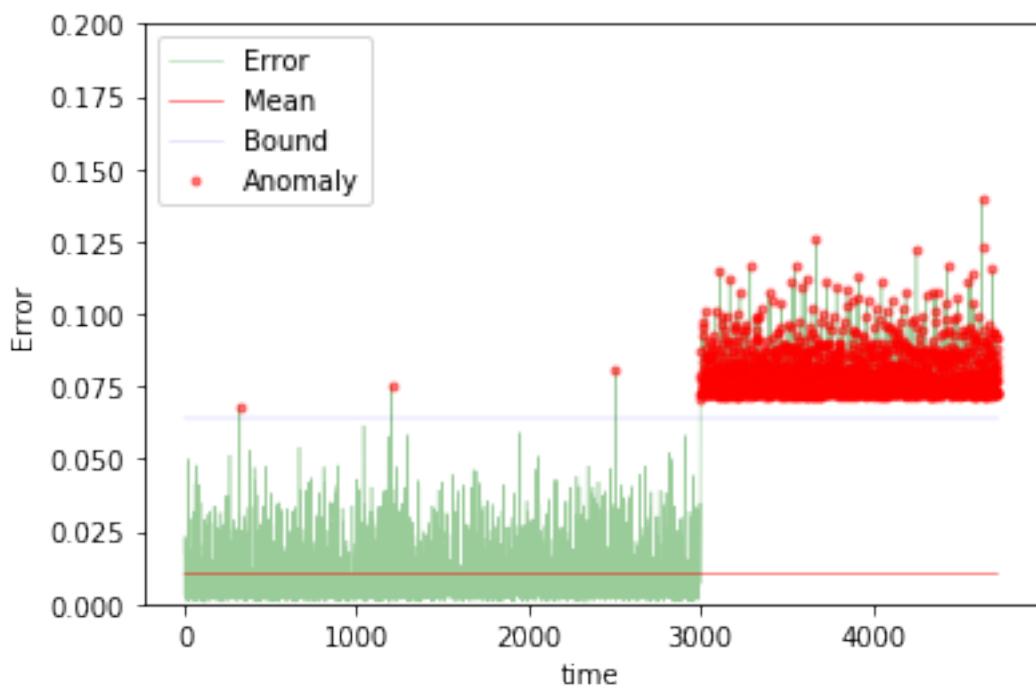
The mean error for nn1_2_anomaly_ is 0.012164337519586114 for length 4727
Testing on different app data.



The mean error for nn1_2_diff_app_ is 0.04288490639575245 for length 4727
Testing on App change synthetic data.



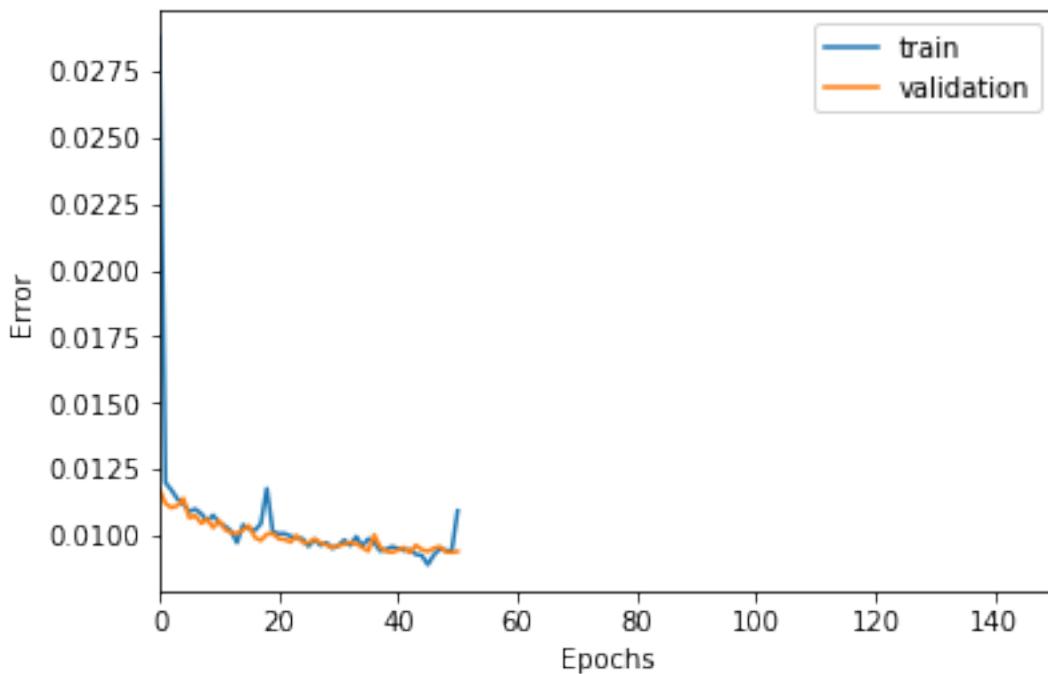
The mean error for nn1_2_app_change_ is 0.018694403838060834 for length 4727
 Testing on Net flood synthetic data.



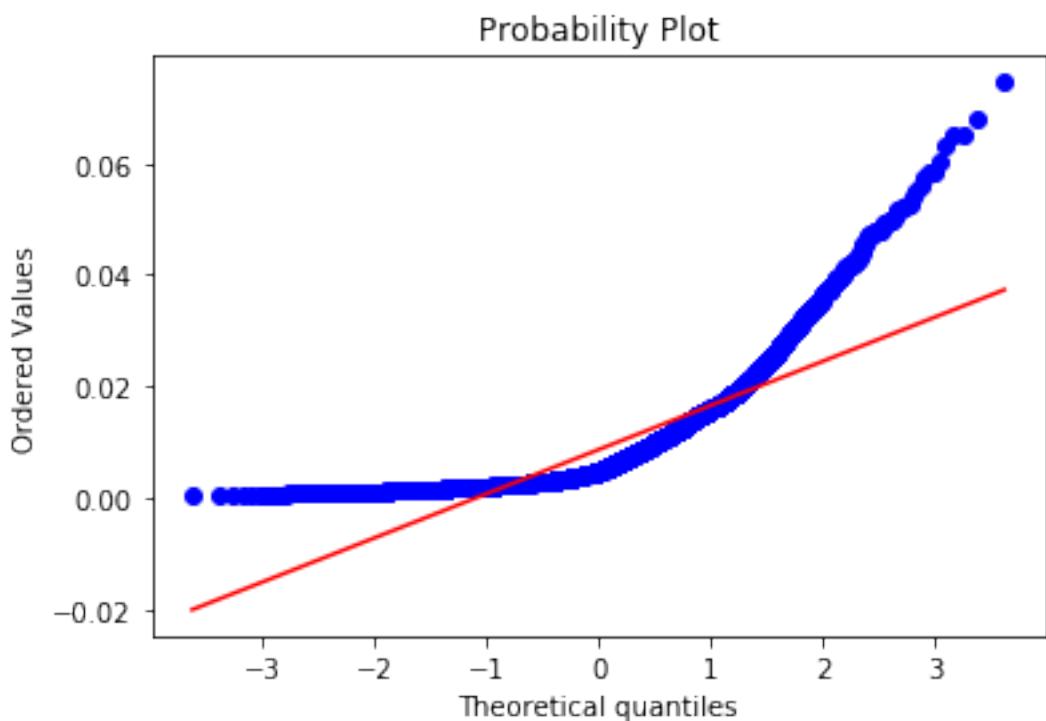
```
The mean error for nn1_2_net_flood_ is 0.0355011529593851 for length 4727  
=====
```

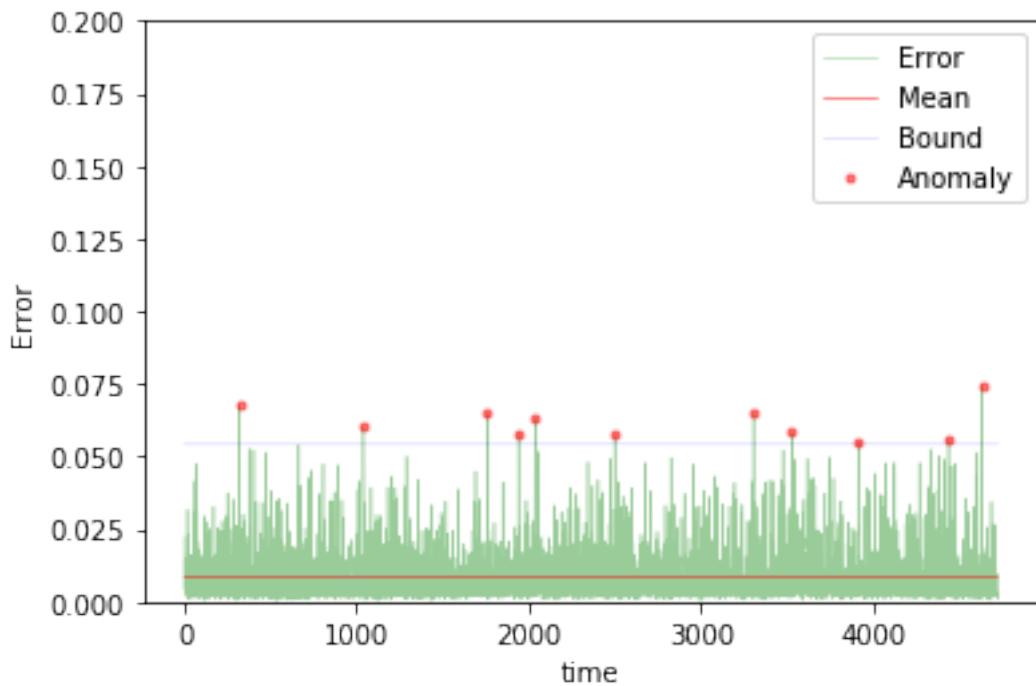
5 steps

```
In [68]: TIMESTEPS = 5  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_5"  
  
In [69]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [70]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [71]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

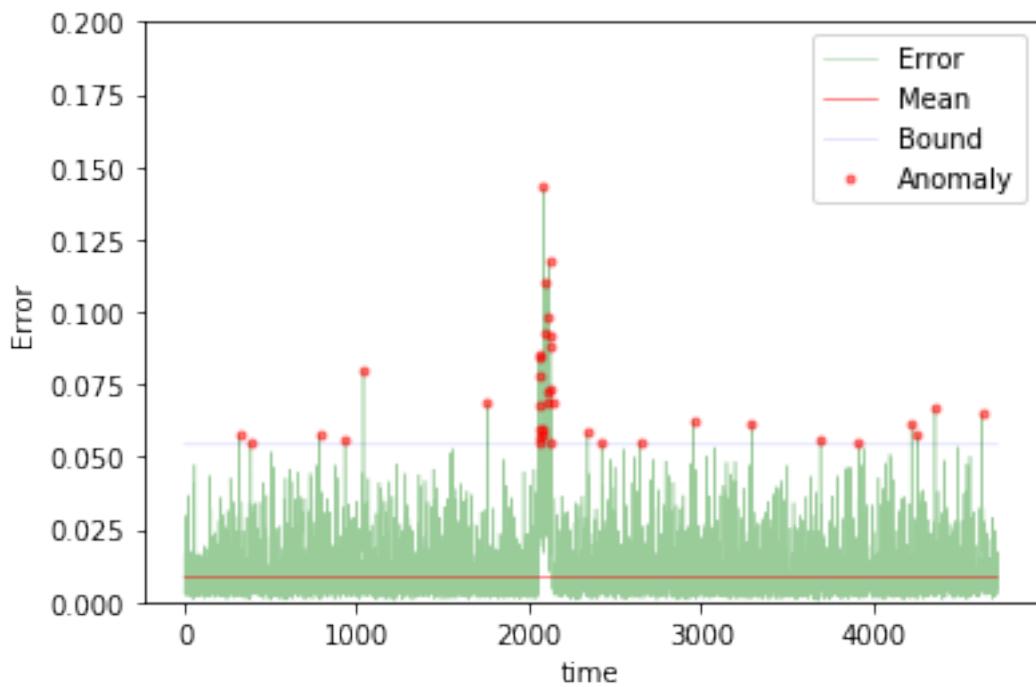


```
Training loss for final epoch is 0.010923392255790532
Validation loss for final epoch is 0.009385330804623664
----- Beginning tests for nn1_5 -----
Testing on normal data.
```

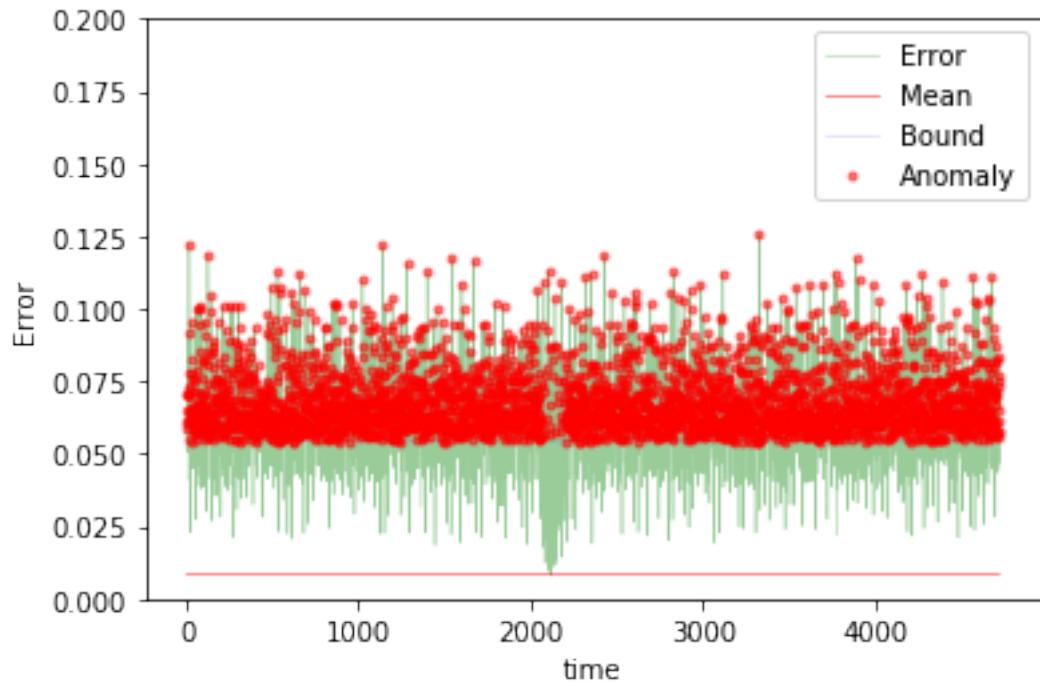




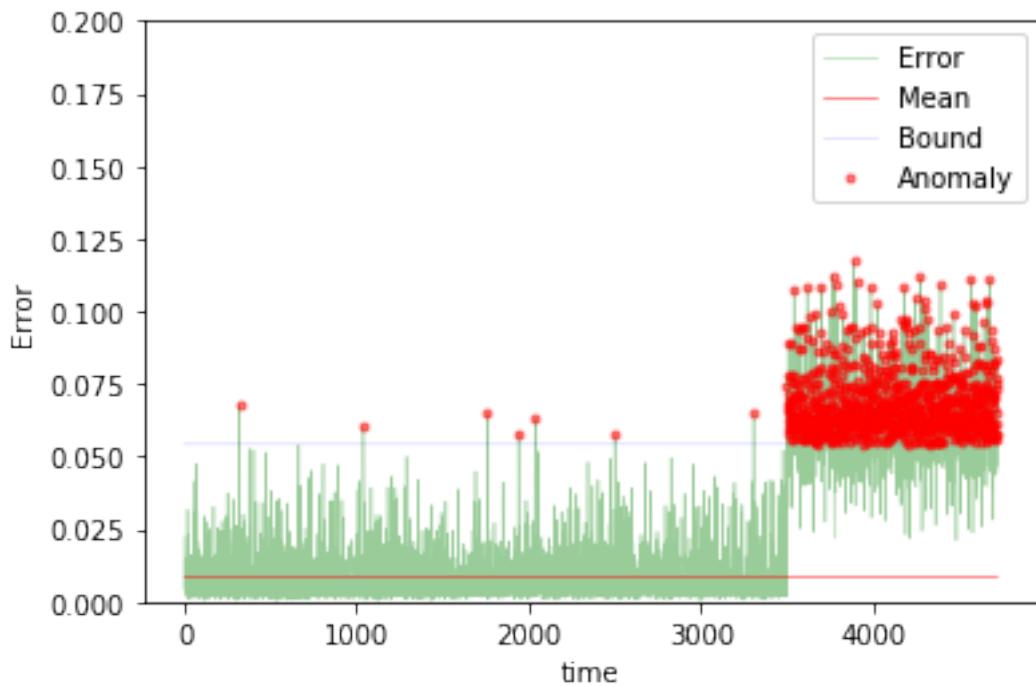
The mean error for nn1_5_normal_ is 0.008623914729641313 for length 4724
Testing on anomaly data.



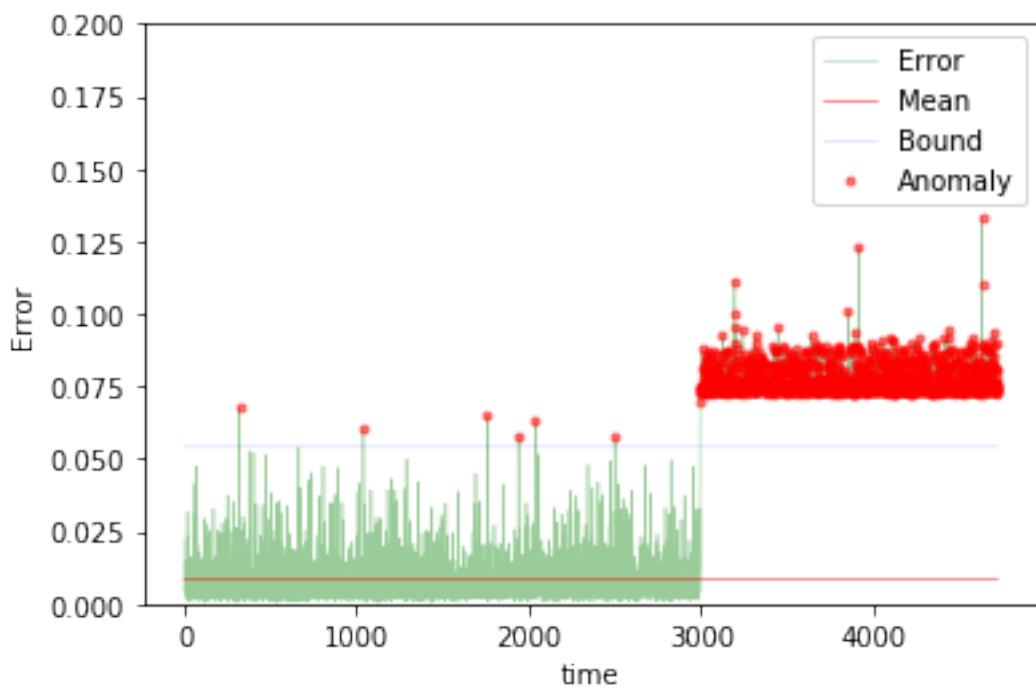
The mean error for nn1_5_anomaly_ is 0.010506085204327099 for length 4724
Testing on different app data.



The mean error for nn1_5_diff_app_ is 0.06226930880634035 for length 4724
Testing on App change synthetic data.



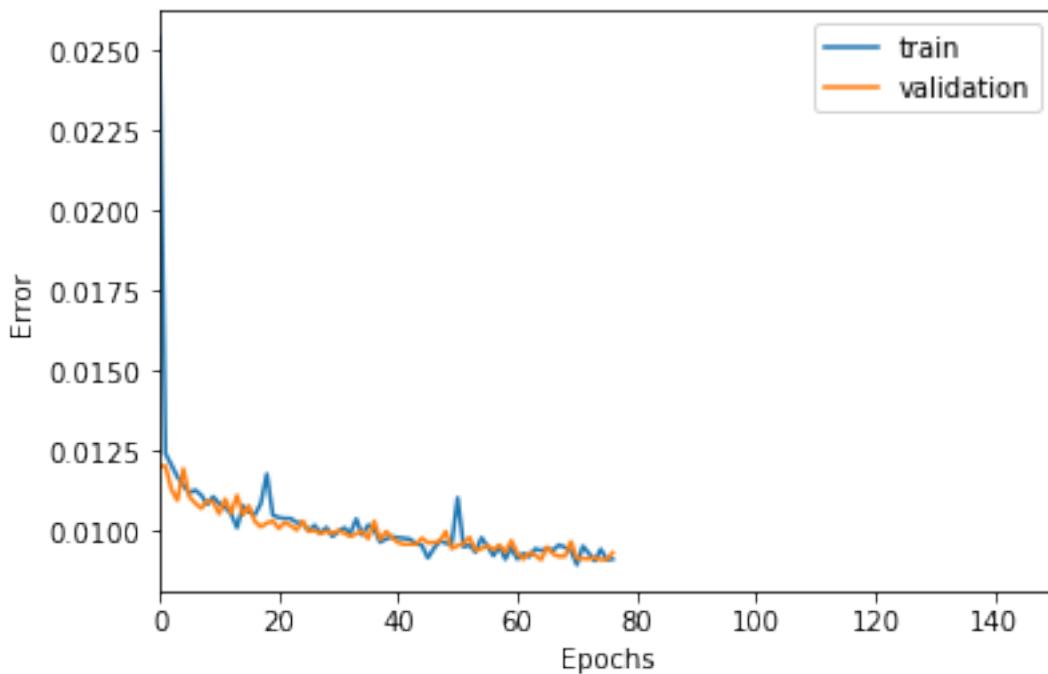
The mean error for nn1_5_app_change_ is 0.022629163828470328 for length 4724
Testing on Net flood synthetic data.



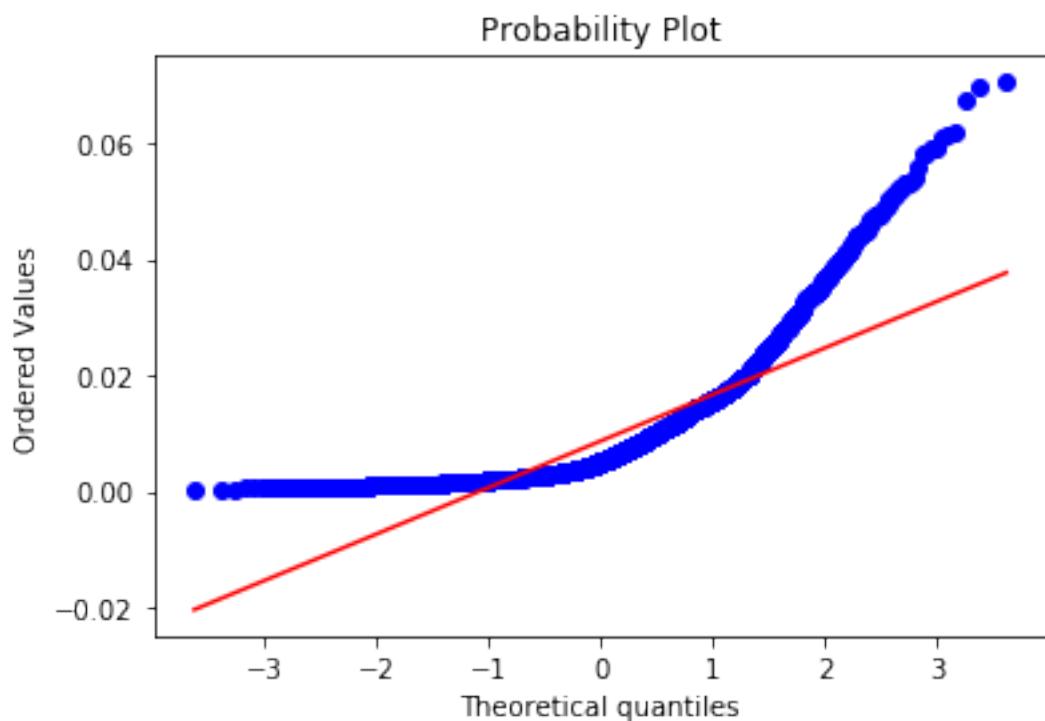
```
The mean error for nn1_5_net_flood_ is 0.03360020521585605 for length 4724  
=====
```

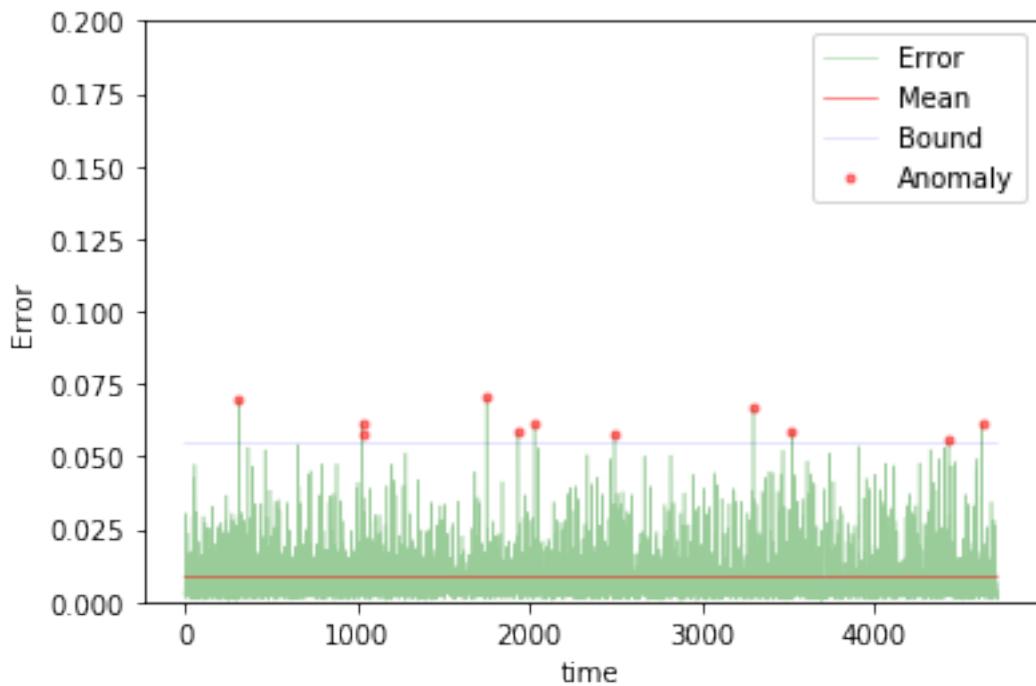
10 steps

```
In [72]: TIMESTEPS = 10  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_10"  
  
In [73]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [74]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [75]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

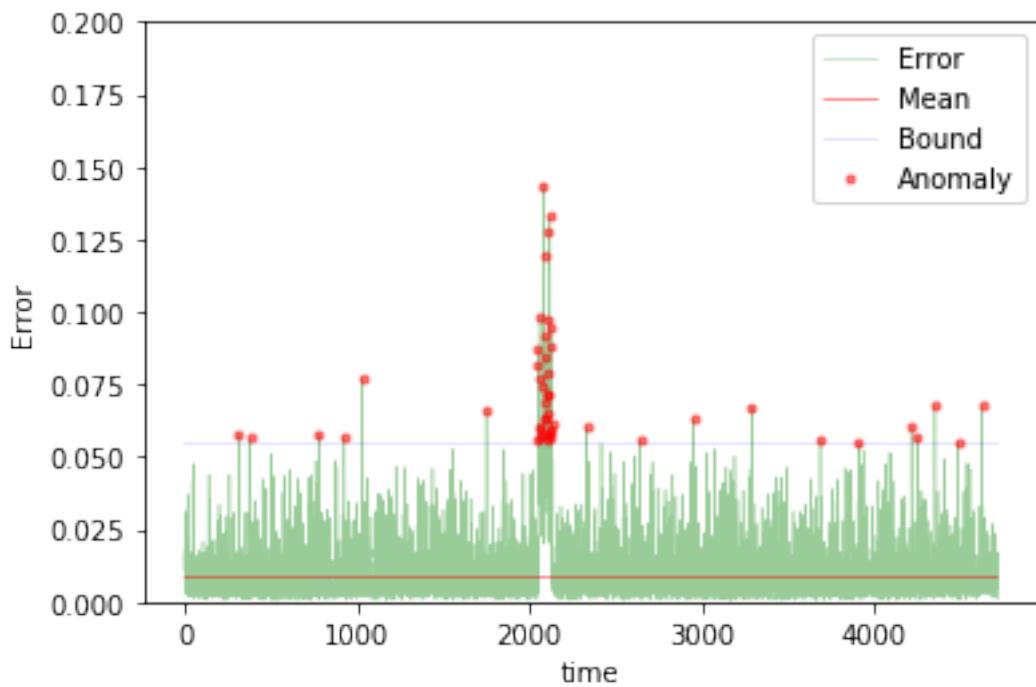


```
Training loss for final epoch is 0.009124995711492374
Validation loss for final epoch is 0.009325276446761564
----- Beginning tests for nn1_10 -----
Testing on normal data.
```

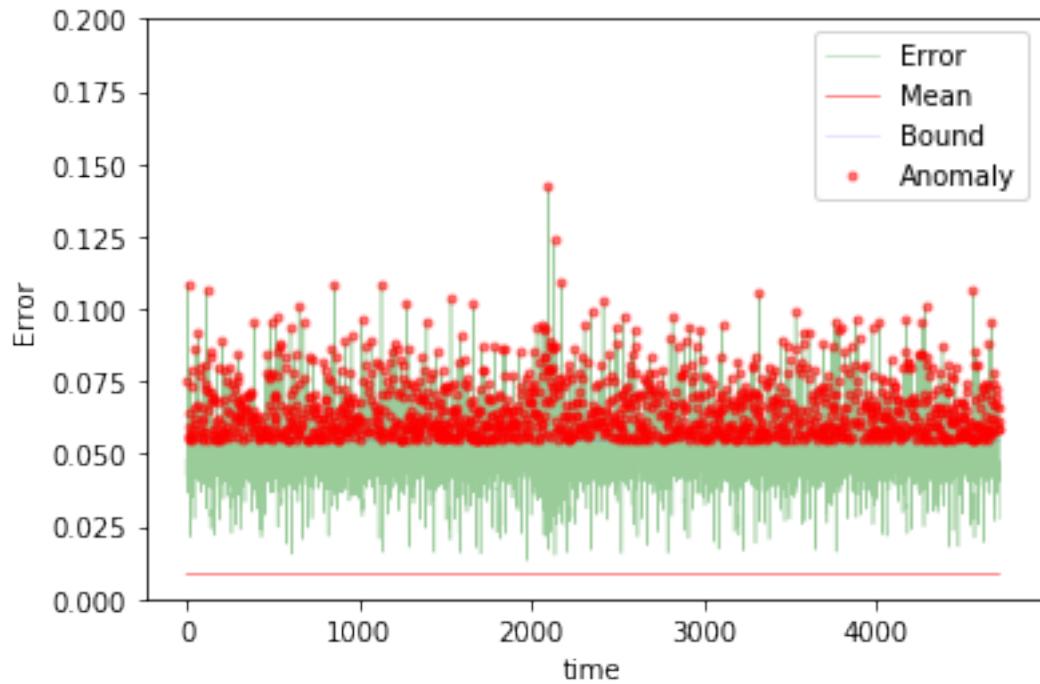




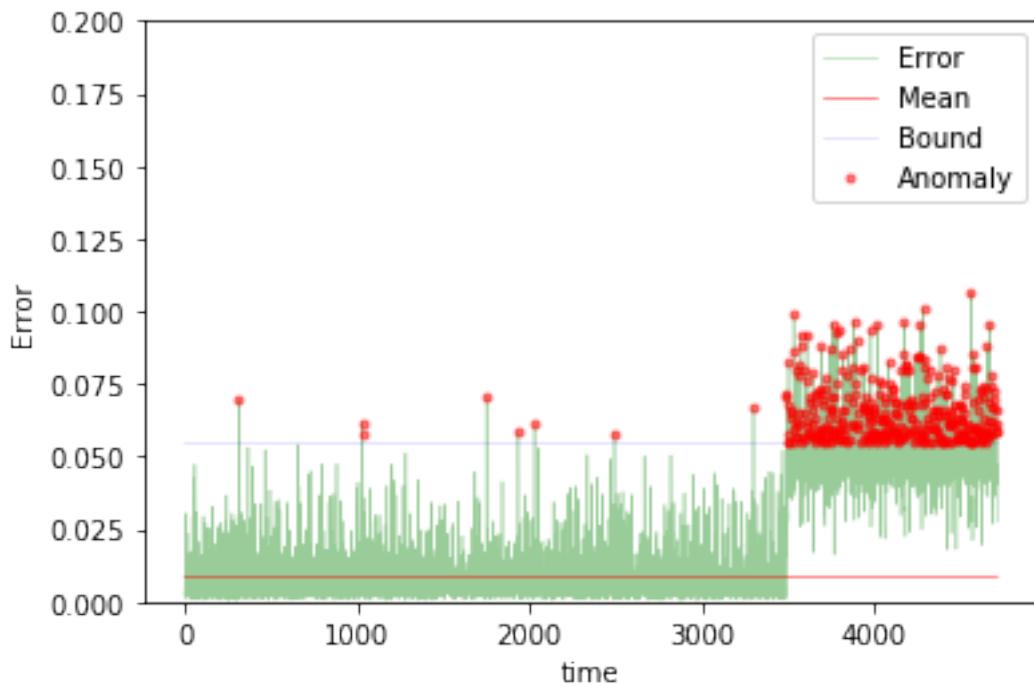
The mean error for nn1_10_normal_ is 0.008634350071451477 for length 4719
Testing on anomaly data.



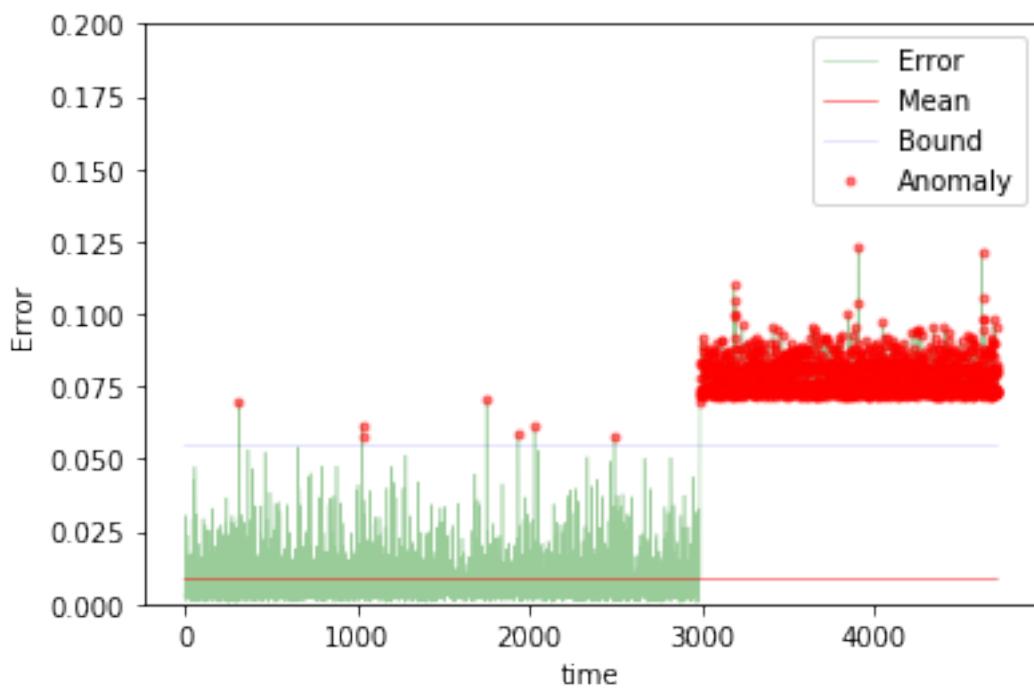
The mean error for nn1_10_anomaly_ is 0.01046592411186496 for length 4719
Testing on different app data.



The mean error for nn1_10_diff_app_ is 0.05050261757309034 for length 4719
Testing on App change synthetic data.



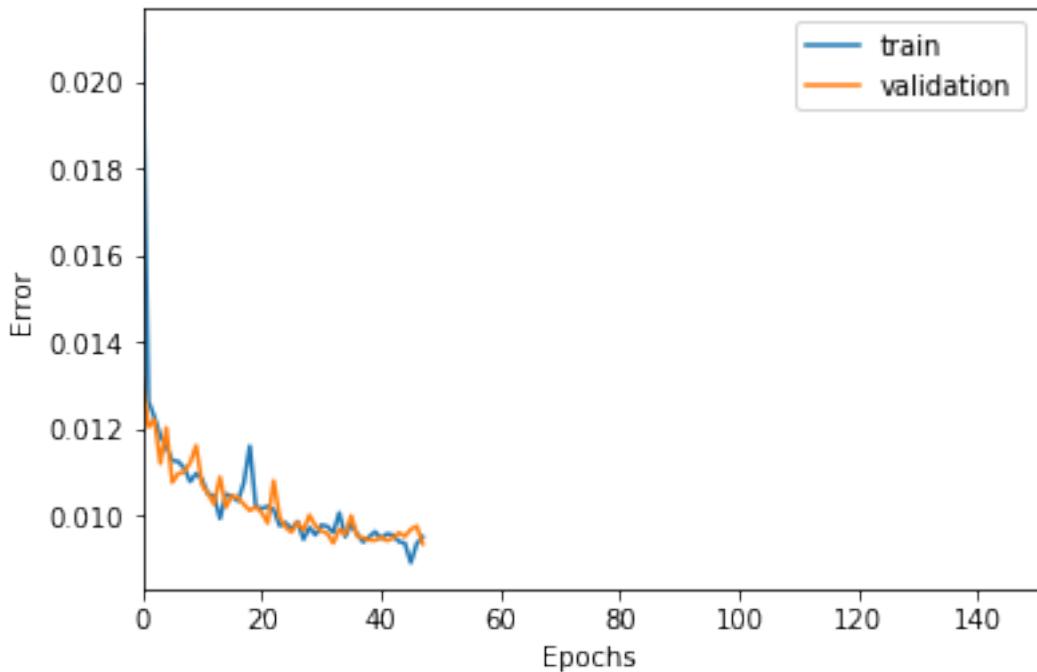
The mean error for nn1_10_app_change_ is 0.019469753865091588 for length 4719
Testing on Net flood synthetic data.



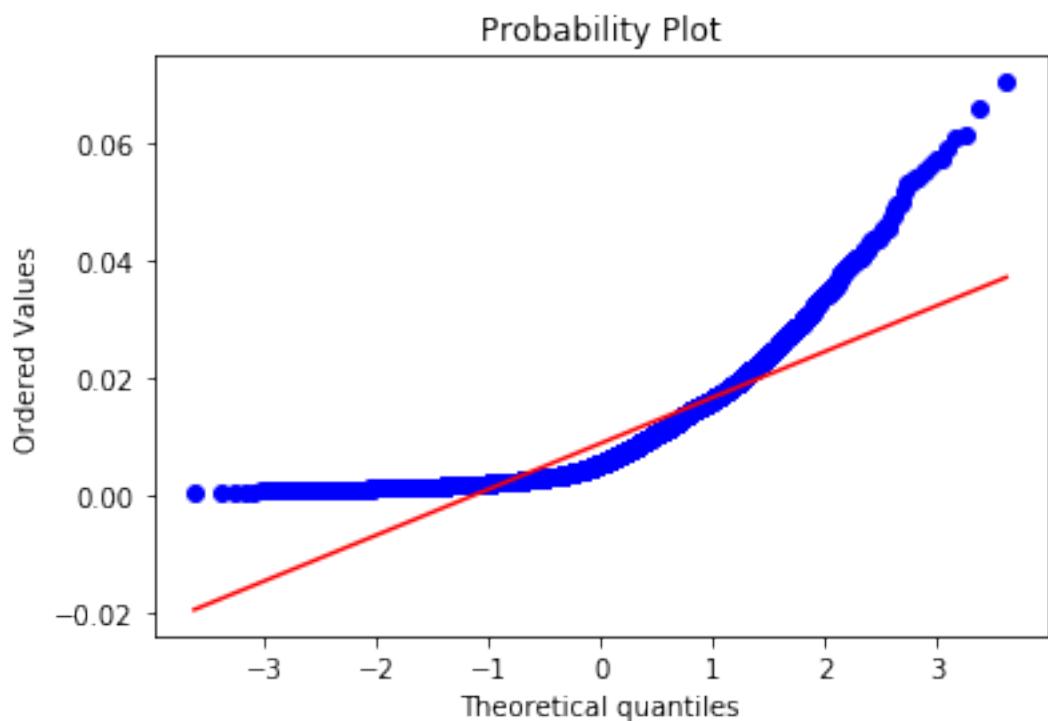
```
The mean error for nn1_10_net_flood_ is 0.03393797911682255 for length 4719  
=====
```

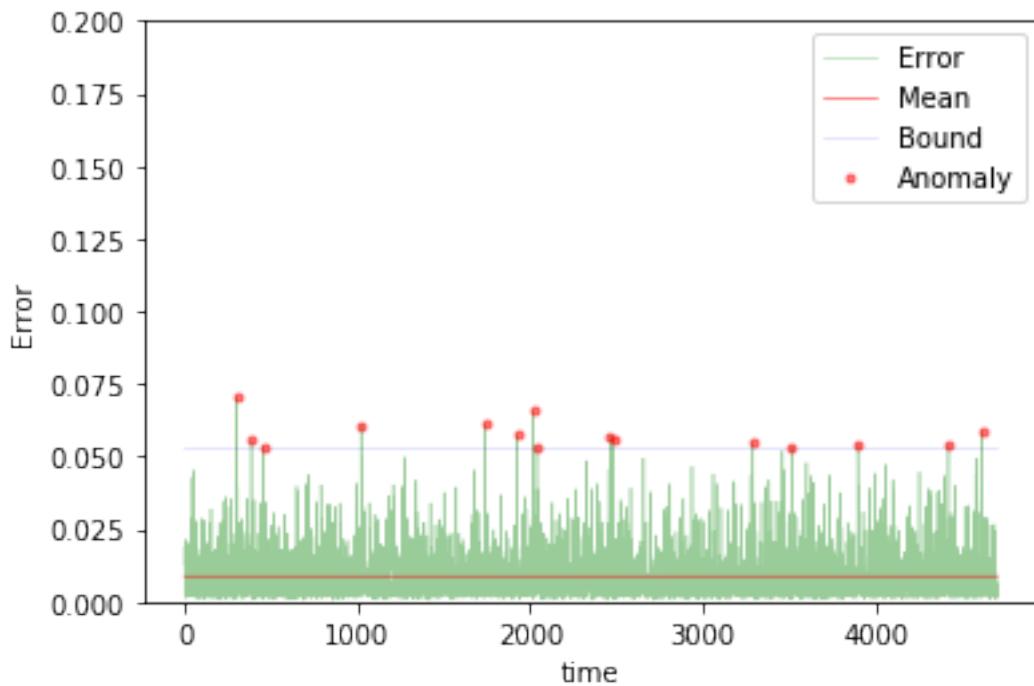
20 steps

```
In [76]: TIMESTEPS = 20  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_20"  
  
In [77]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100,activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [78]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [79]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

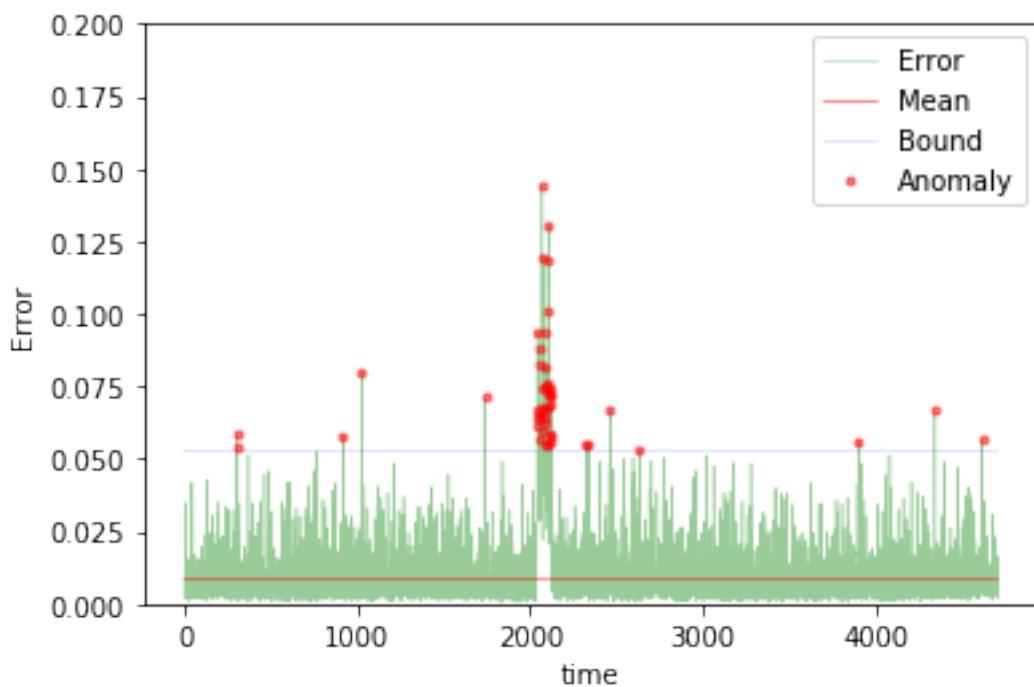


```
Training loss for final epoch is 0.009524953383137472
Validation loss for final epoch is 0.009363794193370268
----- Beginning tests for nn1_20 -----
Testing on normal data.
```

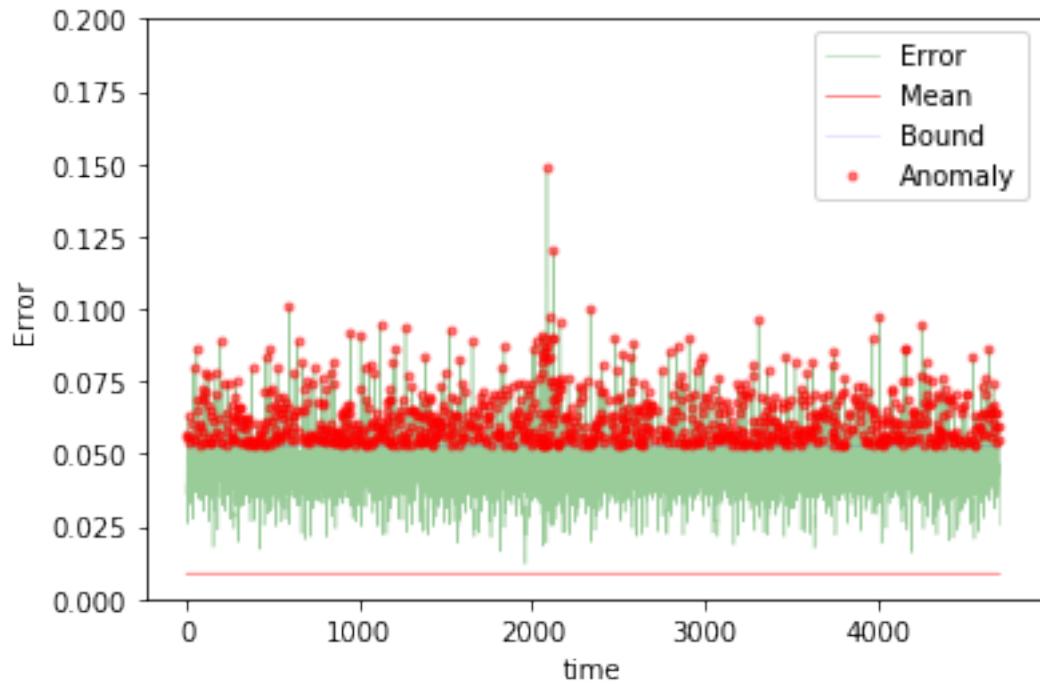




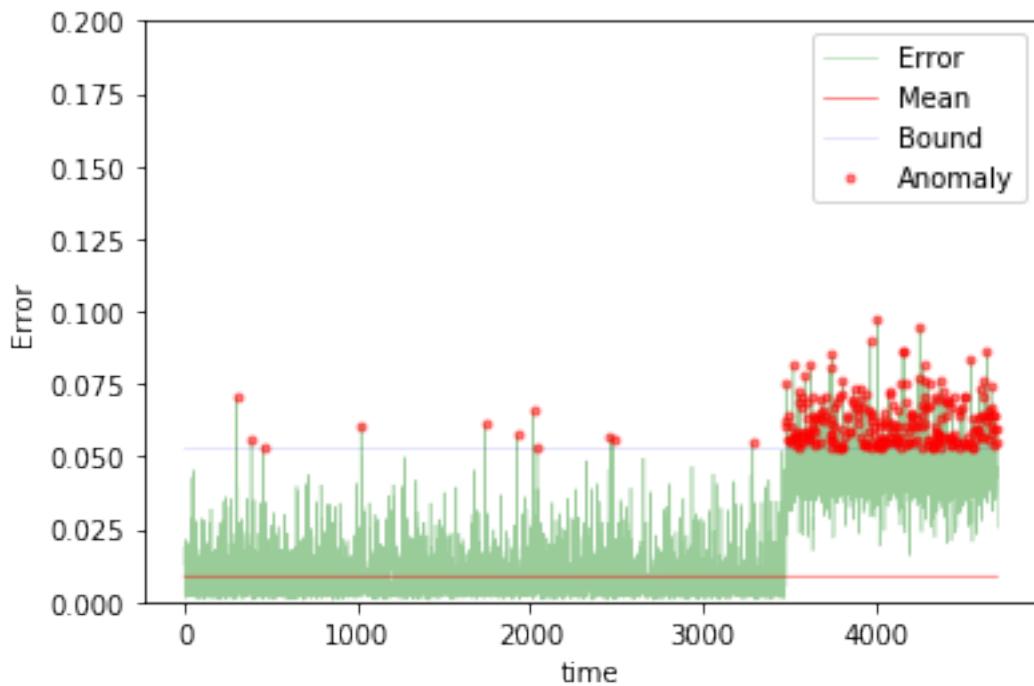
The mean error for nn1_20_normal_ is 0.008833842808512574 for length 4709
Testing on anomaly data.



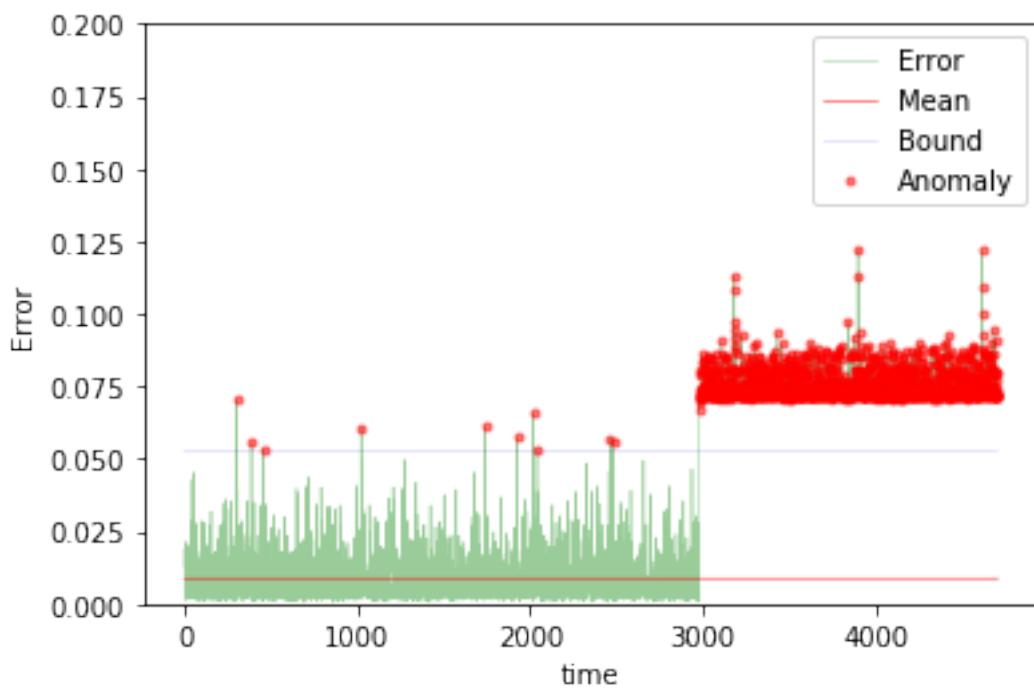
The mean error for nn1_20_anomaly_ is 0.010228371792209713 for length 4709
Testing on different app data.



The mean error for nn1_20_diff_app_ is 0.04552375215924373 for length 4709
Testing on App change synthetic data.



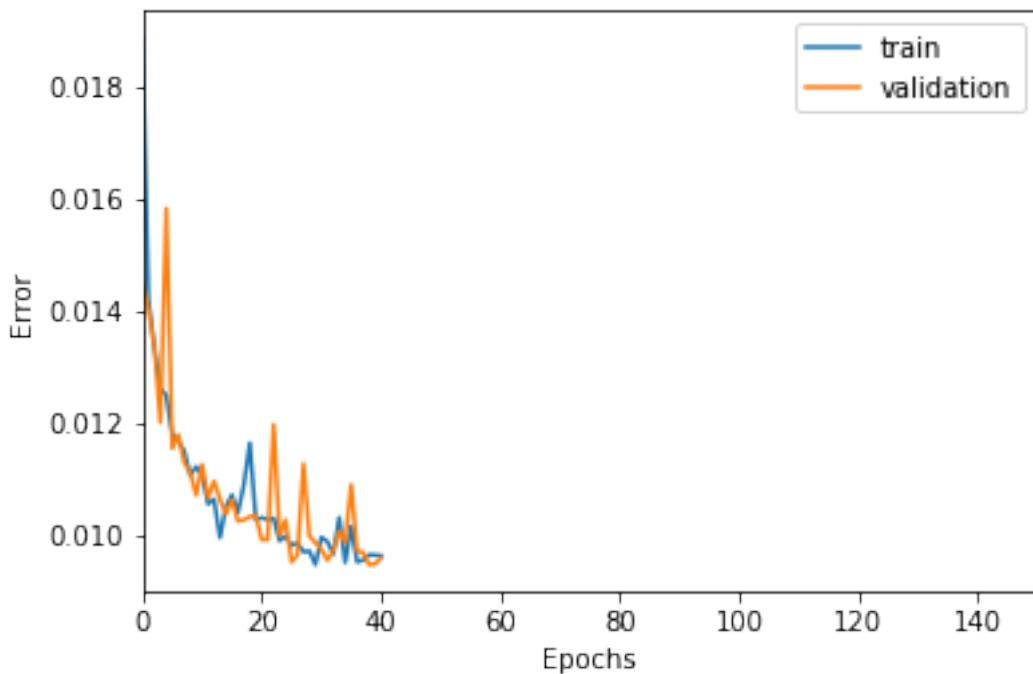
The mean error for nn1_20_app_change_ is 0.018289682179757214 for length 4709
Testing on Net flood synthetic data.



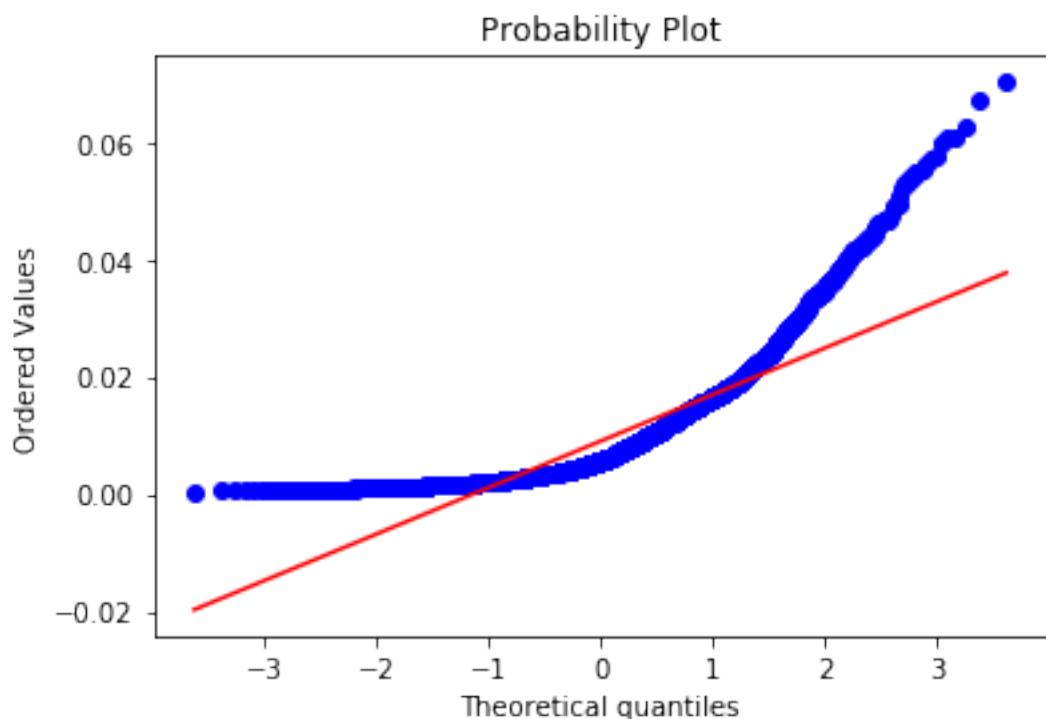
```
The mean error for nn1_20_net_flood_ is 0.03324098603248573 for length 4709  
=====
```

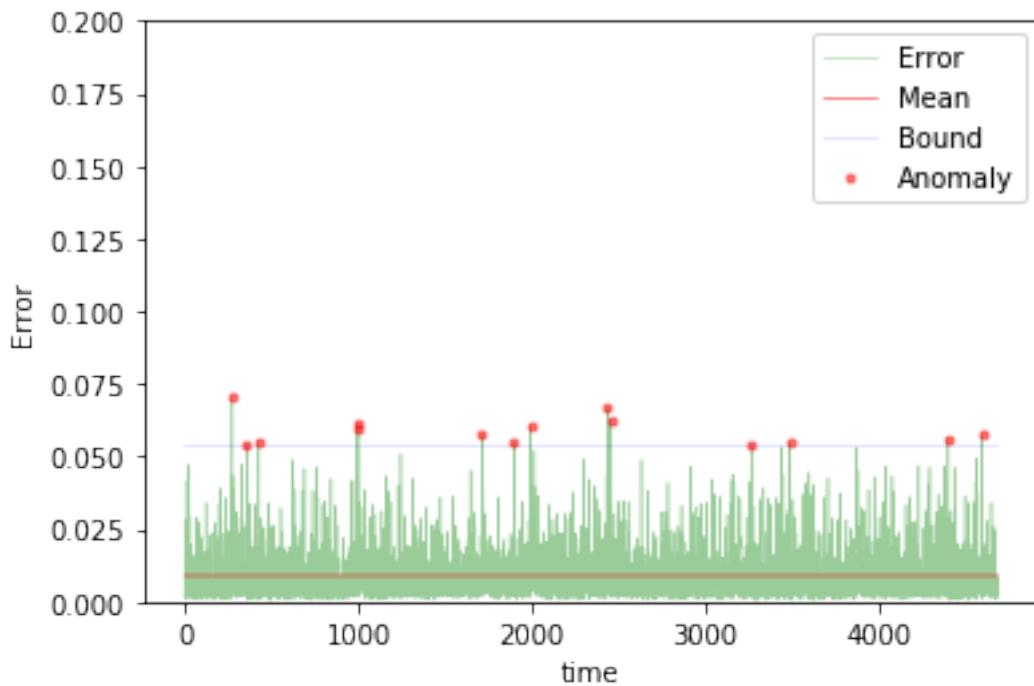
50 steps

```
In [80]: TIMESTEPS = 50  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_50"  
  
In [81]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100,activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [82]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [83]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

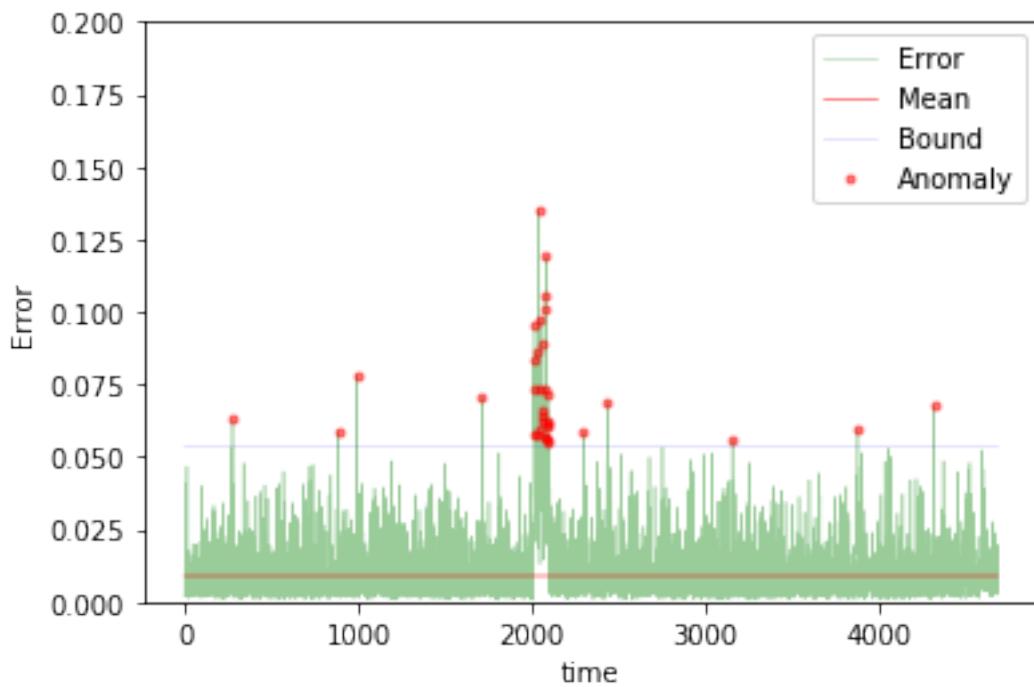


Training loss for final epoch is 0.009625236967694946
Validation loss for final epoch is 0.009584366638562642
----- Beginning tests for nn1_50 -----
Testing on normal data.

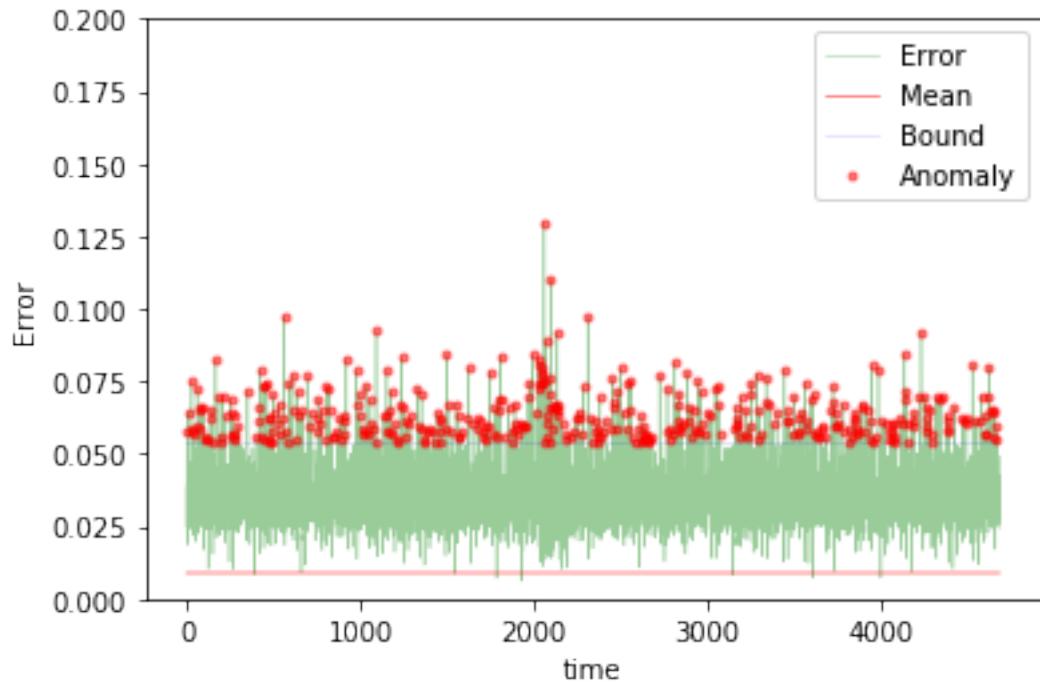




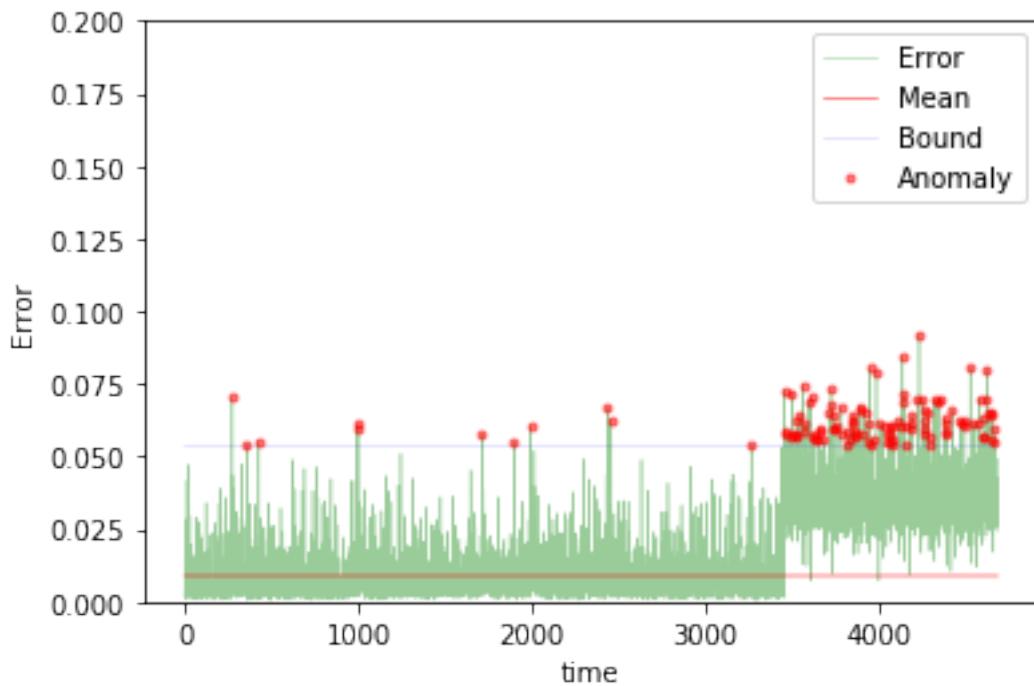
The mean error for nn1_50_normal_ is 0.009011833108668973 for length 4679
Testing on anomaly data.



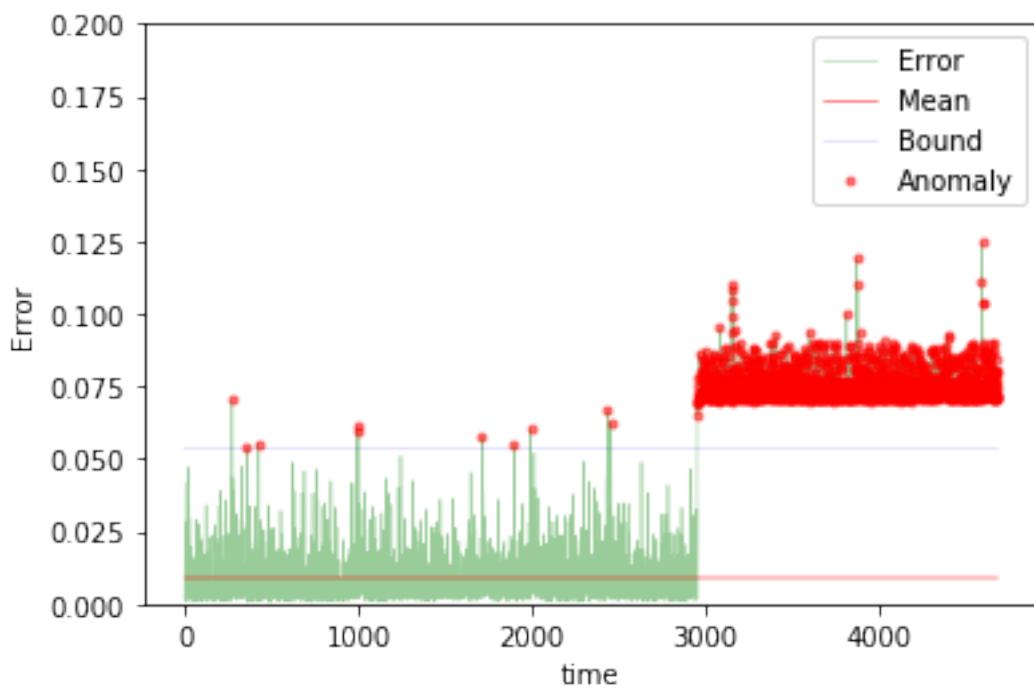
The mean error for nn1_50_anomaly_ is 0.010579345968987408 for length 4679
Testing on different app data.



The mean error for nn1_50_diff_app_ is 0.03562899358450676 for length 4679
Testing on App change synthetic data.



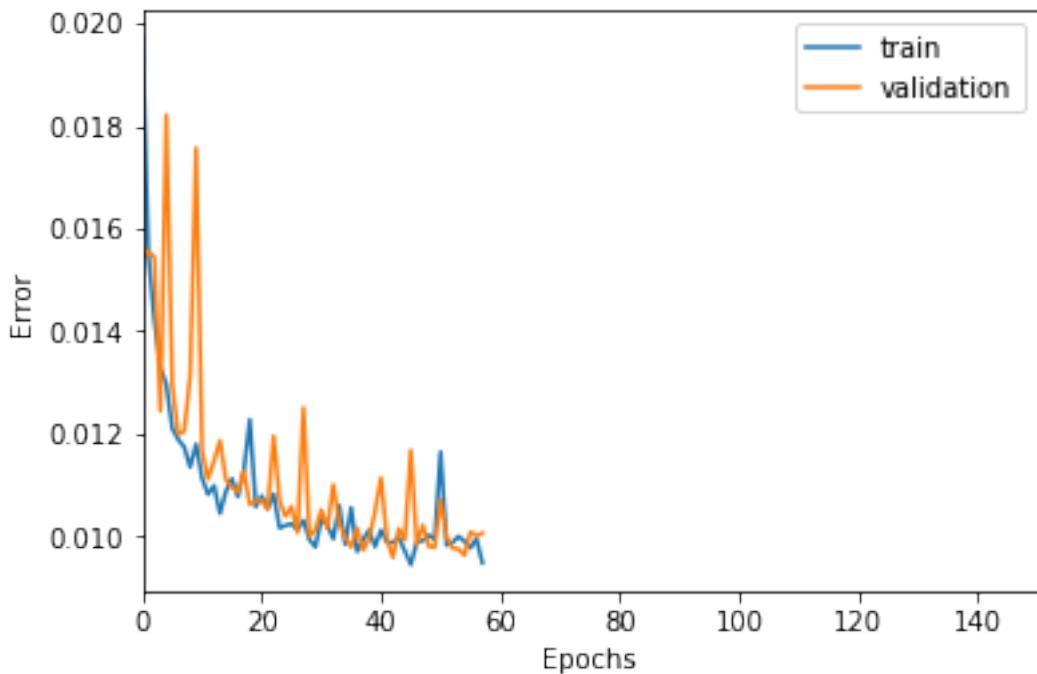
The mean error for nn1_50_app_change_ is 0.0159497086934552 for length 4679
Testing on Net flood synthetic data.



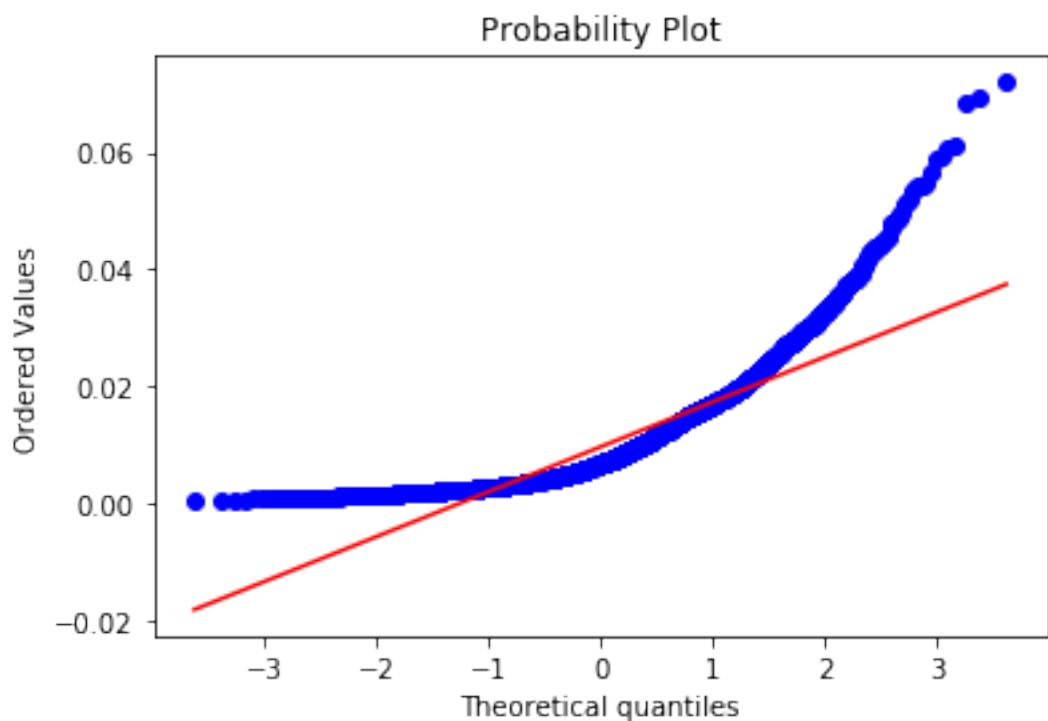
```
The mean error for nn1_50_net_flood_ is 0.03368291176983559 for length 4679  
=====
```

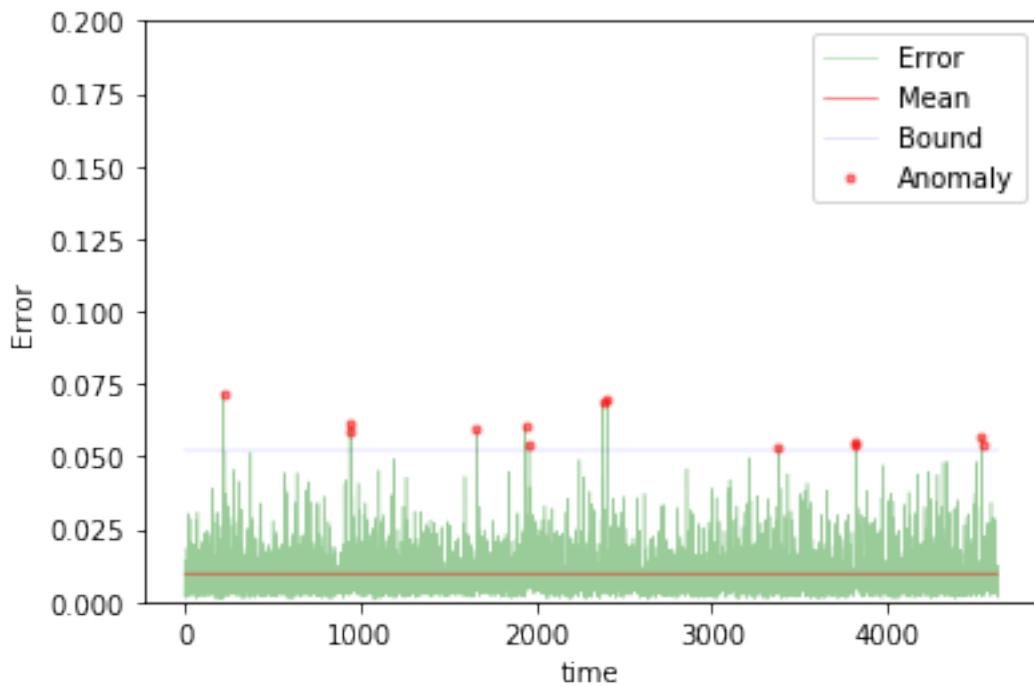
100 steps

```
In [84]: TIMESTEPS = 100  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_100"  
  
In [85]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100,activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [86]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [87]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

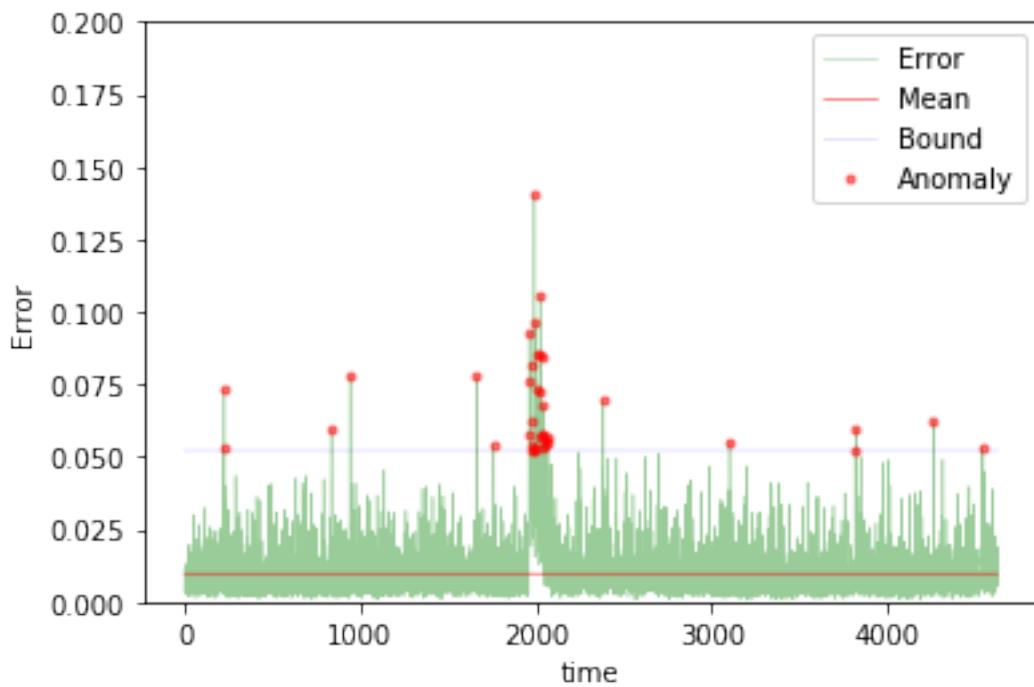


```
Training loss for final epoch is 0.009488005834049545
Validation loss for final epoch is 0.010069758449681103
----- Beginning tests for nn1_100 -----
Testing on normal data.
```

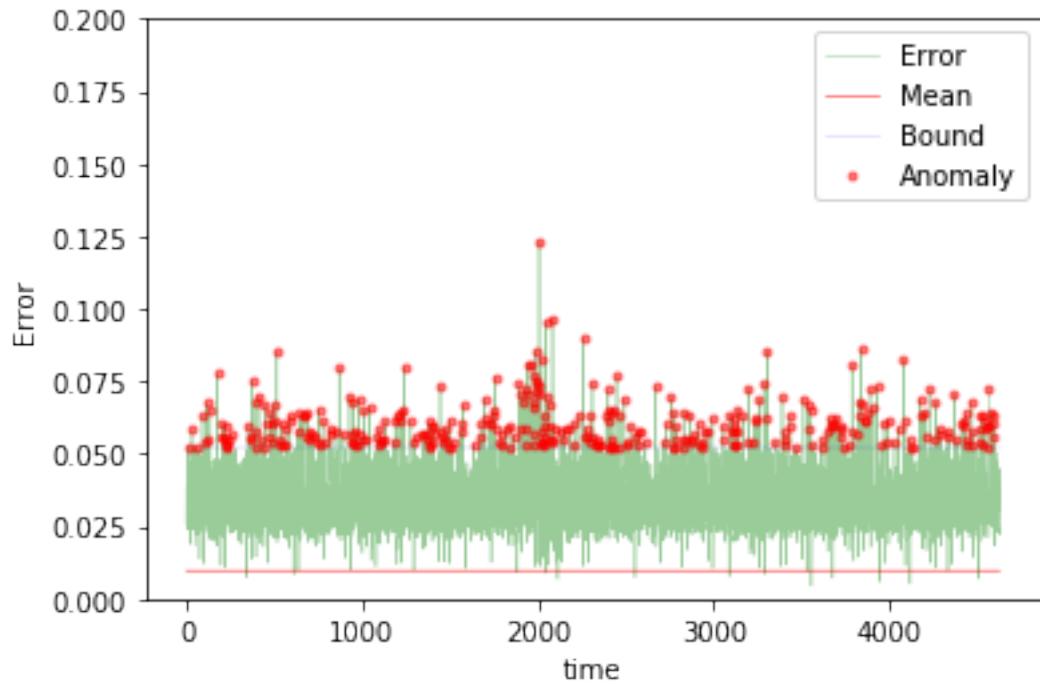




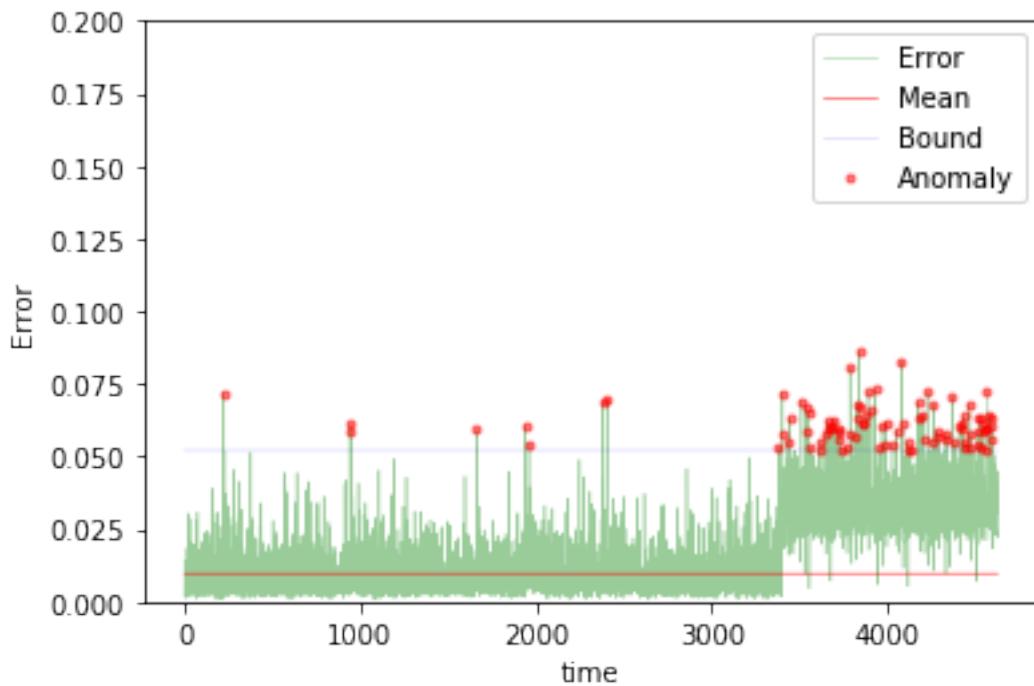
The mean error for nn1_100_normal_ is 0.00977250479489303 for length 4629
Testing on anomaly data.



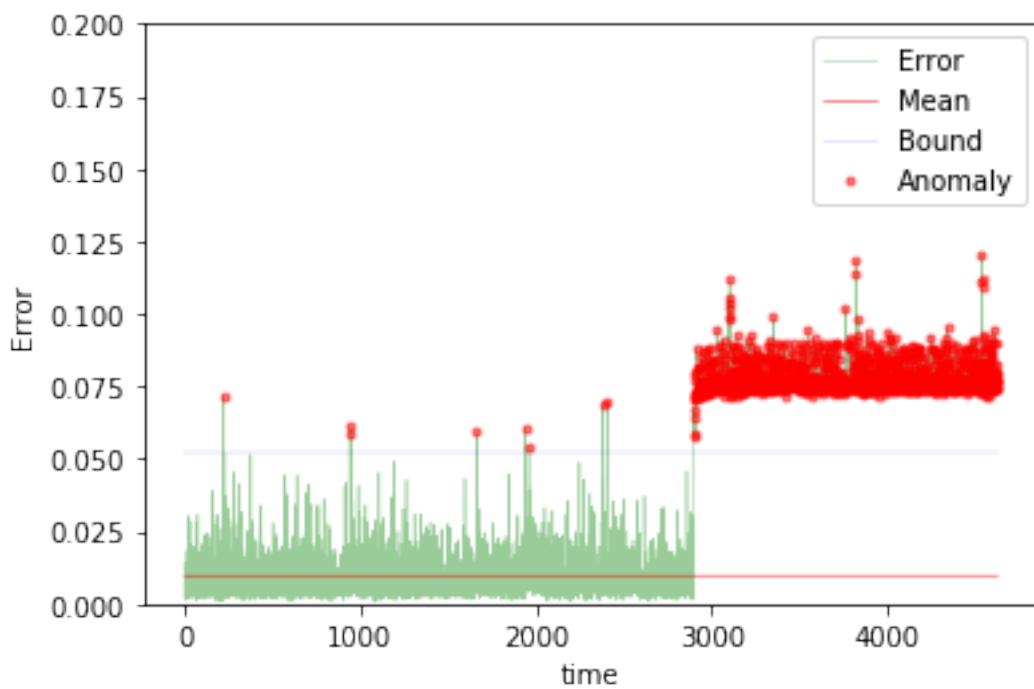
The mean error for nn1_100_anomaly_ is 0.01093115554166877 for length 4629
Testing on different app data.



The mean error for nn1_100_diff_app_ is 0.033843243248906814 for length 4629
Testing on App change synthetic data.



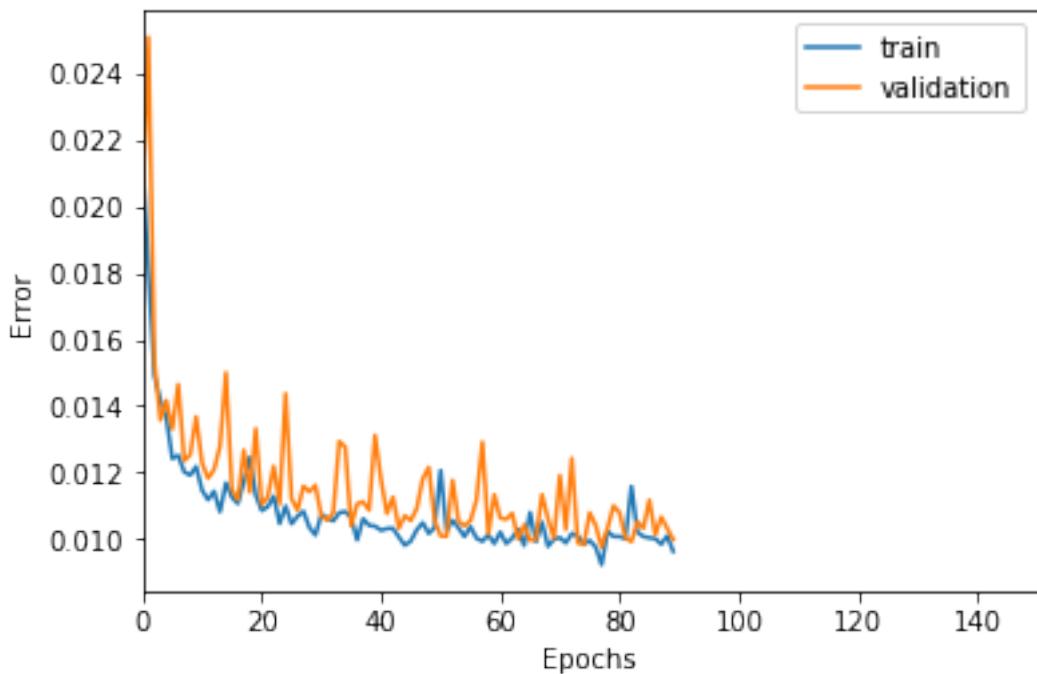
The mean error for nn1_100_app_change_ is 0.016051059233542428 for length 4629
Testing on Net flood synthetic data.



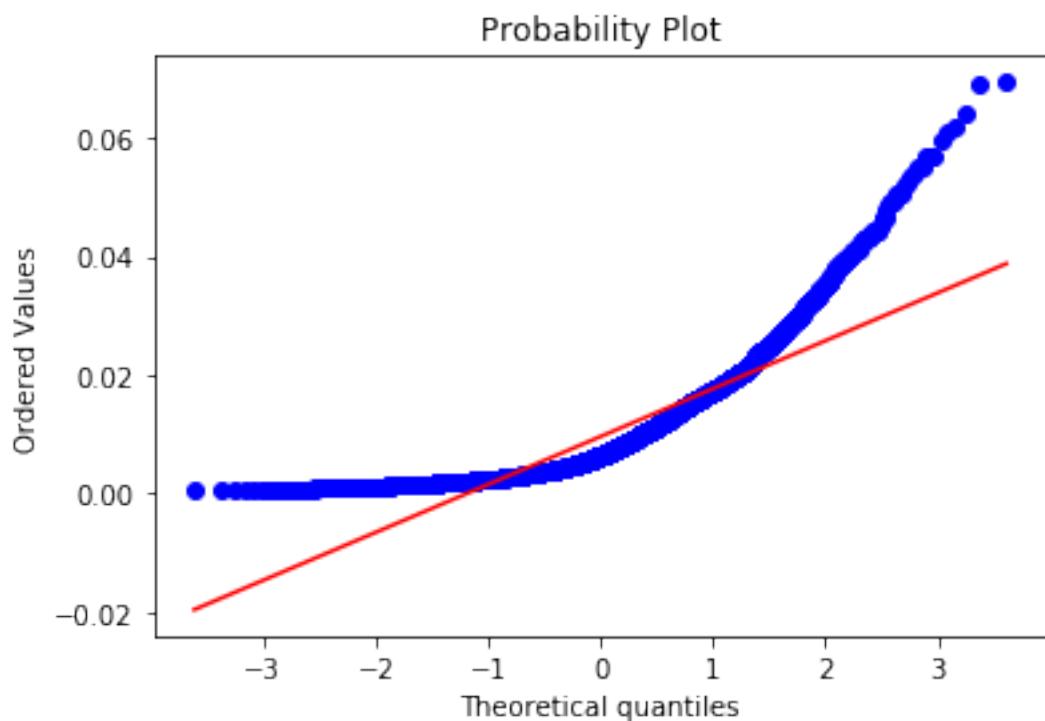
```
The mean error for nn1_100_net_flood_ is 0.035272381941519444 for length 4629  
=====
```

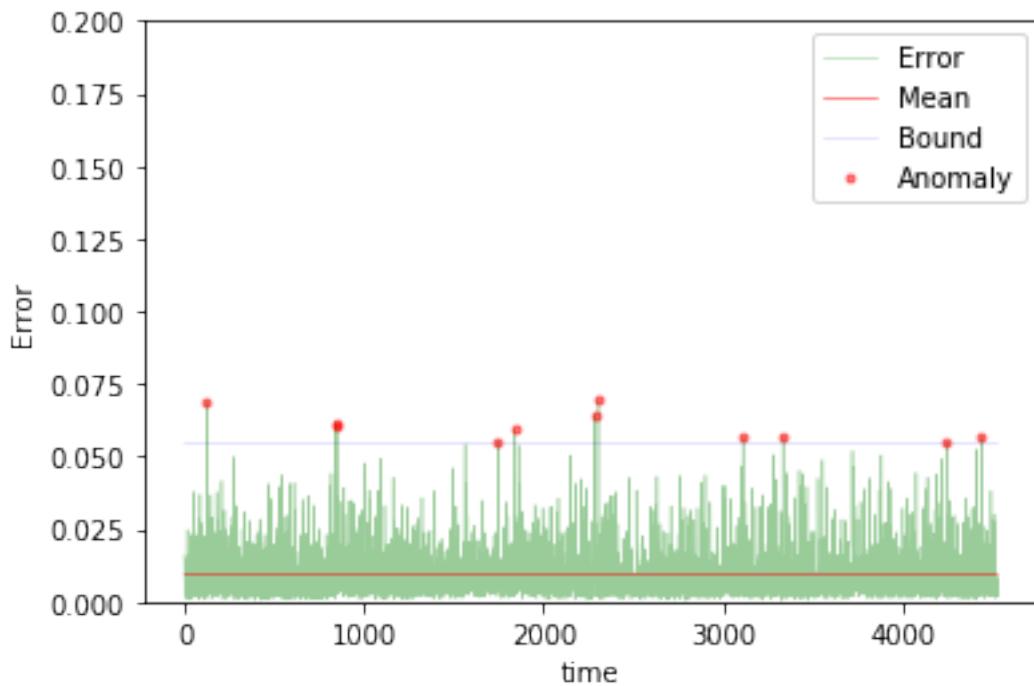
200 steps

```
In [88]: TIMESTEPS = 200  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn1_200"  
  
In [89]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(100,activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [90]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [91]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

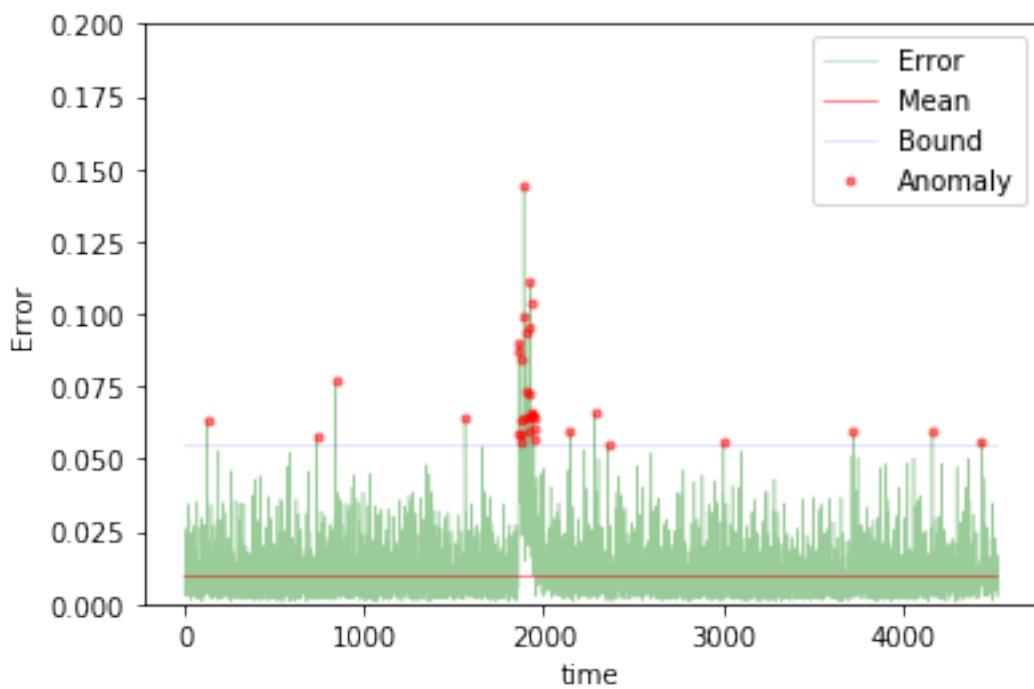


```
Training loss for final epoch is 0.009624548086547292
Validation loss for final epoch is 0.009983150689397008
----- Beginning tests for nn1_200 -----
Testing on normal data.
```

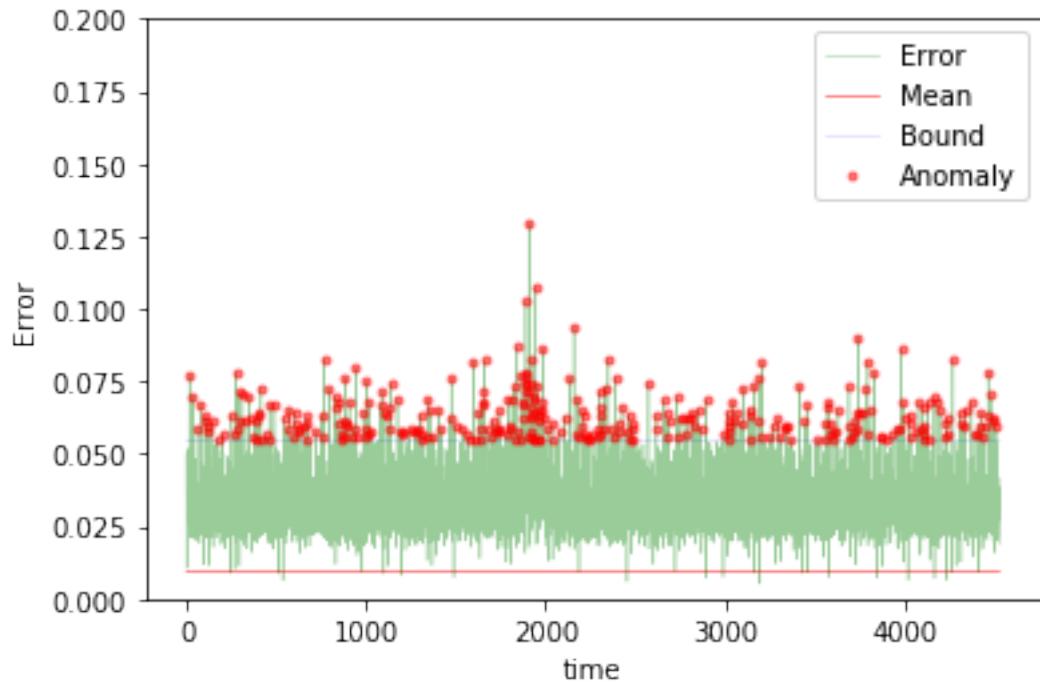




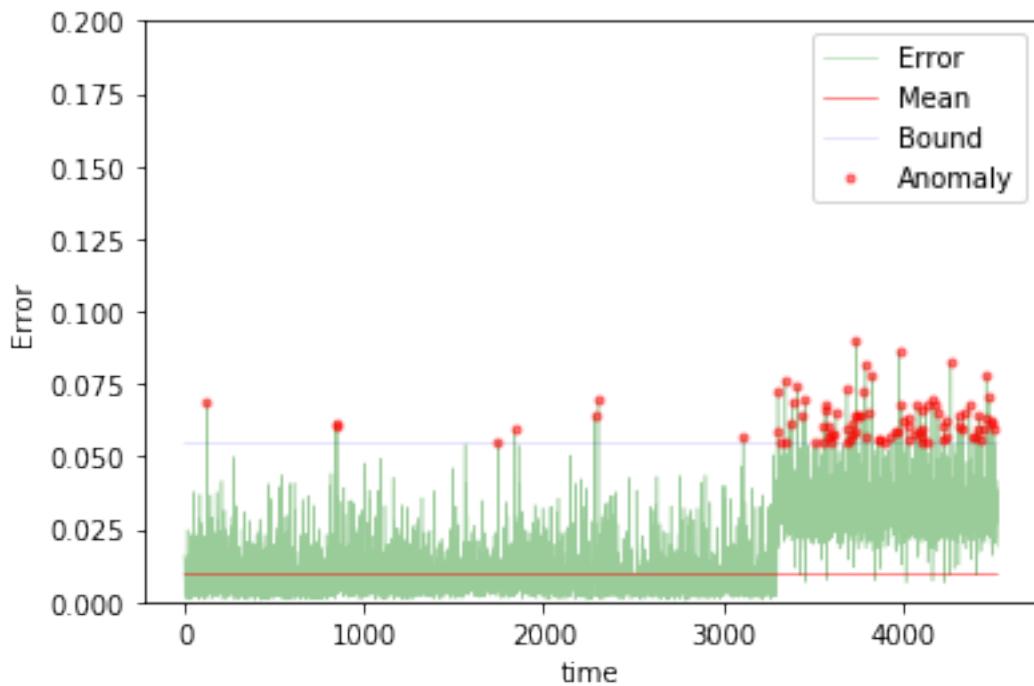
The mean error for nn1_200_normal_ is 0.009625798311277526 for length 4529
Testing on anomaly data.



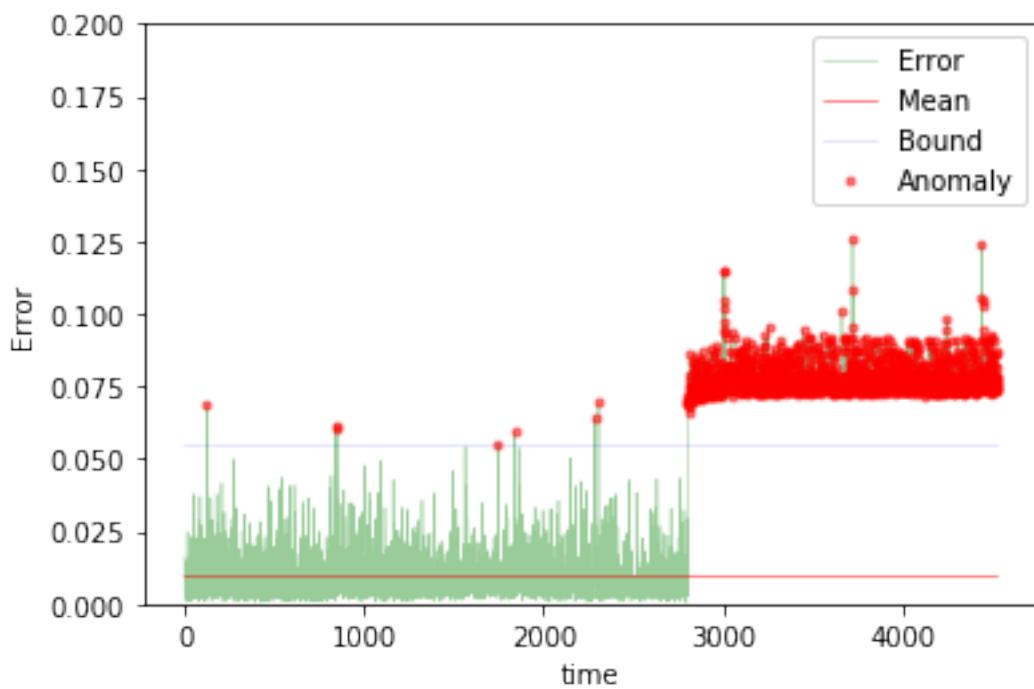
The mean error for nn1_200_anomaly_ is 0.011379258787185945 for length 4529
Testing on different app data.



The mean error for nn1_200_diff_app_ is 0.03383558700101958 for length 4529
Testing on App change synthetic data.



The mean error for nn1_200_app_change_ is 0.016114687034241928 for length 4529
Testing on Net flood synthetic data.



```
The mean error for nn1_200_net_flood_ is 0.03570338998705071 for length 4529
=====
```

2.1.3 NN with 2 hidden layers

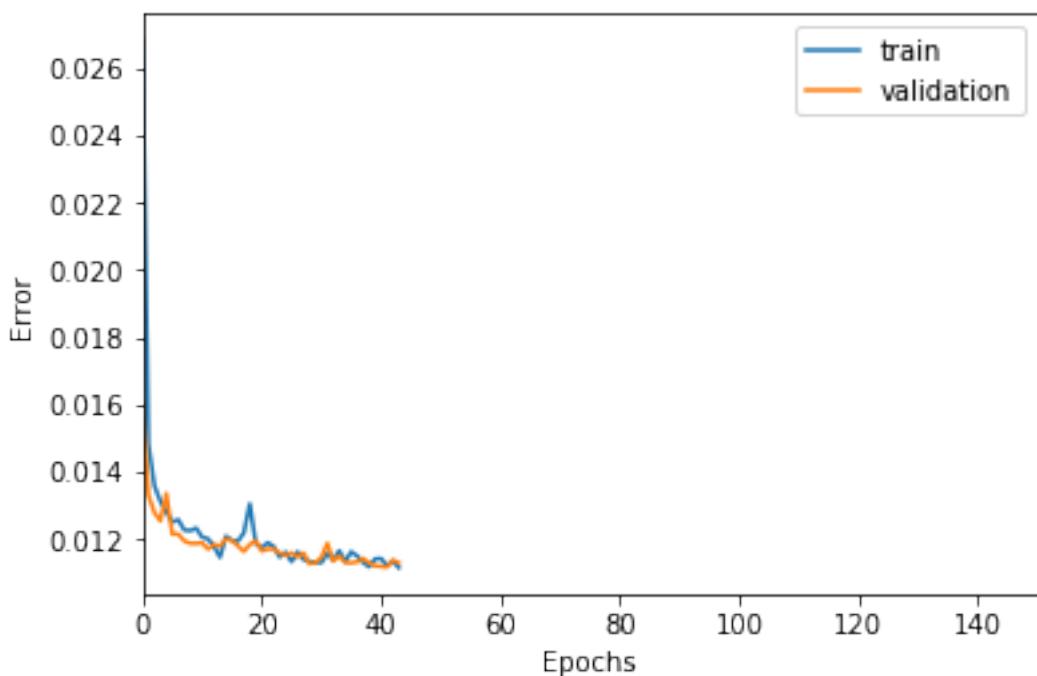
2 steps

```
In [92]: TIMESTEPS = 2
        DIM = 29
        tgen = flat_generator(X, TIMESTEPS)
        vgen = flat_generator(val_X, TIMESTEPS)
        name = "nn2_2"

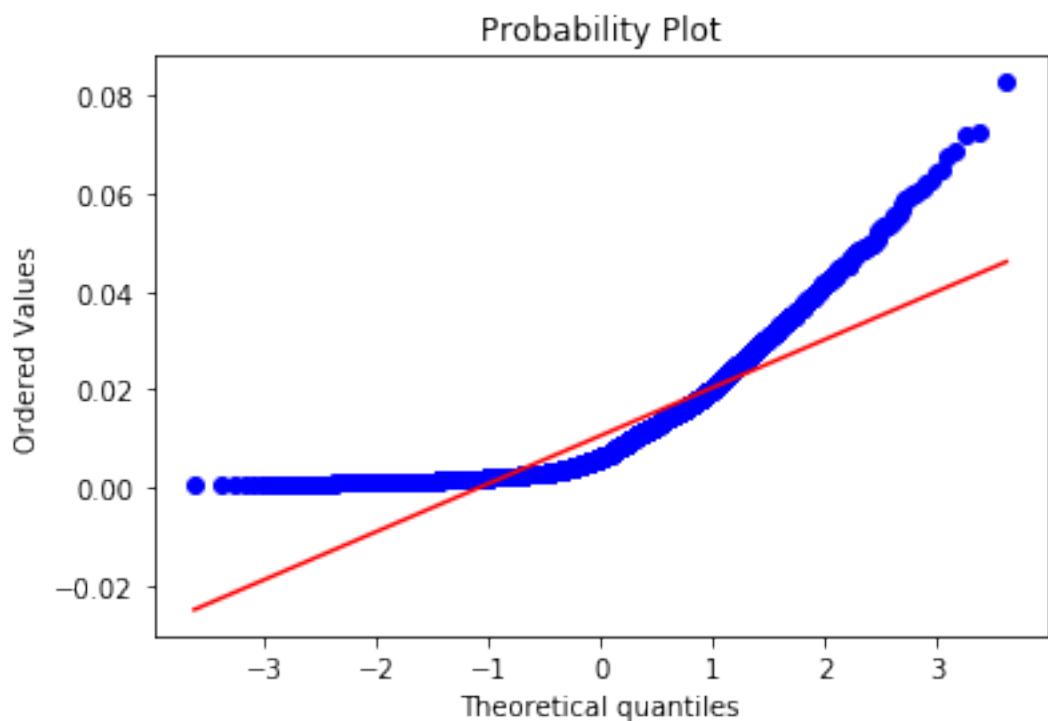
In [93]: input_layer = Input(shape=(TIMESTEPS*DIM,))
        hidden = Dense(500, activation='relu')(input_layer)
        hidden = Dense(100, activation='relu')(hidden)
        output = Dense(DIM, activation='sigmoid')(hidden)

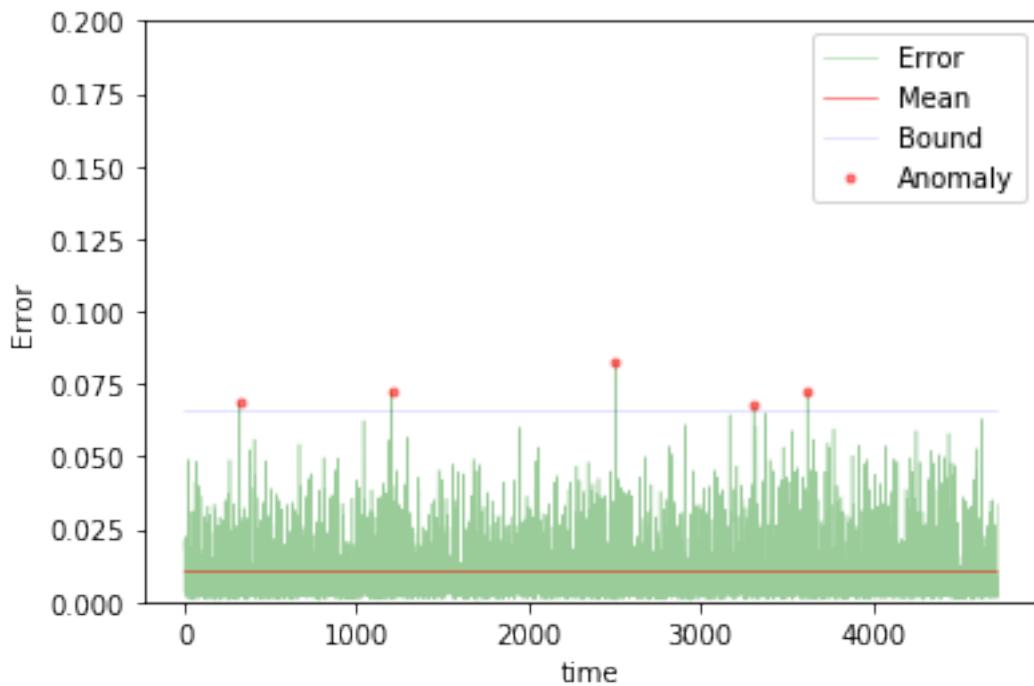
In [94]: model = Model(input_layer, output)
        model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

In [95]: train(model, tgen, vgen, name=name)
        test(model, name=name, window=TIMESTEPS)
```

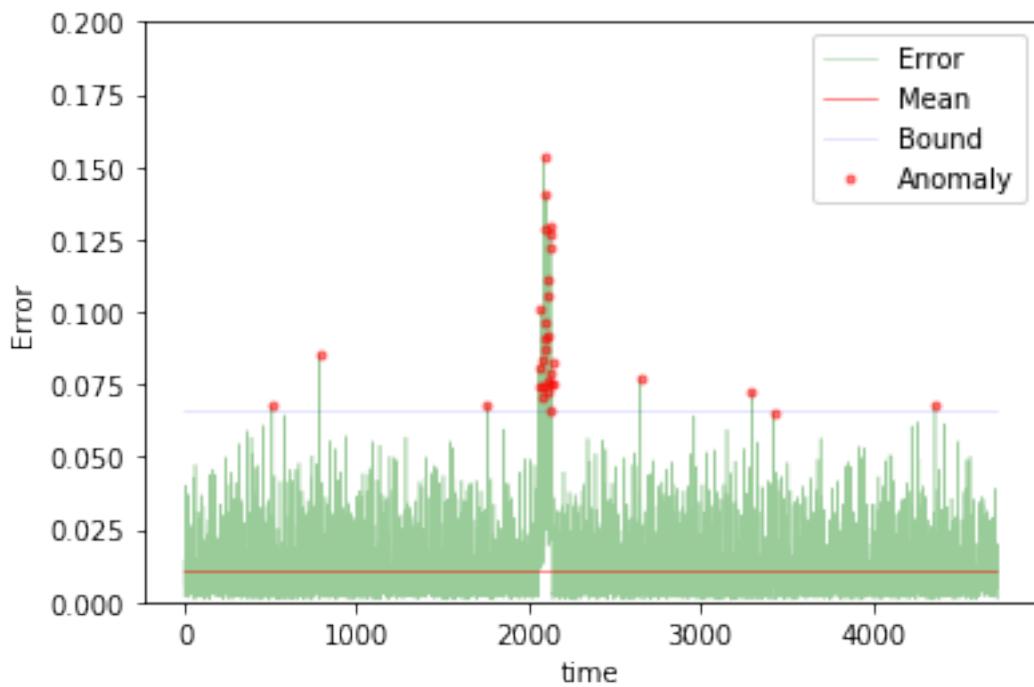


```
Training loss for final epoch is 0.011156654219725169
Validation loss for final epoch is 0.011296930460142903
----- Beginning tests for nn2_2 -----
Testing on normal data.
```

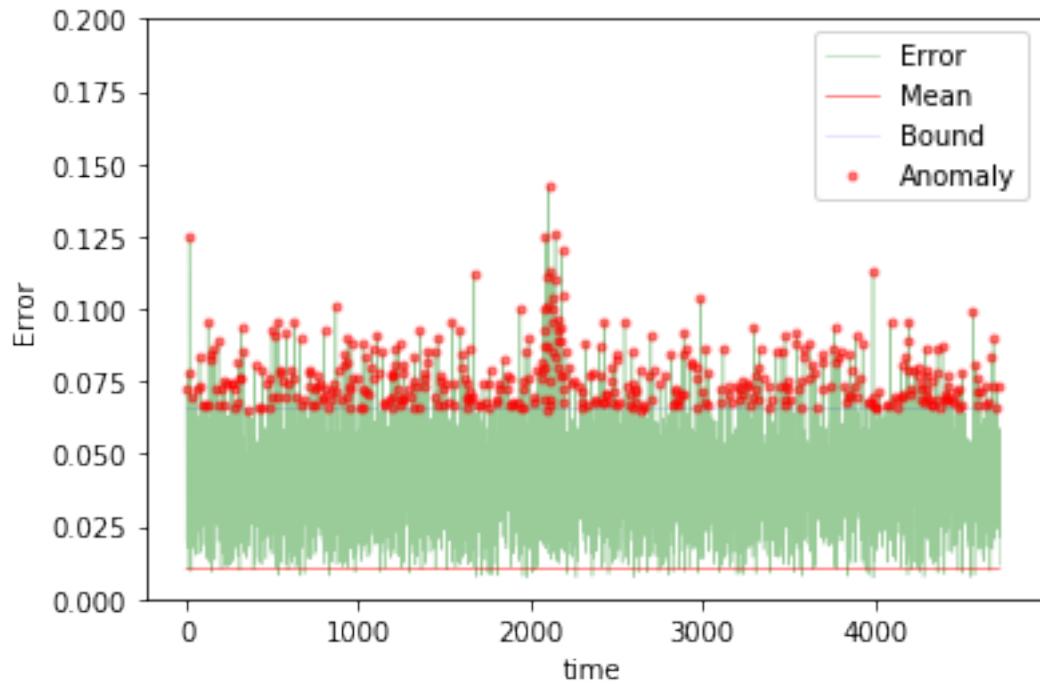




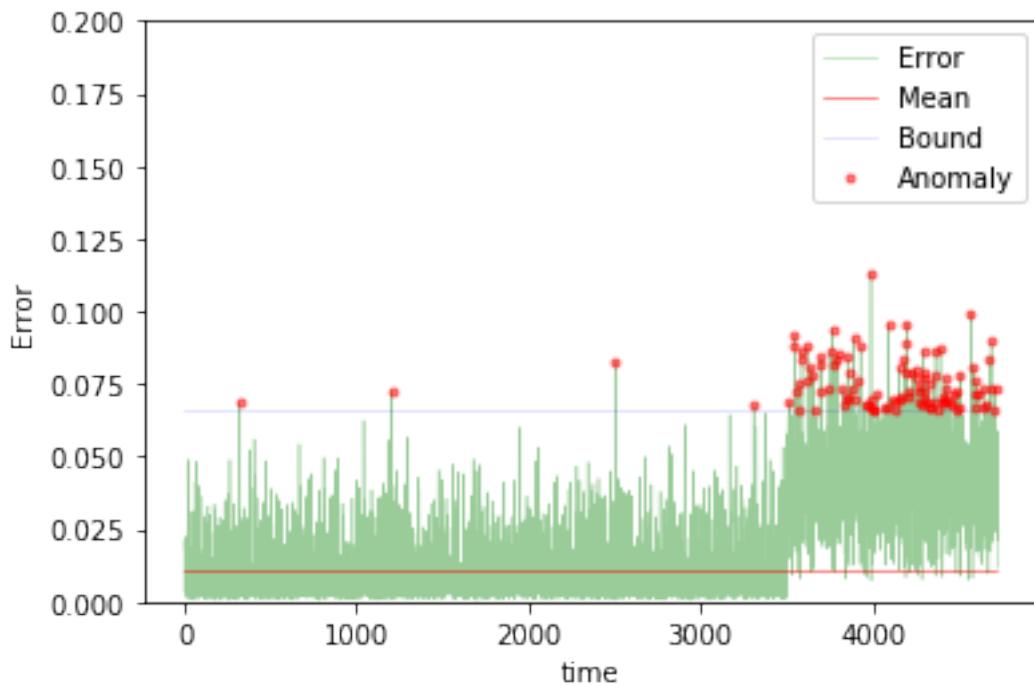
The mean error for nn2_2_normal_ is 0.010587569179211222 for length 4727
Testing on anomaly data.



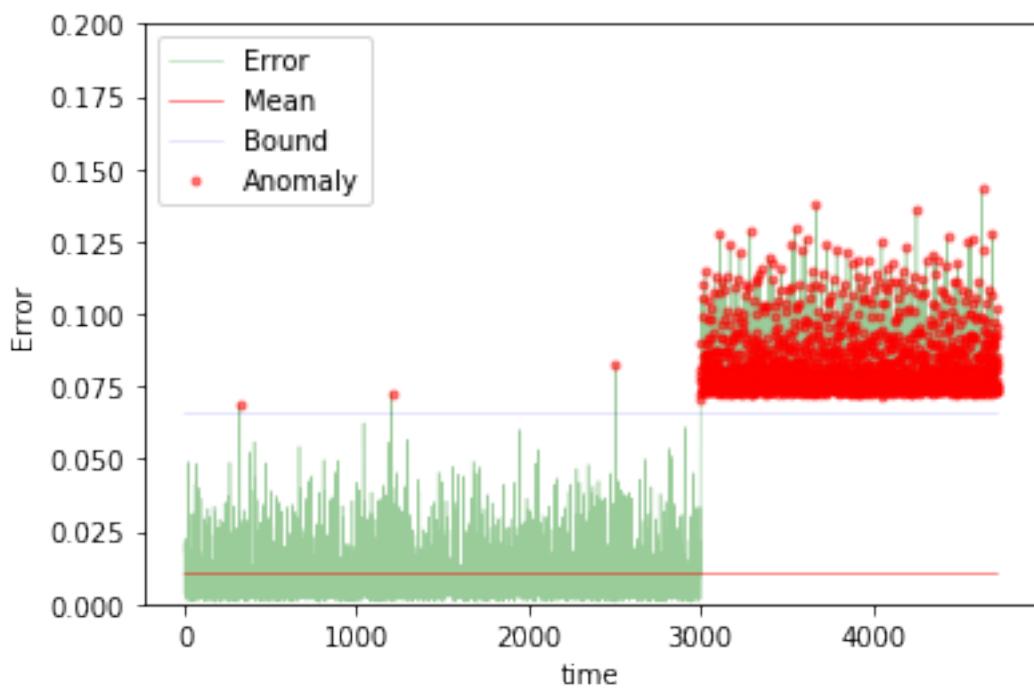
The mean error for nn2_2_anomaly_ is 0.012254363942389134 for length 4727
Testing on different app data.



The mean error for nn2_2_diff_app_ is 0.04242202565286769 for length 4727
Testing on App change synthetic data.



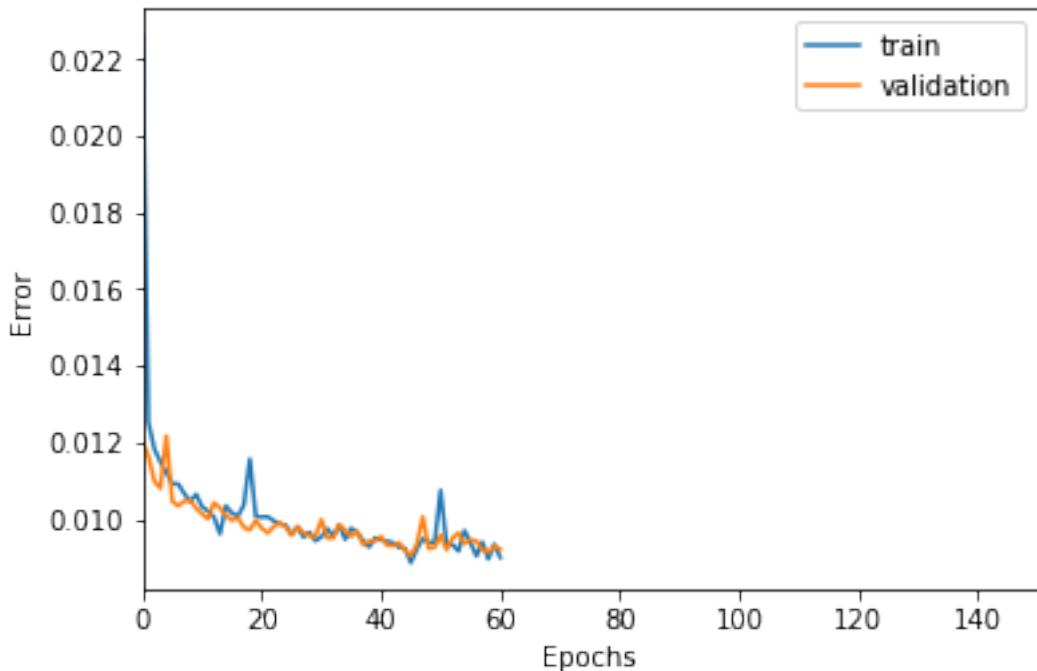
The mean error for nn2_2_app_change_ is 0.018771425029069873 for length 4727
Testing on Net flood synthetic data.



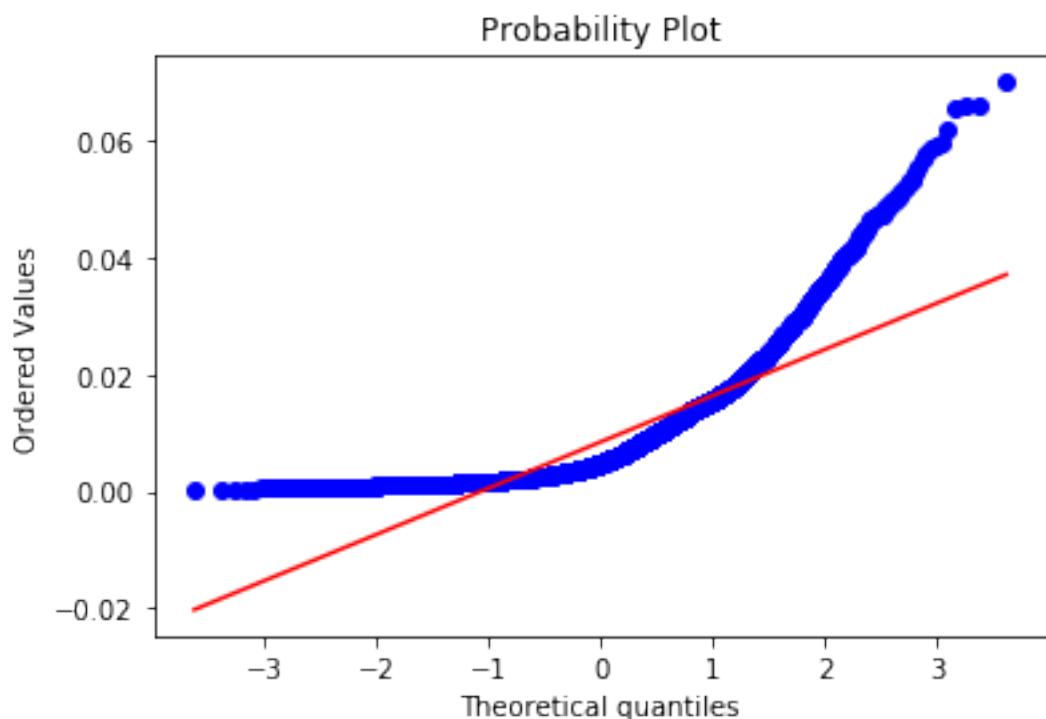
```
The mean error for nn2_2_net_flood_ is 0.03671400594381379 for length 4727  
=====
```

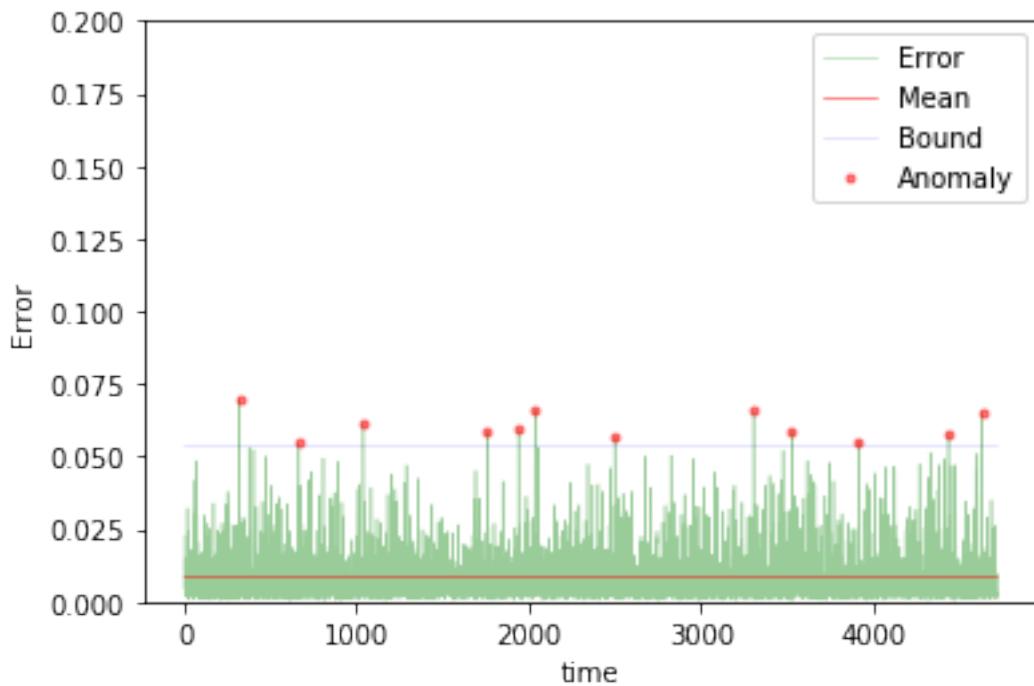
5 steps

```
In [96]: TIMESTEPS = 5  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn2_5"  
  
In [97]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(500, activation='relu')(input_layer)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [98]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [99]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

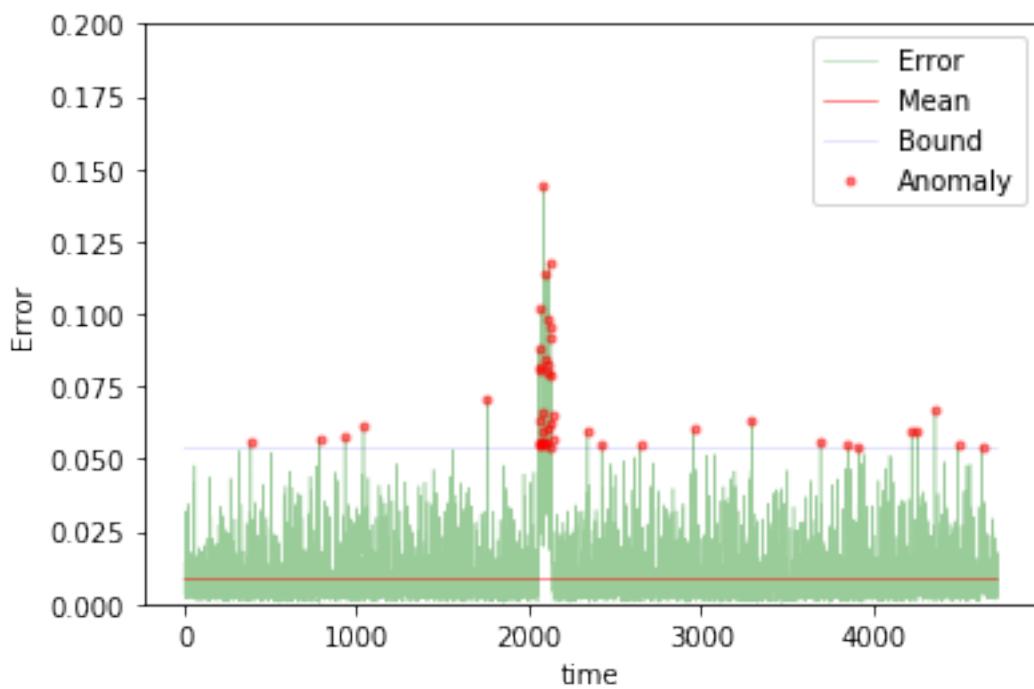


```
Training loss for final epoch is 0.008993440225836822
Validation loss for final epoch is 0.009206569993286394
----- Beginning tests for nn2_5 -----
Testing on normal data.
```

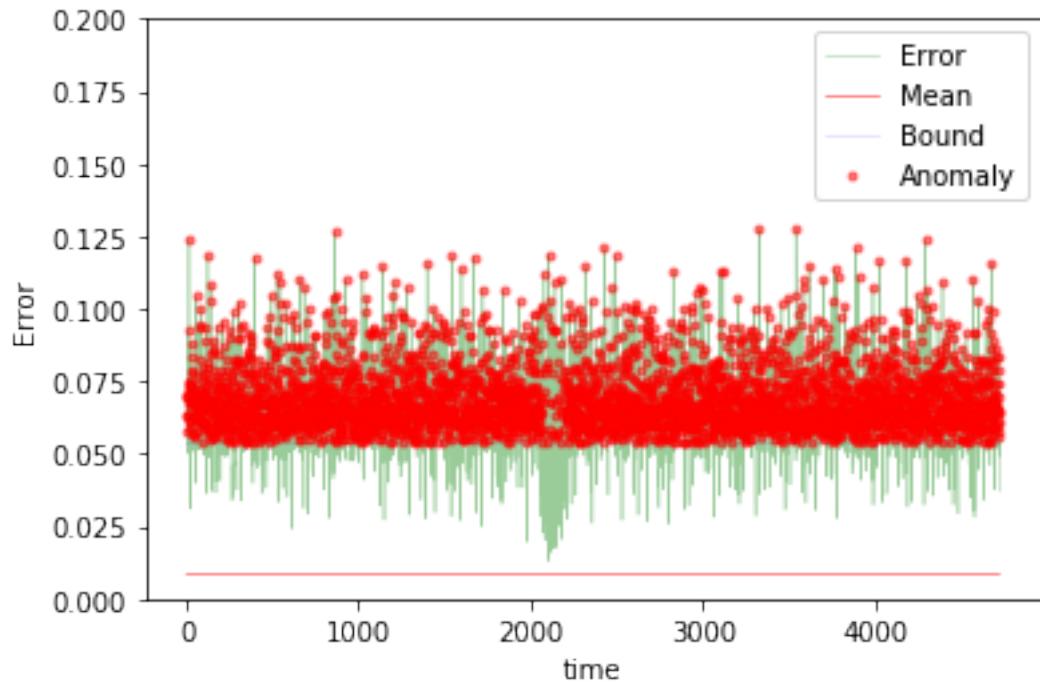




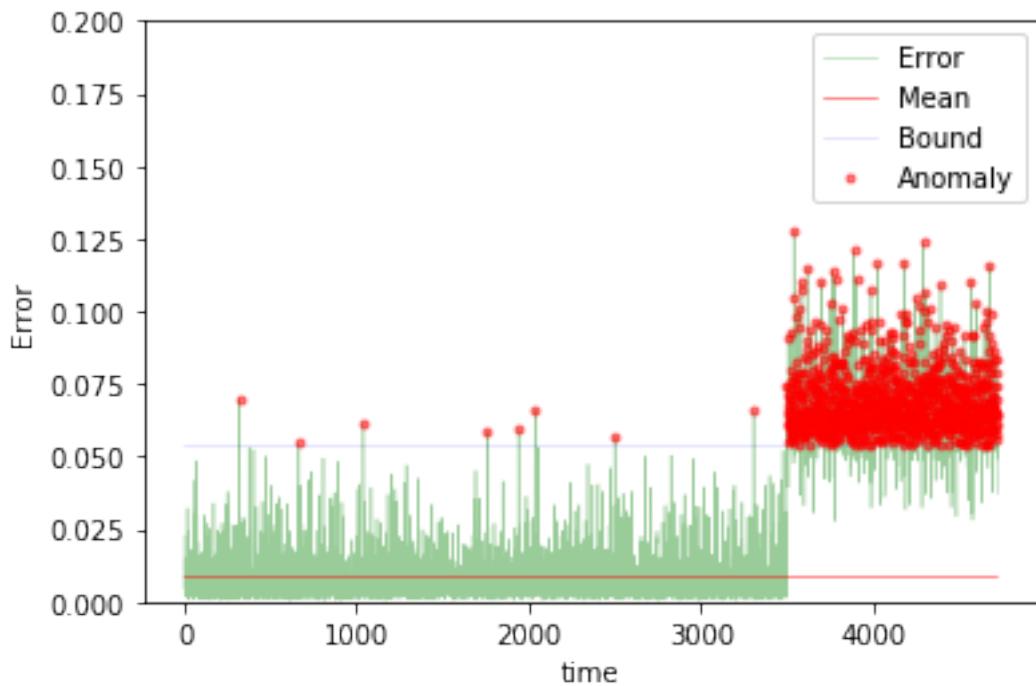
The mean error for nn2_5_normal_ is 0.008423012507313412 for length 4724
Testing on anomaly data.



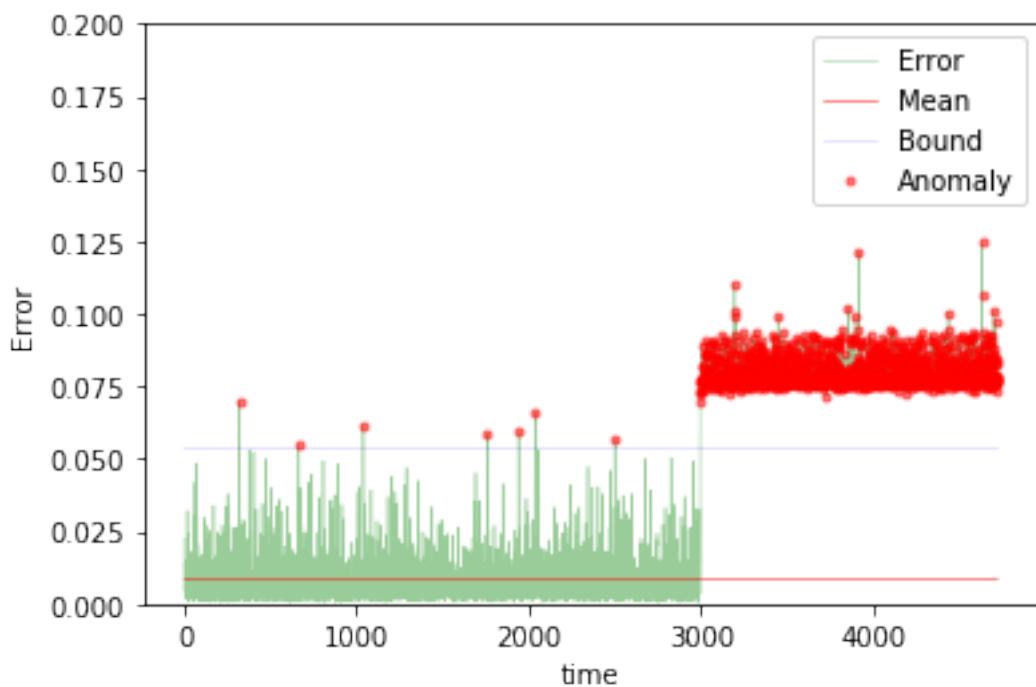
The mean error for nn2_5_anomaly_ is 0.010030639536036988 for length 4724
Testing on different app data.



The mean error for nn2_5_diff_app_ is 0.065525264252749 for length 4724
Testing on App change synthetic data.



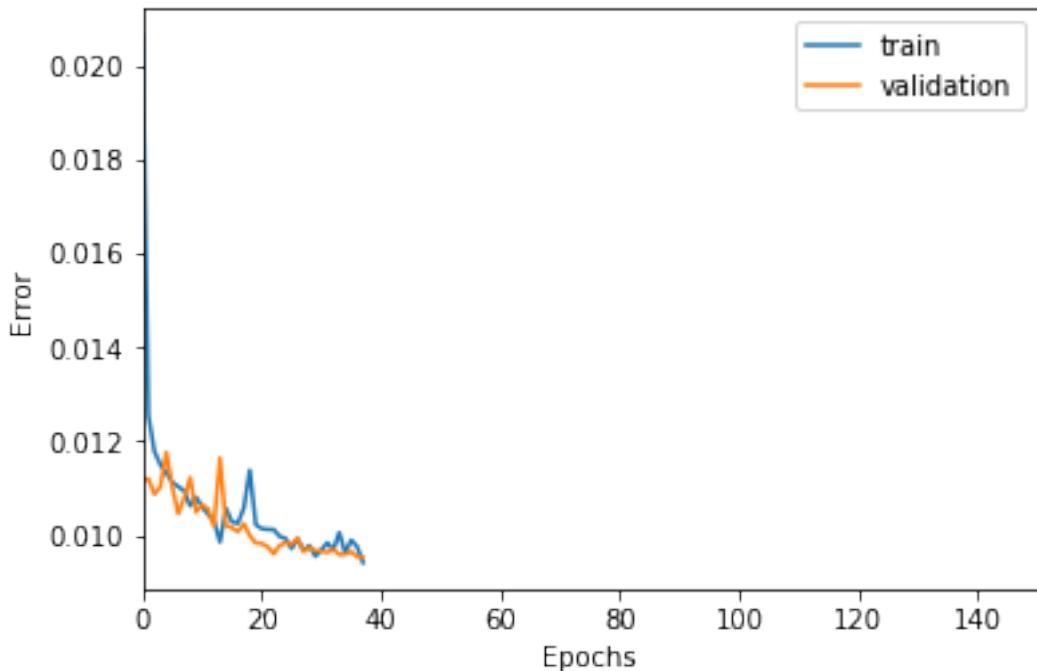
The mean error for nn2_5_app_change_ is 0.023360801284261414 for length 4724
Testing on Net flood synthetic data.



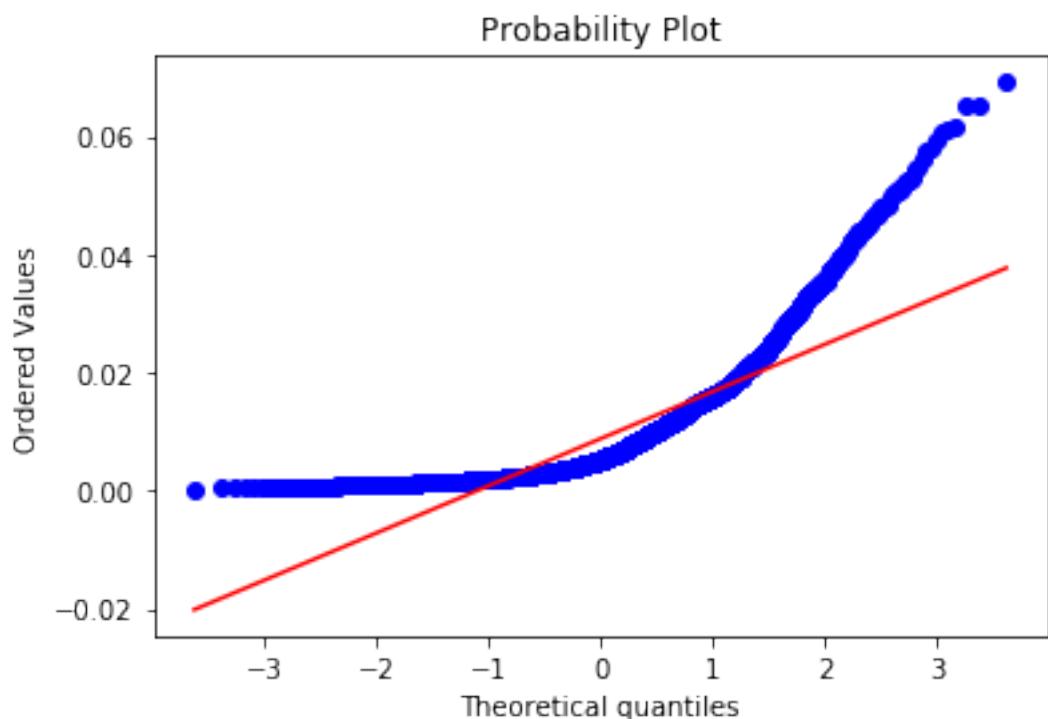
```
The mean error for nn2_5_net_flood_ is 0.034486613447206334 for length 4724  
=====
```

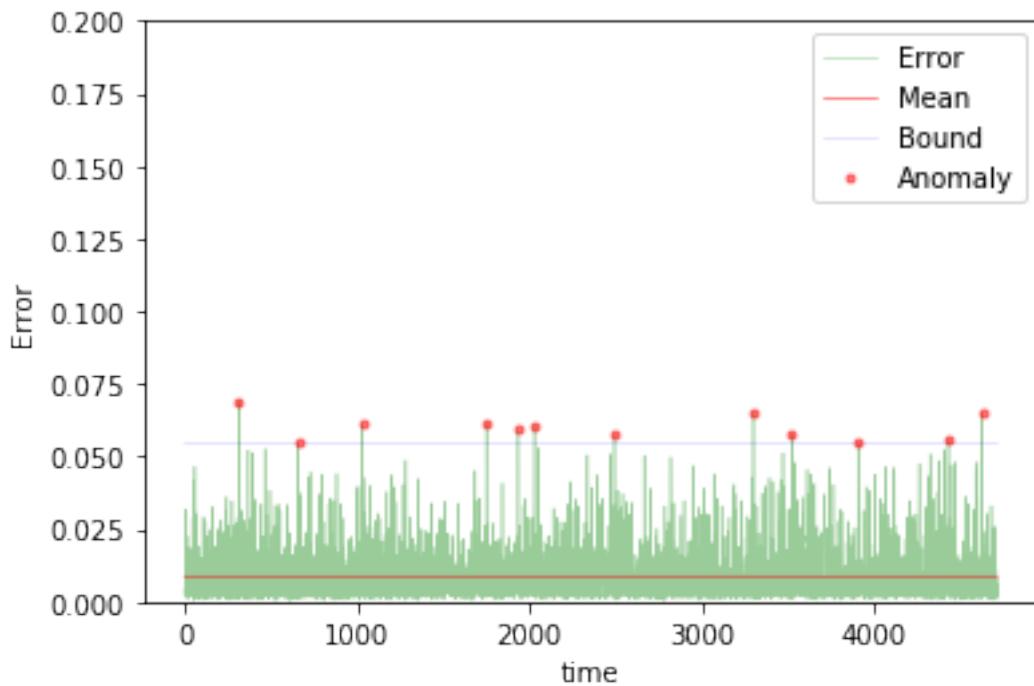
10 steps

```
In [100]: TIMESTEPS = 10  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn2_10"  
  
In [101]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(500, activation='relu')(input_layer)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [102]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [103]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

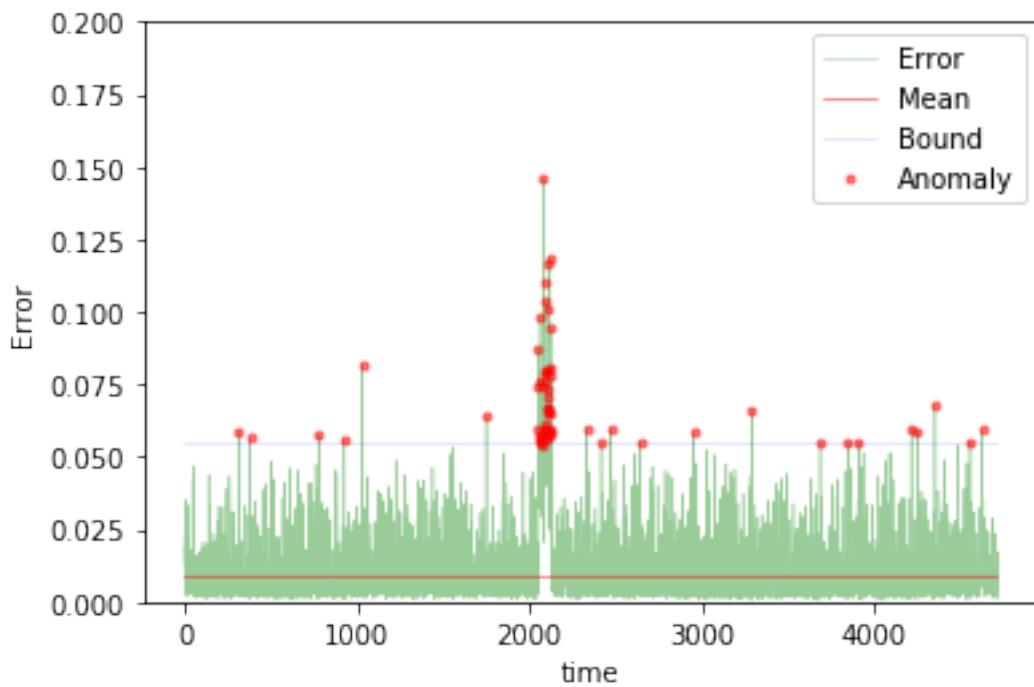


```
Training loss for final epoch is 0.009416980793117546
Validation loss for final epoch is 0.009529023326700553
----- Beginning tests for nn2_10 -----
Testing on normal data.
```

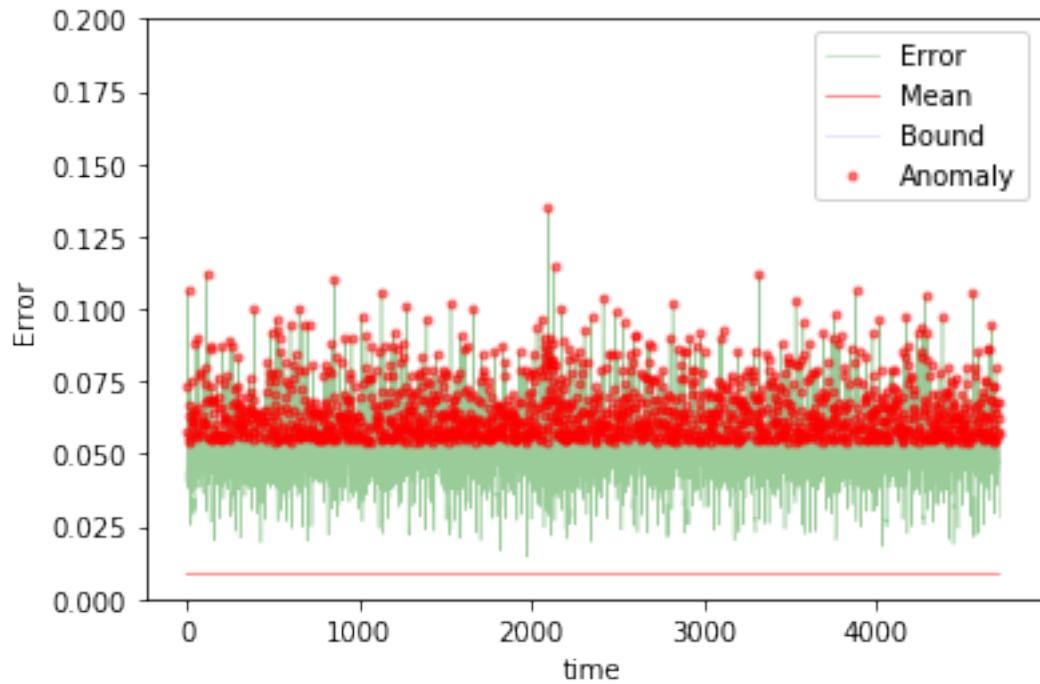




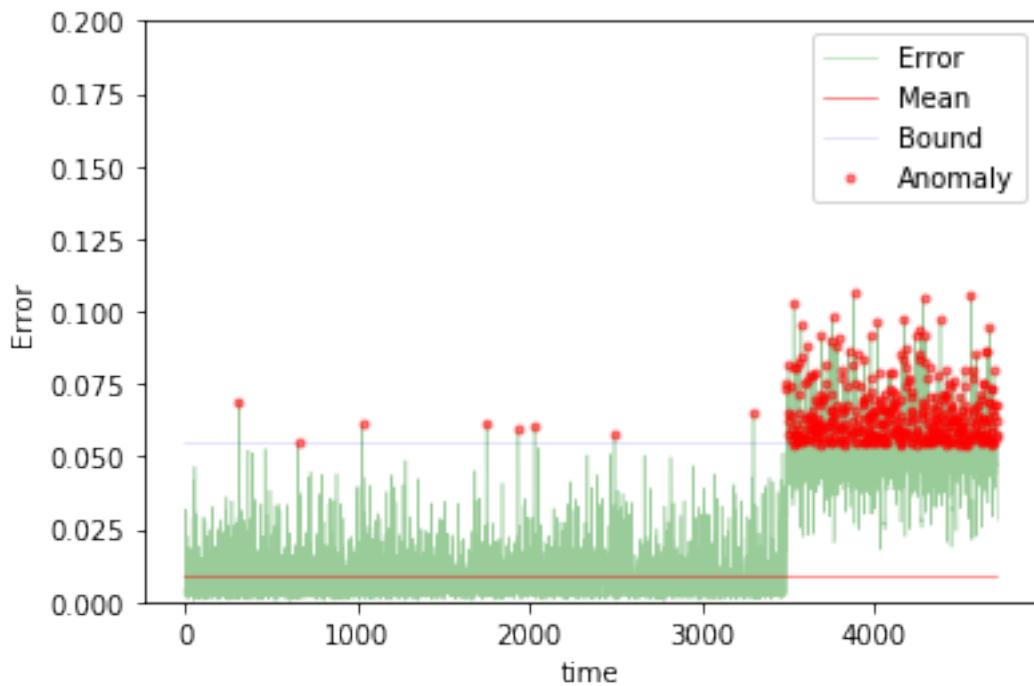
The mean error for nn2_10_normal_ is 0.008811962552416279 for length 4719
Testing on anomaly data.



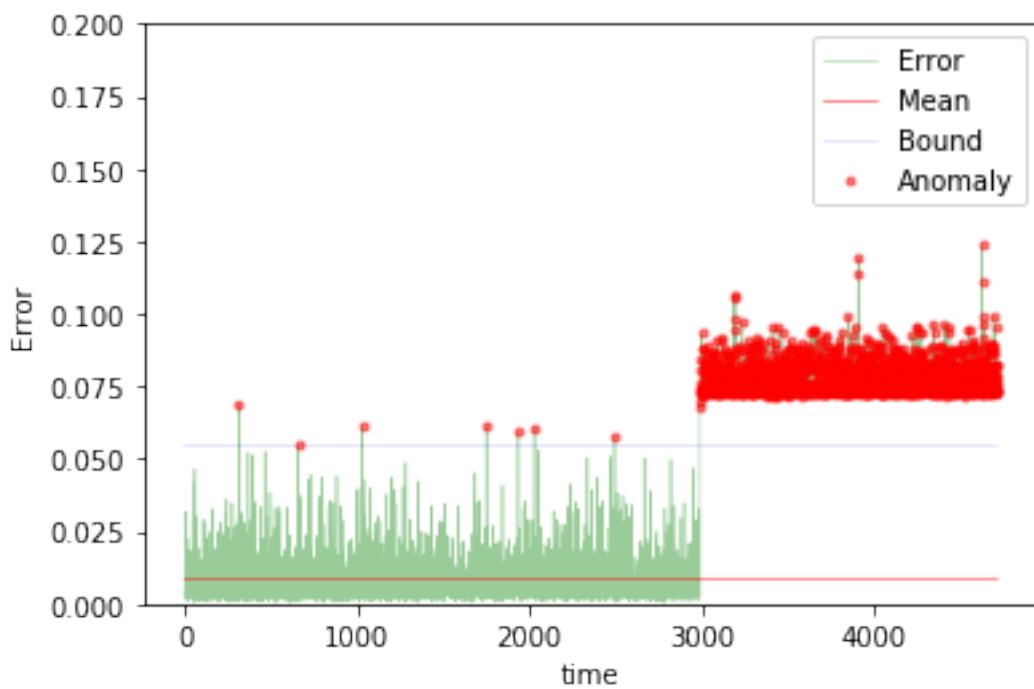
The mean error for nn2_10_anomaly_ is 0.010831795671040723 for length 4719
Testing on different app data.



The mean error for nn2_10_diff_app_ is 0.05146492496784679 for length 4719
Testing on App change synthetic data.



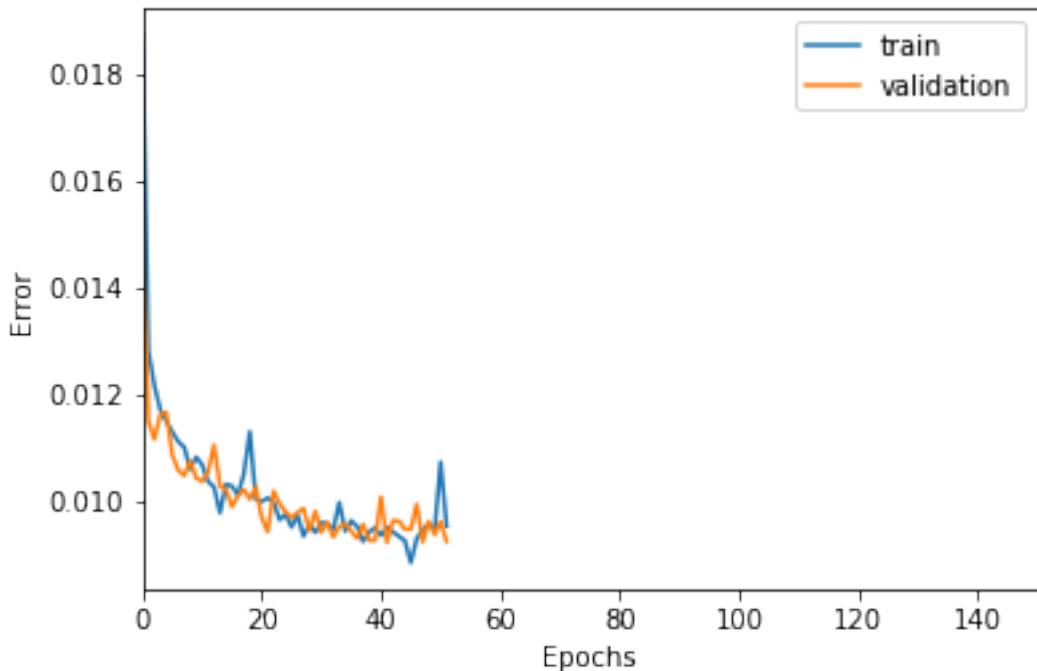
The mean error for nn2_10_app_change_ is 0.01982009193871468 for length 4719
Testing on Net flood synthetic data.



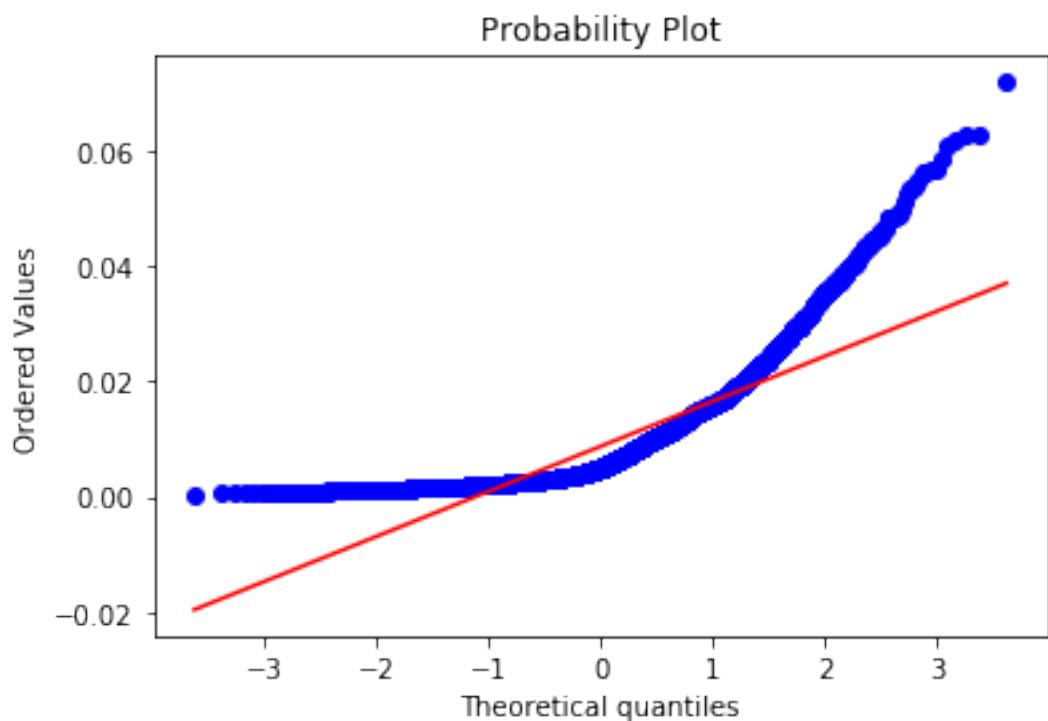
```
The mean error for nn2_10_net_flood_ is 0.03416956469286748 for length 4719  
=====
```

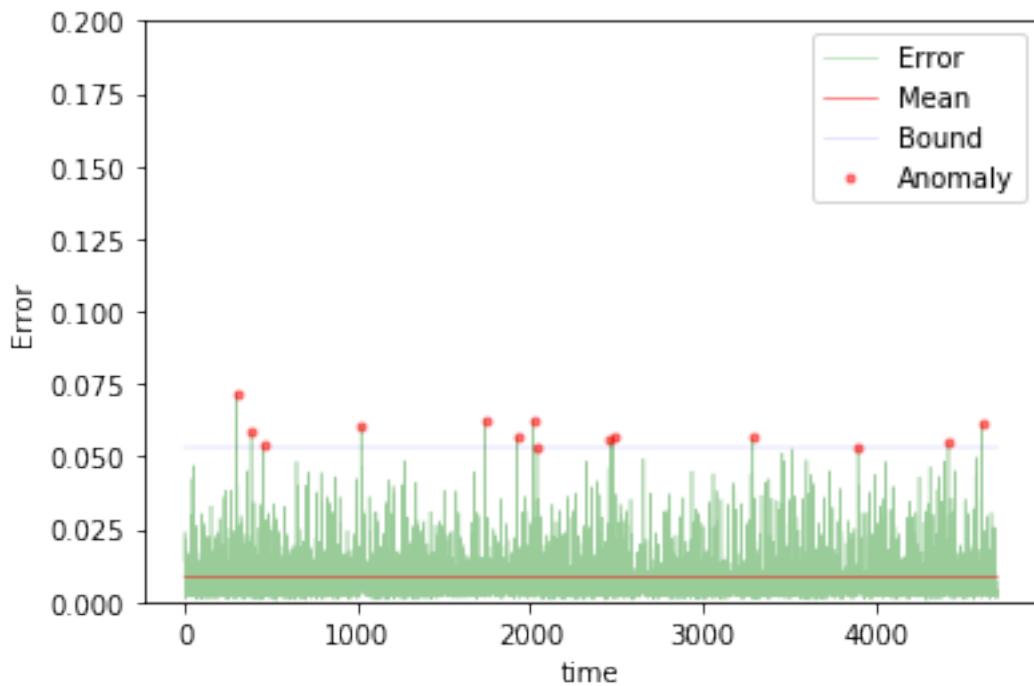
20 steps

```
In [104]: TIMESTEPS = 20  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn2_20"  
  
In [105]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(500, activation='relu')(input_layer)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [106]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [107]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

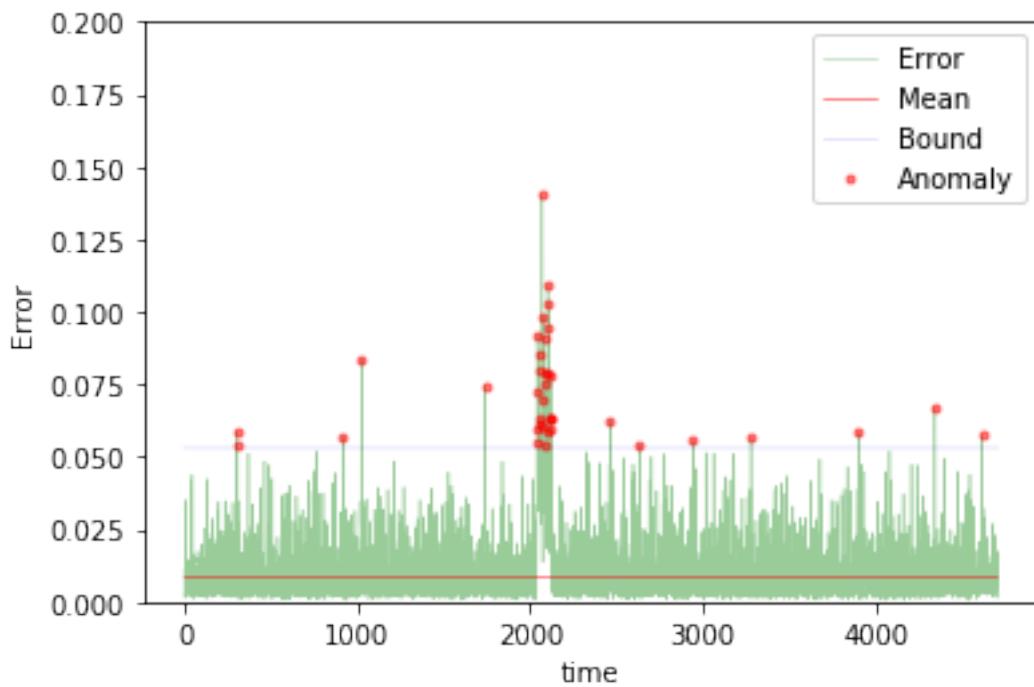


```
Training loss for final epoch is 0.009543817387195303
Validation loss for final epoch is 0.009256752856425009
----- Beginning tests for nn2_20 -----
Testing on normal data.
```

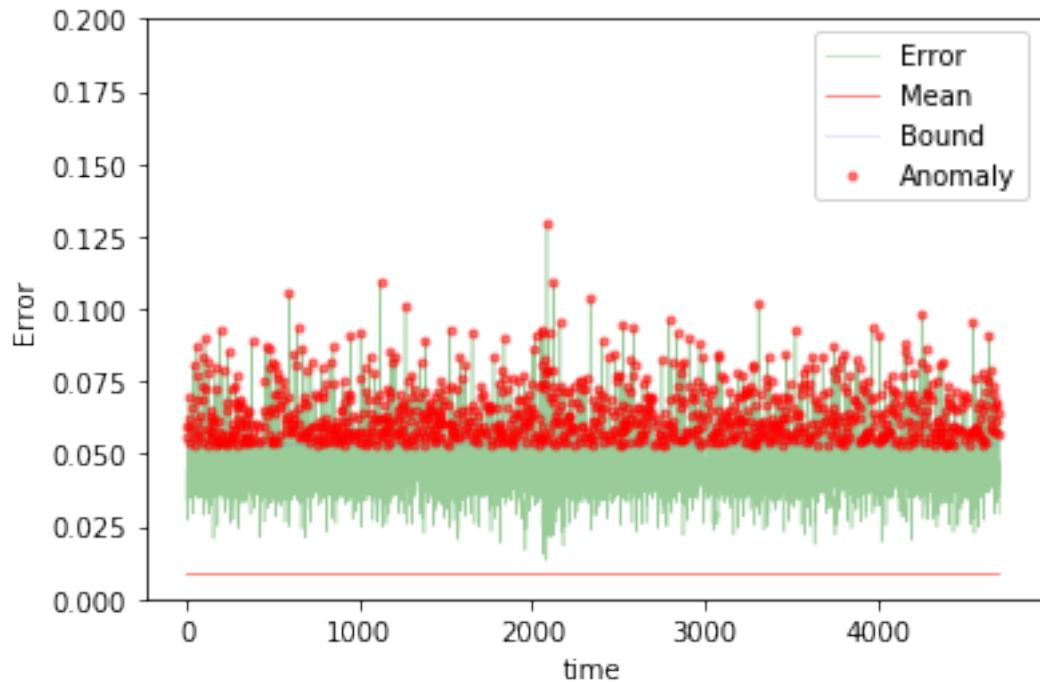




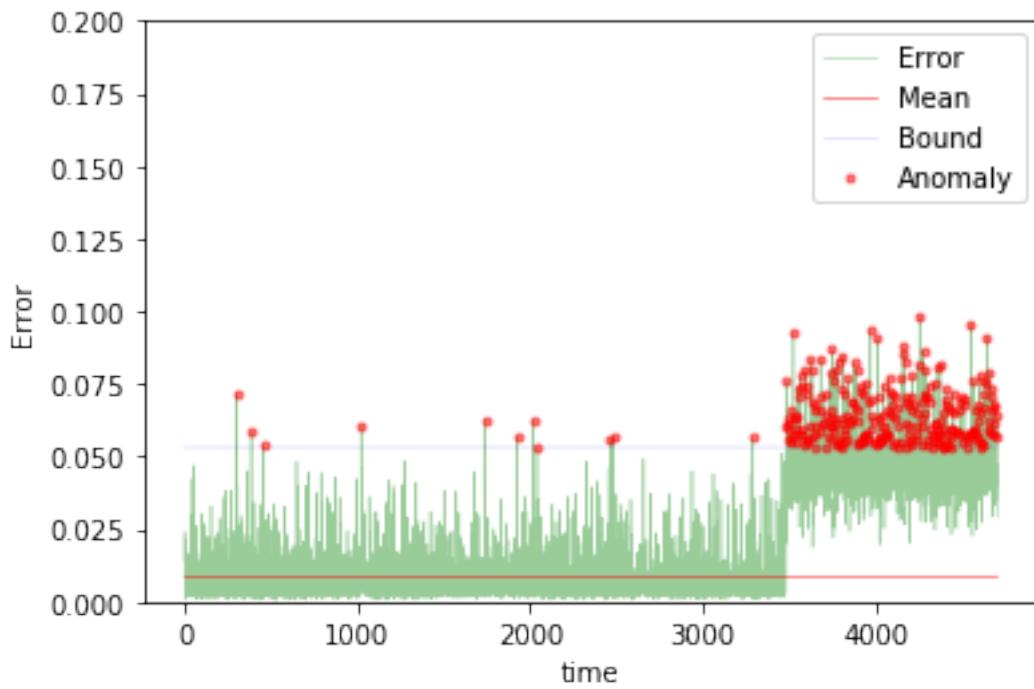
The mean error for nn2_20_normal_ is 0.00869350507728101 for length 4709
Testing on anomaly data.



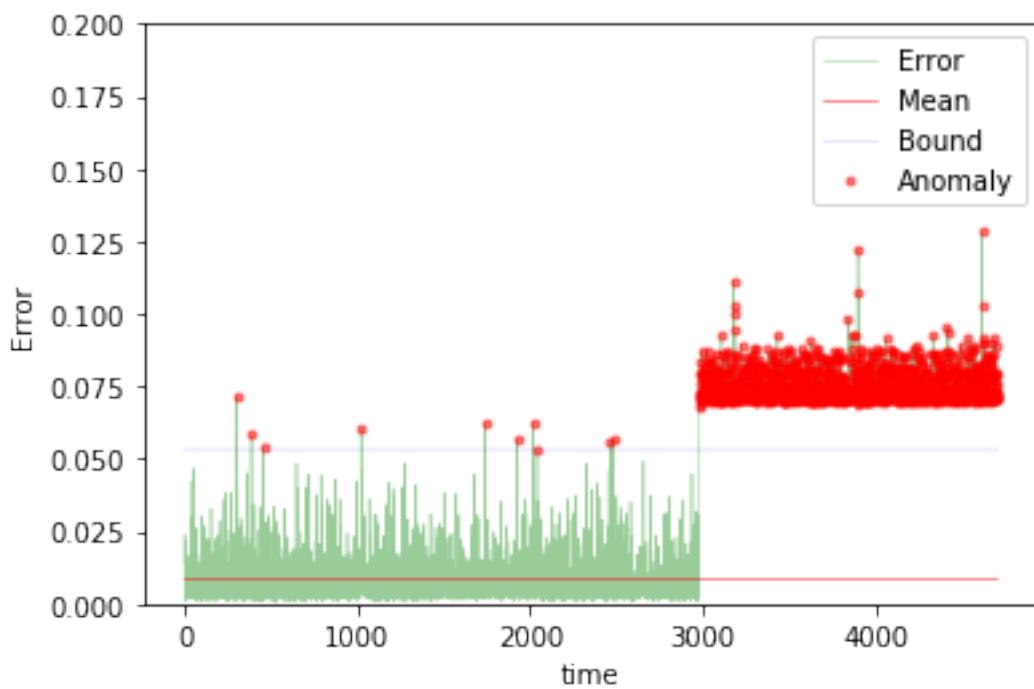
The mean error for nn2_20_anomaly_ is 0.010131714012653342 for length 4709
Testing on different app data.



The mean error for nn2_20_diff_app_ is 0.046129007212994684 for length 4709
Testing on App change synthetic data.



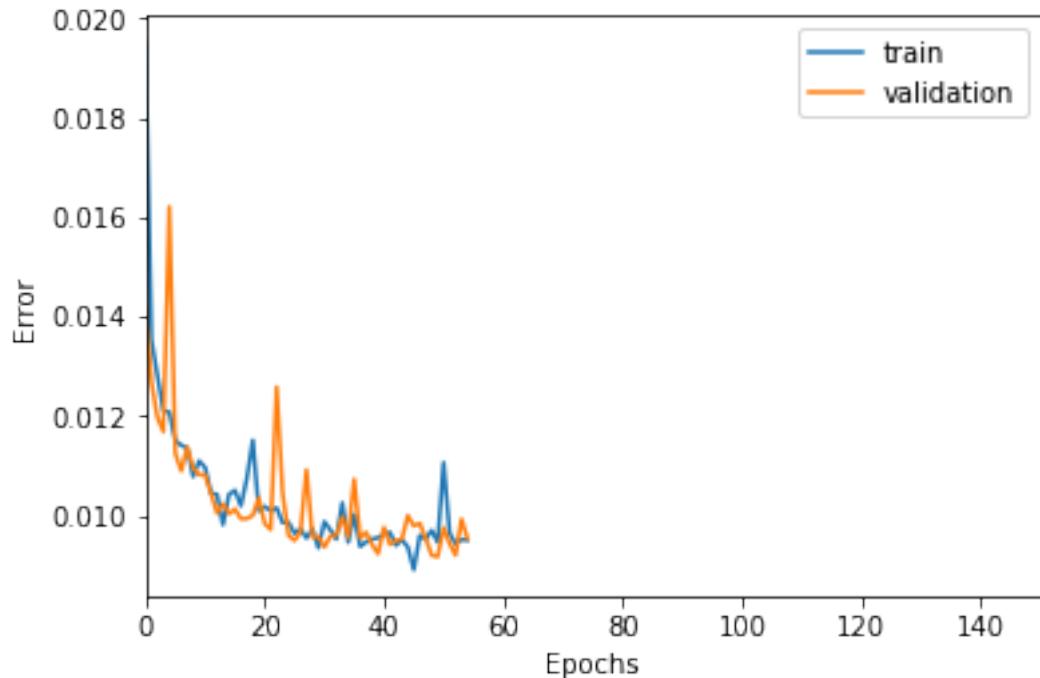
The mean error for nn2_20_app_change_ is 0.018387386533140176 for length 4709
Testing on Net flood synthetic data.



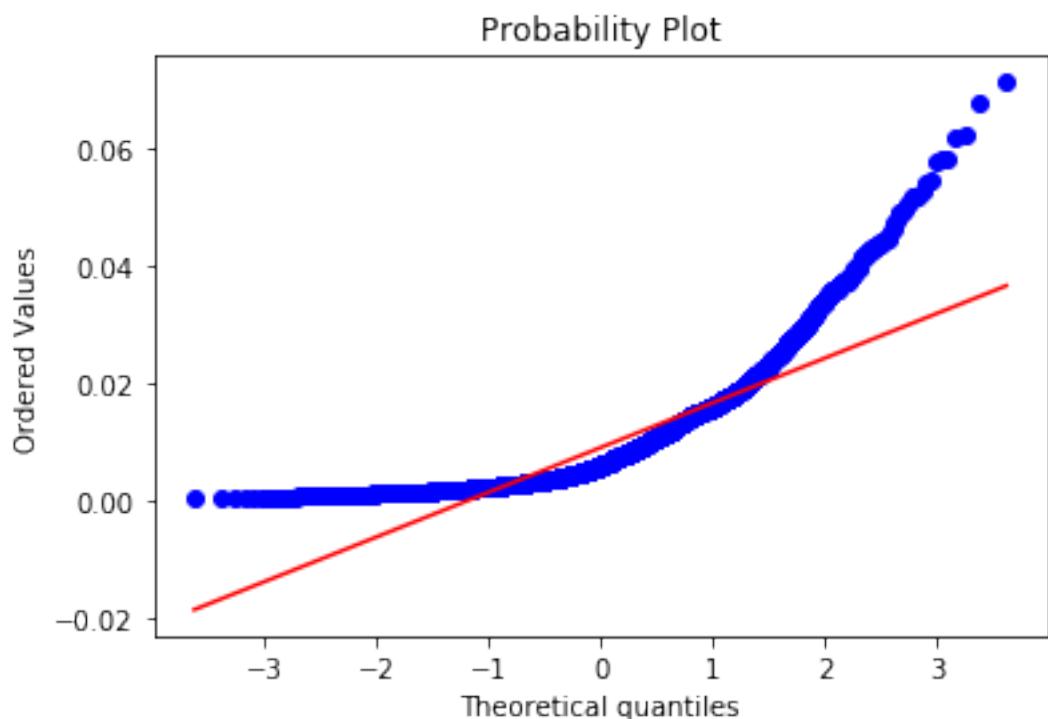
```
The mean error for nn2_20_net_flood_ is 0.033005246243047286 for length 4709  
=====
```

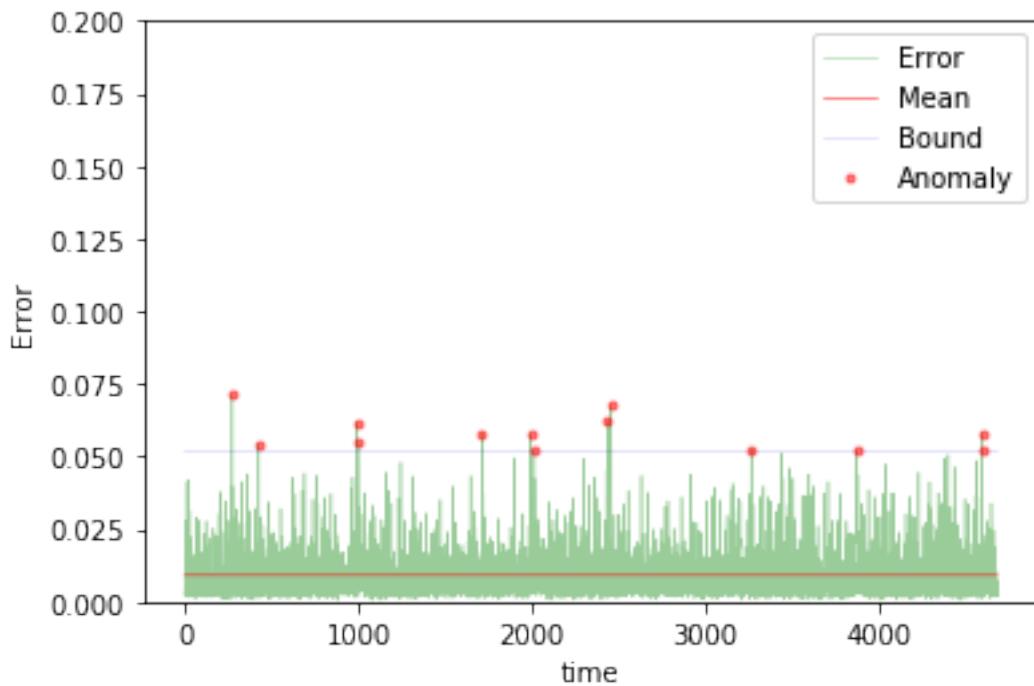
50 steps

```
In [108]: TIMESTEPS = 50  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn2_50"  
  
In [109]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(500, activation='relu')(input_layer)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [110]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [111]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

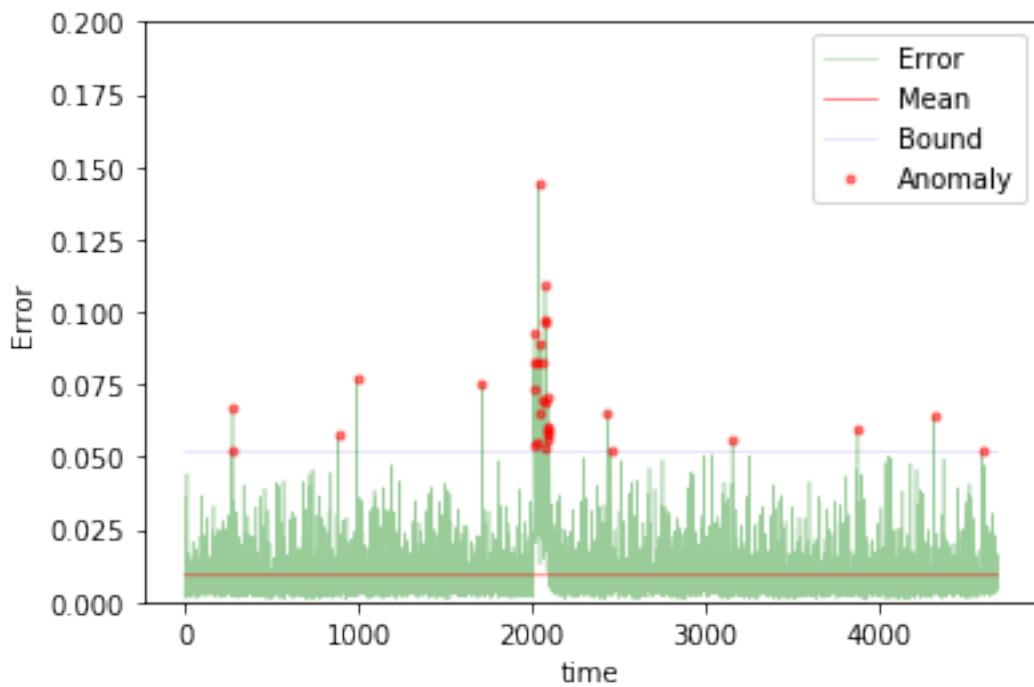


```
Training loss for final epoch is 0.009505581822479143
Validation loss for final epoch is 0.00953334755054675
----- Beginning tests for nn2_50 -----
Testing on normal data.
```

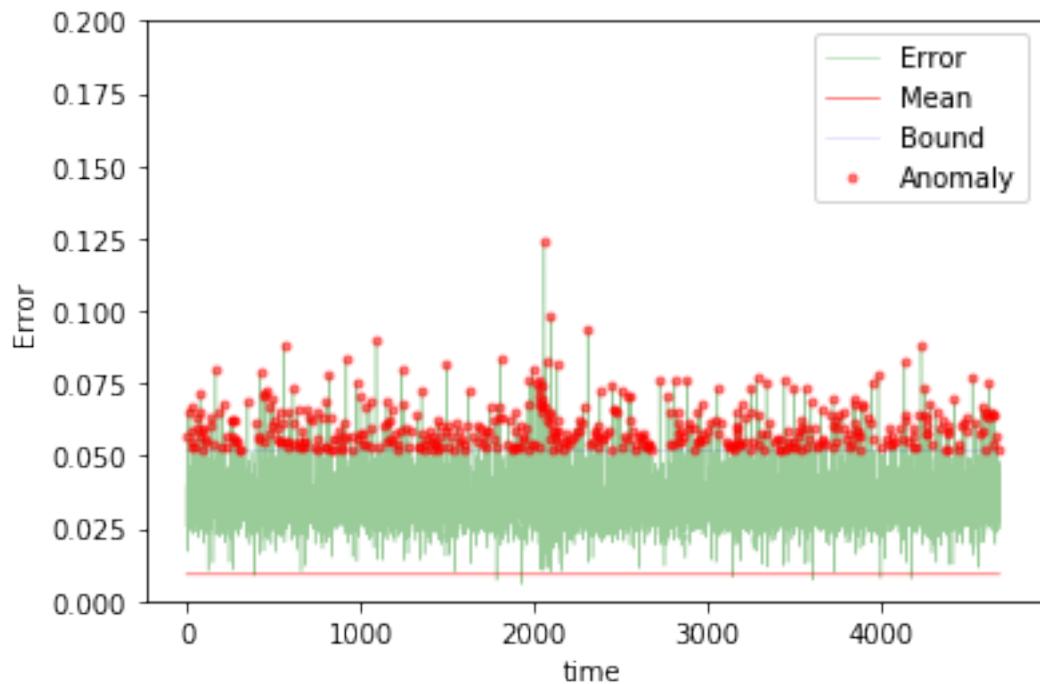




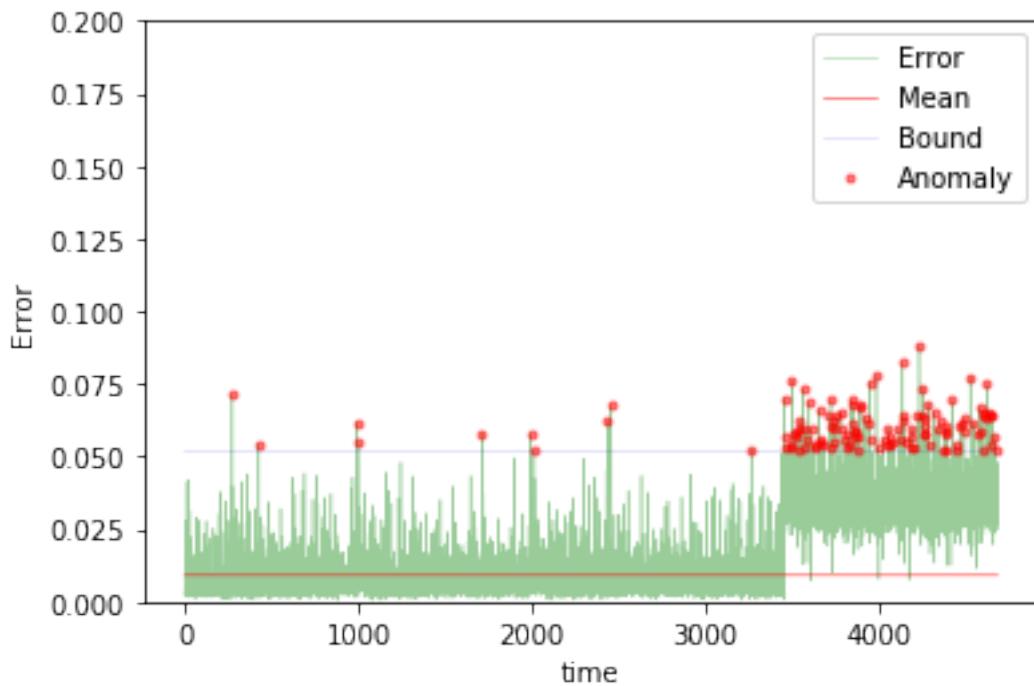
The mean error for nn2_50_normal_ is 0.009161311721198991 for length 4679
Testing on anomaly data.



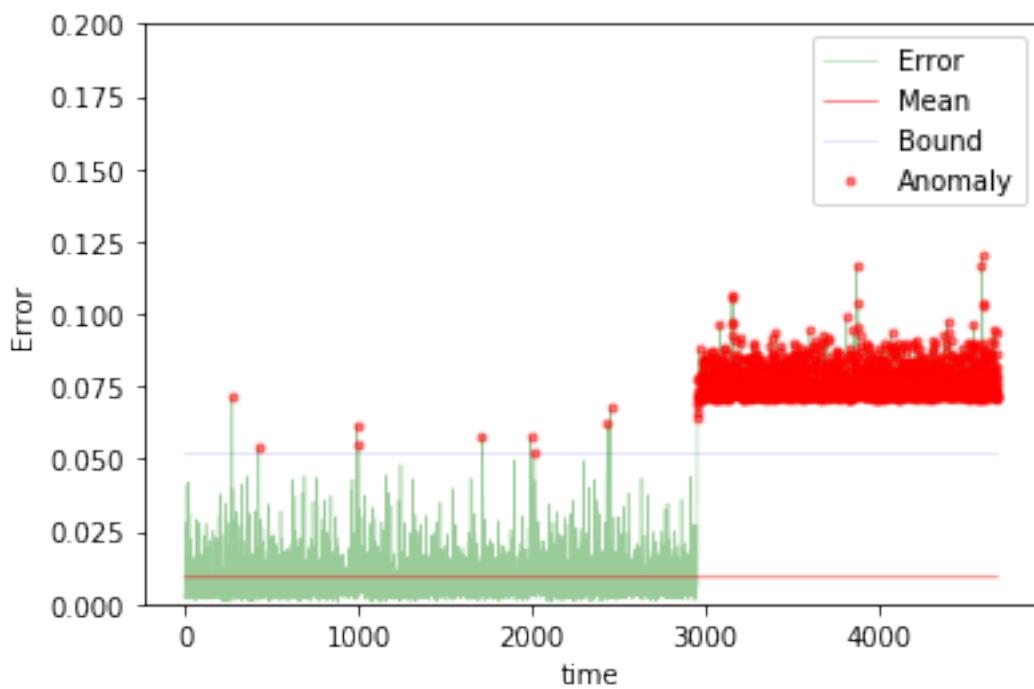
The mean error for nn2_50_anomaly_ is 0.010321838315465239 for length 4679
Testing on different app data.



The mean error for nn2_50_diff_app_ is 0.03530901951418092 for length 4679
Testing on App change synthetic data.



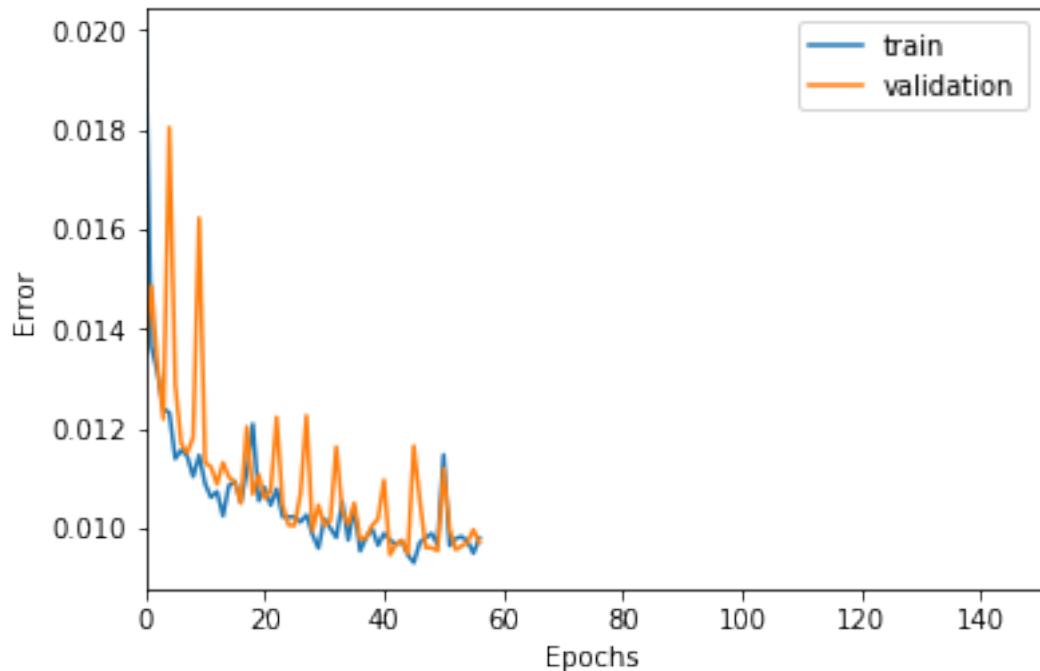
The mean error for nn2_50_app_change_ is 0.015982929440127653 for length 4679
Testing on Net flood synthetic data.



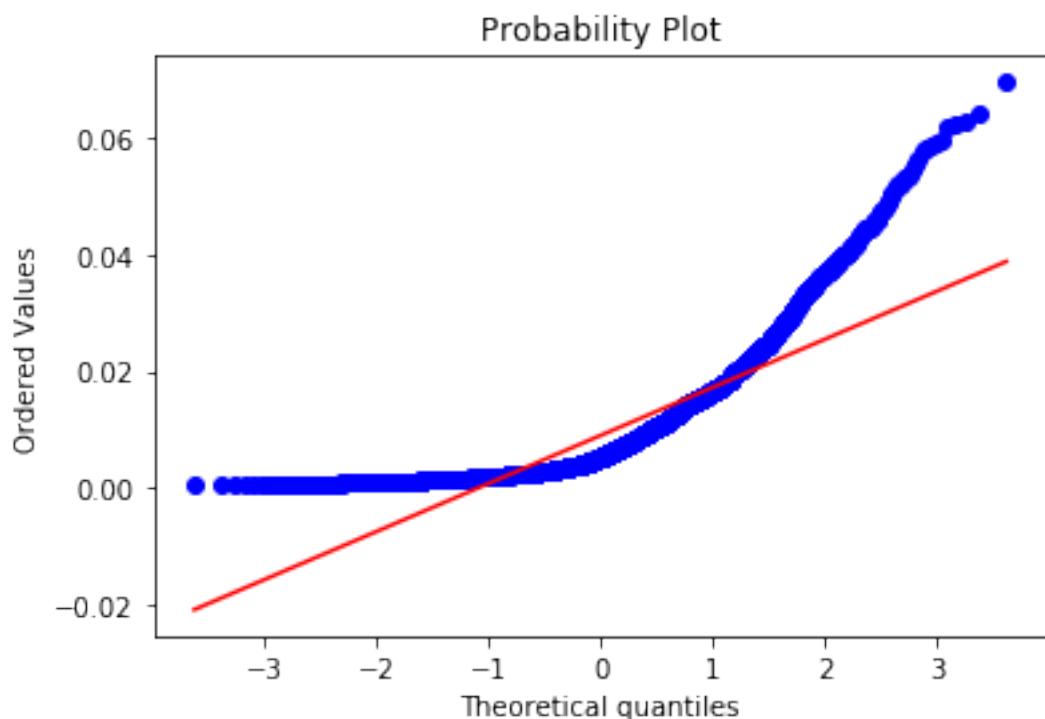
```
The mean error for nn2_50_net_flood_ is 0.03408509906166005 for length 4679  
=====
```

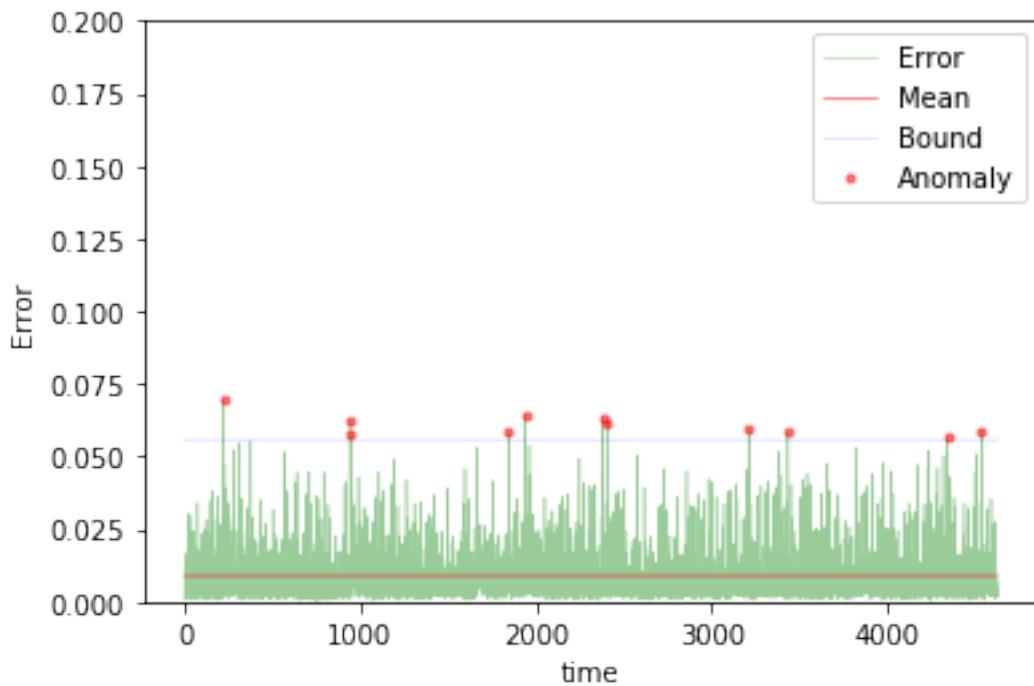
100 steps

```
In [112]: TIMESTEPS = 100  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn2_100"  
  
In [113]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(500, activation='relu')(input_layer)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [114]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [115]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

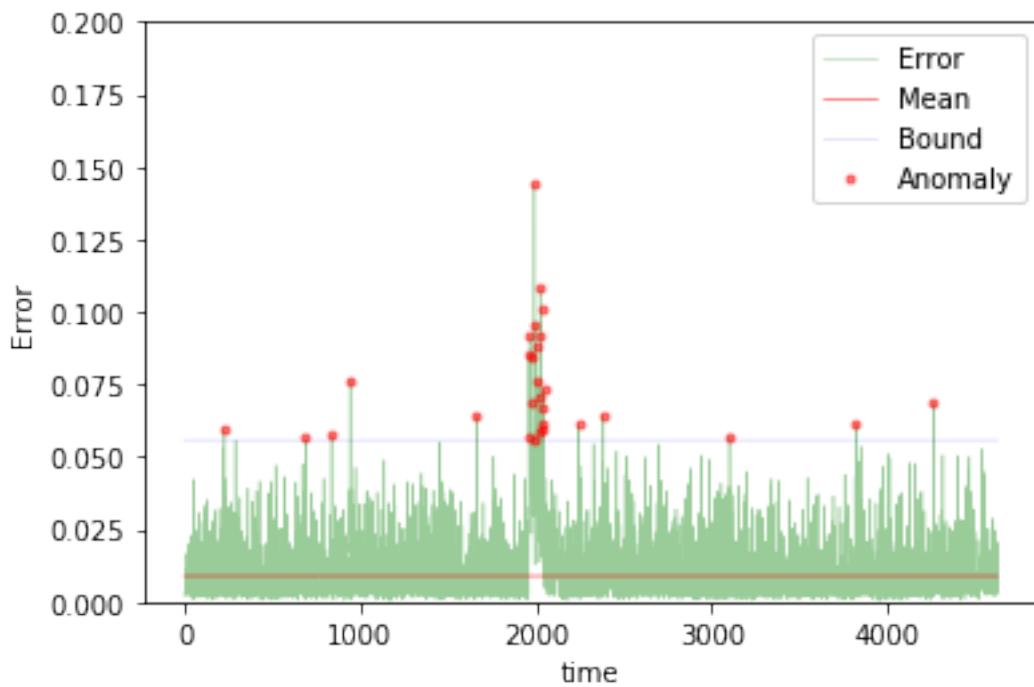


```
Training loss for final epoch is 0.009792730300454423
Validation loss for final epoch is 0.00970095186214894
----- Beginning tests for nn2_100 -----
Testing on normal data.
```

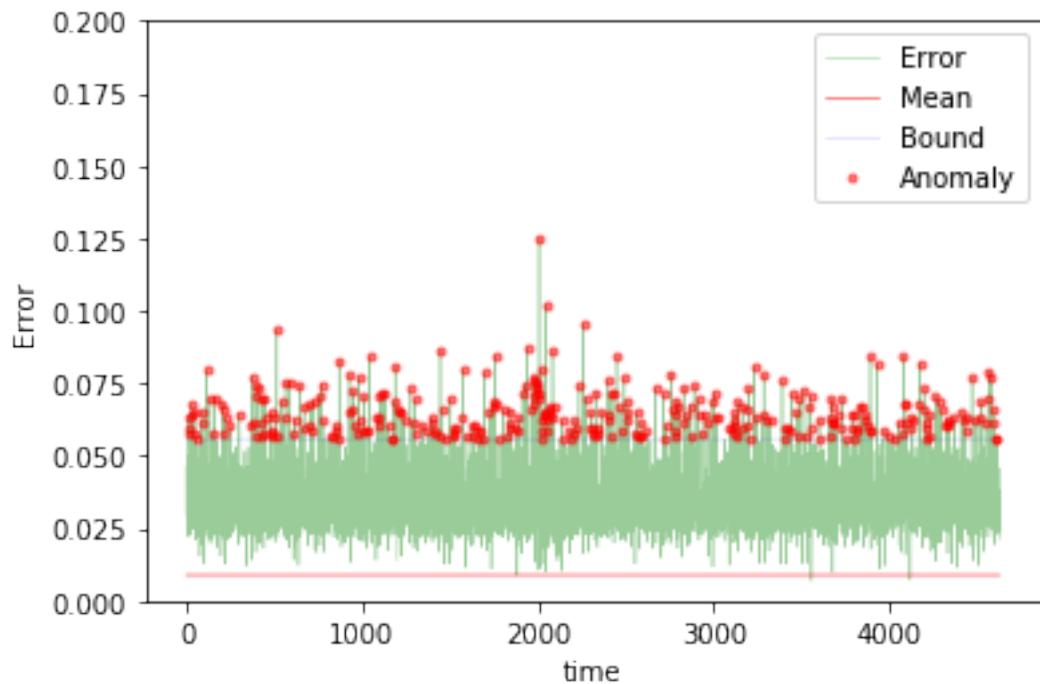




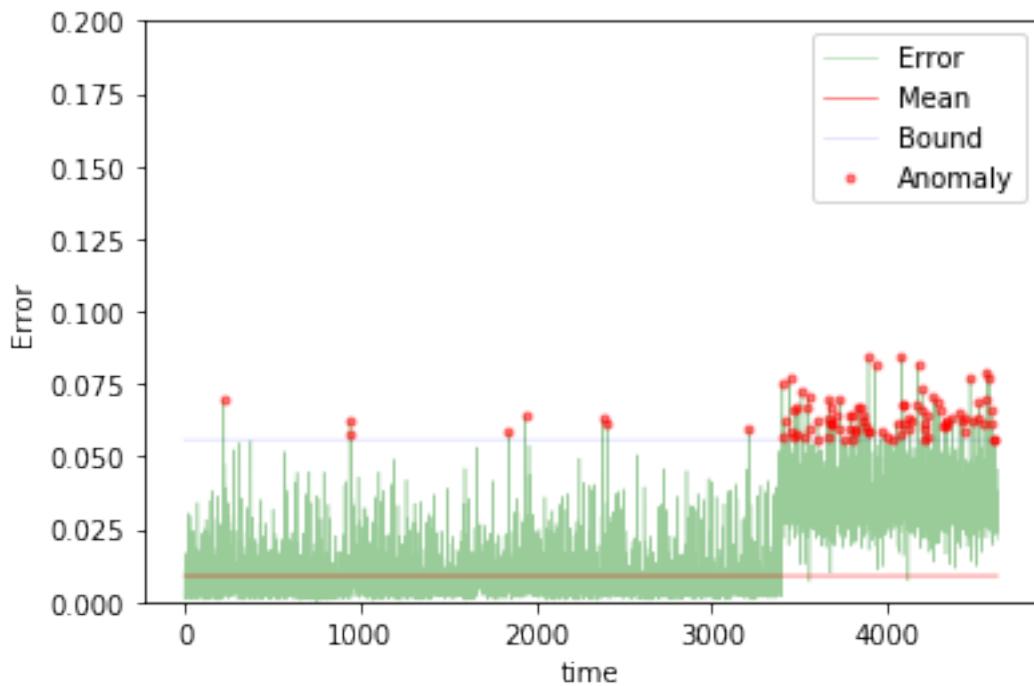
The mean error for nn2_100_normal_ is 0.008981598229170641 for length 4629
Testing on anomaly data.



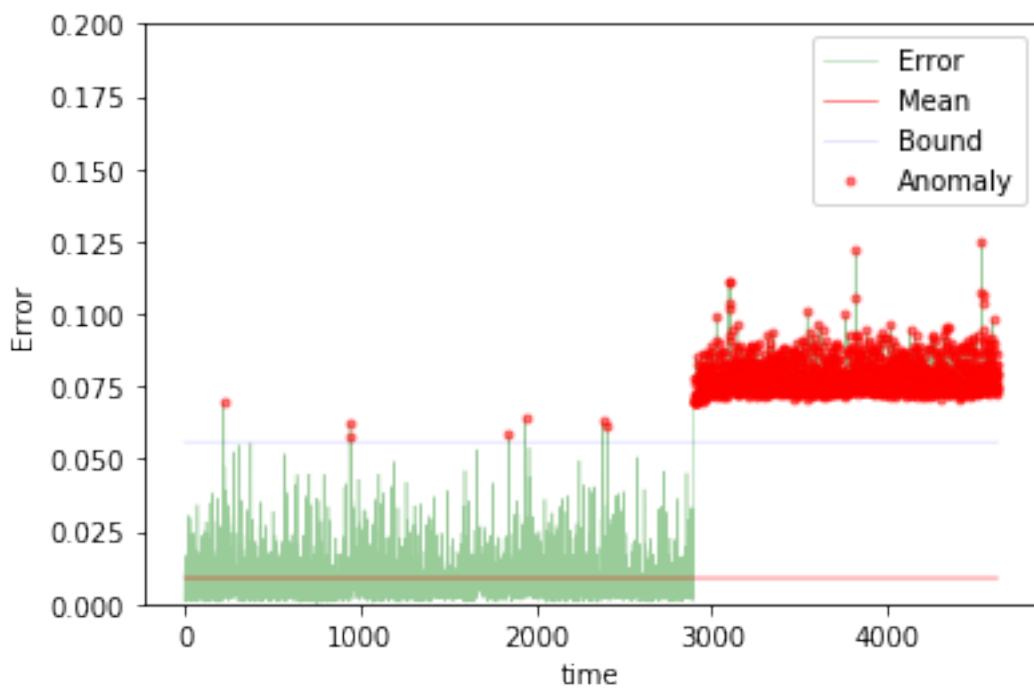
The mean error for nn2_100_anomaly_ is 0.010714489106242956 for length 4629
Testing on different app data.



The mean error for nn2_100_diff_app_ is 0.03570722856054408 for length 4629
Testing on App change synthetic data.



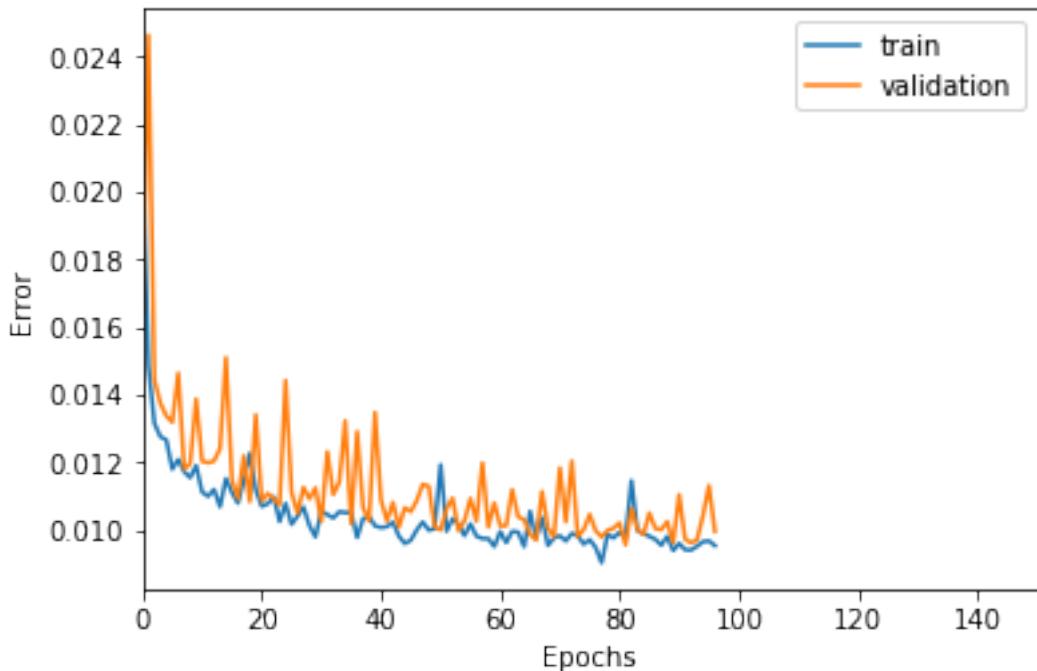
The mean error for nn2_100_app_change_ is 0.01605372827660438 for length 4629
Testing on Net flood synthetic data.



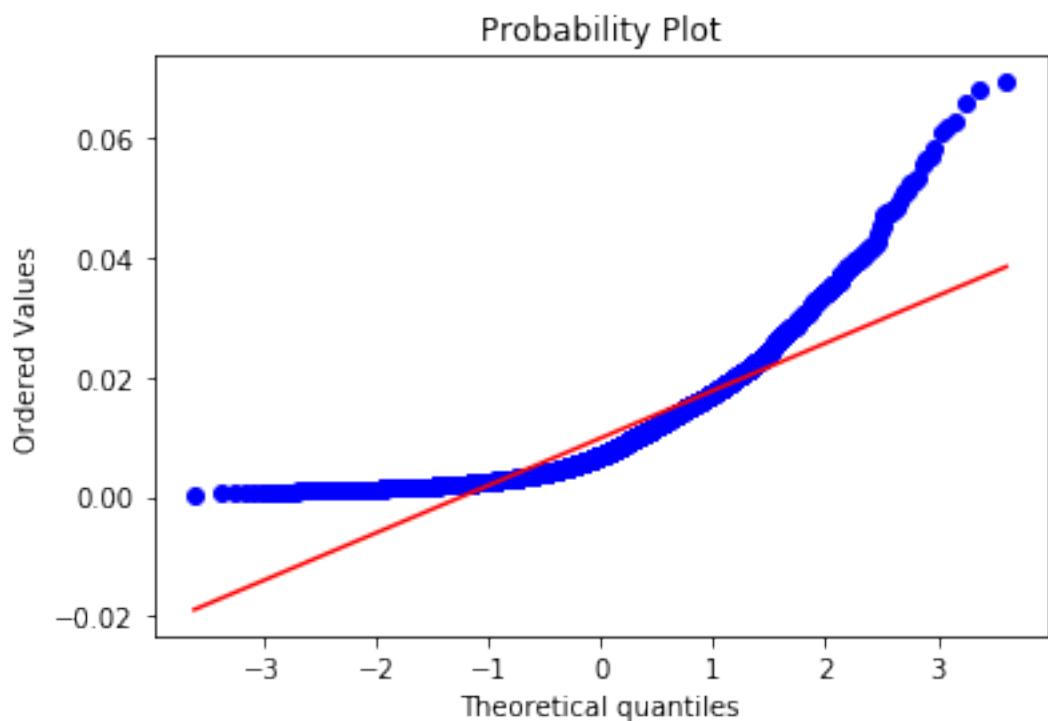
```
The mean error for nn2_100_net_flood_ is 0.03471105570901872 for length 4629  
=====
```

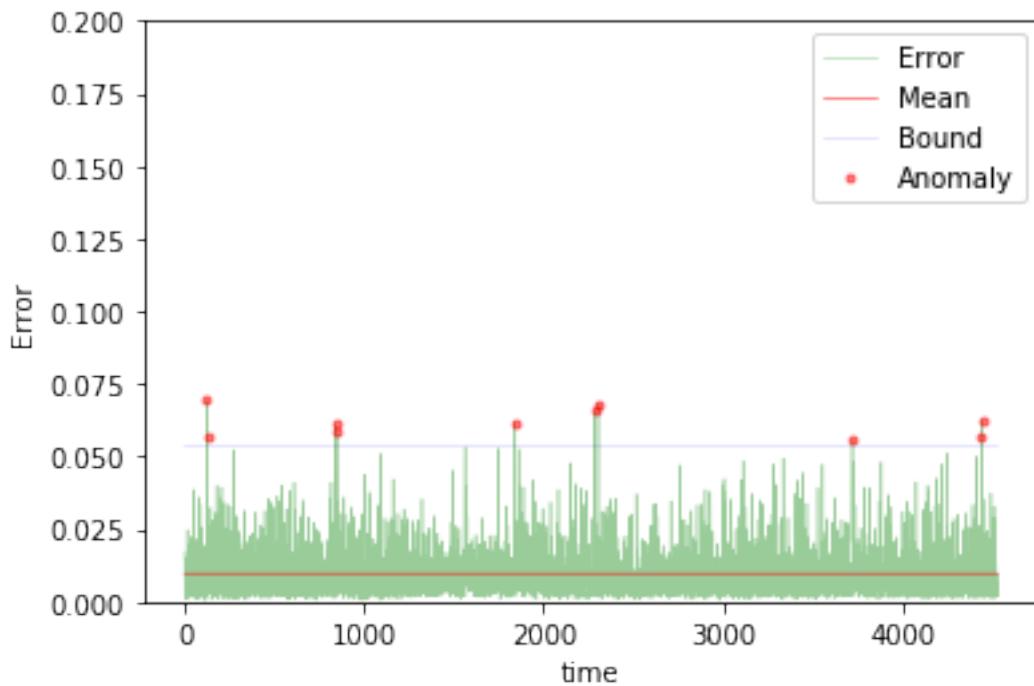
200 steps

```
In [116]: TIMESTEPS = 200  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn2_200"  
  
In [117]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(500, activation='relu')(input_layer)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [118]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [119]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```

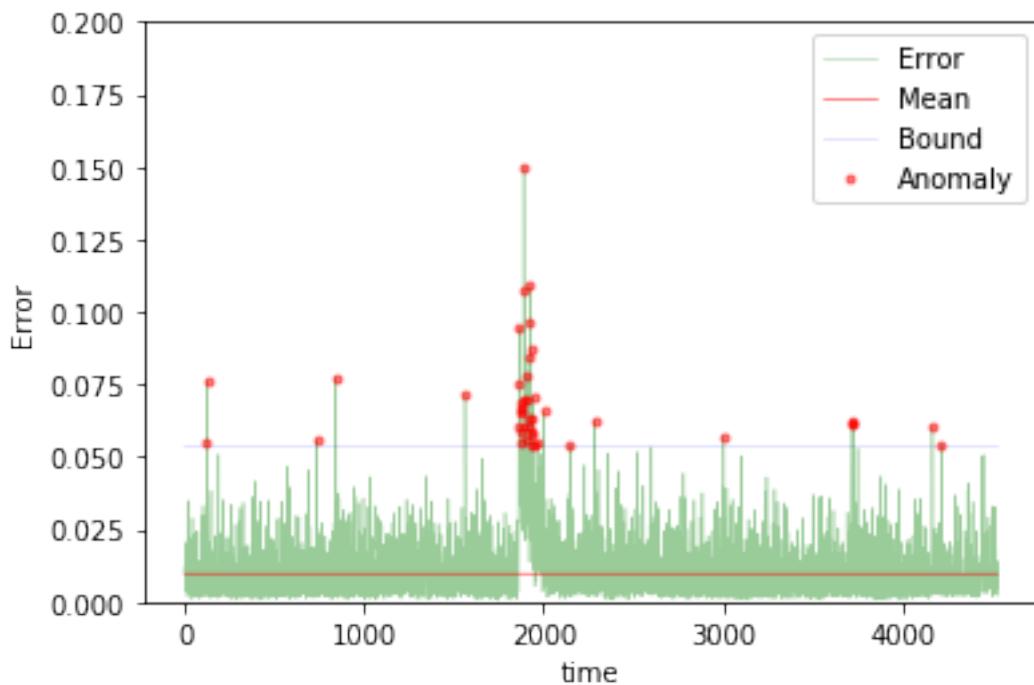


```
Training loss for final epoch is 0.00952873109781649
Validation loss for final epoch is 0.00995271112327464
----- Beginning tests for nn2_200 -----
Testing on normal data.
```

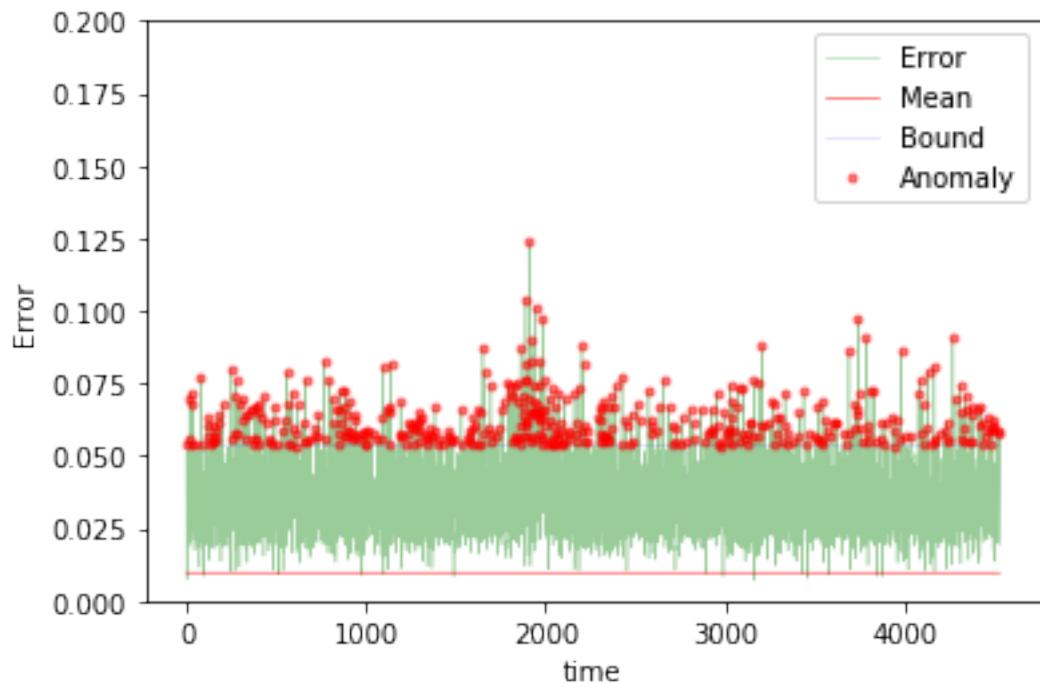




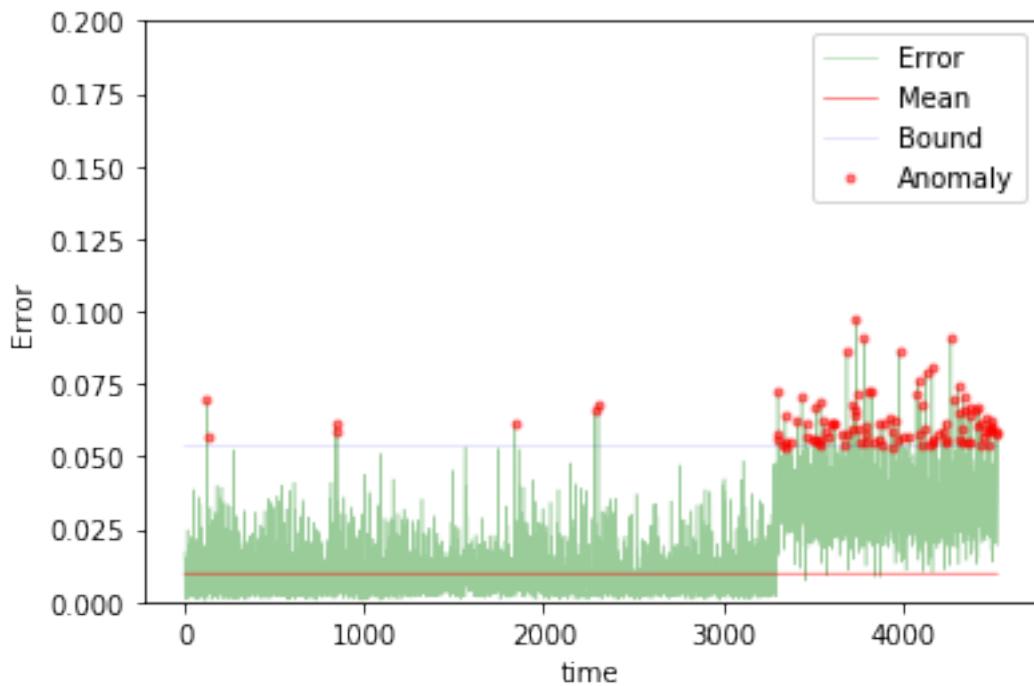
The mean error for nn2_200_normal_ is 0.009769785385386188 for length 4529
Testing on anomaly data.



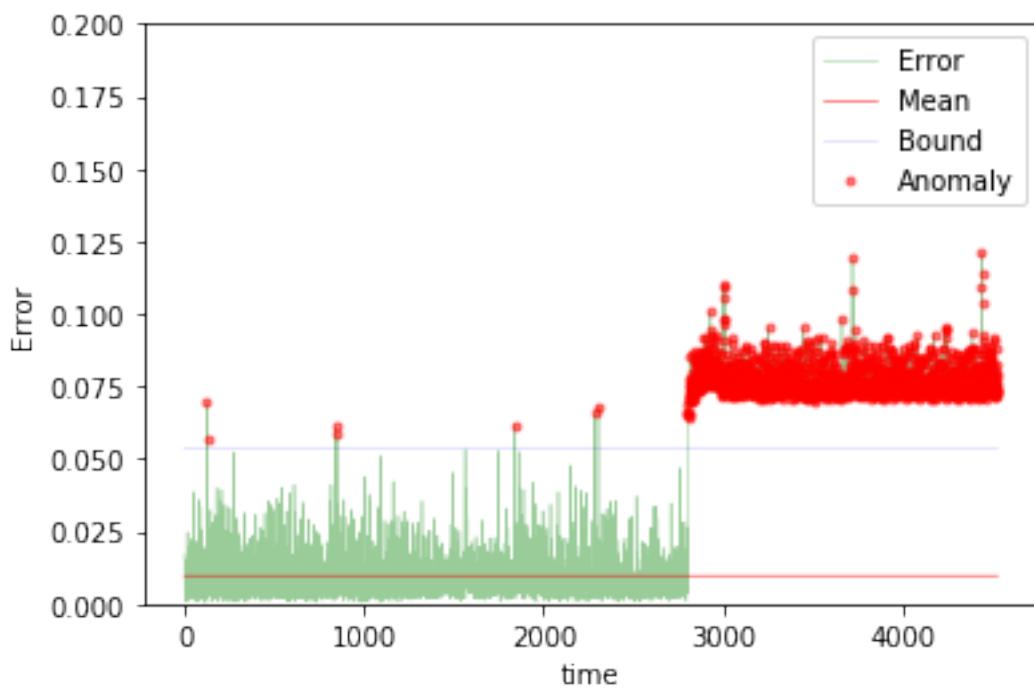
The mean error for nn2_200_anomaly_ is 0.011375198885782166 for length 4529
Testing on different app data.



The mean error for nn2_200_diff_app_ is 0.03551135454008039 for length 4529
Testing on App change synthetic data.



The mean error for nn2_200_app_change_ is 0.016634465068386232 for length 4529
Testing on Net flood synthetic data.

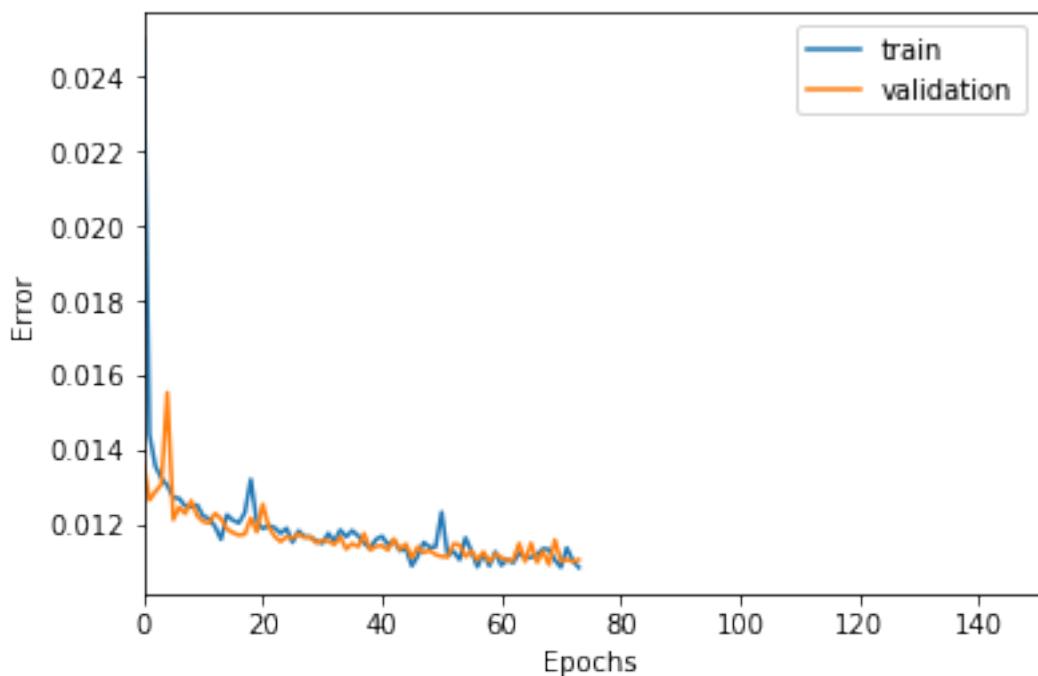


```
The mean error for nn2_200_net_flood_ is 0.03560348380619449 for length 4529  
=====
```

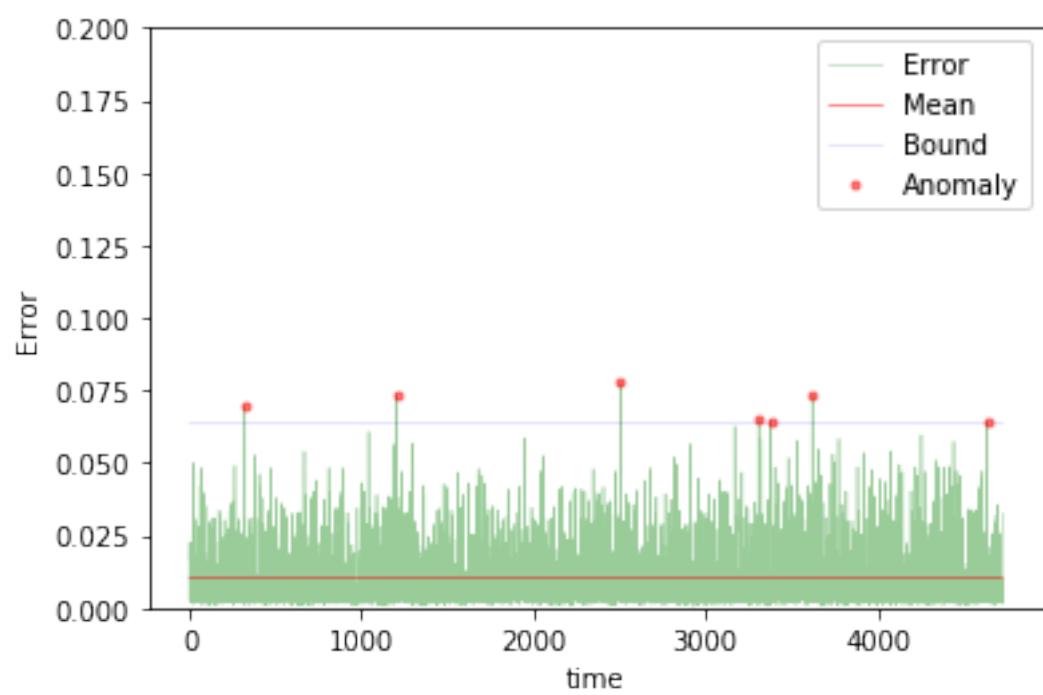
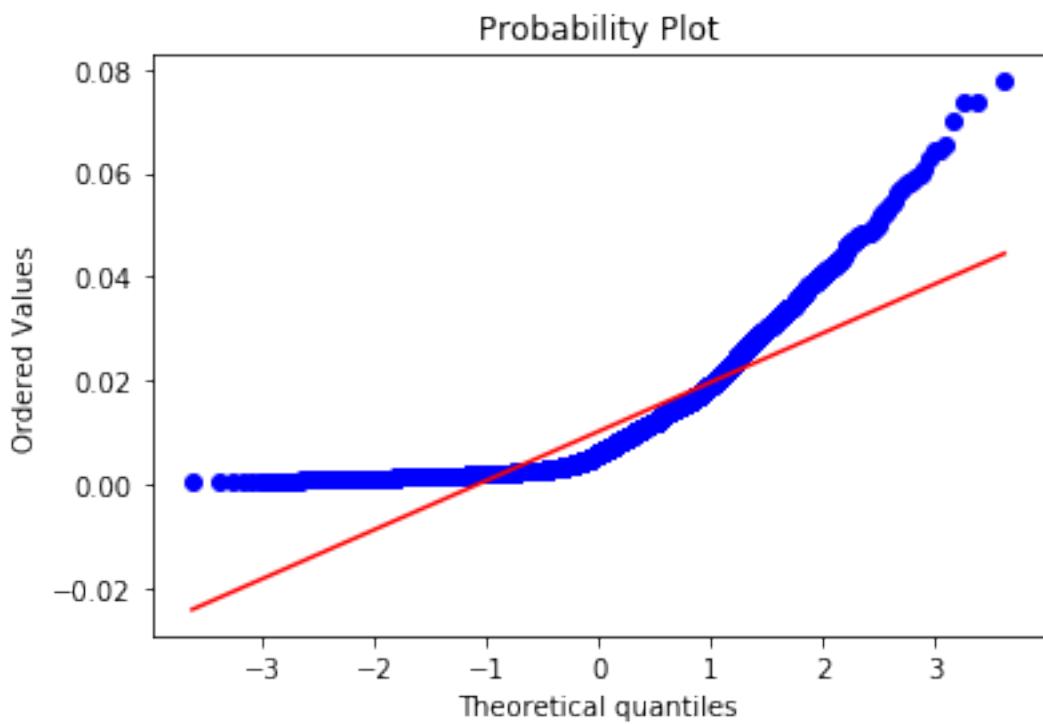
2.1.4 NN with 3 hidden layers

2 steps

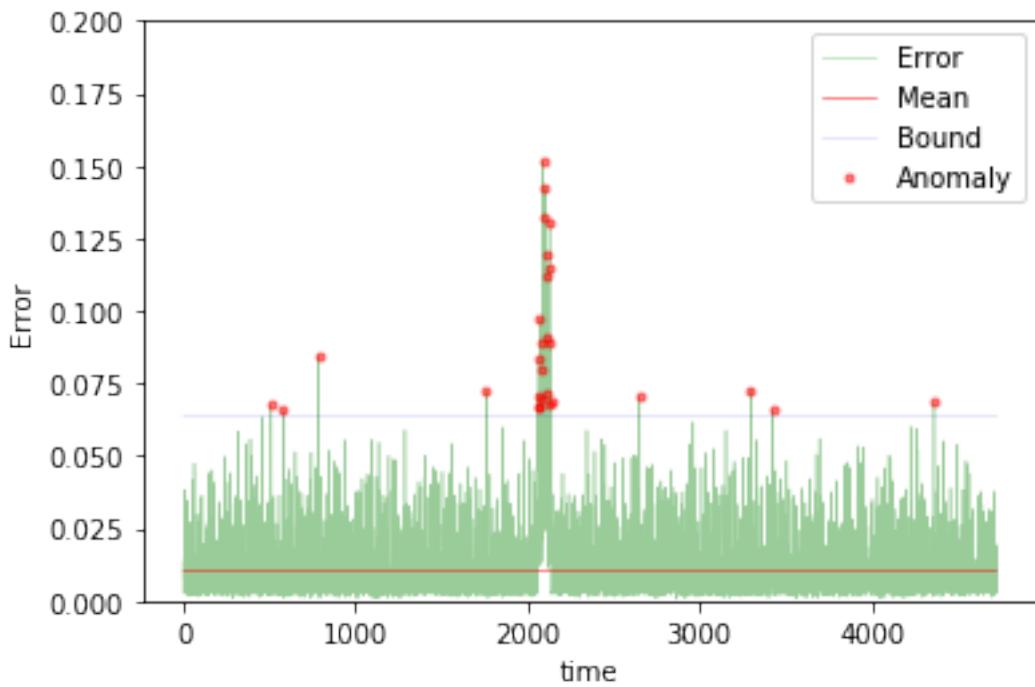
```
In [120]: TIMESTEPS = 2  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn3_2"  
  
In [121]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(1000, activation='relu')(input_layer)  
hidden = Dense(500, activation='relu')(hidden)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)  
  
In [122]: model = Model(input_layer, output)  
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])  
  
In [123]: train(model, tgen, vgen, name=name)  
test(model, name=name, window=TIMESTEPS)
```



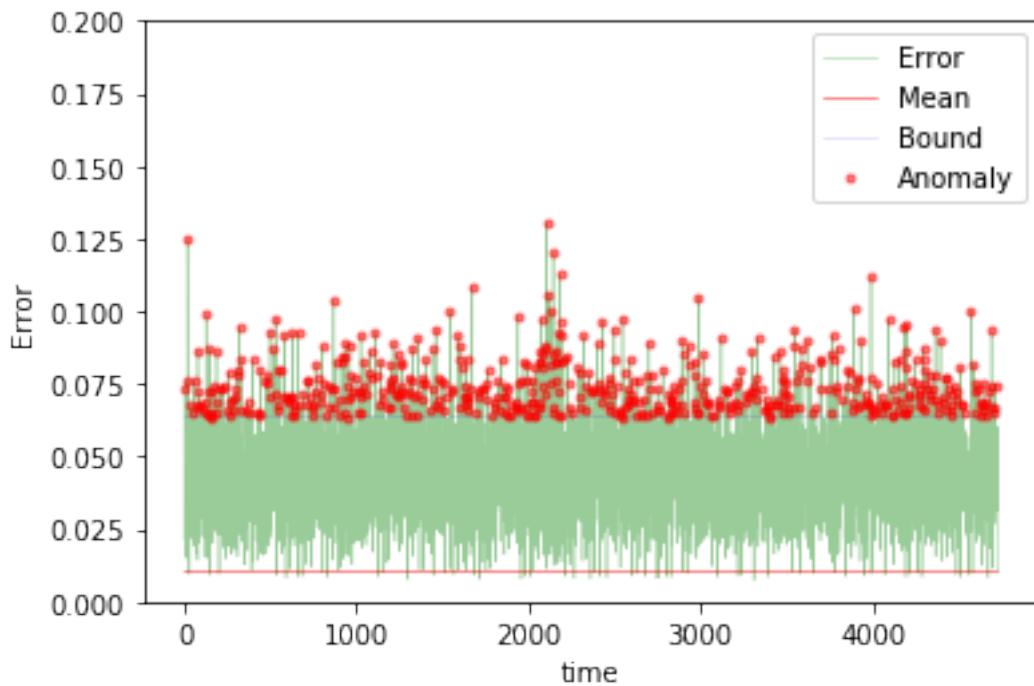
Training loss for final epoch is 0.010866318519343622
Validation loss for final epoch is 0.011076284590992145
----- Beginning tests for nn3_2 -----
Testing on normal data.



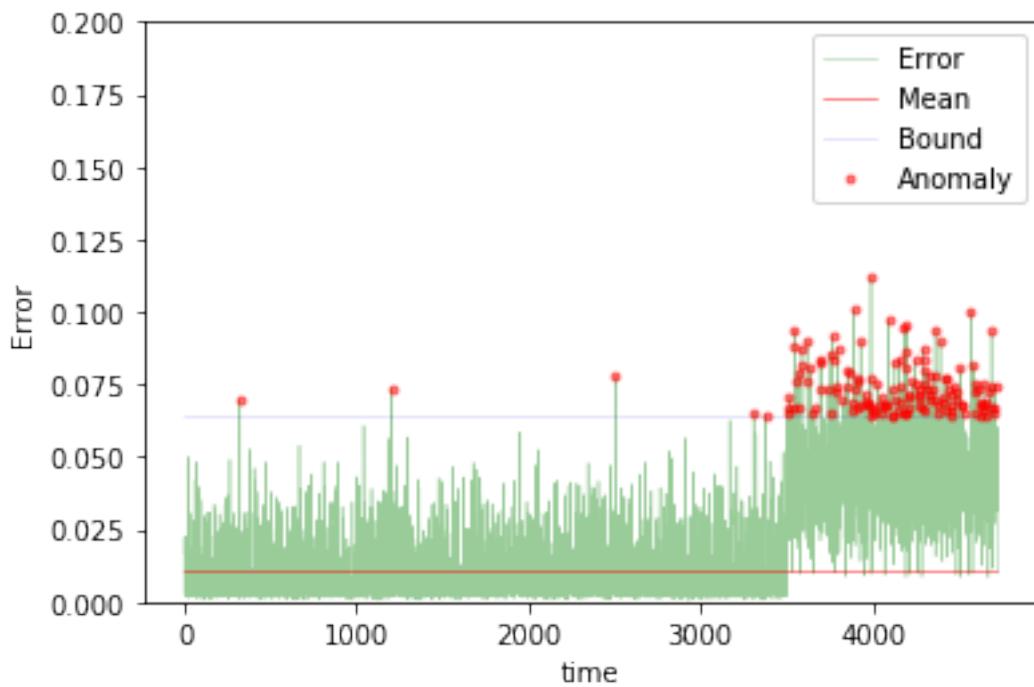
The mean error for nn3_2_normal_ is 0.010238071247660187 for length 4727
Testing on anomaly data.



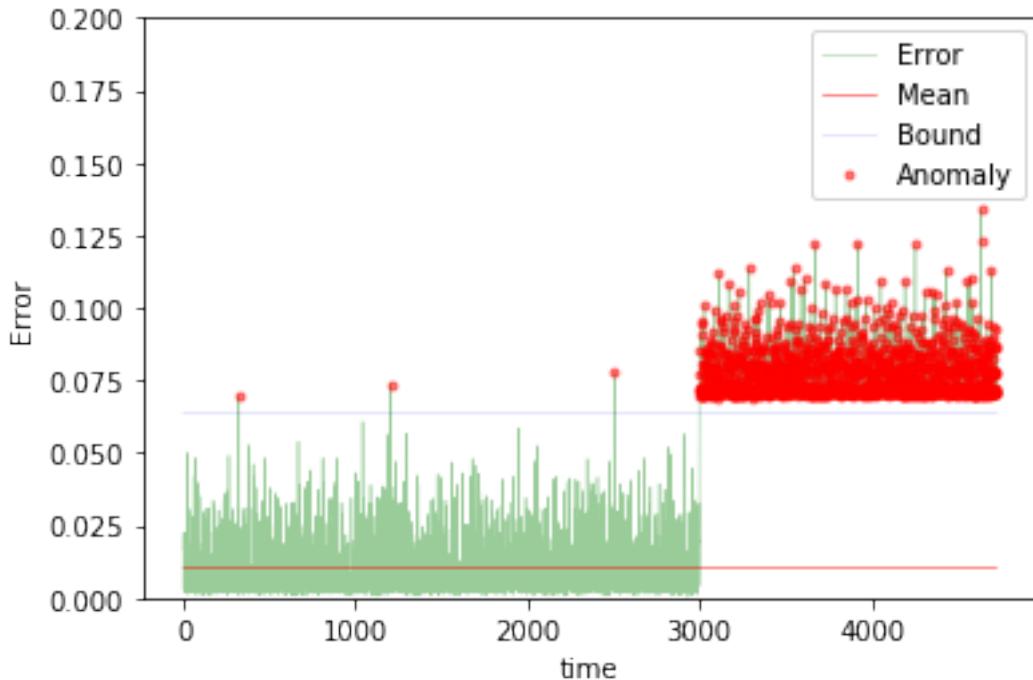
The mean error for nn3_2_anomaly_ is 0.01205387903275973 for length 4727
Testing on different app data.



The mean error for nn3_2_diff_app_ is 0.0451358807631471 for length 4727
Testing on App change synthetic data.



The mean error for nn3_2_app_change_ is 0.019270191742930935 for length 4727
Testing on Net flood synthetic data.



The mean error for nn3_2_net_flood_ is 0.03452772106217477 for length 4727
=====

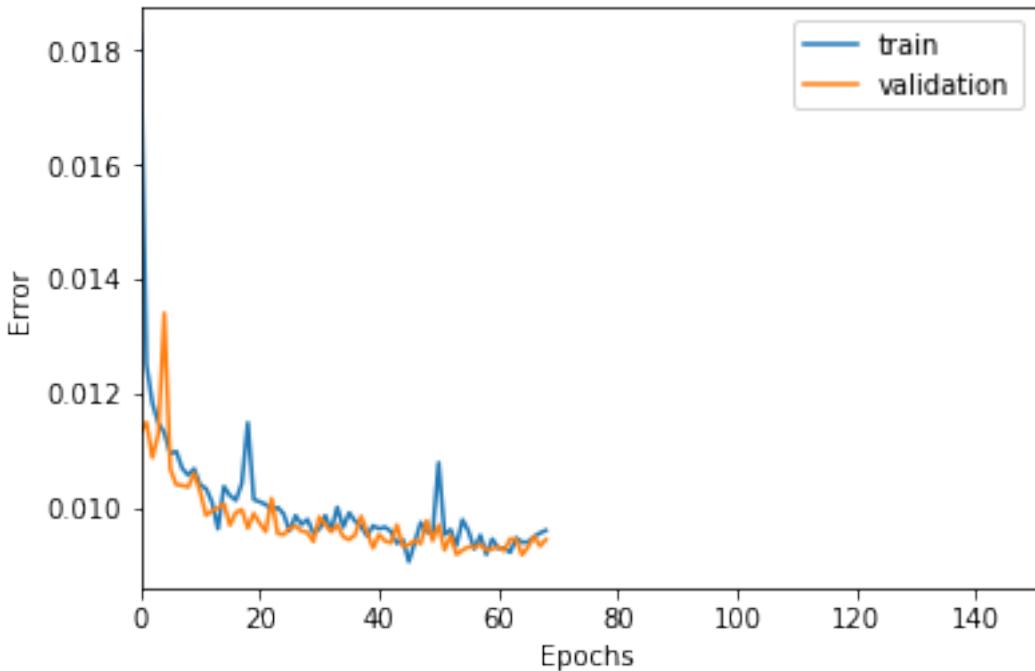
5 steps

```
In [124]: TIMESTEPS = 5
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS)
          vgen = flat_generator(val_X, TIMESTEPS)
          name = "nn3_5"

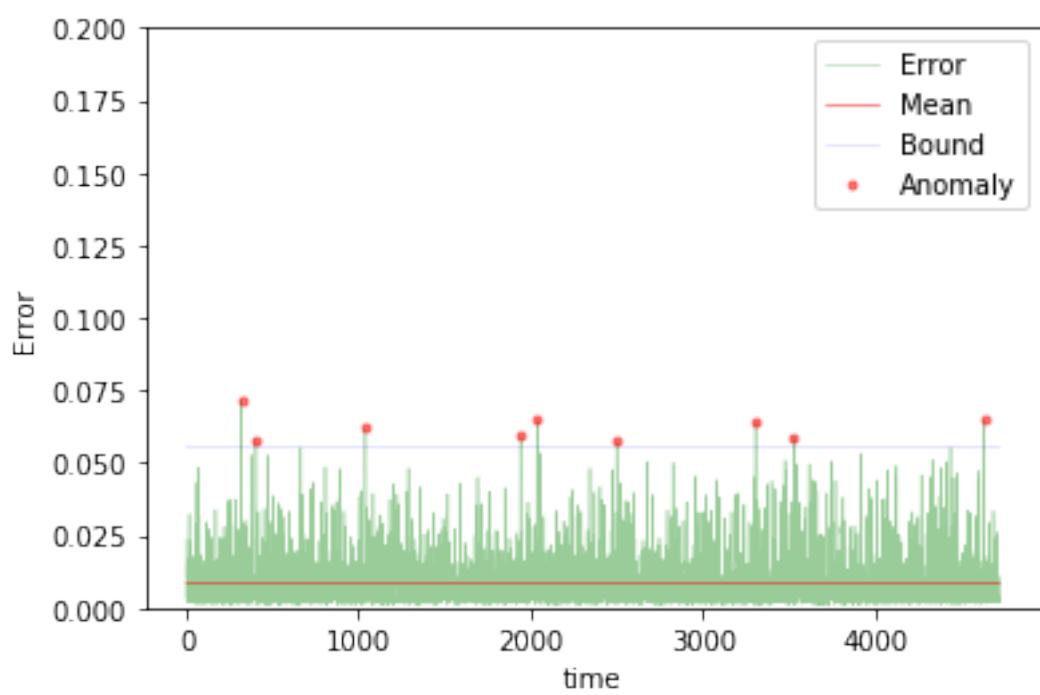
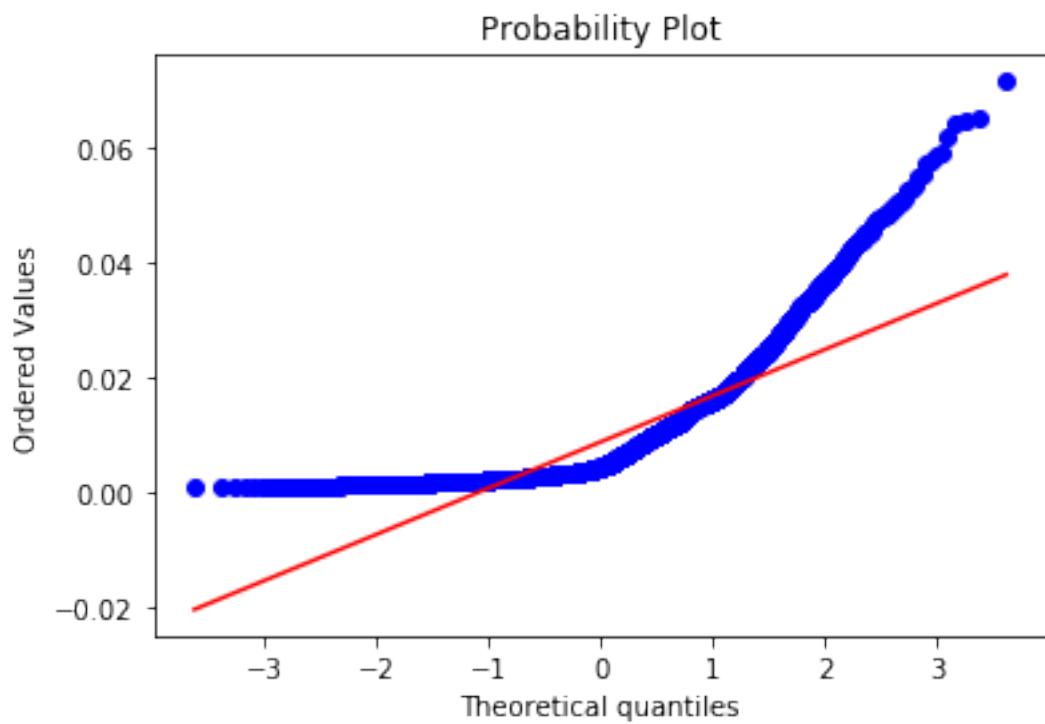
In [125]: input_layer = Input(shape=(TIMESTEPS*DIM,))
          hidden = Dense(1000, activation='relu')(input_layer)
          hidden = Dense(500, activation='relu')(hidden)
          hidden = Dense(100, activation='relu')(hidden)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [126]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

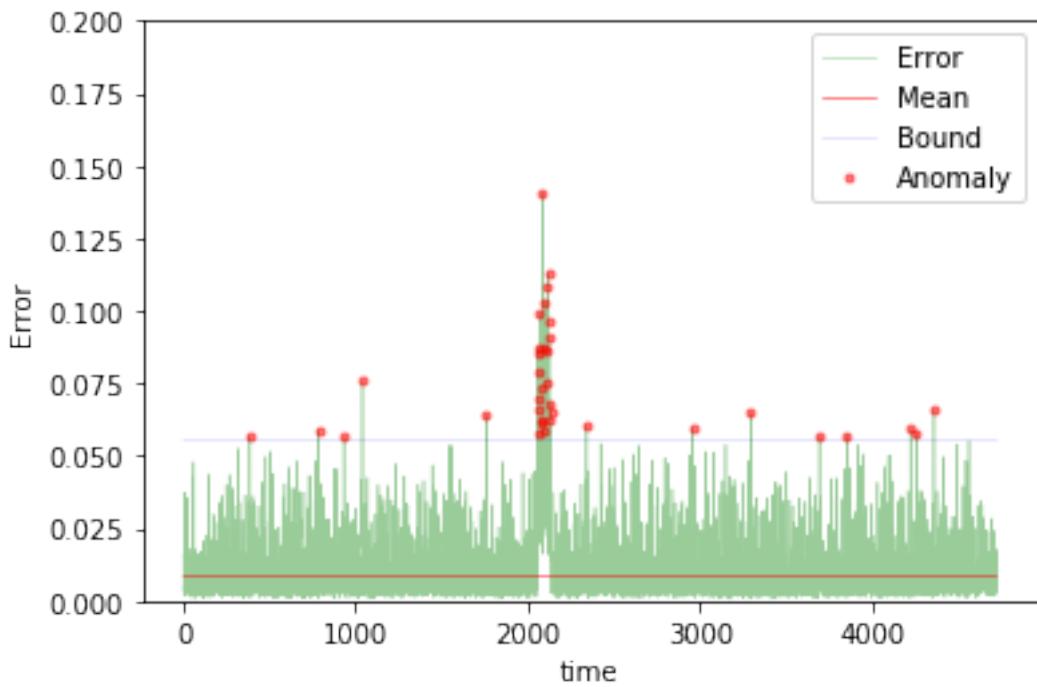
In [127]: train(model, tgen, vgen, name=name)
test(model, name=name, window=TIMESTEPS)
```



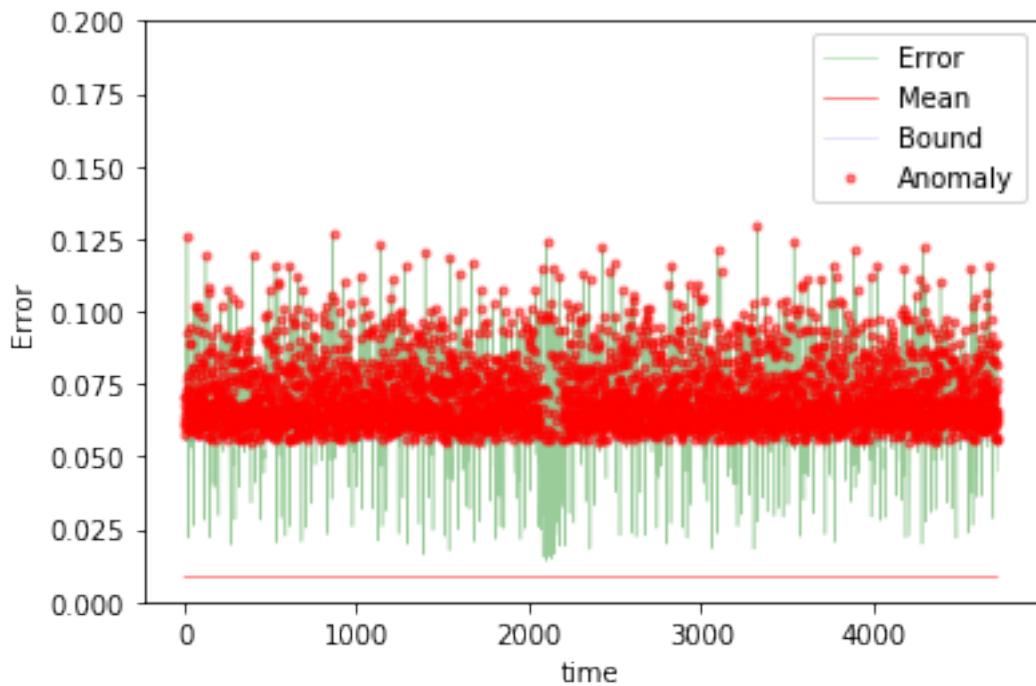
```
Training loss for final epoch is 0.00959795557463076
Validation loss for final epoch is 0.009435736372601242
----- Beginning tests for nn3_5 -----
Testing on normal data.
```



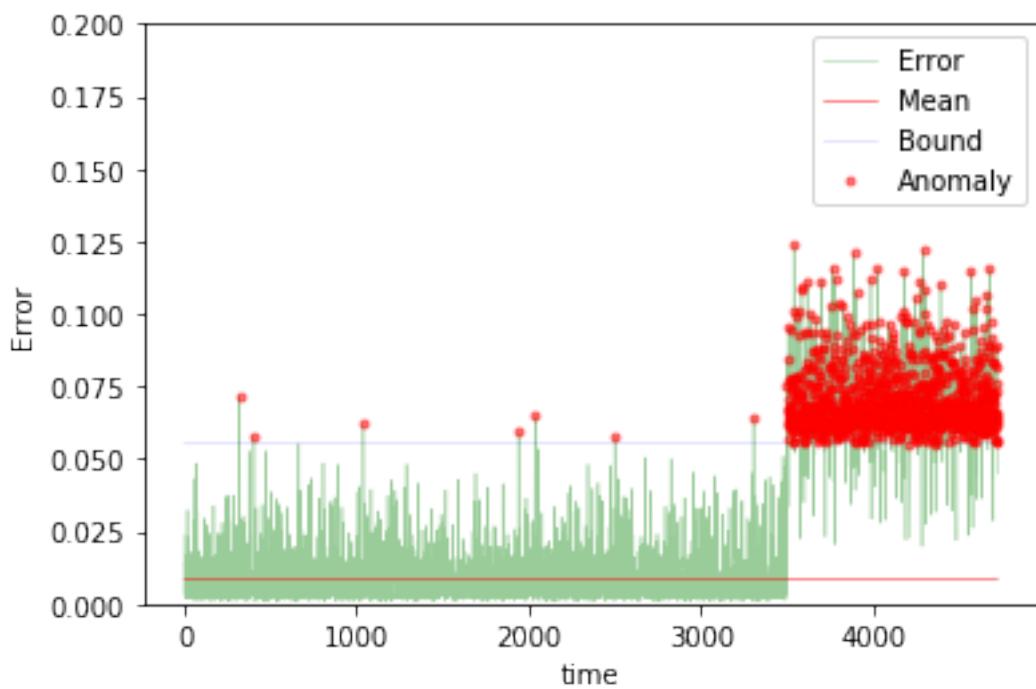
The mean error for nn3_5_normal_ is 0.008726265523861248 for length 4724
Testing on anomaly data.



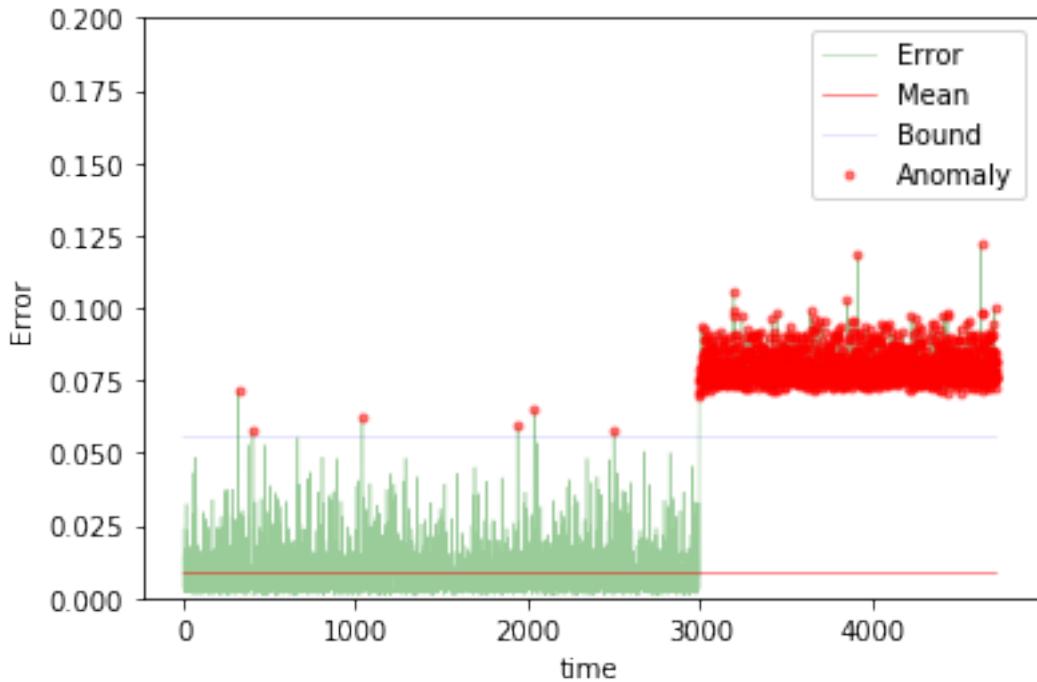
The mean error for nn3_5_anomaly_ is 0.010463631794435097 for length 4724
Testing on different app data.



The mean error for nn3_5_diff_app_ is 0.06711722259439397 for length 4724
Testing on App change synthetic data.



The mean error for nn3_5_app_change_ is 0.024000183008989515 for length 4724
Testing on Net flood synthetic data.



The mean error for nn3_5_net_flood_ is 0.03459319613735467 for length 4724
=====

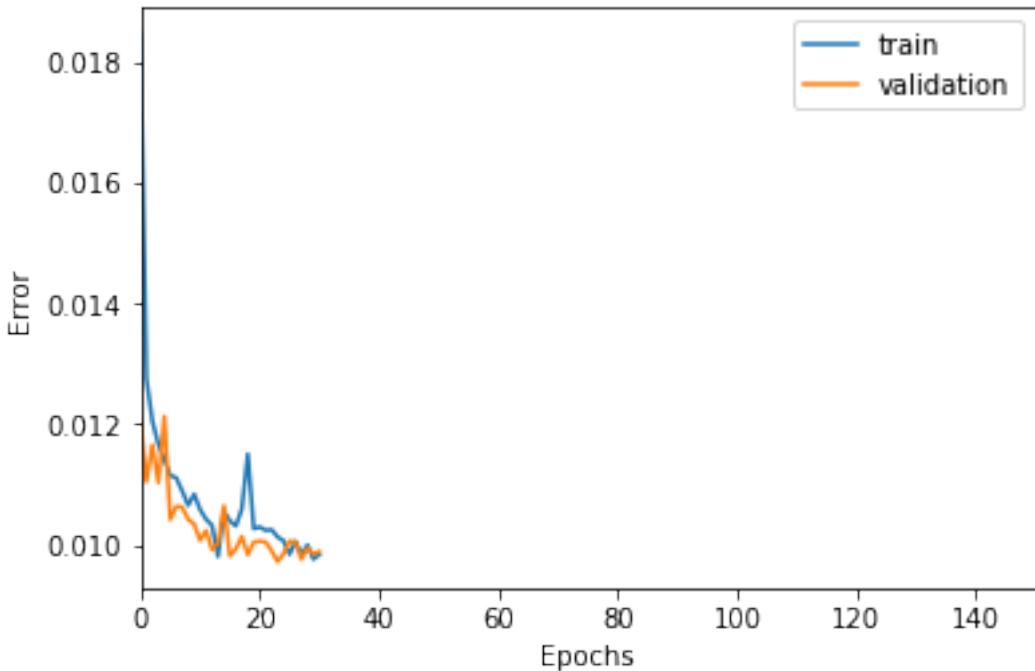
10 steps

```
In [128]: TIMESTEPS = 10
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS)
          vgen = flat_generator(val_X, TIMESTEPS)
          name = "nn3_10"

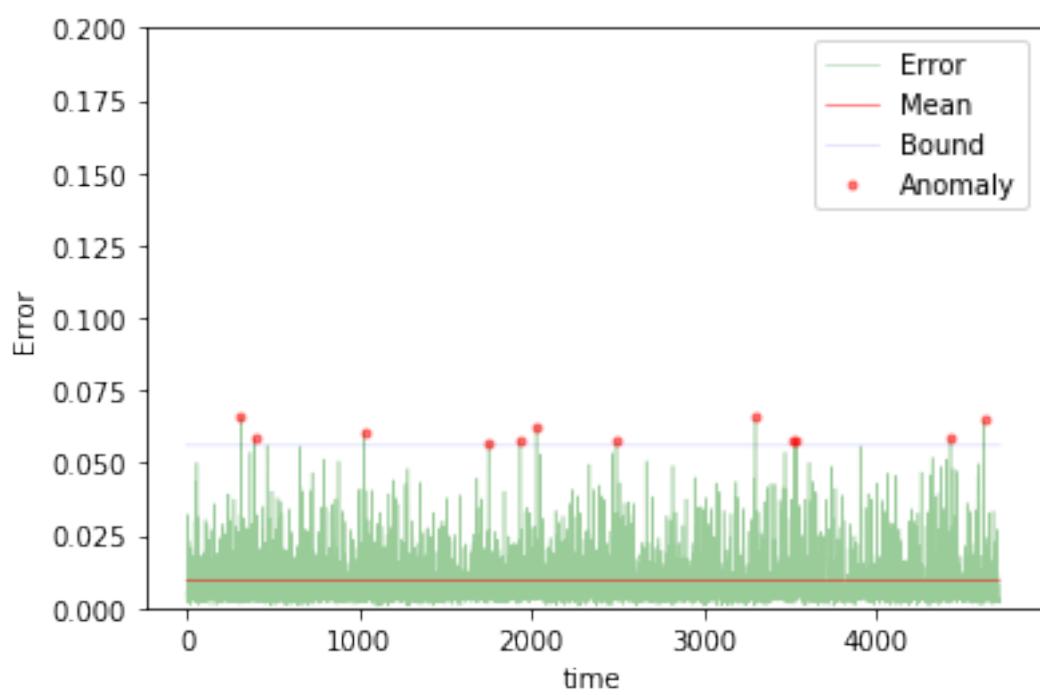
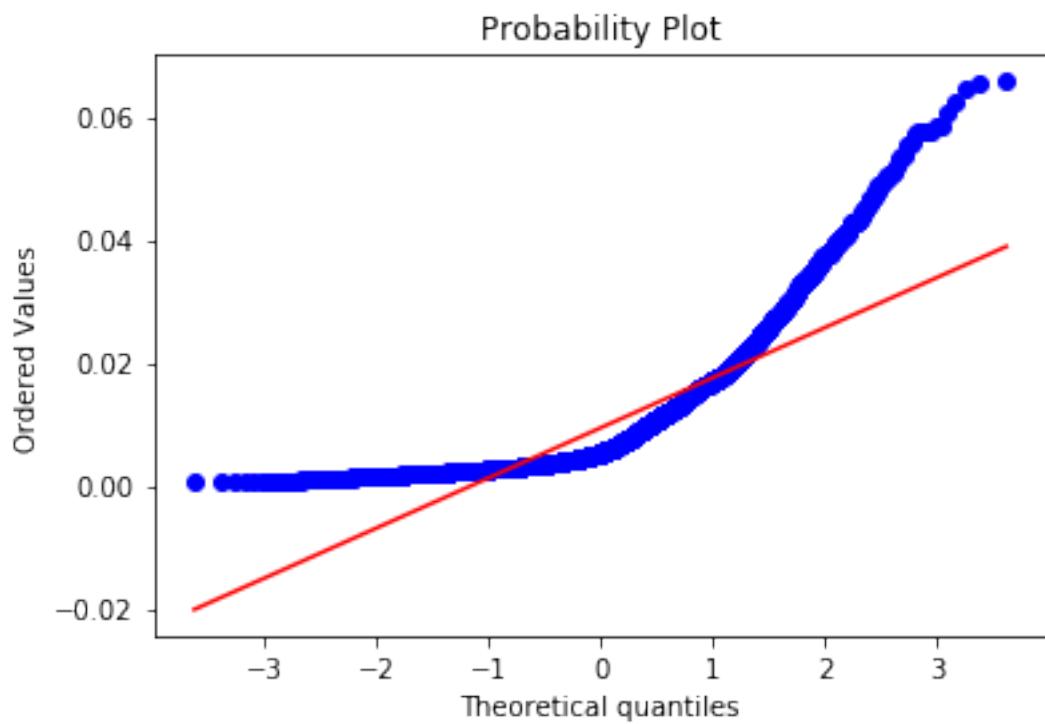
In [129]: input_layer = Input(shape=(TIMESTEPS*DIM,))
          hidden = Dense(1000, activation='relu')(input_layer)
          hidden = Dense(500, activation='relu')(hidden)
          hidden = Dense(100, activation='relu')(hidden)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [130]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

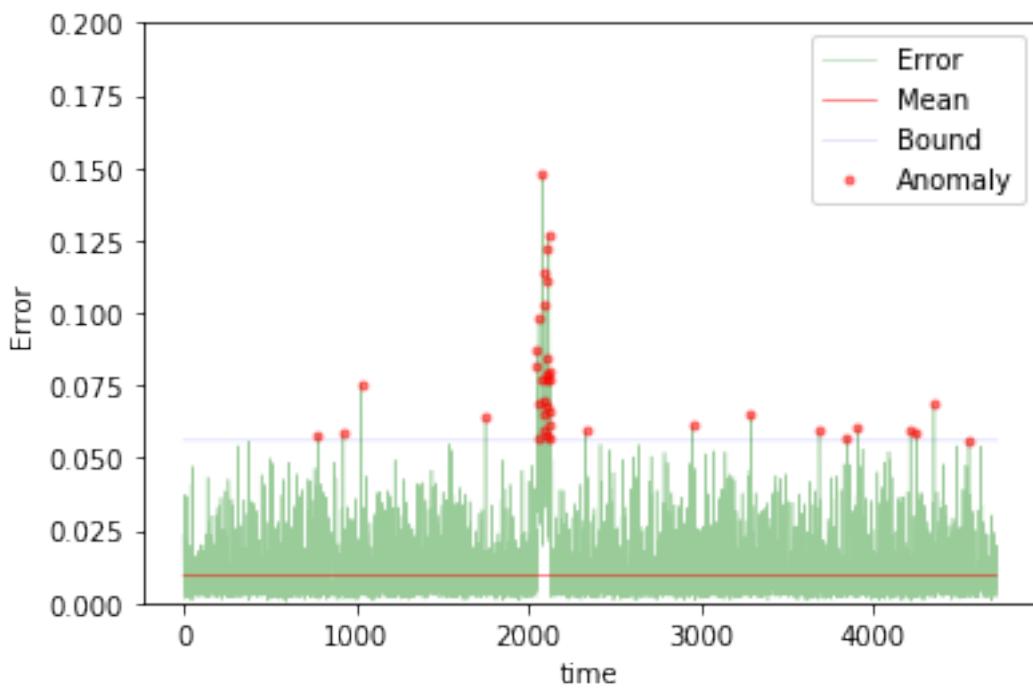
In [131]: train(model, tgen, vgen, name=name)
          test(model, name=name, window=TIMESTEPS)
```



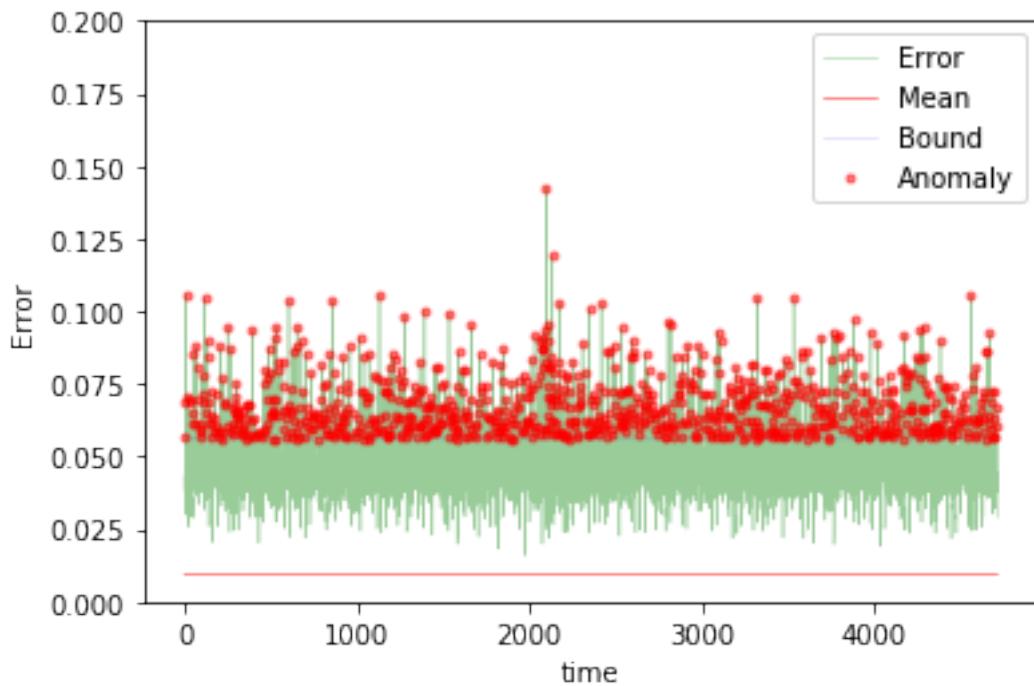
```
Training loss for final epoch is 0.009841431740438566
Validation loss for final epoch is 0.00988885405624751
----- Beginning tests for nn3_10 -----
Testing on normal data.
```



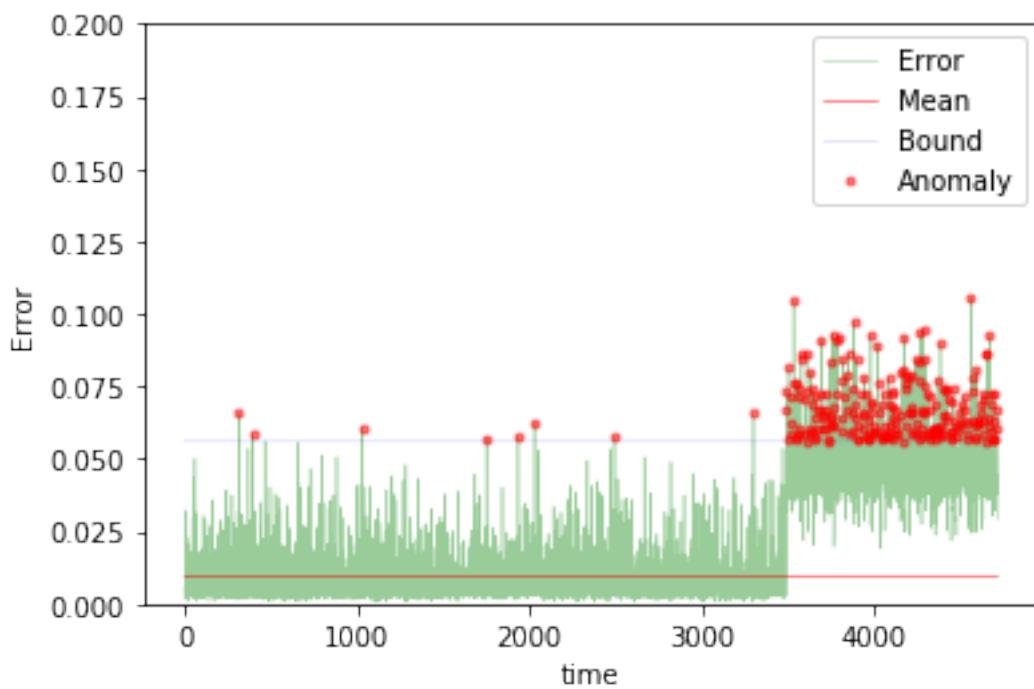
The mean error for nn3_10_normal_ is 0.00937045241451355 for length 4719
Testing on anomaly data.



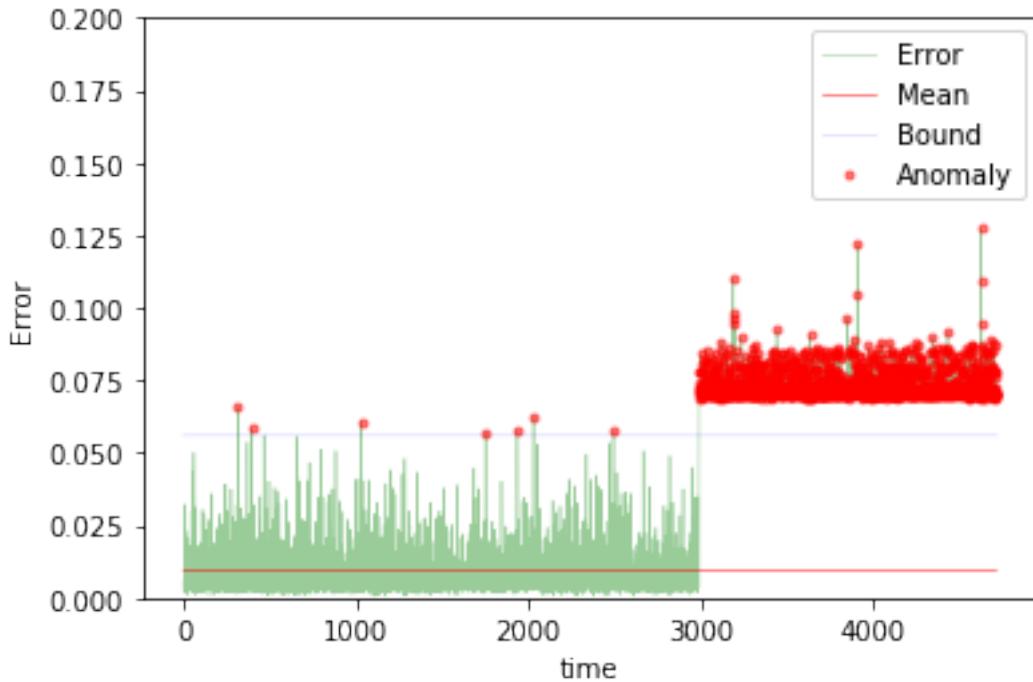
The mean error for nn3_10_anomaly_ is 0.011124507398845707 for length 4719
Testing on different app data.



The mean error for nn3_10_diff_app_ is 0.047638491397877104 for length 4719
Testing on App change synthetic data.



```
The mean error for nn3_10_app_change_ is 0.01925040610901939 for length 4719
Testing on Net flood synthetic data.
```



```
The mean error for nn3_10_net_flood_ is 0.032924405471116455 for length 4719
=====
=====
```

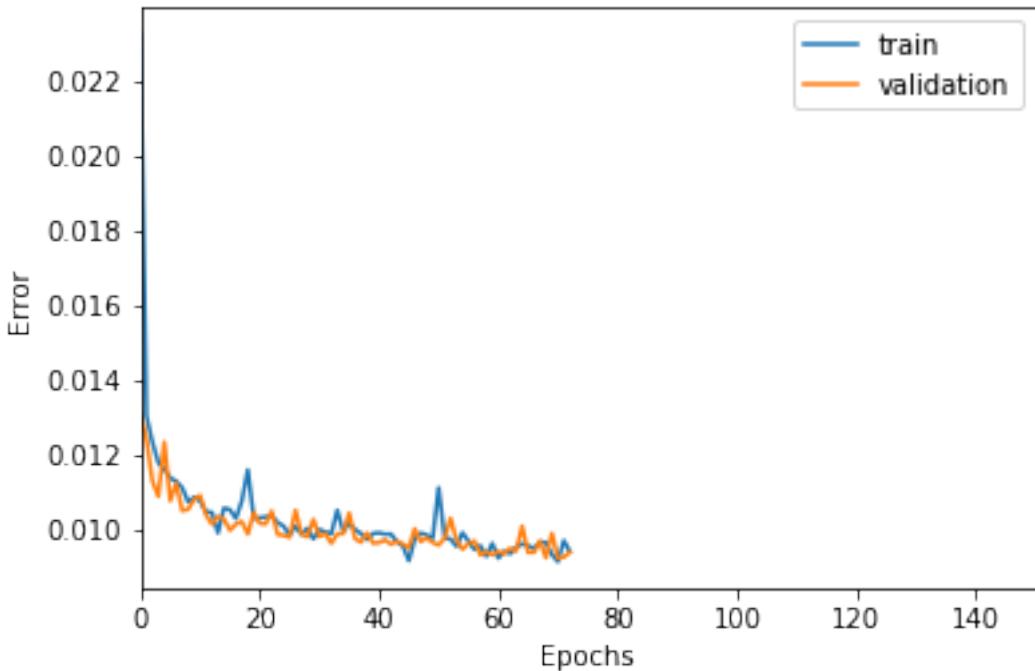
20 steps

```
In [132]: TIMESTEPS = 20
DIM = 29
tgen = flat_generator(X, TIMESTEPS)
vgen = flat_generator(val_X, TIMESTEPS)
name = "nn3_20"
```

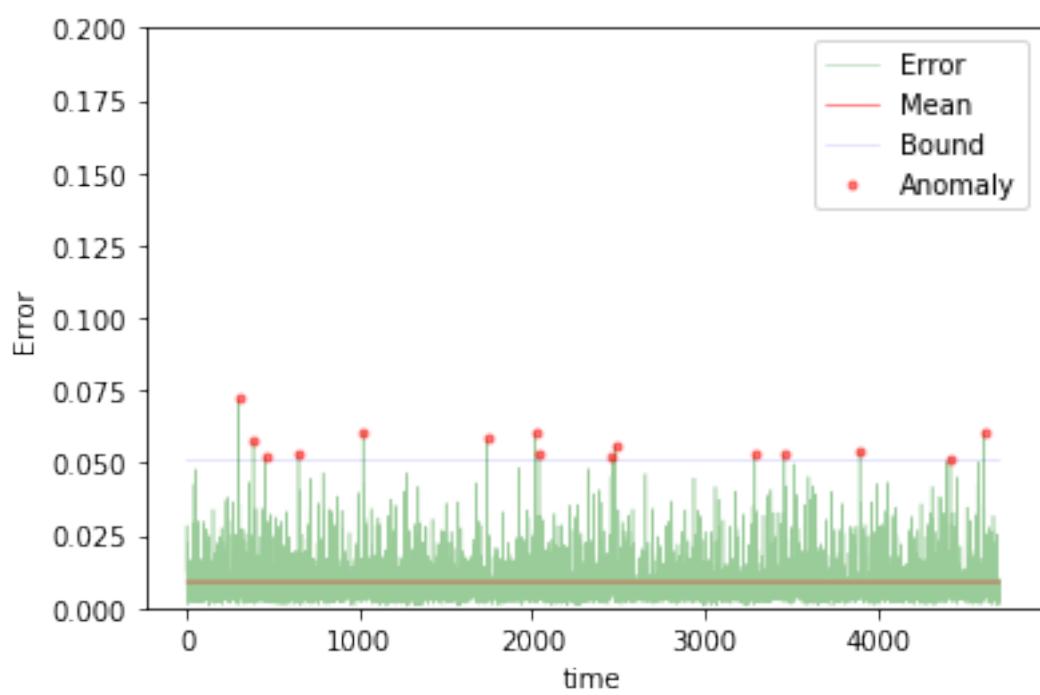
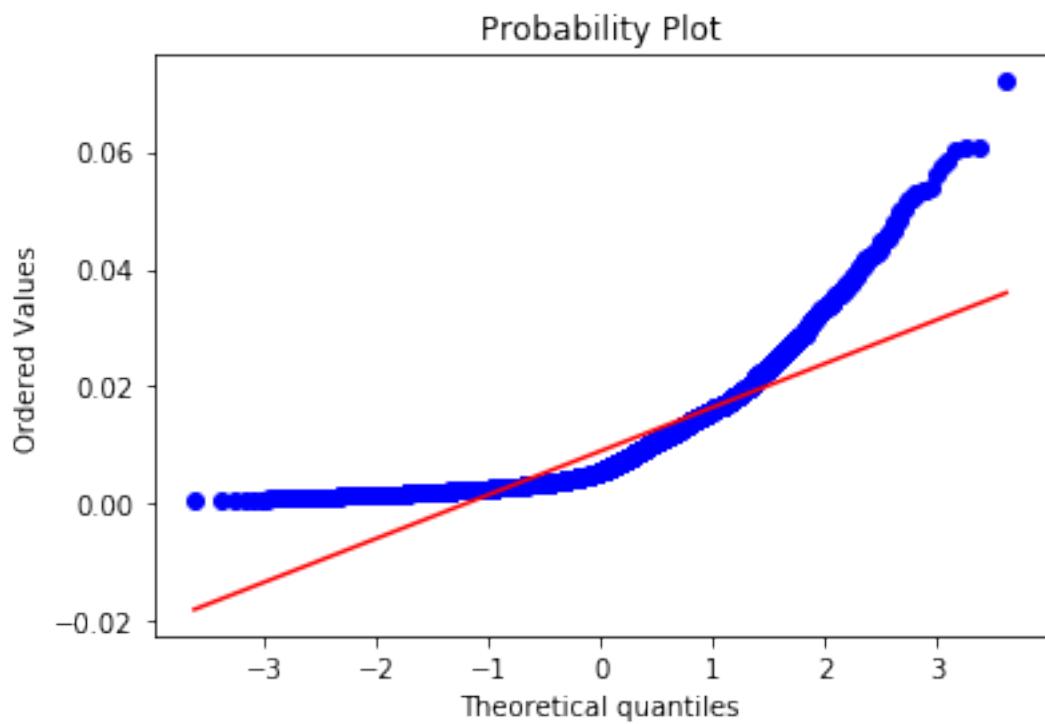
```
In [133]: input_layer = Input(shape=(TIMESTEPS*DIM,))
hidden = Dense(1000, activation='relu')(input_layer)
hidden = Dense(500, activation='relu')(hidden)
hidden = Dense(100, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [134]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

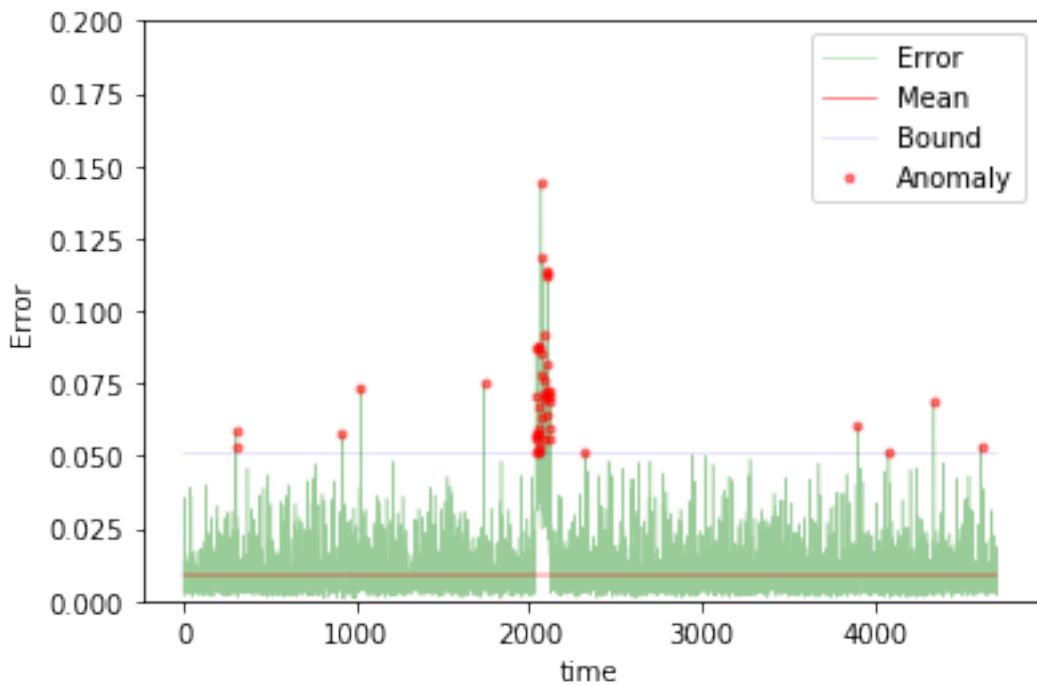
In [135]: train(model, tgen, vgen, name=name)
test(model, name=name, window=TIMESTEPS)
```



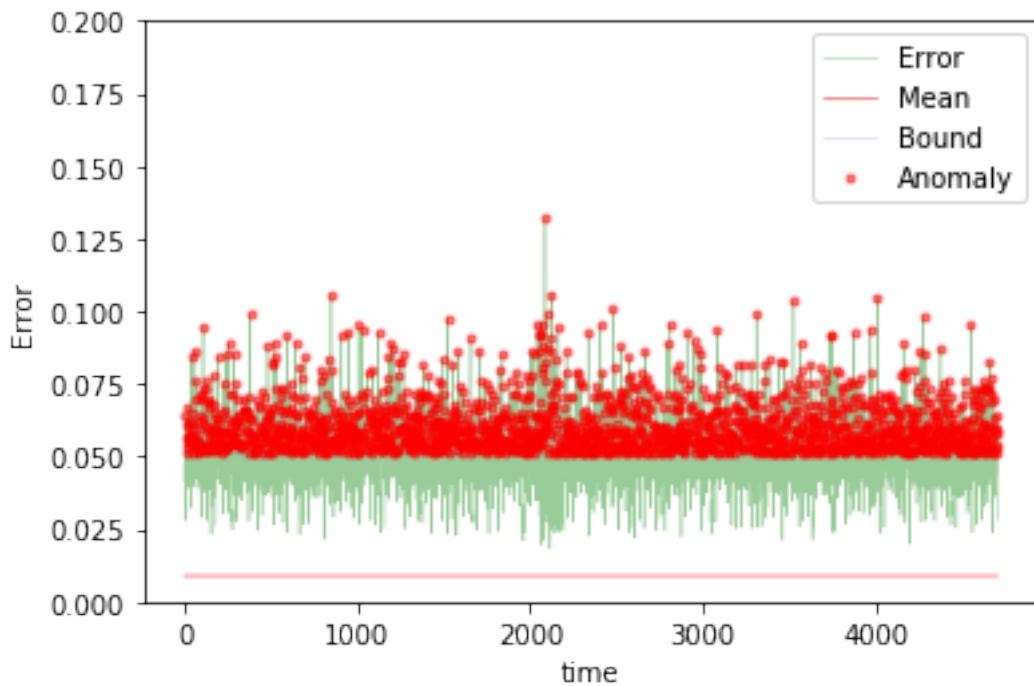
```
Training loss for final epoch is 0.009421228910330683
Validation loss for final epoch is 0.009401601006276905
----- Beginning tests for nn3_20 -----
Testing on normal data.
```



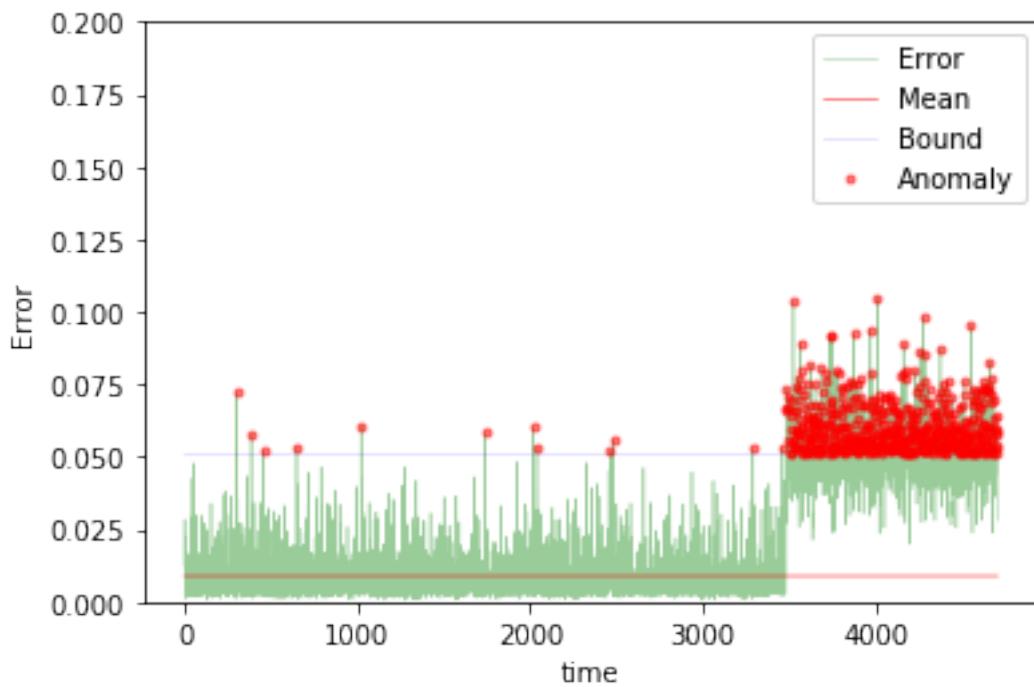
The mean error for nn3_20_normal_ is 0.008986661717485876 for length 4709
Testing on anomaly data.



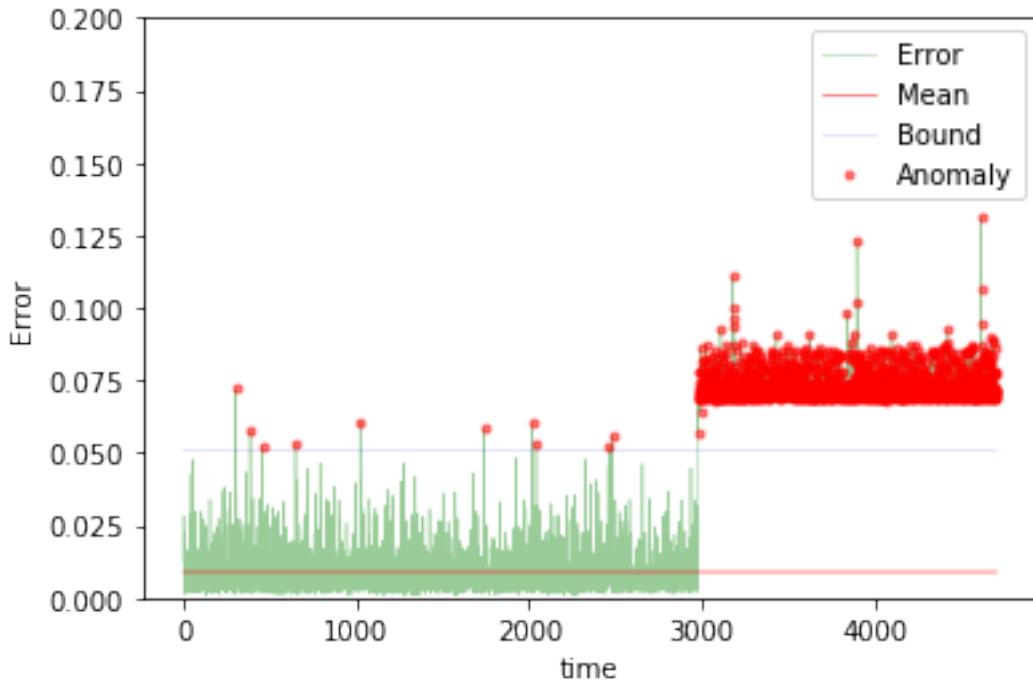
The mean error for nn3_20_anomaly_ is 0.010385211018122736 for length 4709
Testing on different app data.



The mean error for nn3_20_diff_app_ is 0.05085714442787182 for length 4709
Testing on App change synthetic data.



```
The mean error for nn3_20_app_change_ is 0.019829236966379334 for length 4709  
Testing on Net flood synthetic data.
```



```
The mean error for nn3_20_net_flood_ is 0.032723324049685636 for length 4709  
=====
```

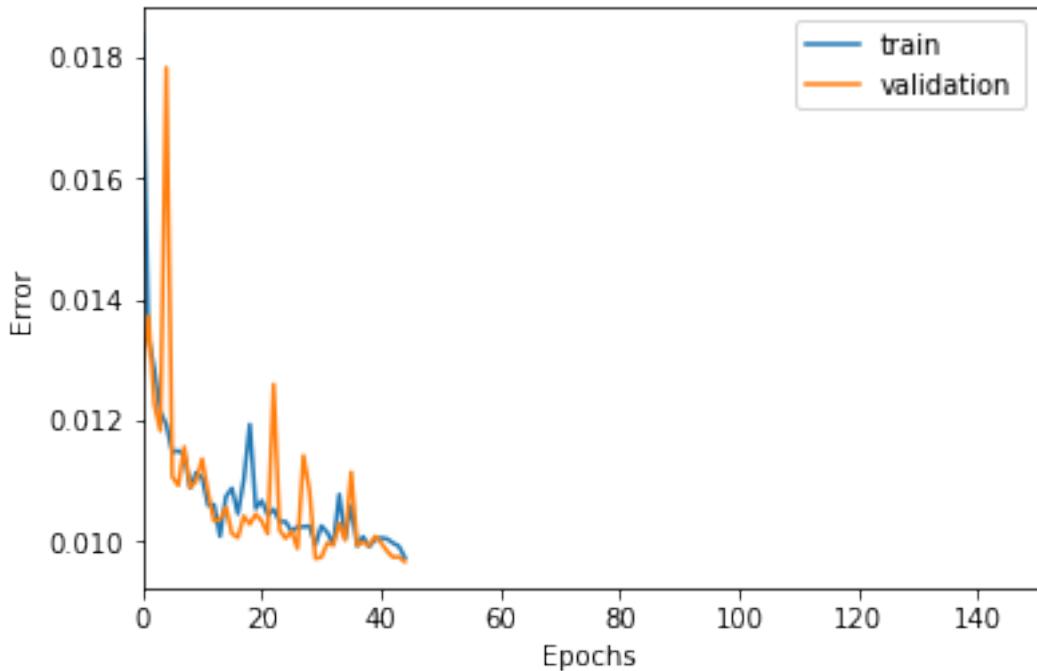
50 steps

```
In [136]: TIMESTEPS = 50  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn3_50"
```

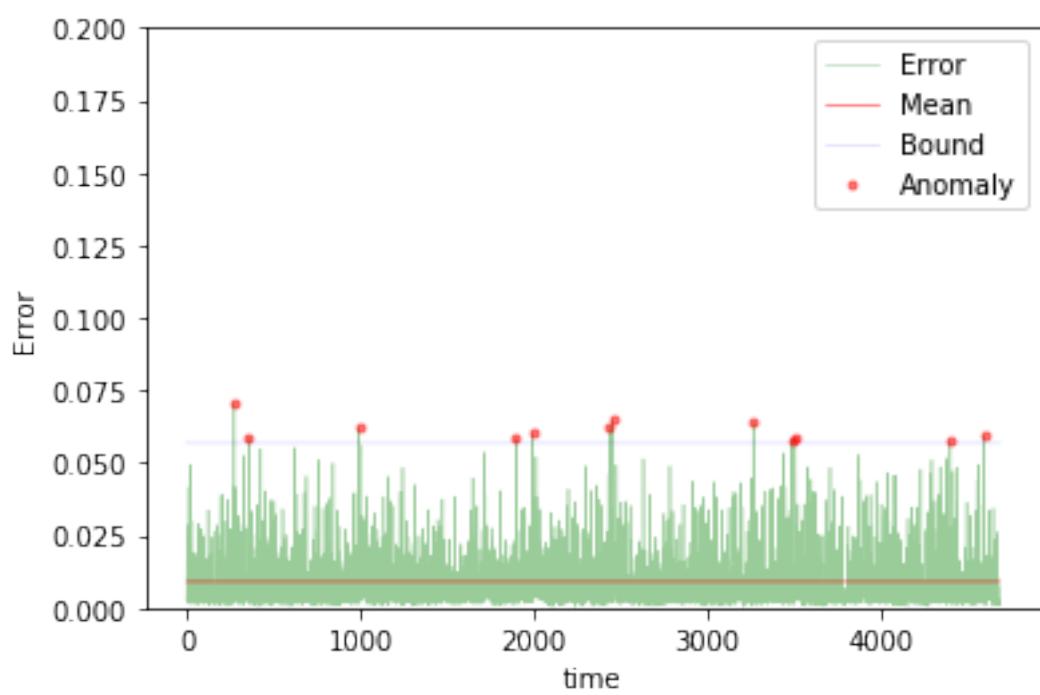
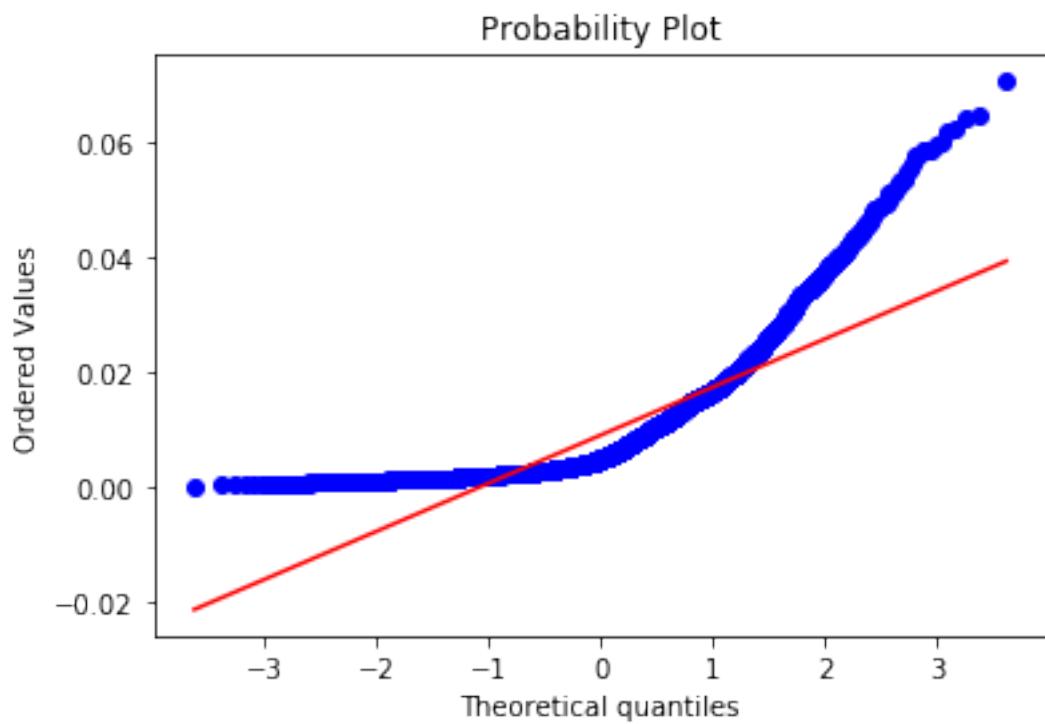
```
In [137]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(1000, activation='relu')(input_layer)  
hidden = Dense(500, activation='relu')(hidden)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [138]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

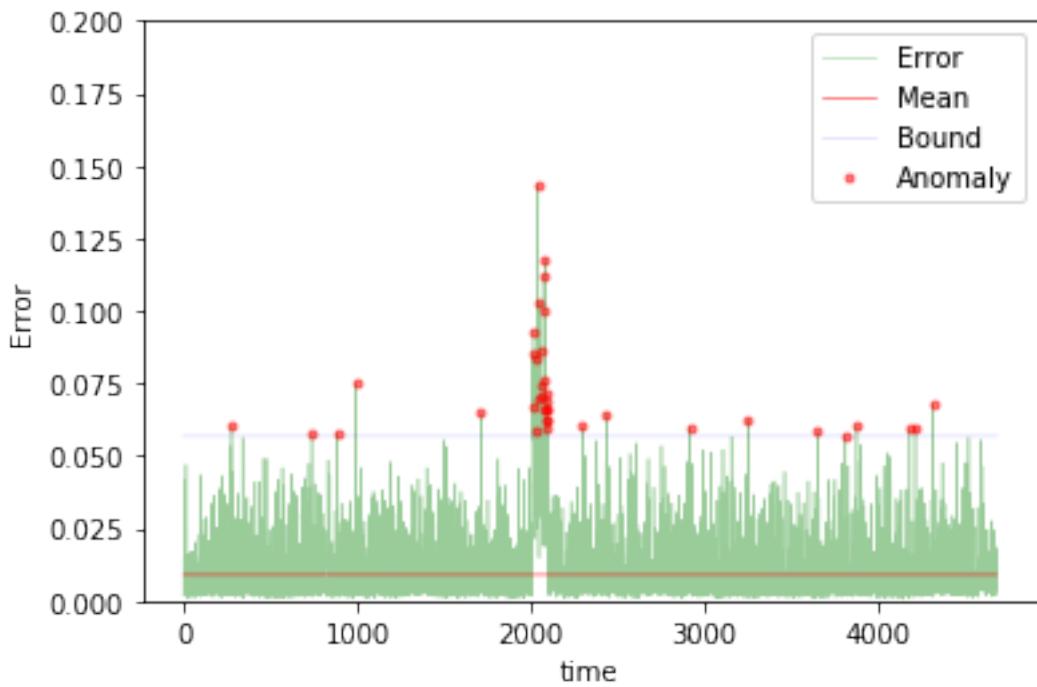
In [139]: train(model, tgen, vgen, name=name)
test(model, name=name, window=TIMESTEPS)
```



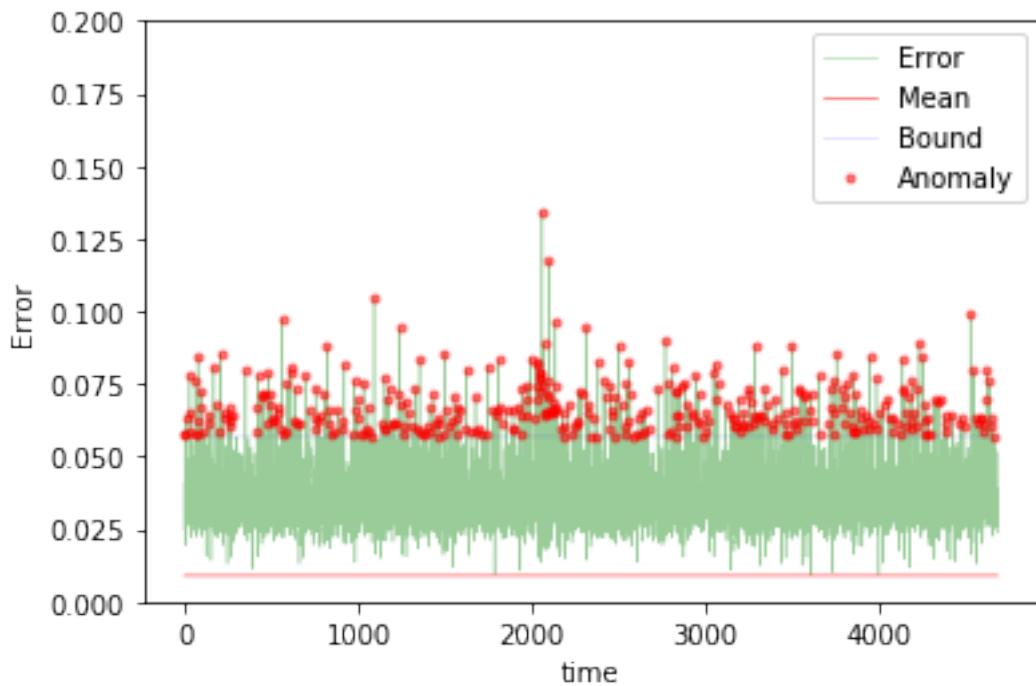
```
Training loss for final epoch is 0.009731387729989365
Validation loss for final epoch is 0.009663918917882257
----- Beginning tests for nn3_50 -----
Testing on normal data.
```



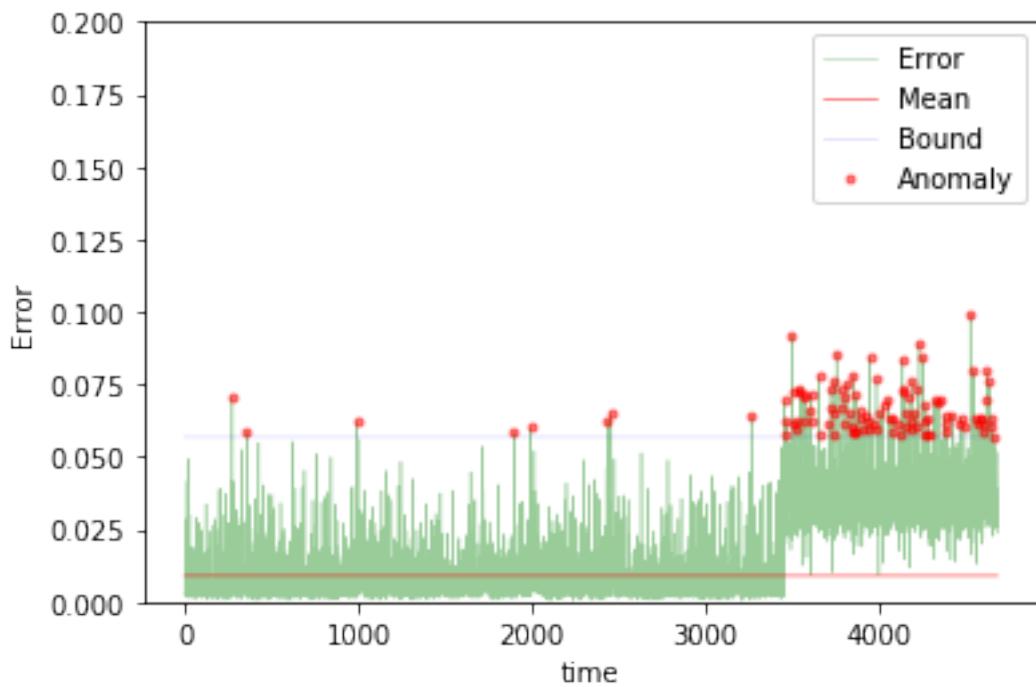
The mean error for nn3_50_normal_ is 0.009066710774178421 for length 4679
Testing on anomaly data.



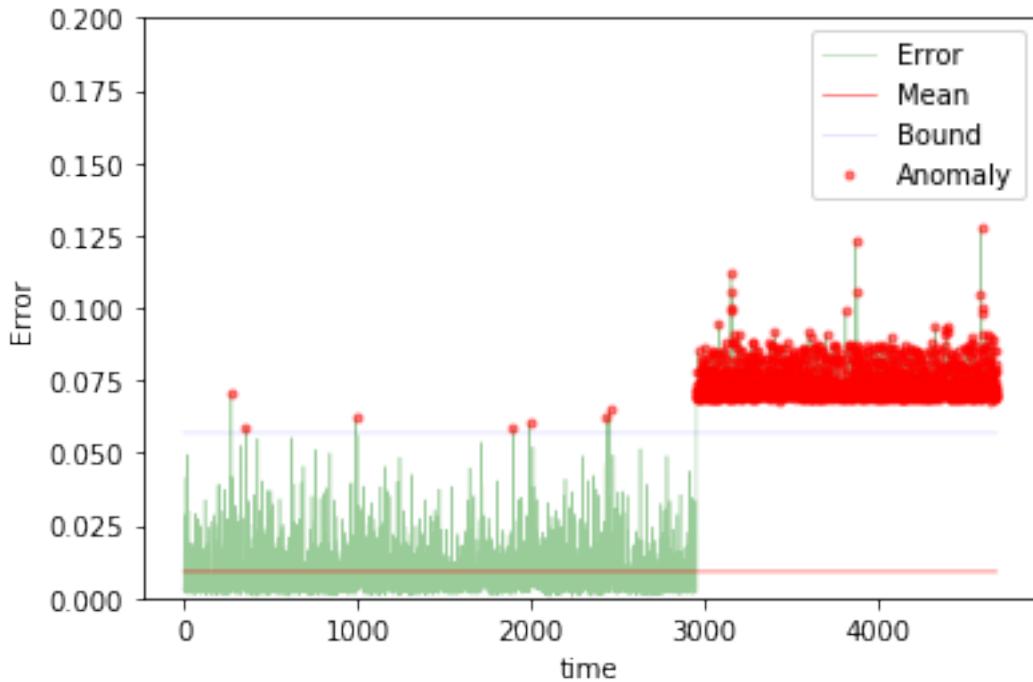
The mean error for nn3_50_anomaly_ is 0.010898157918746657 for length 4679
Testing on different app data.



The mean error for nn3_50_diff_app_ is 0.03665488972230804 for length 4679
Testing on App change synthetic data.



```
The mean error for nn3_50_app_change_ is 0.01625167965928034 for length 4679
Testing on Net flood synthetic data.
```



```
The mean error for nn3_50_net_flood_ is 0.033279751561734856 for length 4679
=====
=====
```

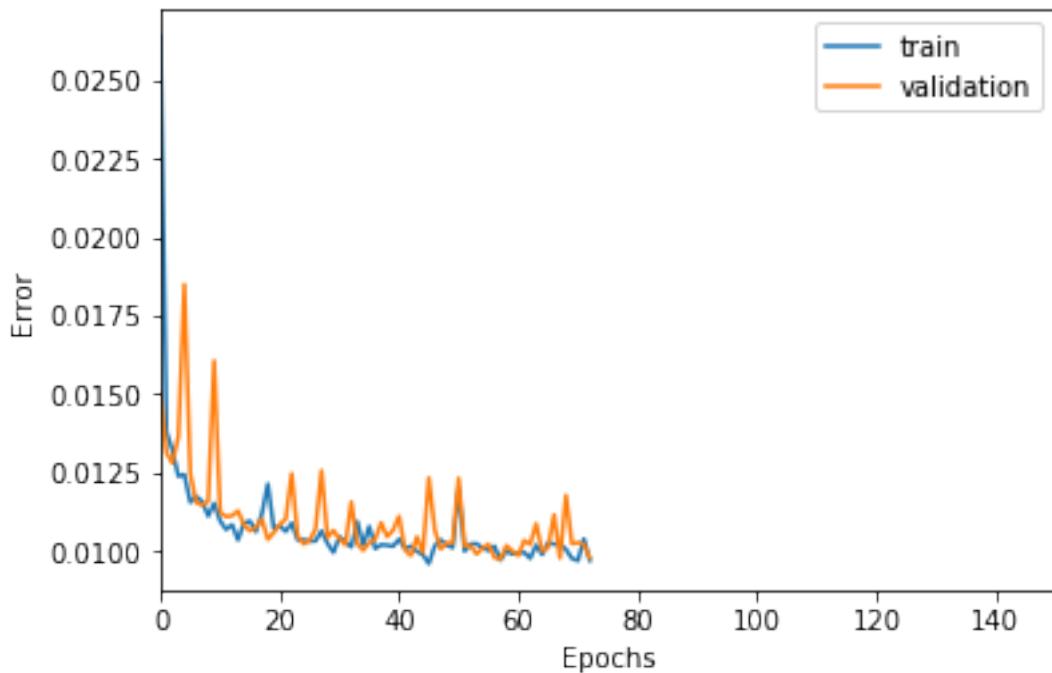
100 steps

```
In [140]: TIMESTEPS = 100
DIM = 29
tgen = flat_generator(X, TIMESTEPS)
vgen = flat_generator(val_X, TIMESTEPS)
name = "nn3_100"
```

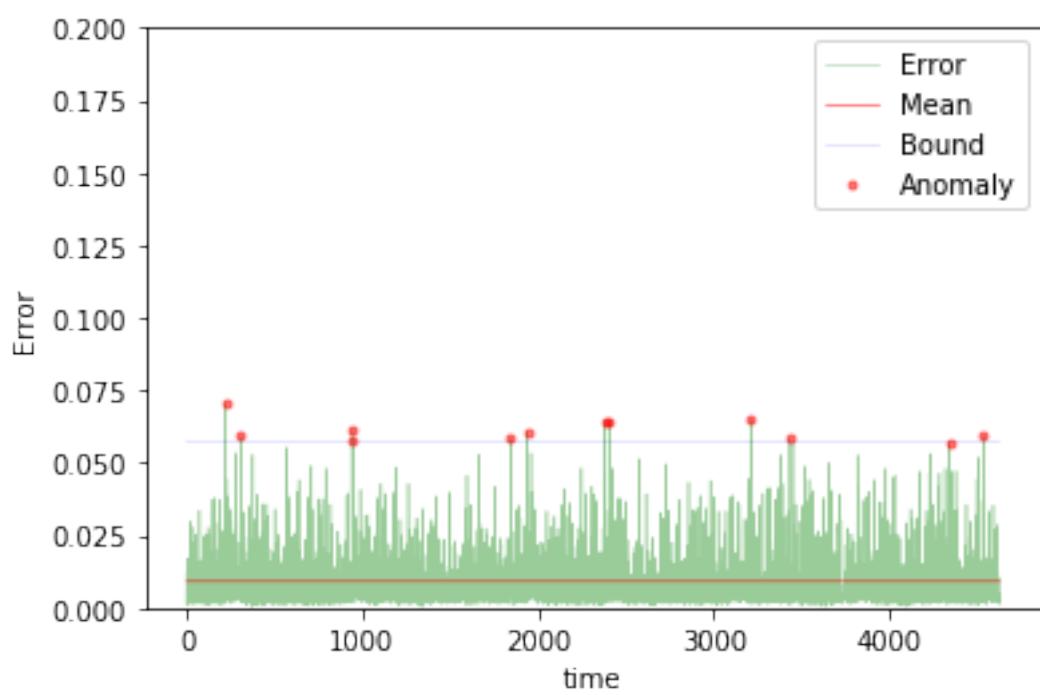
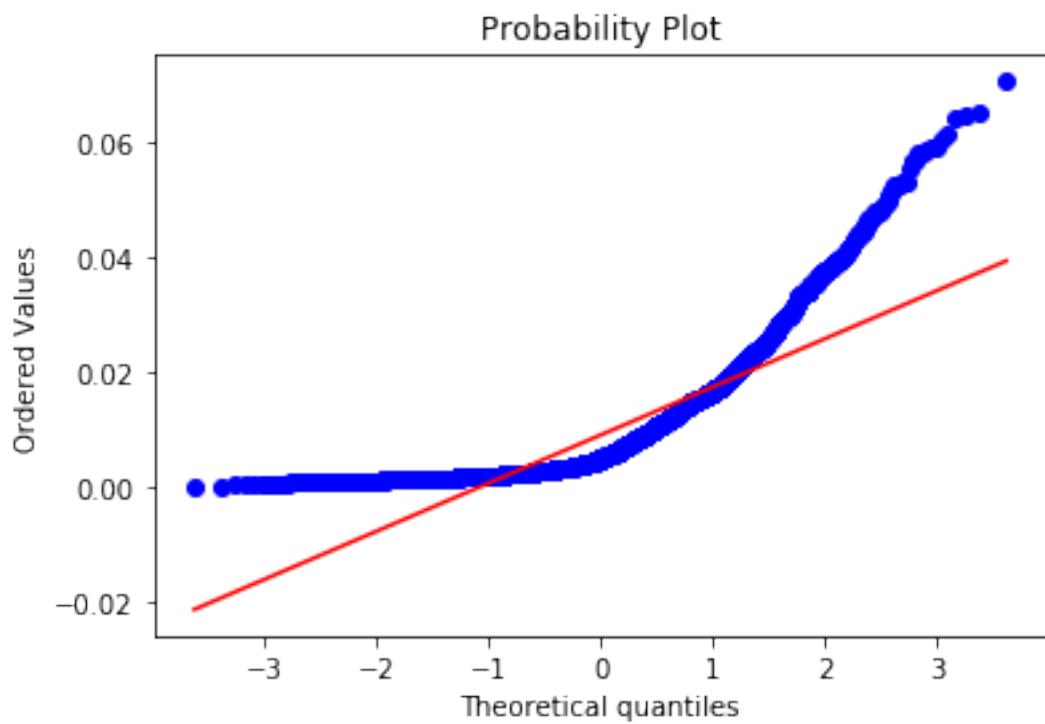
```
In [141]: input_layer = Input(shape=(TIMESTEPS*DIM,))
hidden = Dense(1000, activation='relu')(input_layer)
hidden = Dense(500, activation='relu')(hidden)
hidden = Dense(100, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [142]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

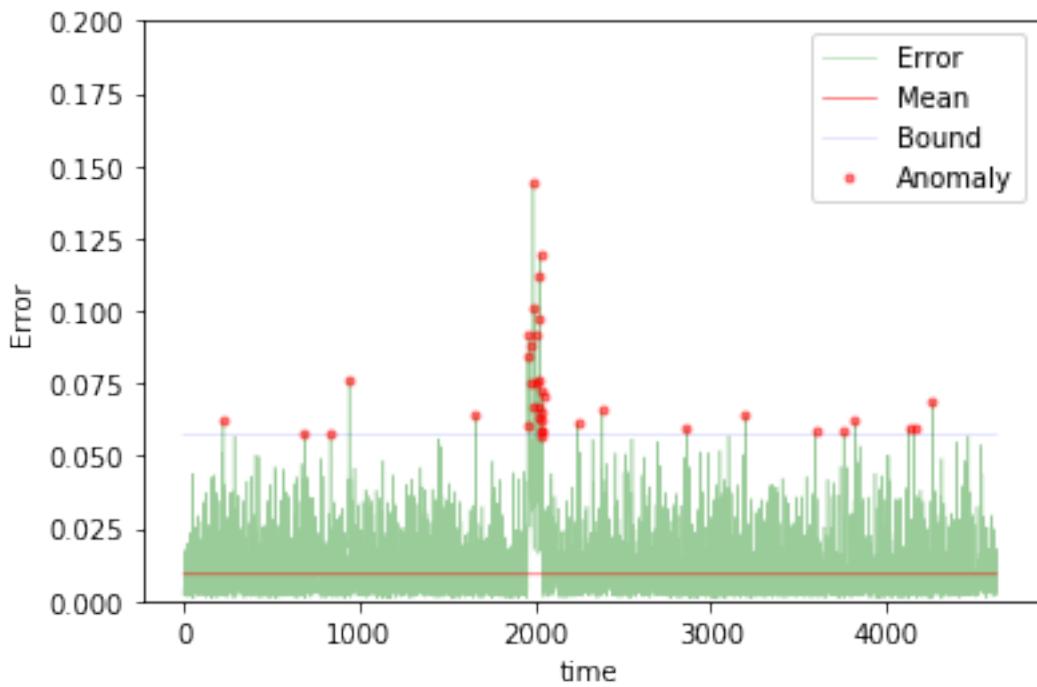
In [143]: train(model, tgen, vgen, name=name)
          test(model, name=name, window=TIMESTEPS)
```



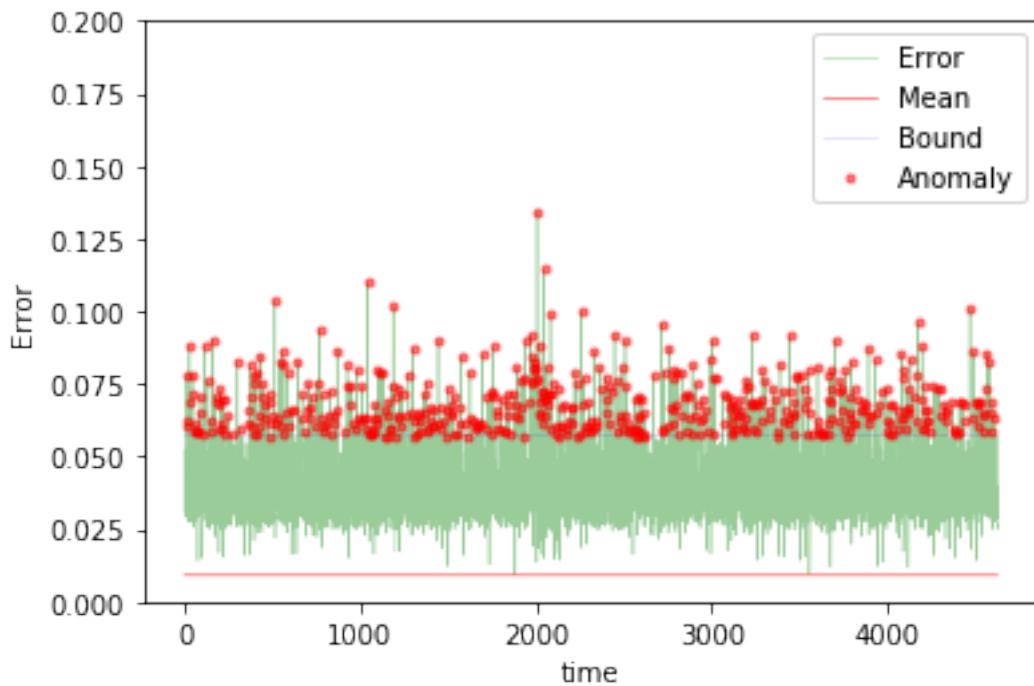
```
Training loss for final epoch is 0.009704973791376687
Validation loss for final epoch is 0.009796592995291576
----- Beginning tests for nn3_100 -----
Testing on normal data.
```



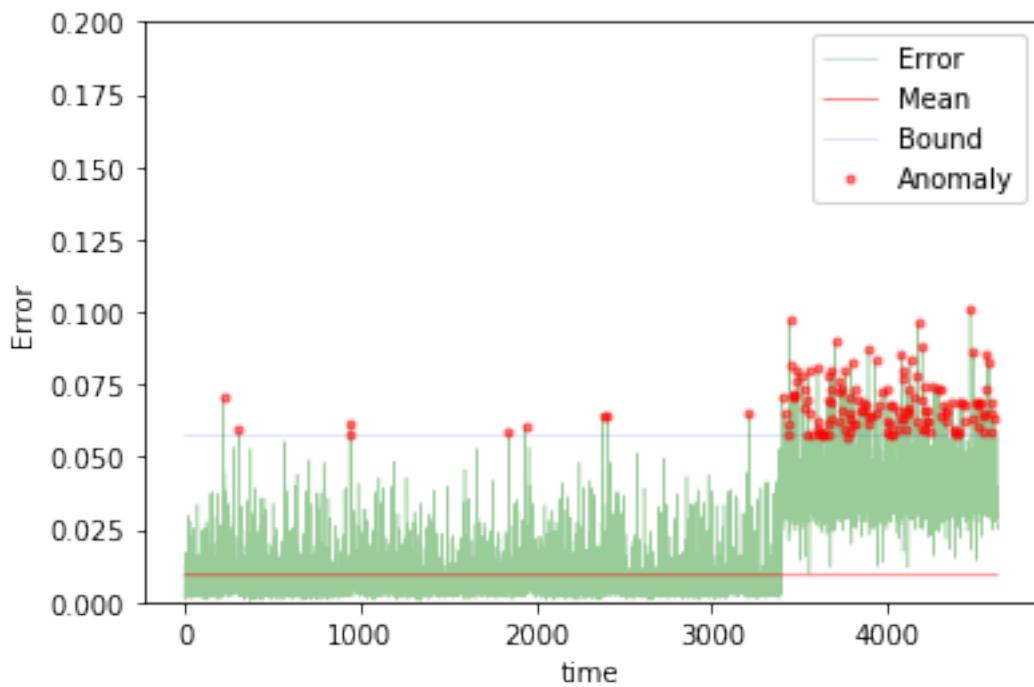
The mean error for nn3_100_normal_ is 0.009161981219870441 for length 4629
Testing on anomaly data.



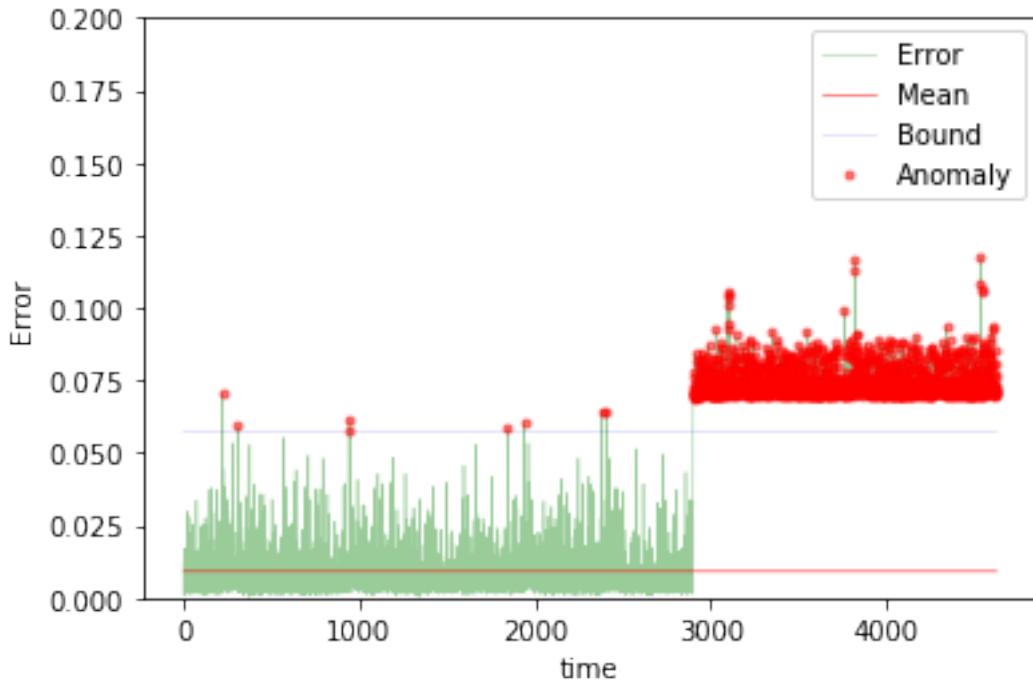
The mean error for nn3_100_anomaly_ is 0.011083110642850806 for length 4629
Testing on different app data.



The mean error for nn3_100_diff_app_ is 0.04040772048764713 for length 4629
Testing on App change synthetic data.



```
The mean error for nn3_100_app_change_ is 0.01740037367261588 for length 4629  
Testing on Net flood synthetic data.
```



```
The mean error for nn3_100_net_flood_ is 0.03361851544487802 for length 4629  
=====
```

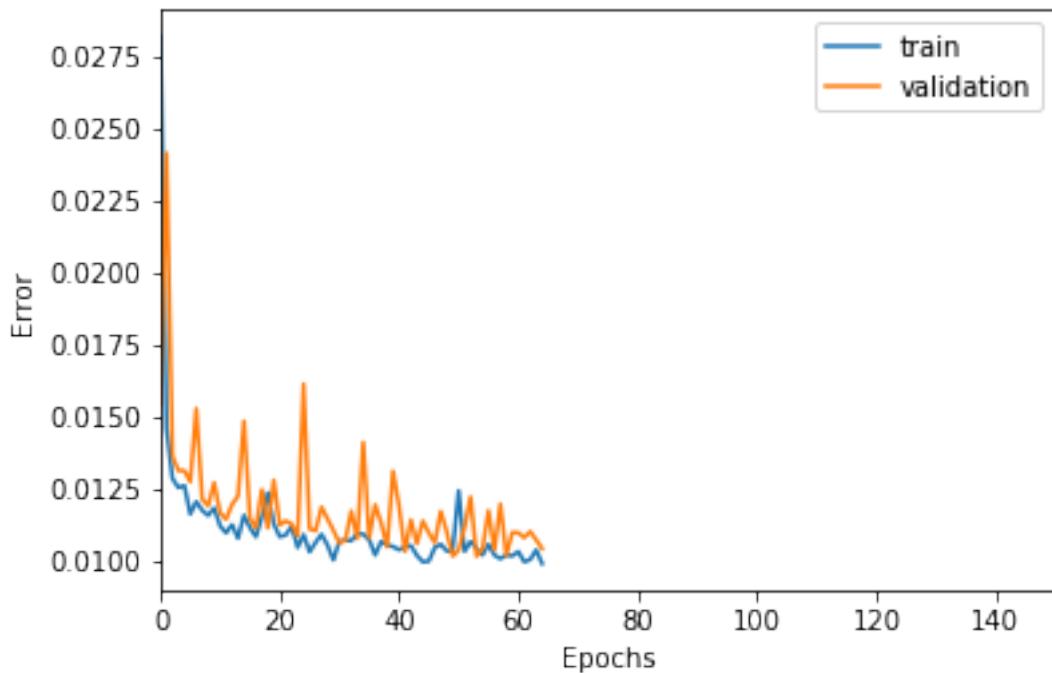
200 steps

```
In [144]: TIMESTEPS = 200  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS)  
vgen = flat_generator(val_X, TIMESTEPS)  
name = "nn3_200"
```

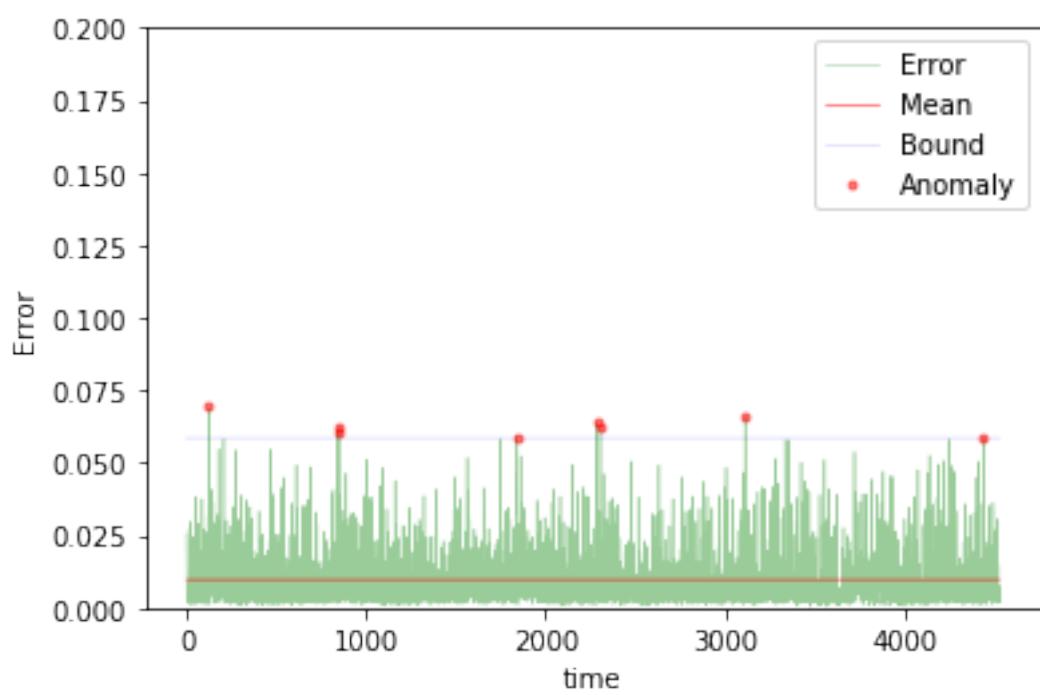
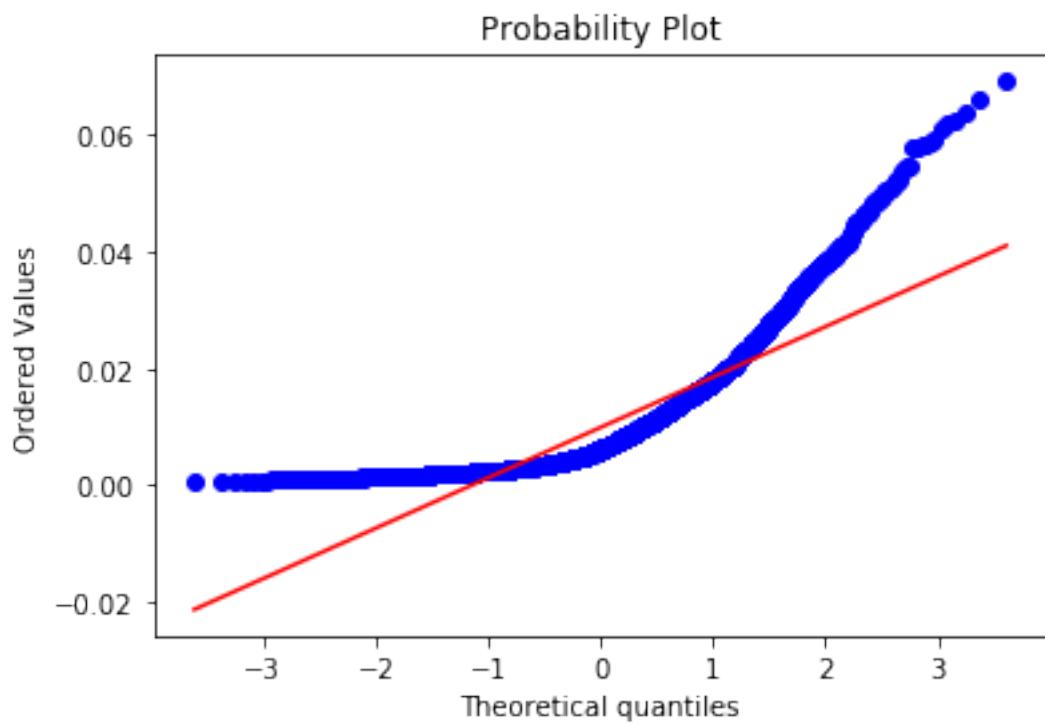
```
In [145]: input_layer = Input(shape=(TIMESTEPS*DIM,))  
hidden = Dense(1000, activation='relu')(input_layer)  
hidden = Dense(500, activation='relu')(hidden)  
hidden = Dense(100, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [146]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

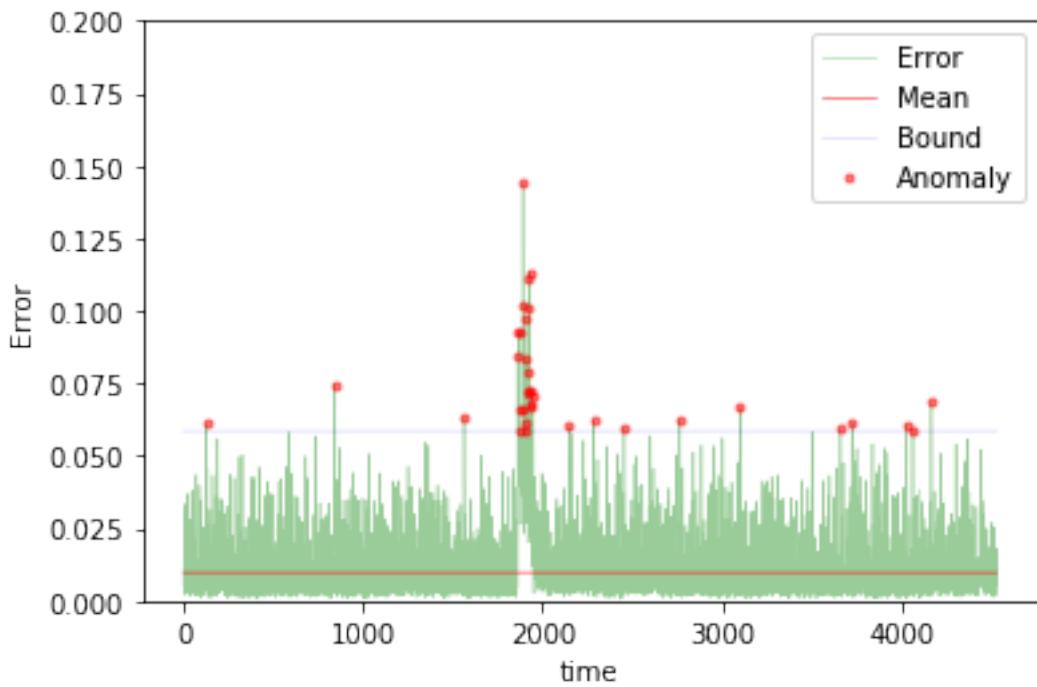
In [147]: train(model, tgen, vgen, name=name)
          test(model, name=name, window=TIMESTEPS)
```



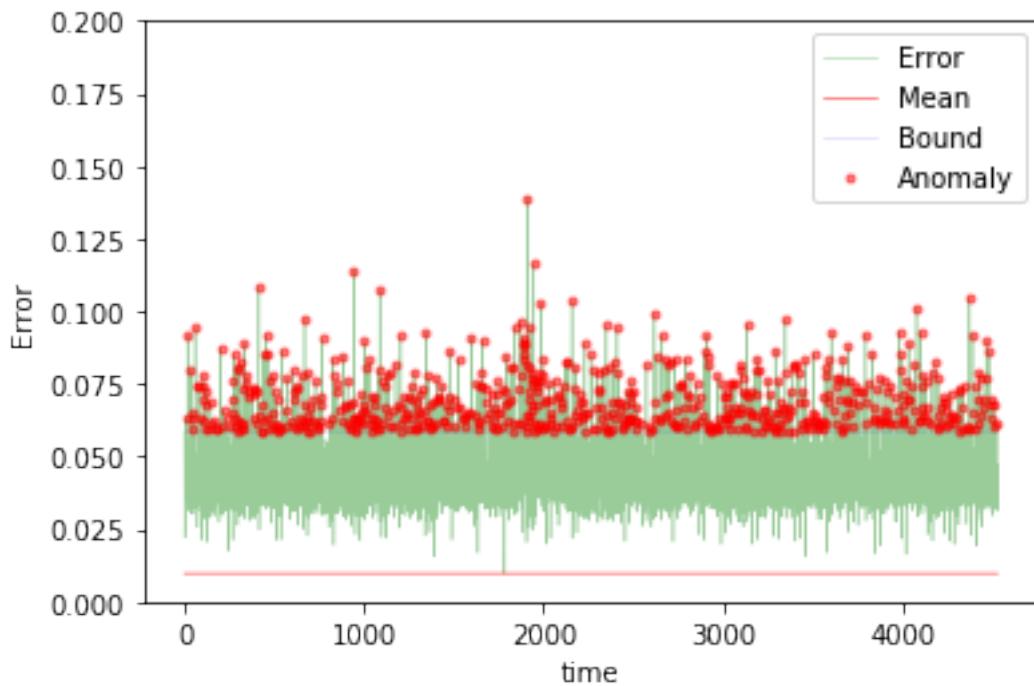
```
Training loss for final epoch is 0.009892589452210814
Validation loss for final epoch is 0.010416171515011228
----- Beginning tests for nn3_200 -----
Testing on normal data.
```



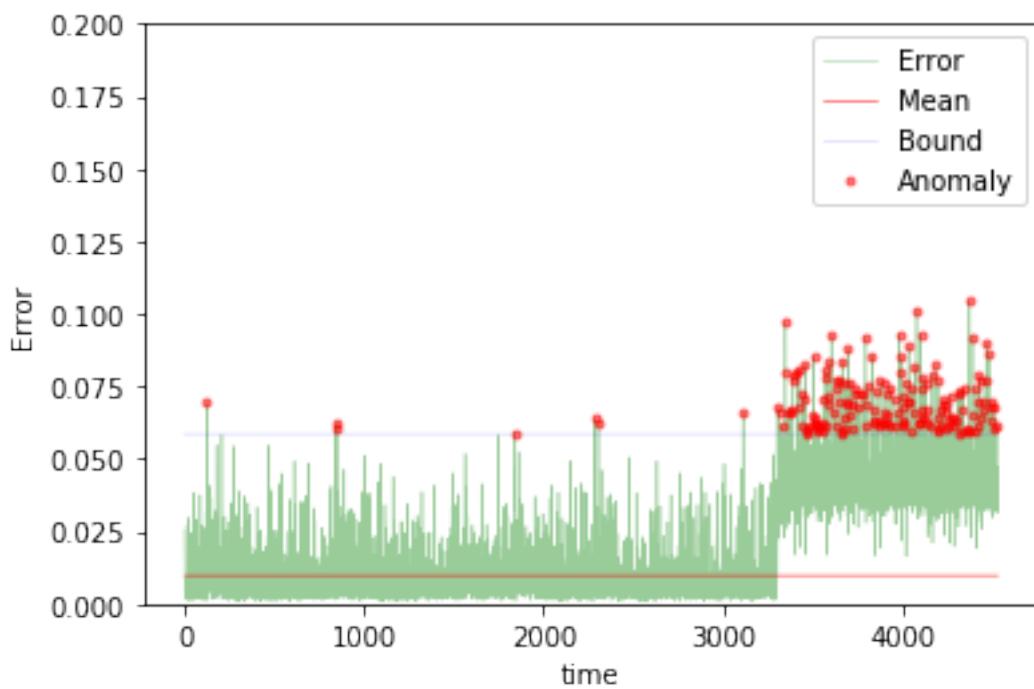
The mean error for nn3_200_normal_ is 0.009838348950549896 for length 4529
Testing on anomaly data.



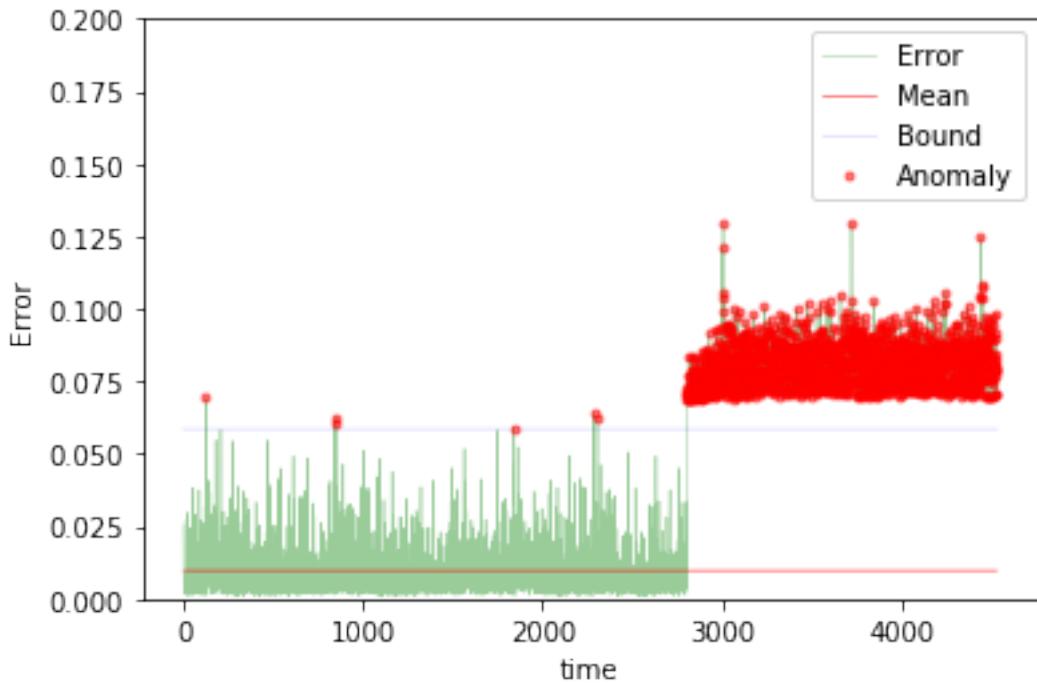
The mean error for nn3_200_anomaly_ is 0.012125486243249729 for length 4529
Testing on different app data.



The mean error for nn3_200_diff_app_ is 0.044716475911693115 for length 4529
Testing on App change synthetic data.



```
The mean error for nn3_200_app_change_ is 0.019173034538692634 for length 4529
Testing on Net flood synthetic data.
```



```
The mean error for nn3_200_net_flood_ is 0.03685002912805364 for length 4529
=====
=====
```

2.1.5 RNN with 1 GRU layers

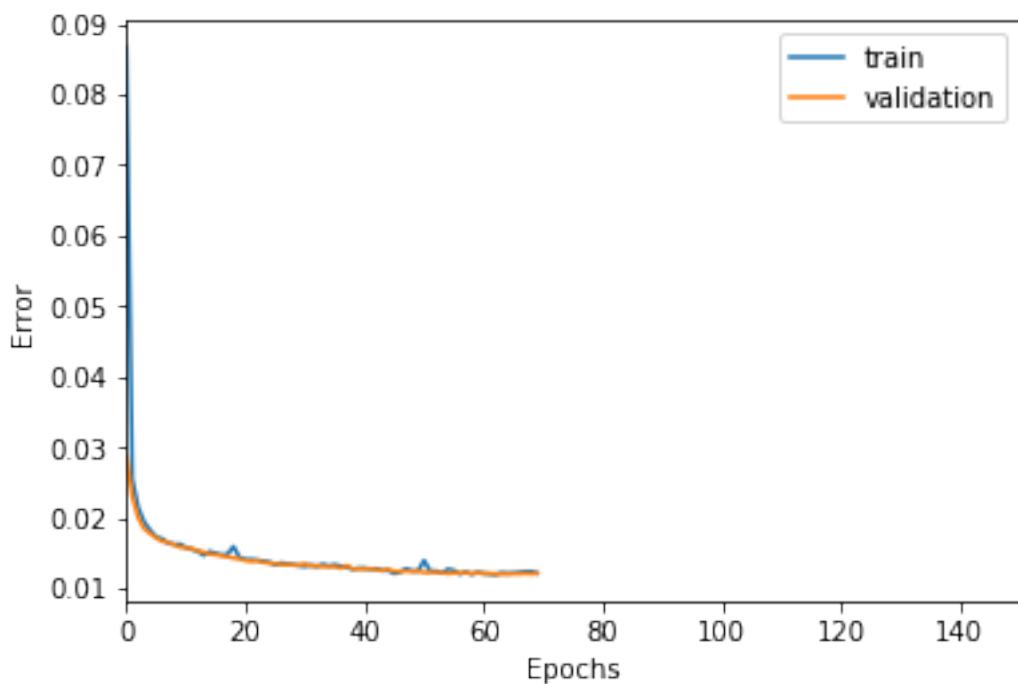
2 steps

```
In [148]: TIMESTEPS = 2
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS,0)
          vgen = flat_generator(val_X, TIMESTEPS,0)
          name = "gru1_2"
```

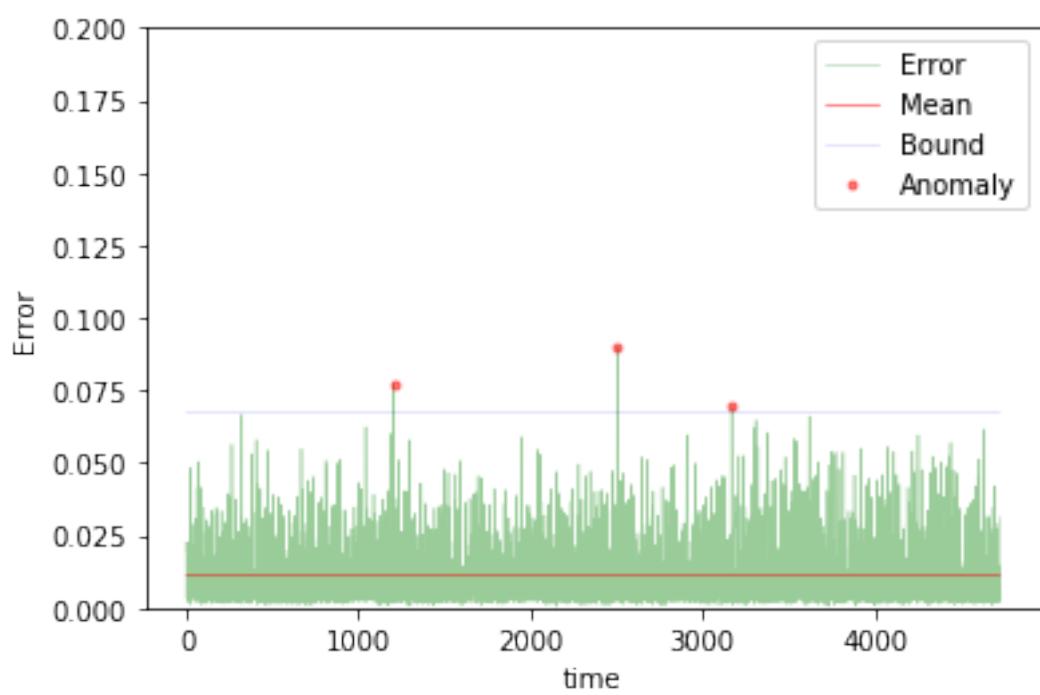
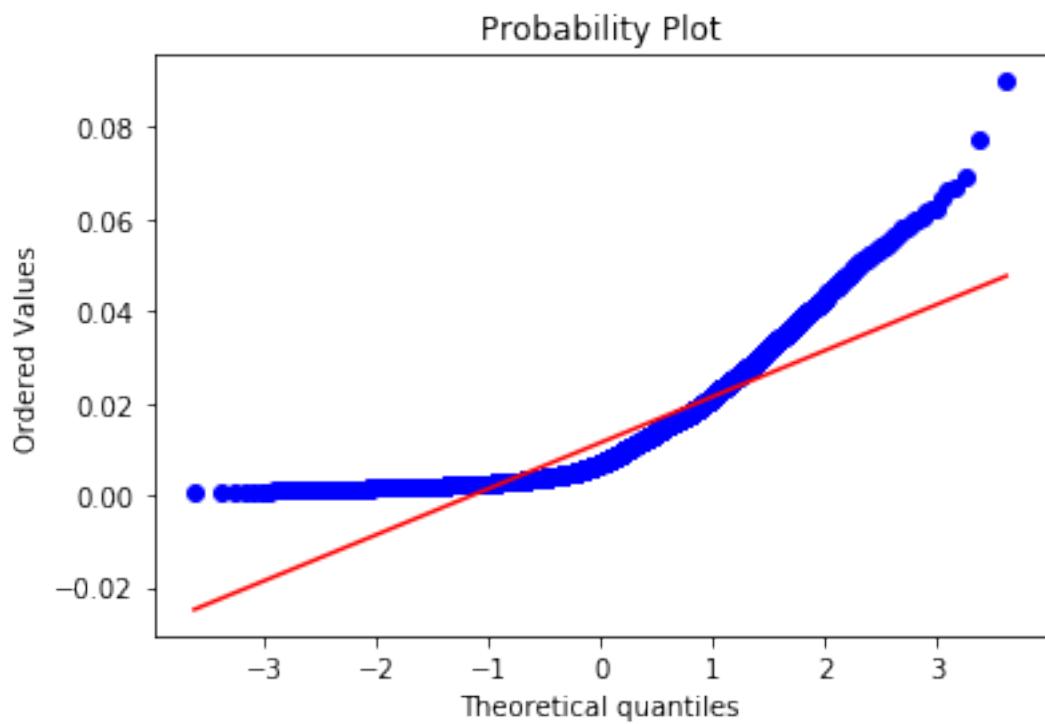
```
In [149]: input_layer = Input(shape=(TIMESTEPS,DIM))
          hidden = GRU(10, activation='relu')(input_layer)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [150]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

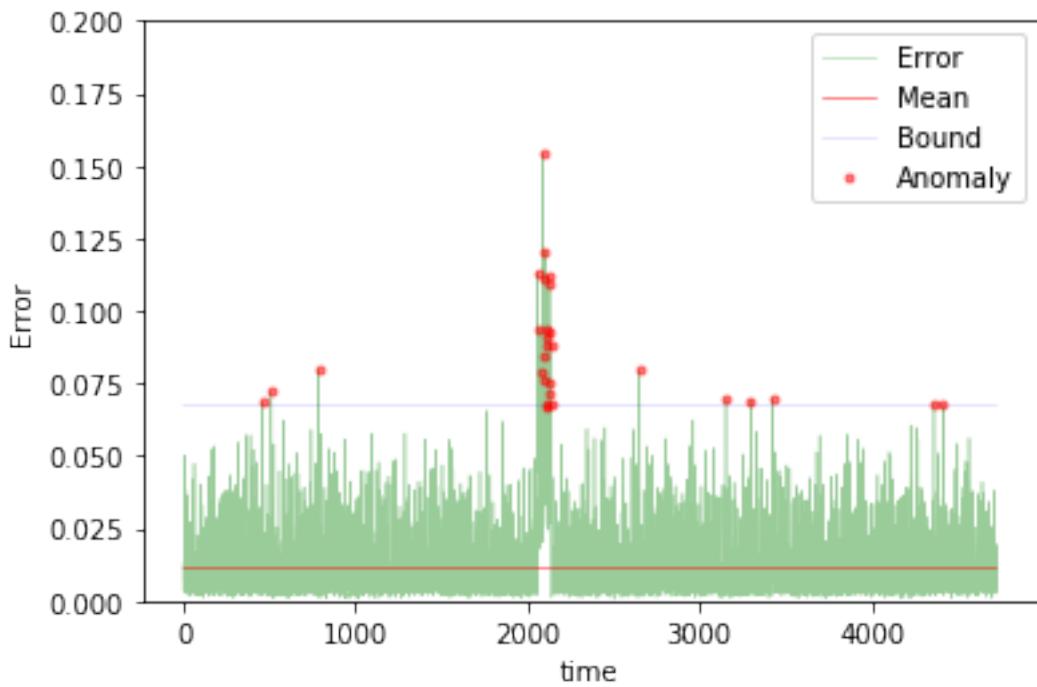
In [151]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



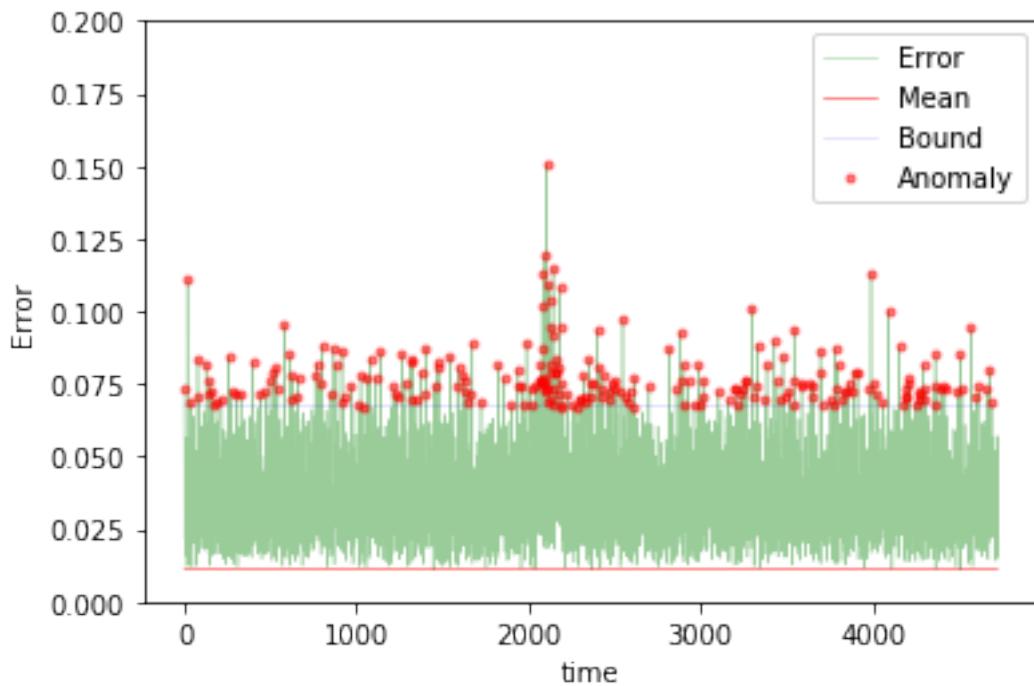
```
Training loss for final epoch is 0.012161327028064989
Validation loss for final epoch is 0.012145240509416908
----- Beginning tests for gru1_2 -----
Testing on normal data.
```



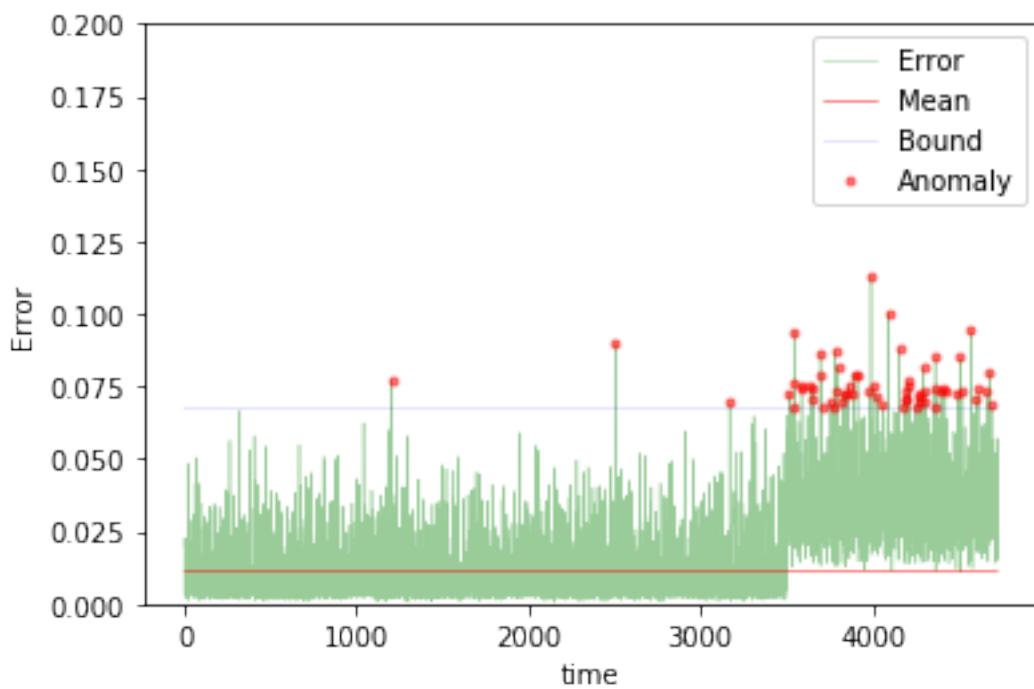
The mean error for gru1_2_normal_ is 0.011387075376896295 for length 4727
Testing on anomaly data.



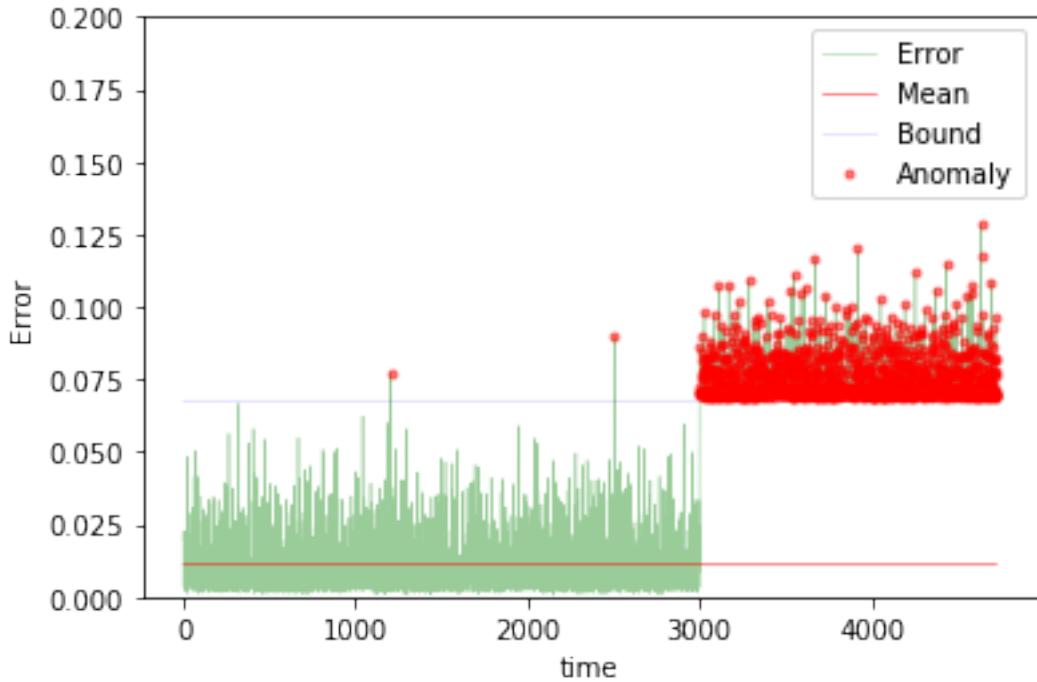
The mean error for gru1_2_anomaly_ is 0.013163933656669414 for length 4727
Testing on different app data.



The mean error for gru1_2_diff_app_ is 0.036216279168074524 for length 4727
Testing on App change synthetic data.



```
The mean error for gru1_2_app_change_ is 0.017687933724965795 for length 4727  
Testing on Net flood synthetic data.
```



```
The mean error for gru1_2_net_flood_ is 0.034572378779422264 for length 4727  
=====
```

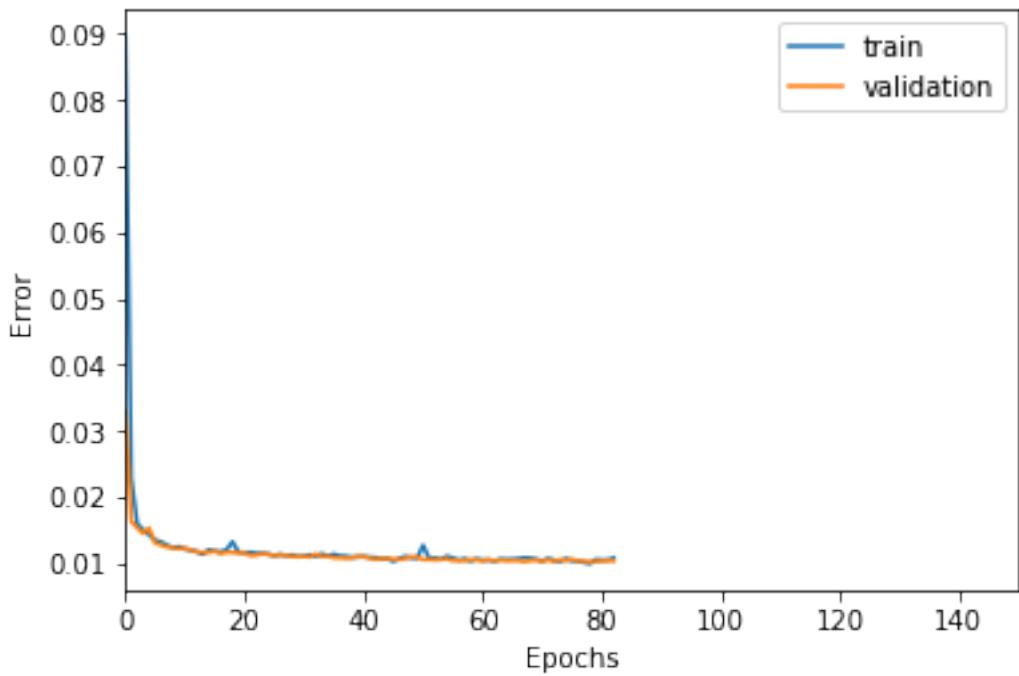
5 steps

```
In [152]: TIMESTEPS = 5  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS, 0)  
vgen = flat_generator(val_X, TIMESTEPS, 0)  
name = "gru1_5"
```

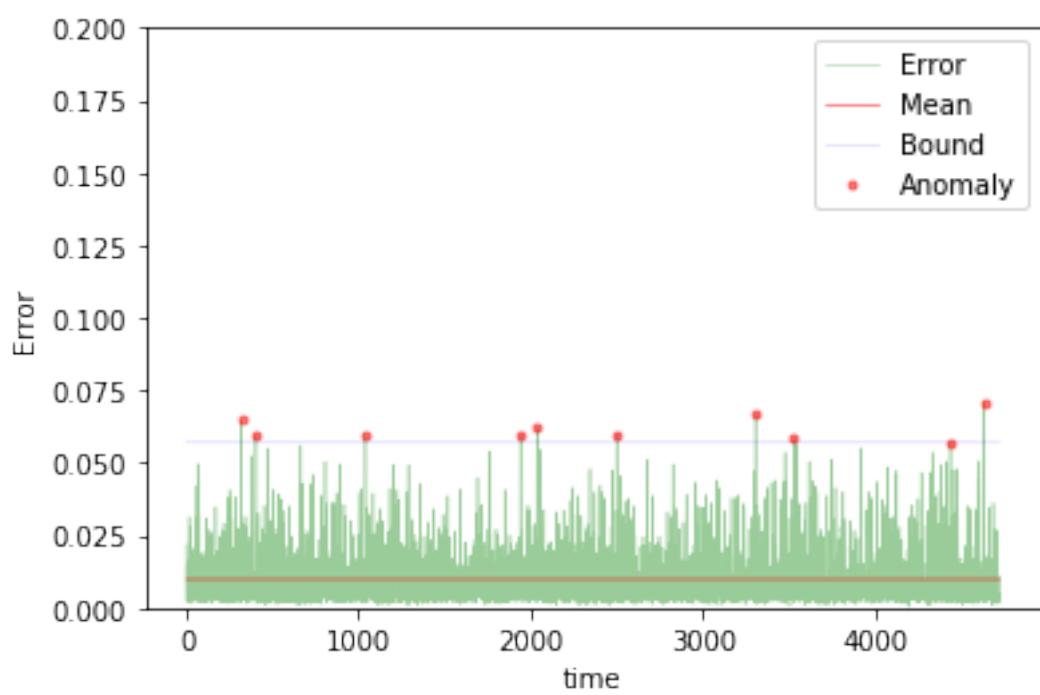
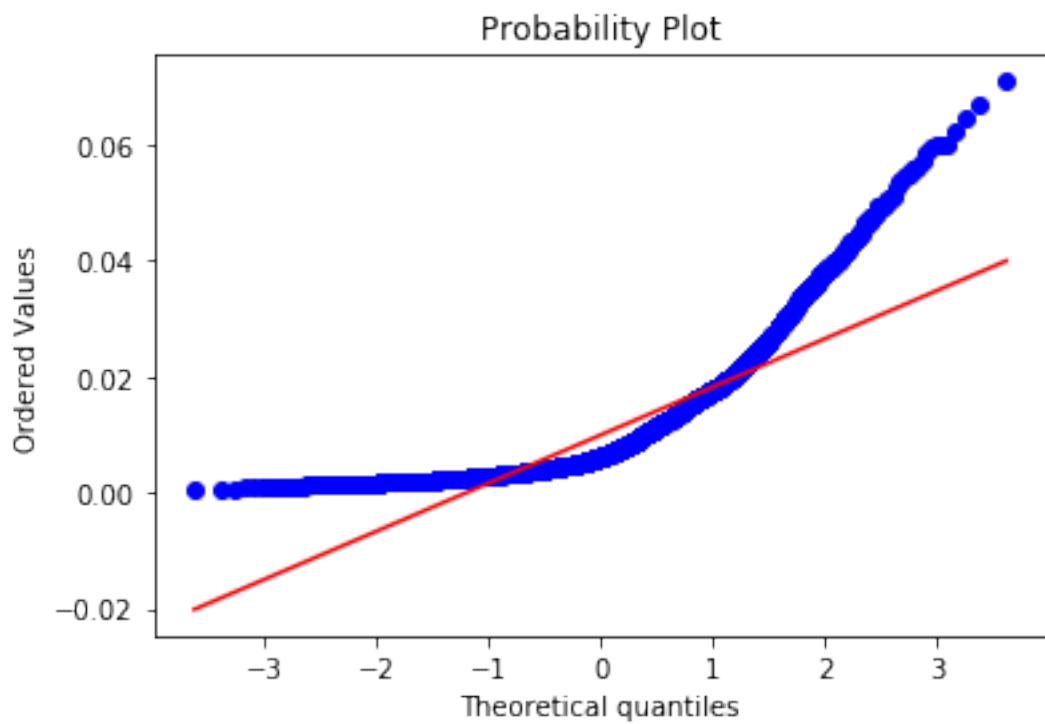
```
In [153]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [154]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

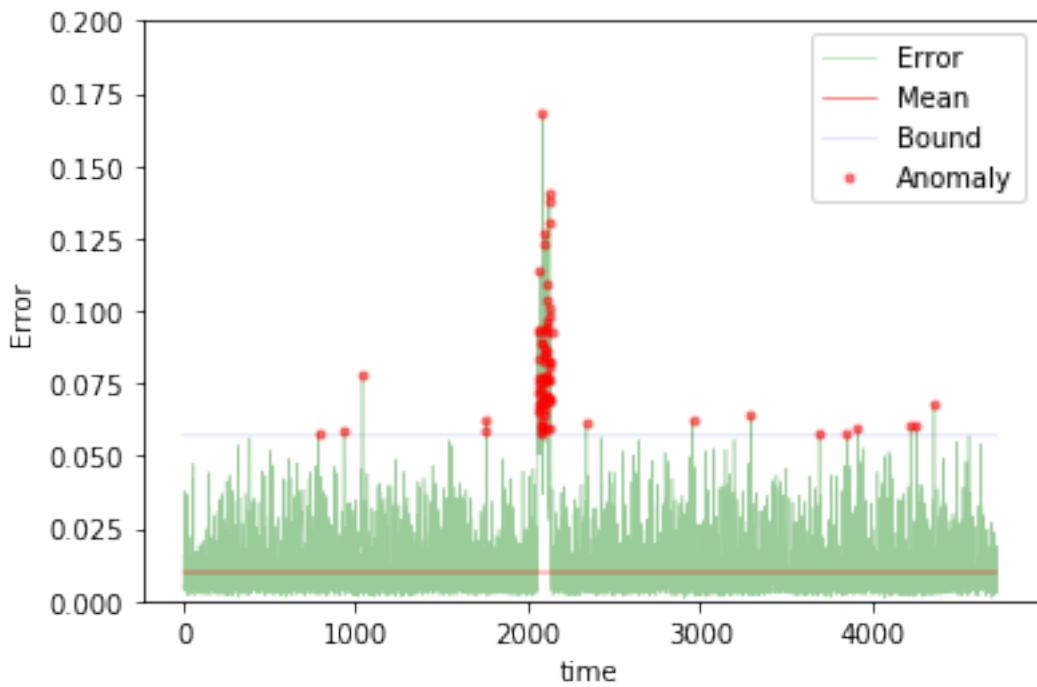
In [155]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



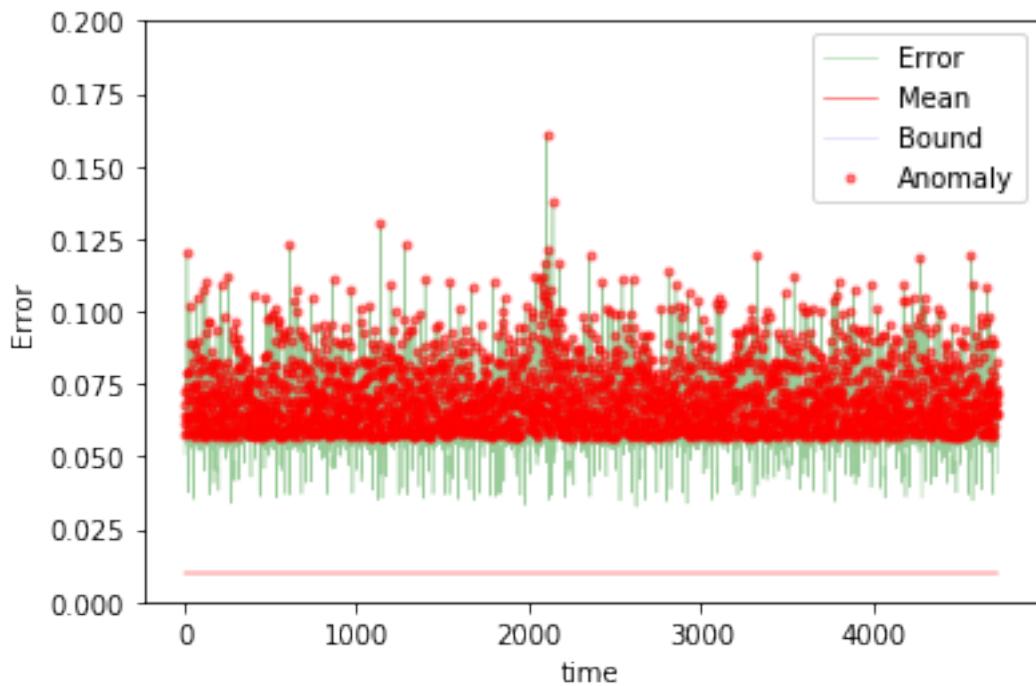
```
Training loss for final epoch is 0.010798511168221012
Validation loss for final epoch is 0.01034308428701479
----- Beginning tests for gru1_5 -----
Testing on normal data.
```



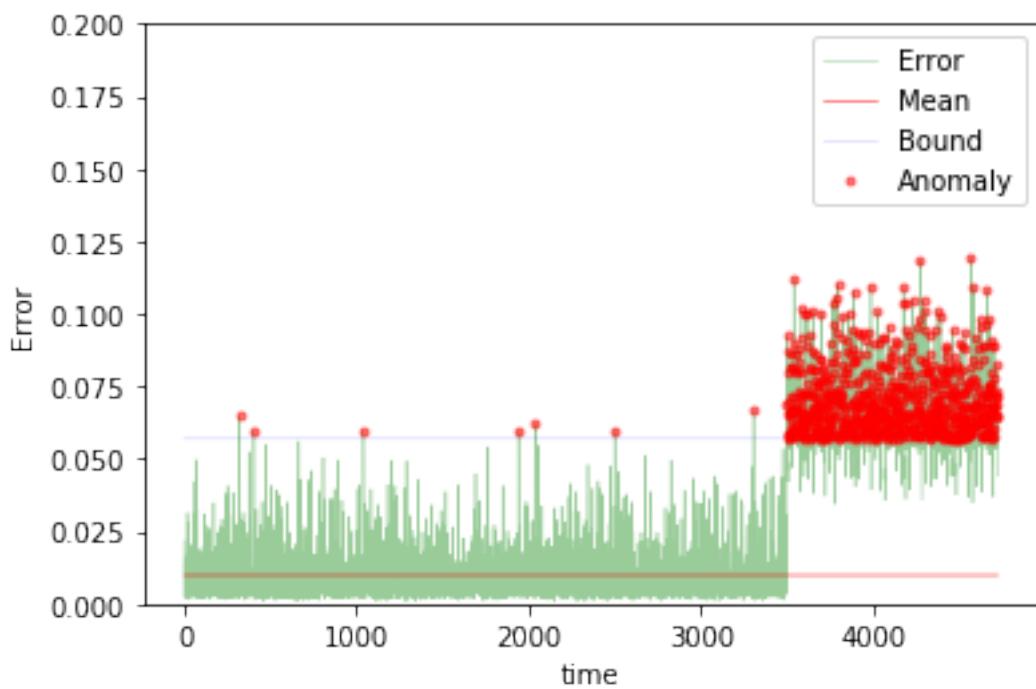
The mean error for gru1_5_normal_ is 0.009928258158379805 for length 4724
Testing on anomaly data.



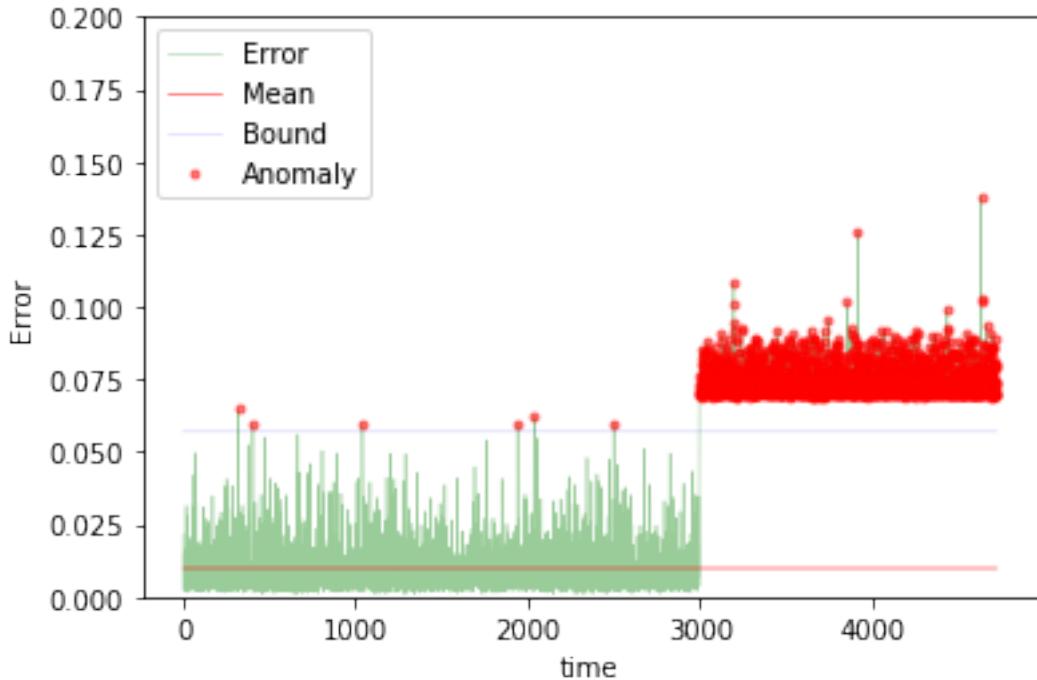
The mean error for gru1_5_anomaly_ is 0.011890586656430495 for length 4724
Testing on different app data.



The mean error for gru1_5_diff_app_ is 0.0640609905694641 for length 4724
Testing on App change synthetic data.



```
The mean error for gru1_5_app_change_ is 0.023859676873422206 for length 4724  
Testing on Net flood synthetic data.
```



```
The mean error for gru1_5_net_flood_ is 0.033965973539462485 for length 4724  
=====
```

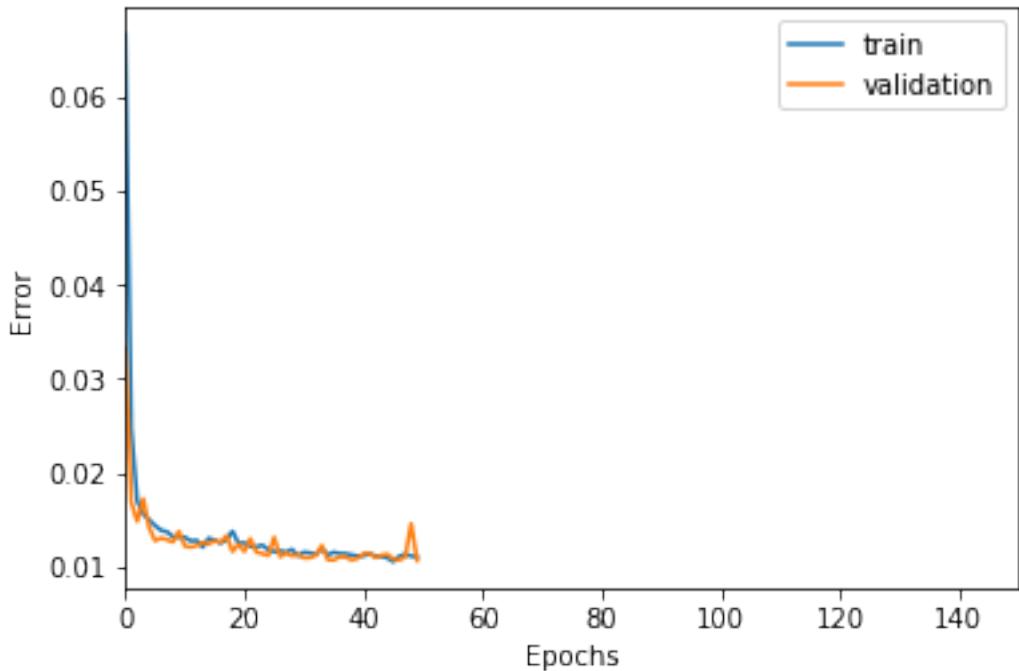
10 steps

```
In [156]: TIMESTEPS = 10  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS, 0)  
vgen = flat_generator(val_X, TIMESTEPS, 0)  
name = "gru1_10"
```

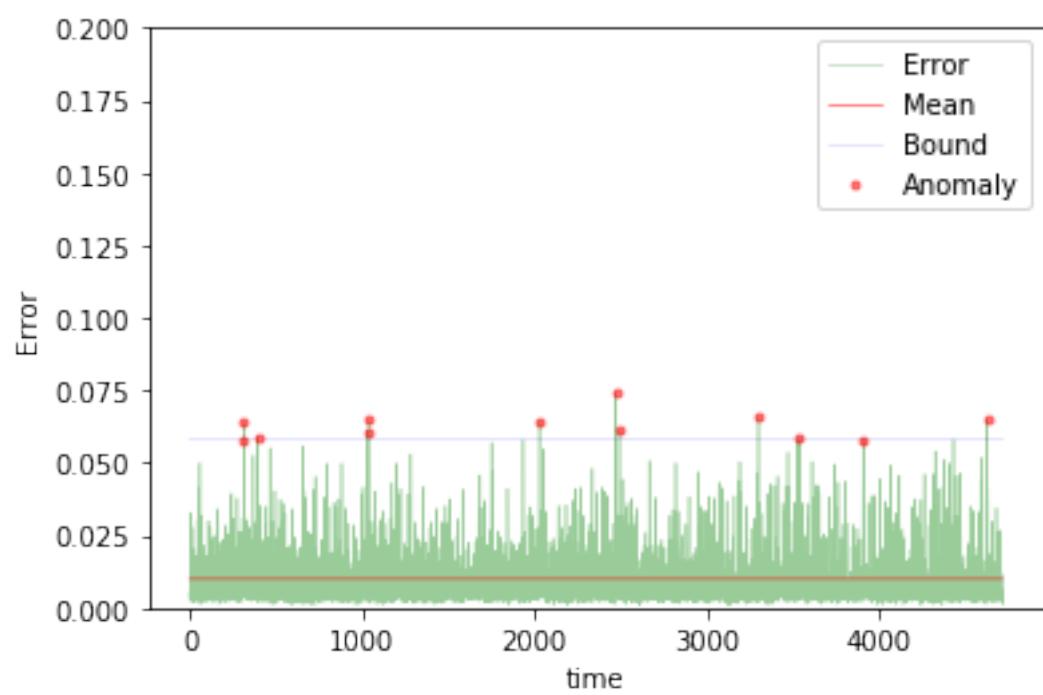
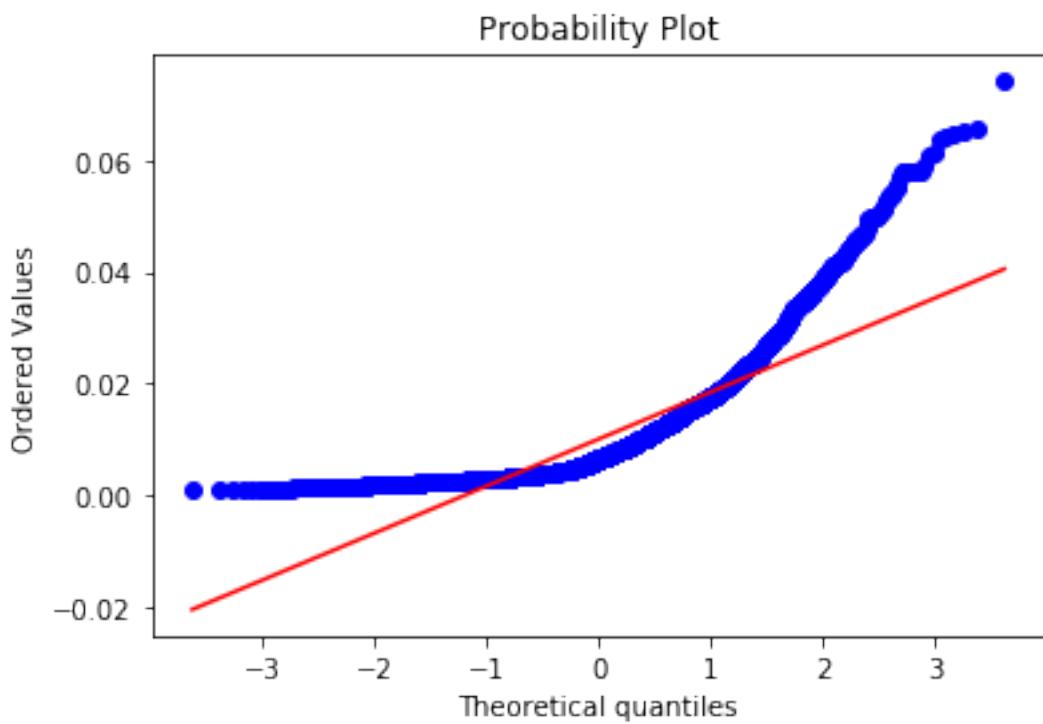
```
In [157]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [158]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

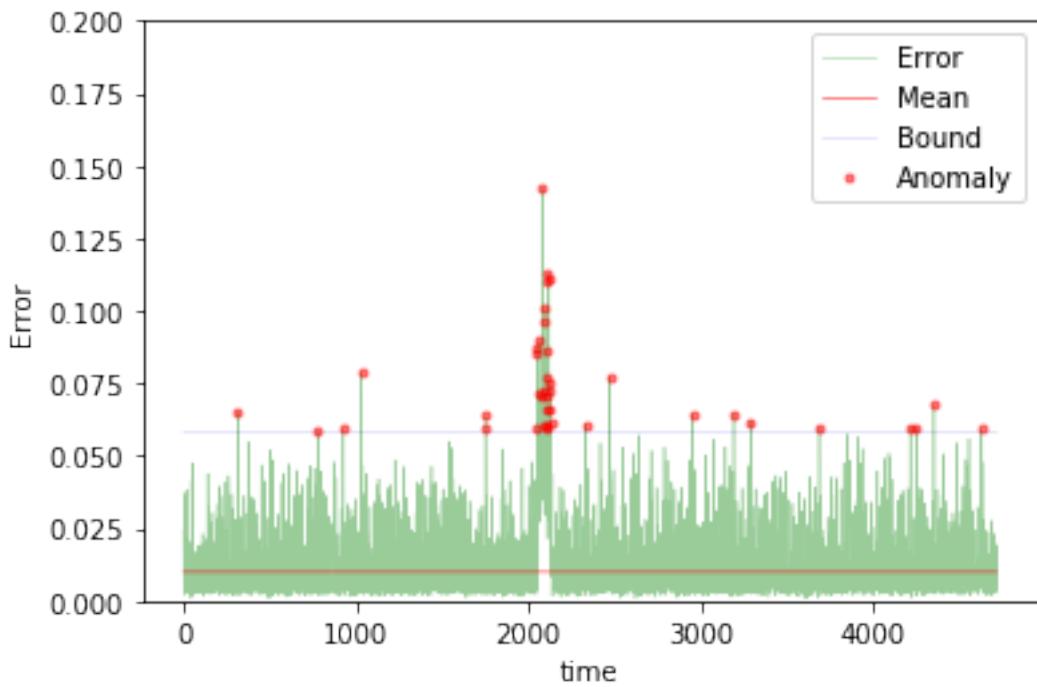
In [159]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



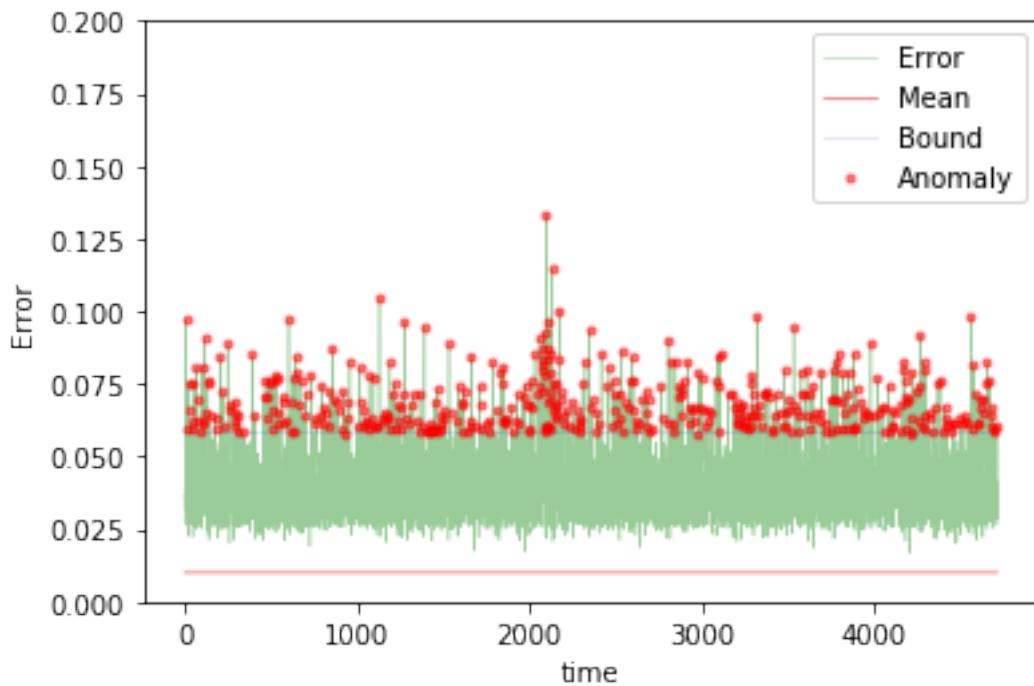
```
Training loss for final epoch is 0.011058549219742418
Validation loss for final epoch is 0.010694127397146076
----- Beginning tests for gru1_10 -----
Testing on normal data.
```



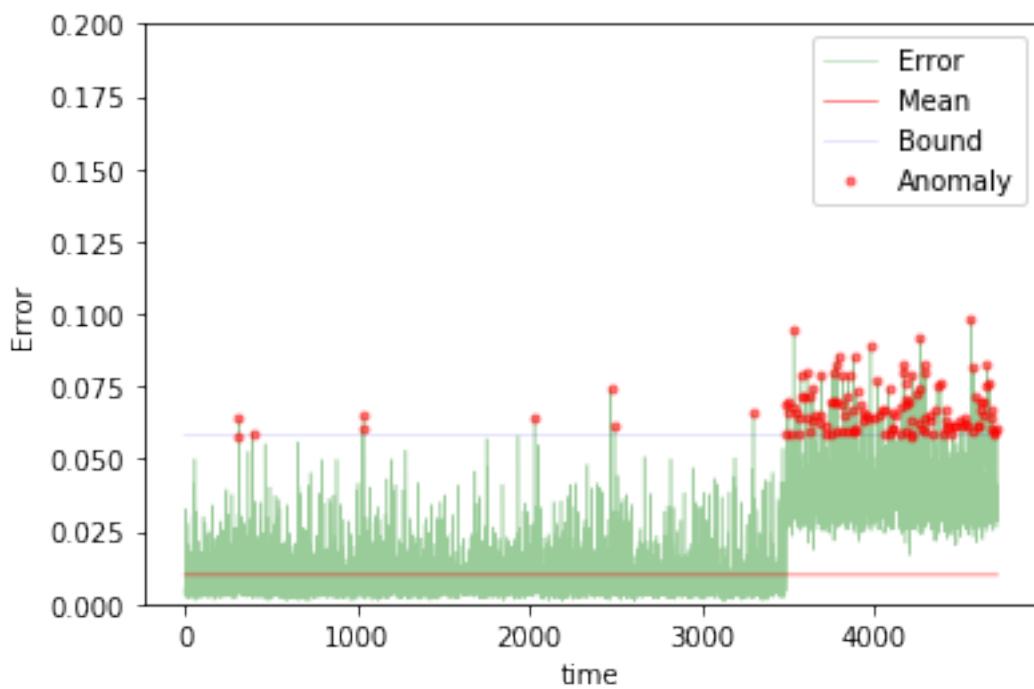
The mean error for gru1_10_normal_ is 0.010023852634204607 for length 4719
Testing on anomaly data.



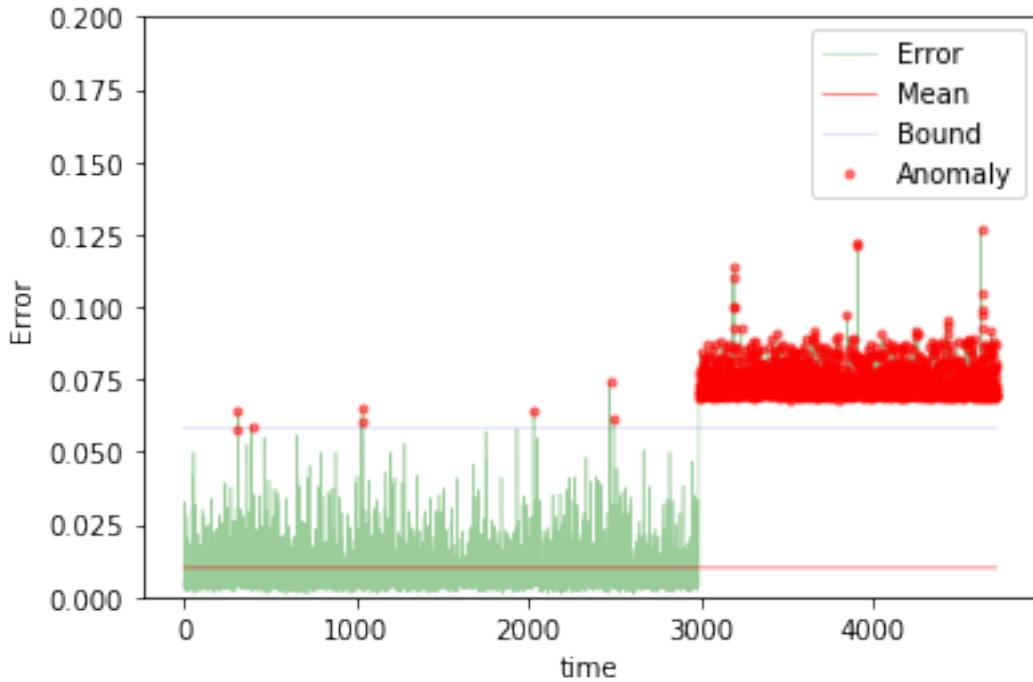
The mean error for gru1_10_anomaly_ is 0.011916608485632262 for length 4719
Testing on different app data.



The mean error for gru1_10_diff_app_ is 0.03949417060614817 for length 4719
Testing on App change synthetic data.



```
The mean error for gru1_10_app_change_ is 0.01758672896164563 for length 4719  
Testing on Net flood synthetic data.
```



```
The mean error for gru1_10_net_flood_ is 0.033687993532819156 for length 4719  
=====
```

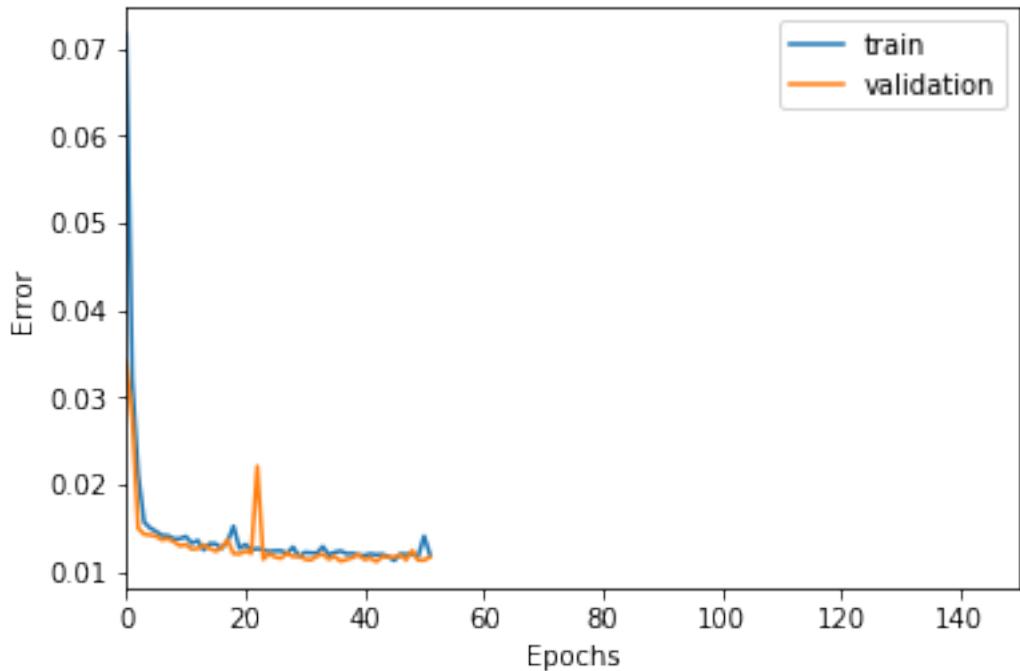
20 steps

```
In [160]: TIMESTEPS = 20  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru1_20"
```

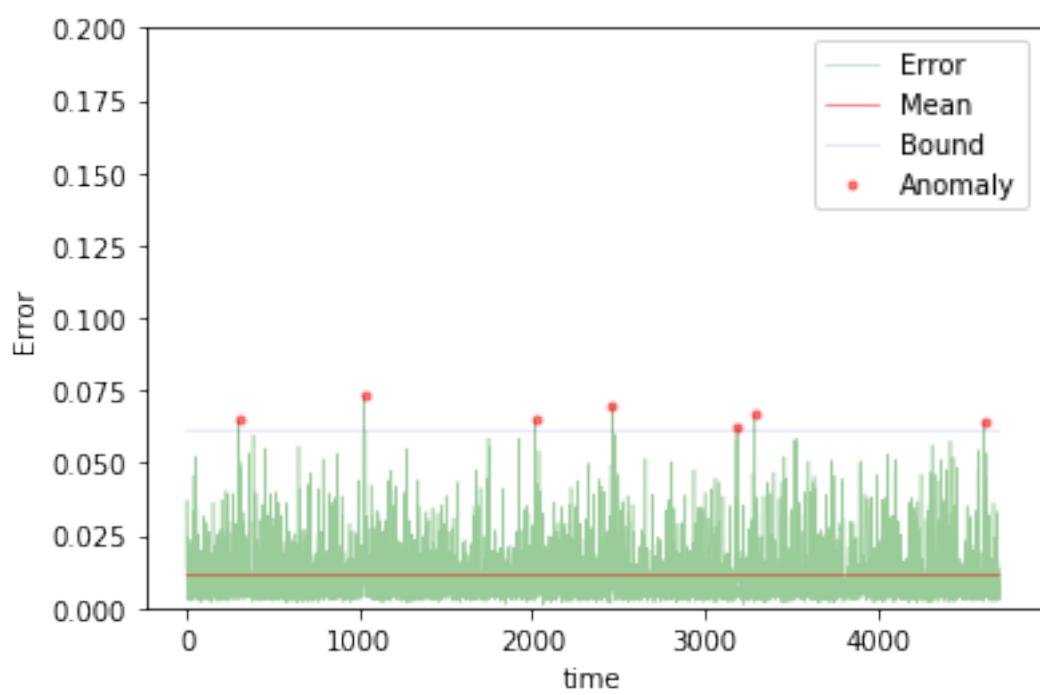
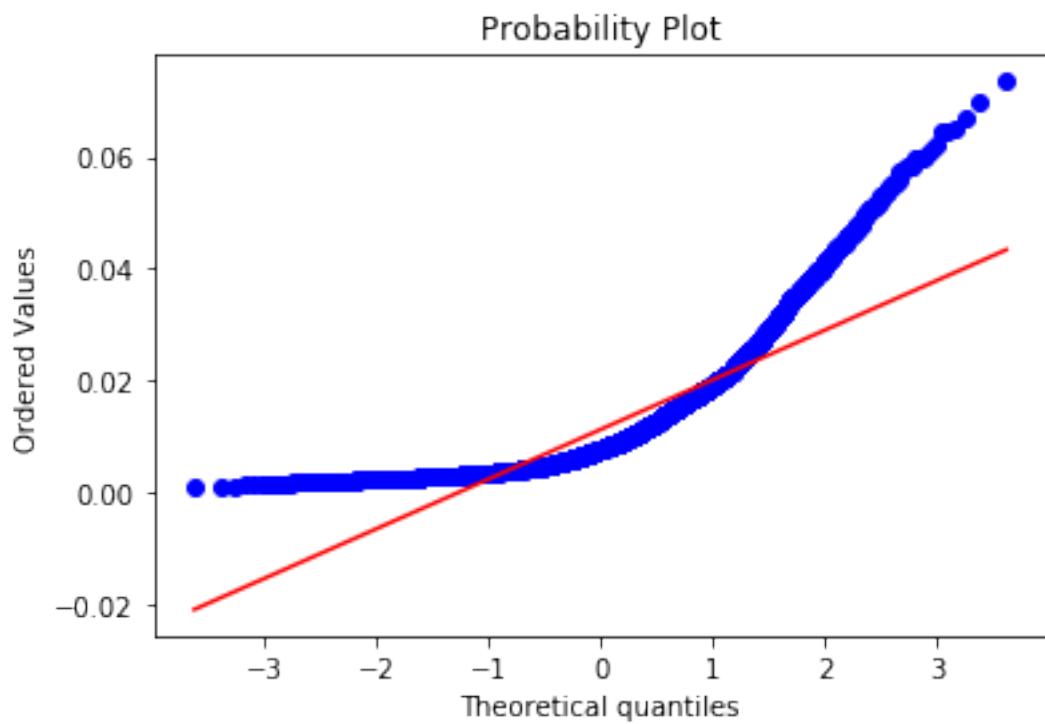
```
In [161]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [162]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

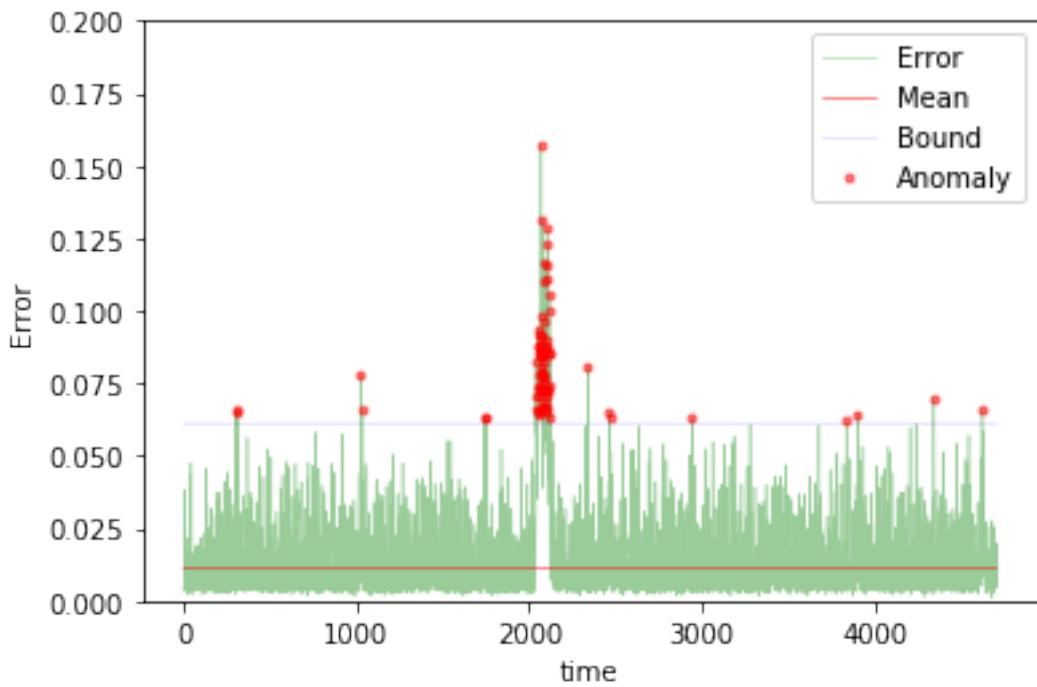
In [163]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



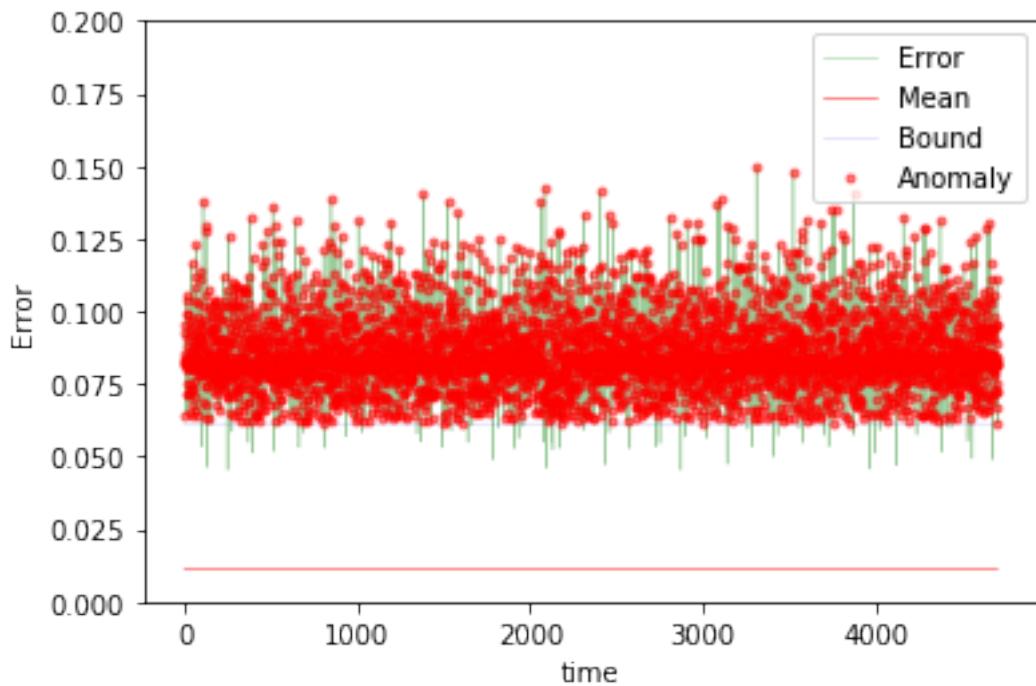
```
Training loss for final epoch is 0.011937156777596101
Validation loss for final epoch is 0.011663857864681632
----- Beginning tests for gru1_20 -----
Testing on normal data.
```



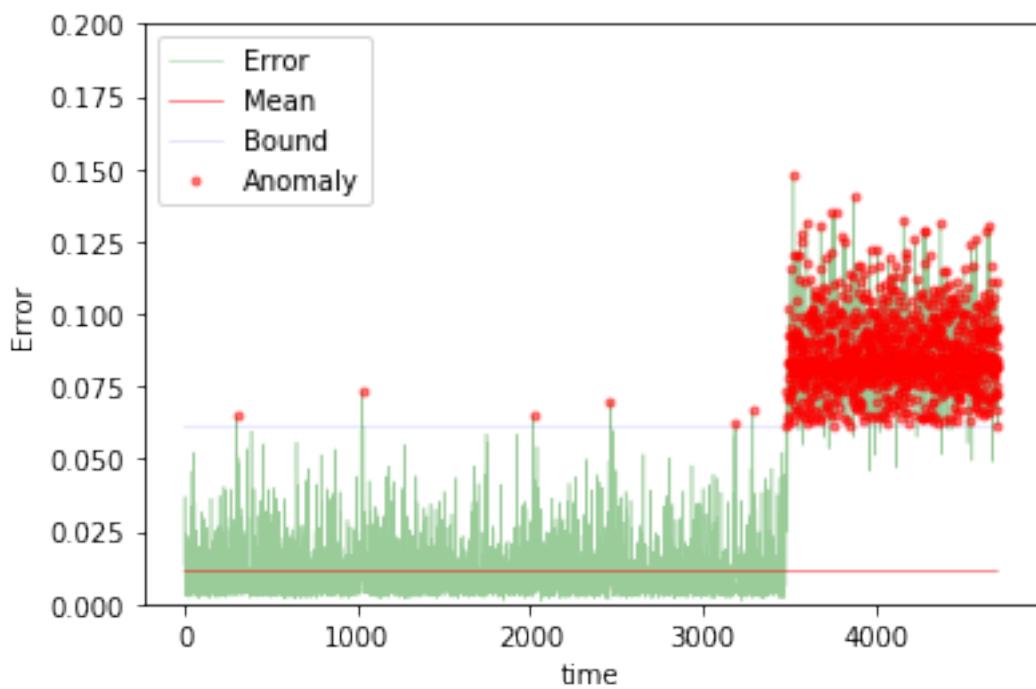
The mean error for gru1_20_normal_ is 0.011309675499037178 for length 4709
Testing on anomaly data.



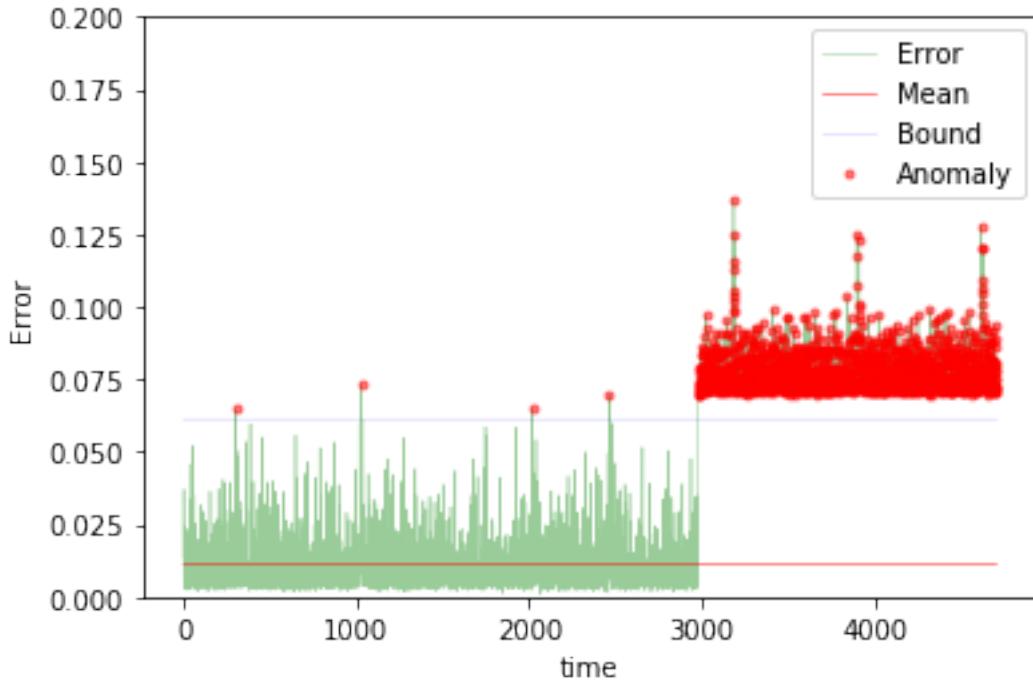
The mean error for gru1_20_anomaly_ is 0.013487180245208006 for length 4709
Testing on different app data.



The mean error for gru1_20_diff_app_ is 0.08567203199399698 for length 4709
Testing on App change synthetic data.



```
The mean error for gru1_20_app_change_ is 0.030569031227939254 for length 4709  
Testing on Net flood synthetic data.
```



```
The mean error for gru1_20_net_flood_ is 0.035618702778092134 for length 4709  
=====
```

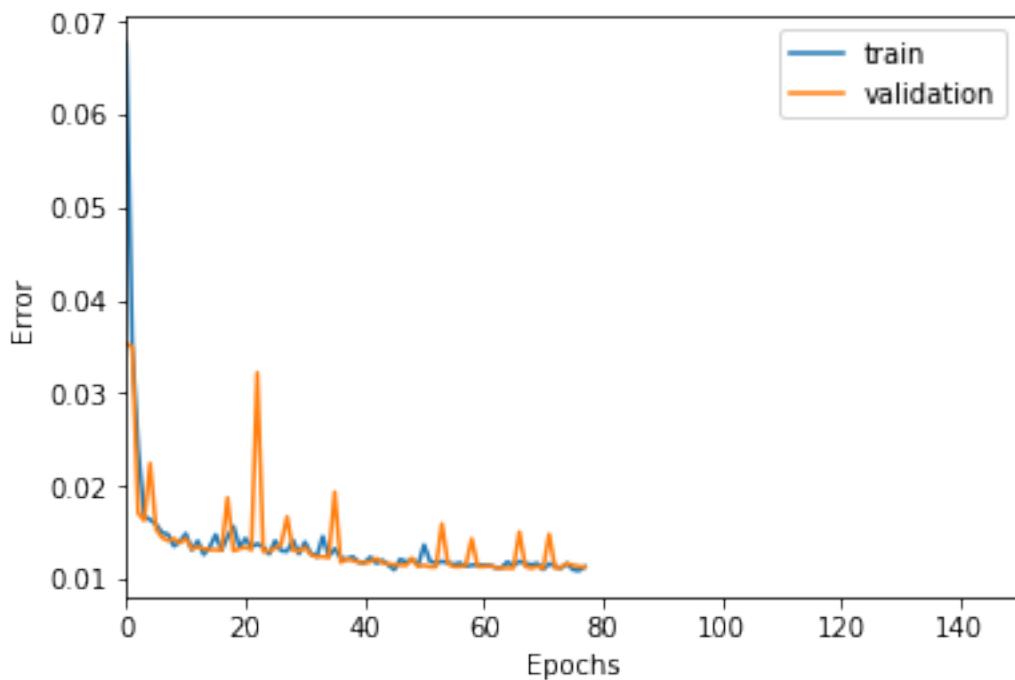
50 steps

```
In [164]: TIMESTEPS = 50  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru1_50"
```

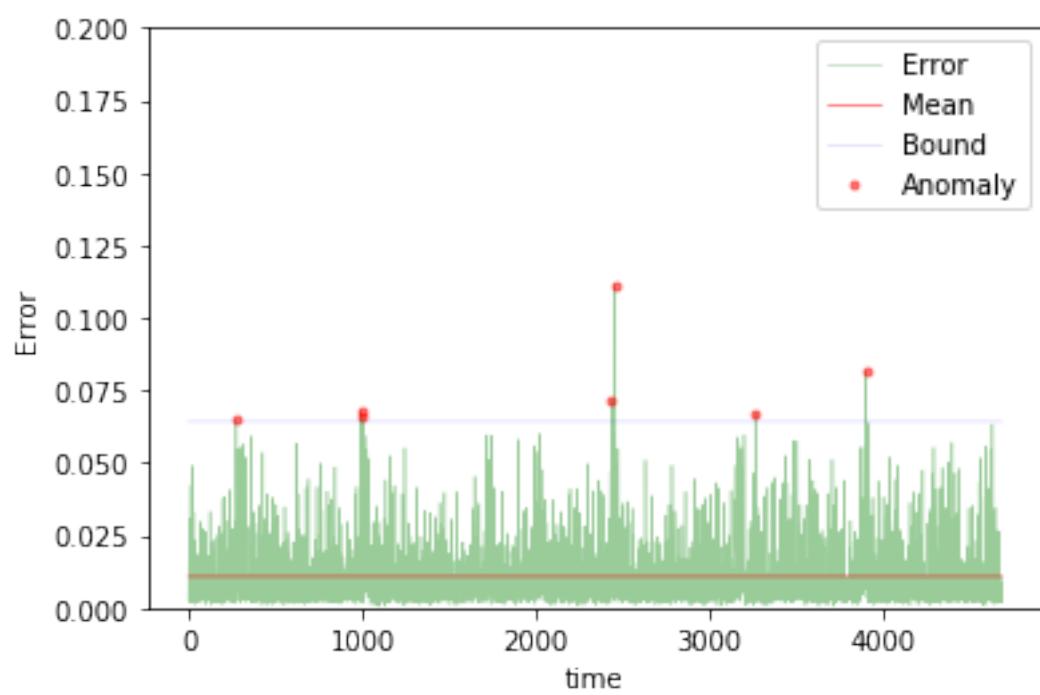
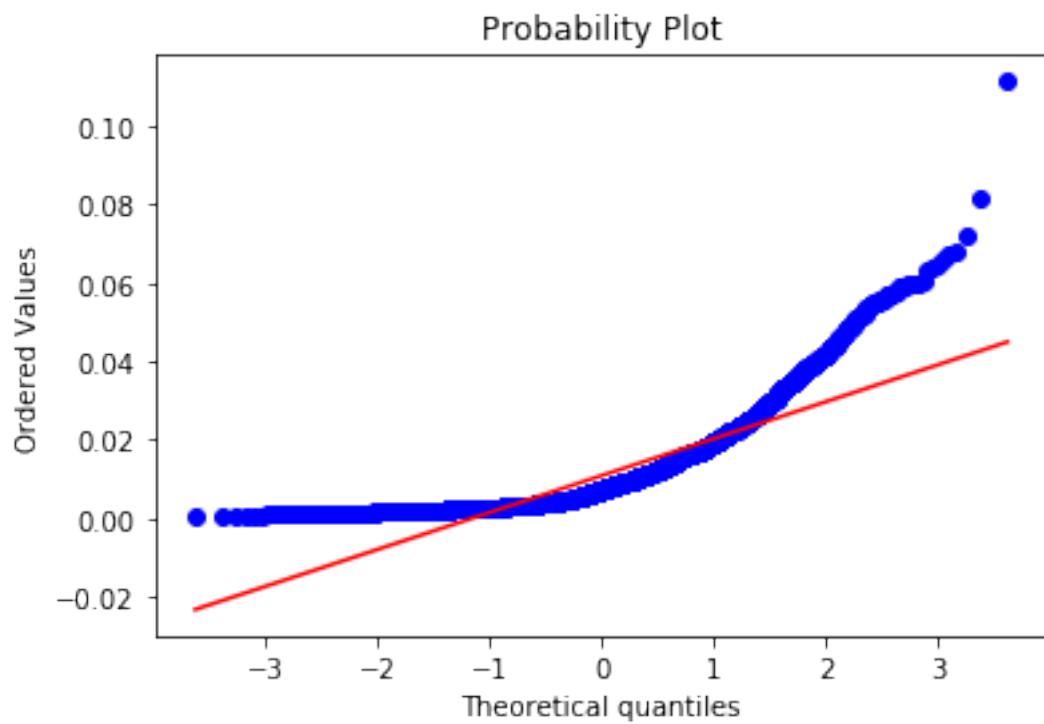
```
In [165]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [166]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

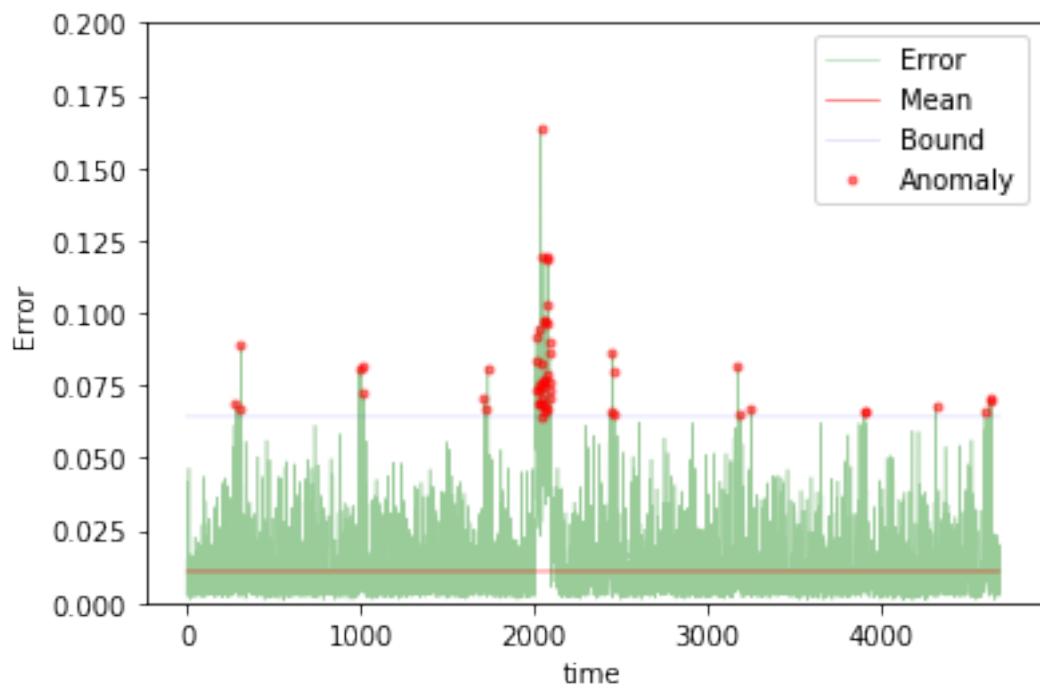
```
In [167]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



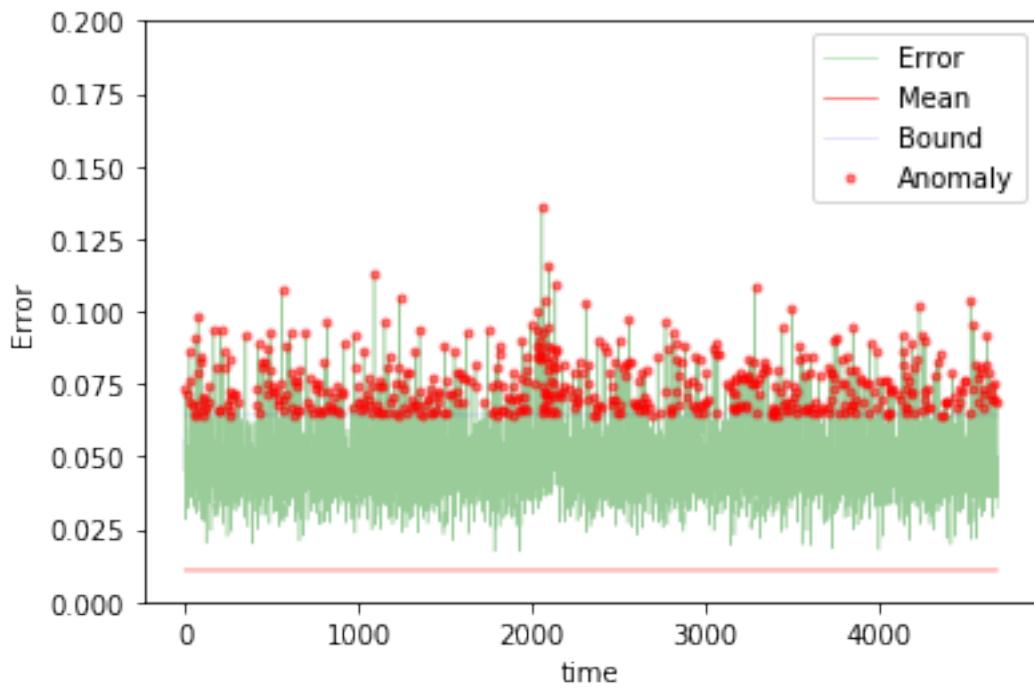
```
Training loss for final epoch is 0.011318125466816127
Validation loss for final epoch is 0.011230091270059347
----- Beginning tests for gru1_50 -----
Testing on normal data.
```



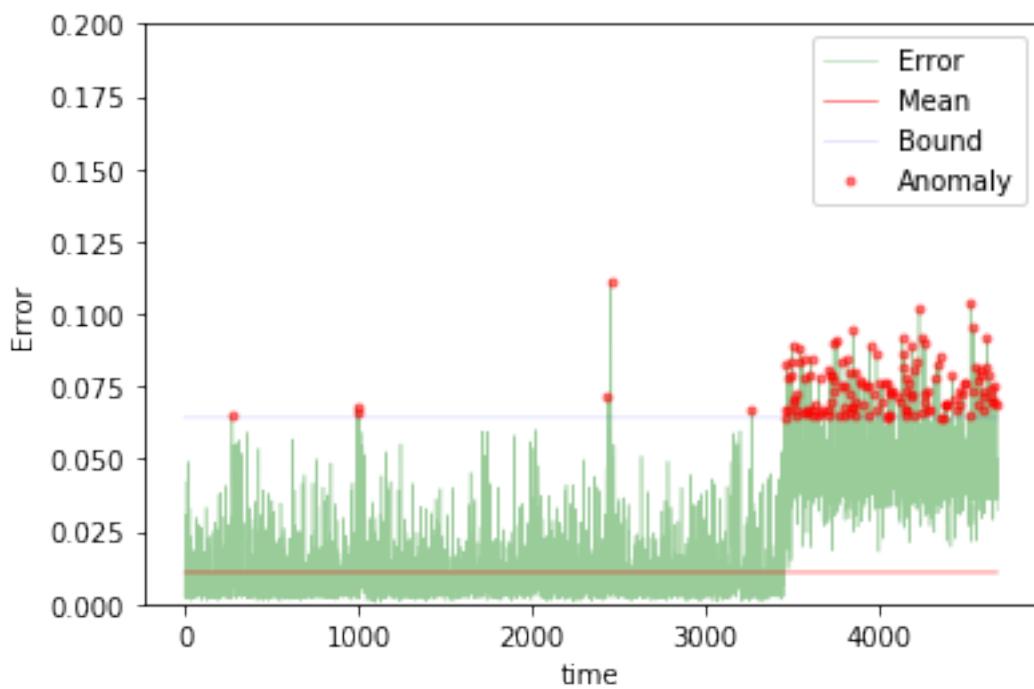
The mean error for gru1_50_normal_ is 0.010889310374524492 for length 4679
Testing on anomaly data.



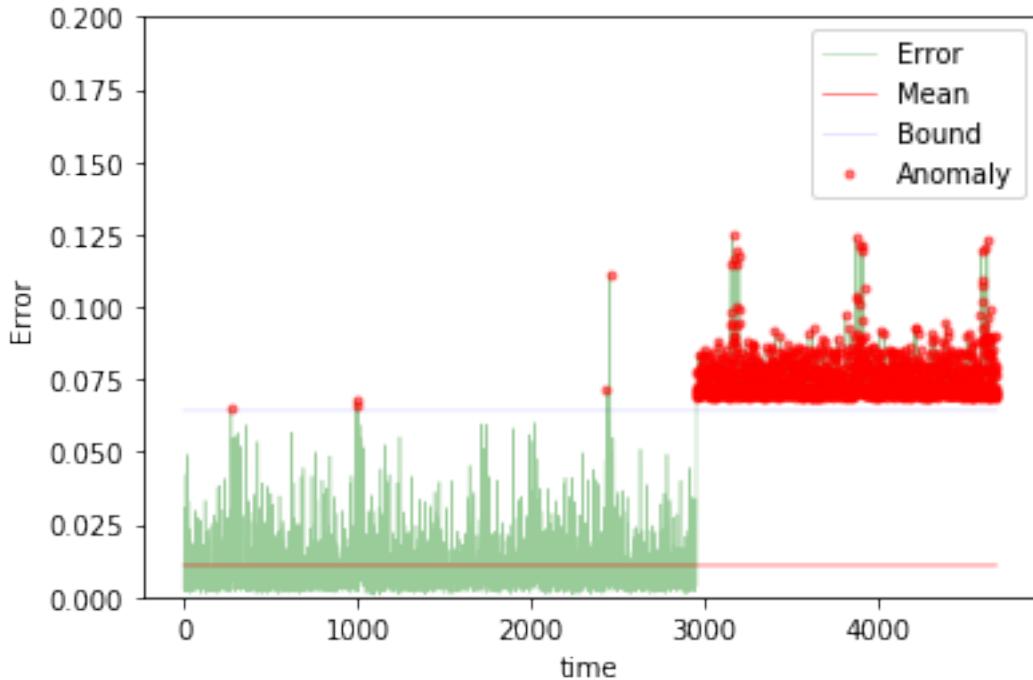
The mean error for gru1_50_anomaly_ is 0.013113472712043012 for length 4679
Testing on different app data.



The mean error for gru1_50_diff_app_ is 0.04681481573964798 for length 4679
Testing on App change synthetic data.



```
The mean error for gru1_50_app_change_ is 0.020224948759447196 for length 4679
Testing on Net flood synthetic data.
```



```
The mean error for gru1_50_net_flood_ is 0.03454686491434641 for length 4679
=====
=====
```

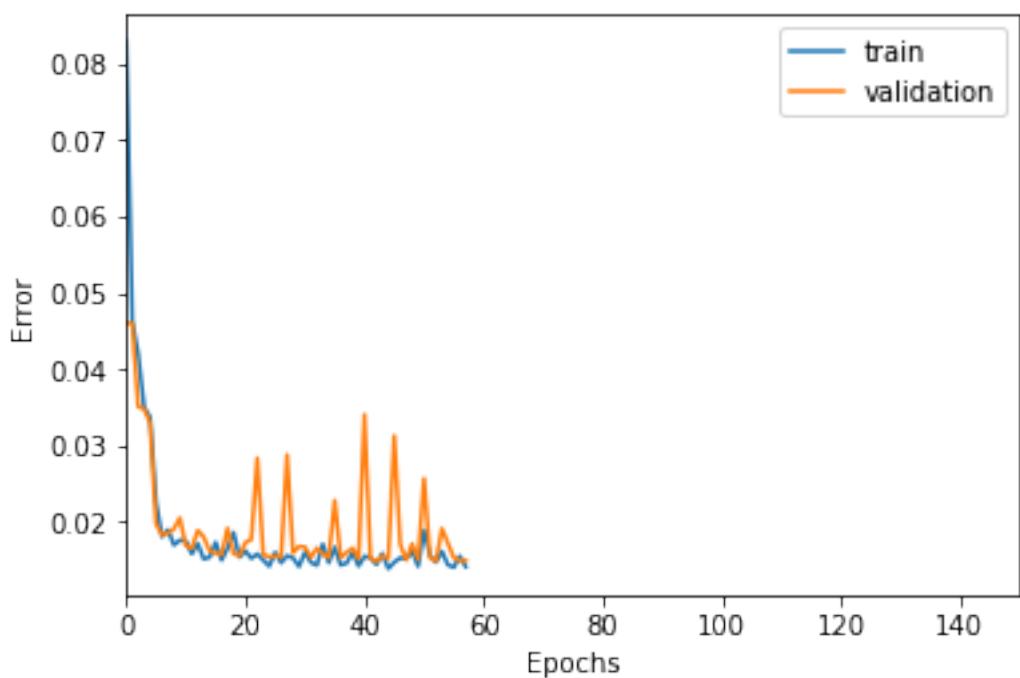
100 steps

```
In [168]: TIMESTEPS = 100
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru1_100"
```

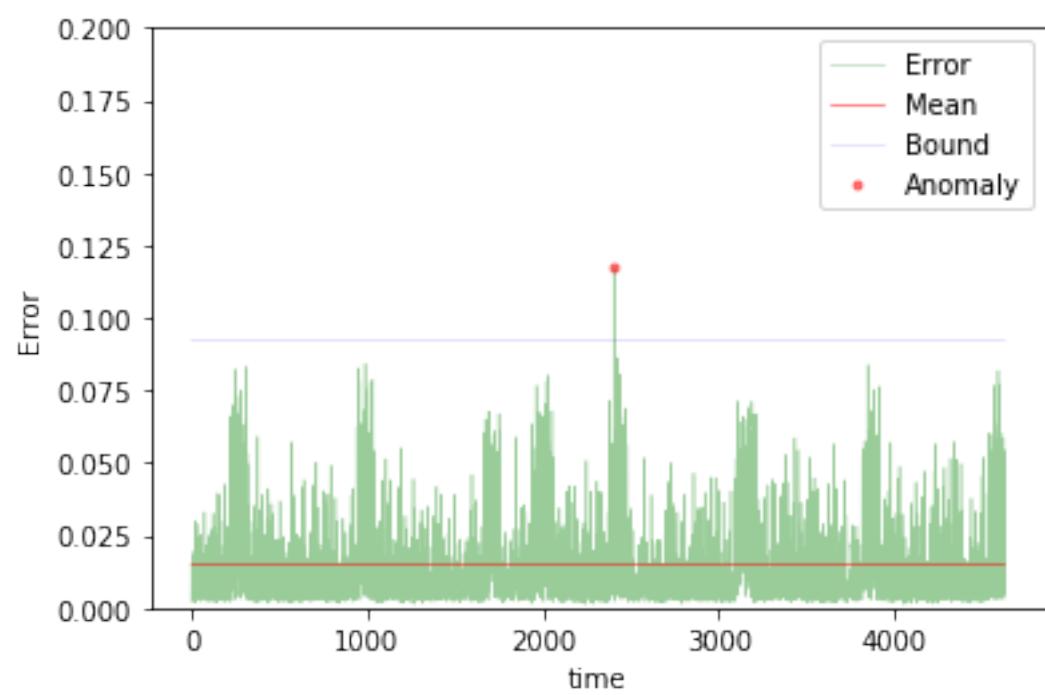
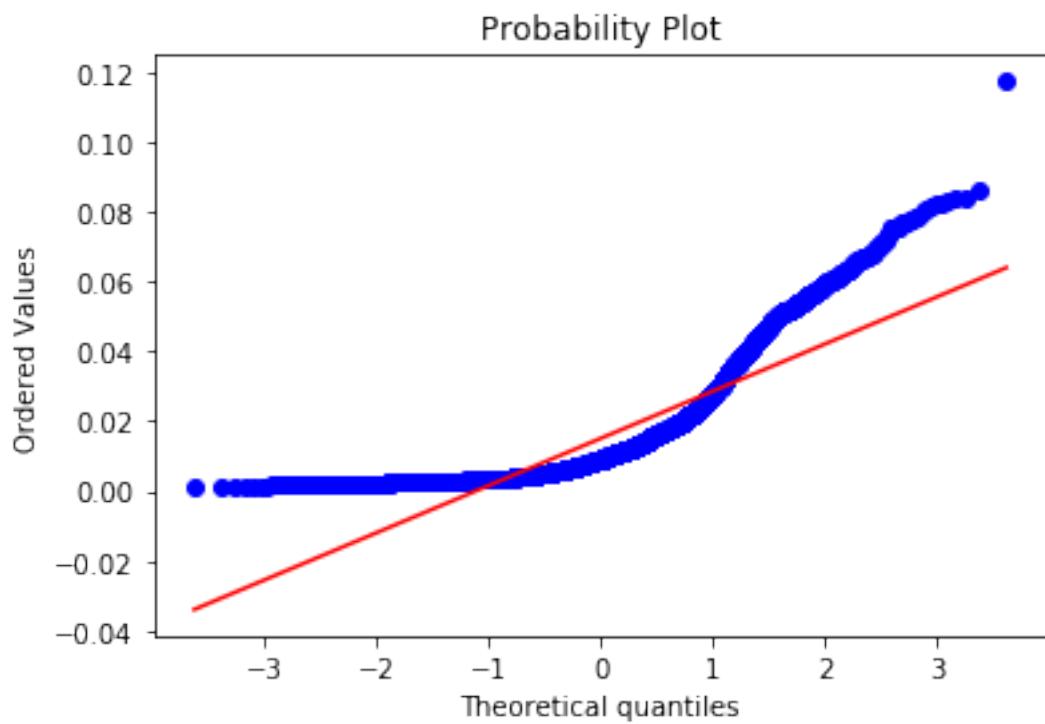
```
In [169]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu')(input_layer)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [170]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

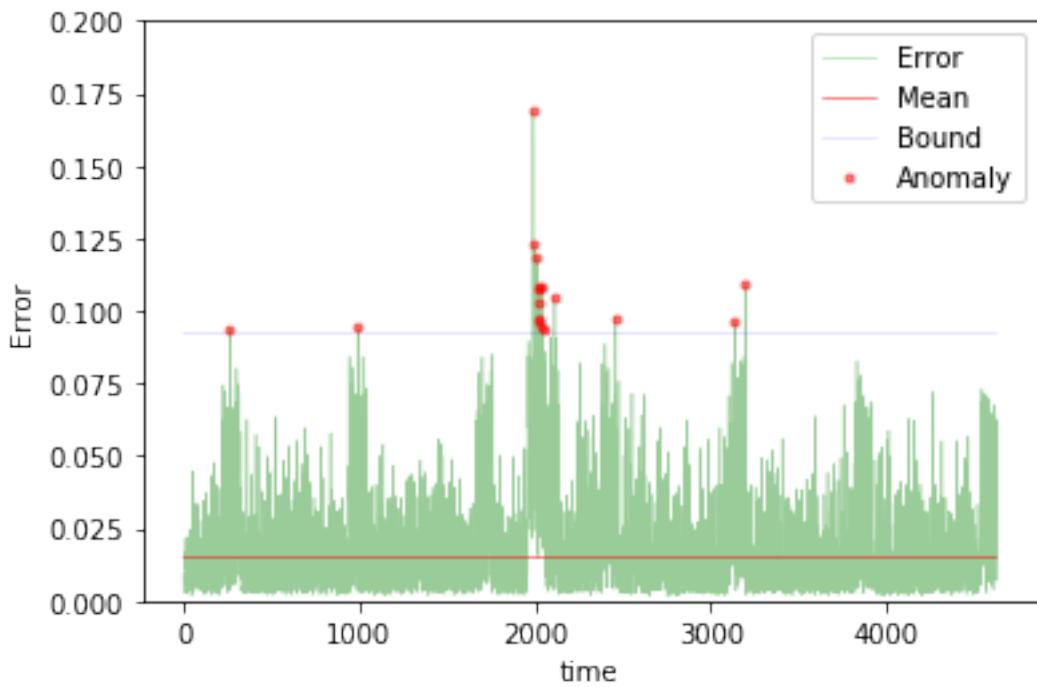
In [171]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



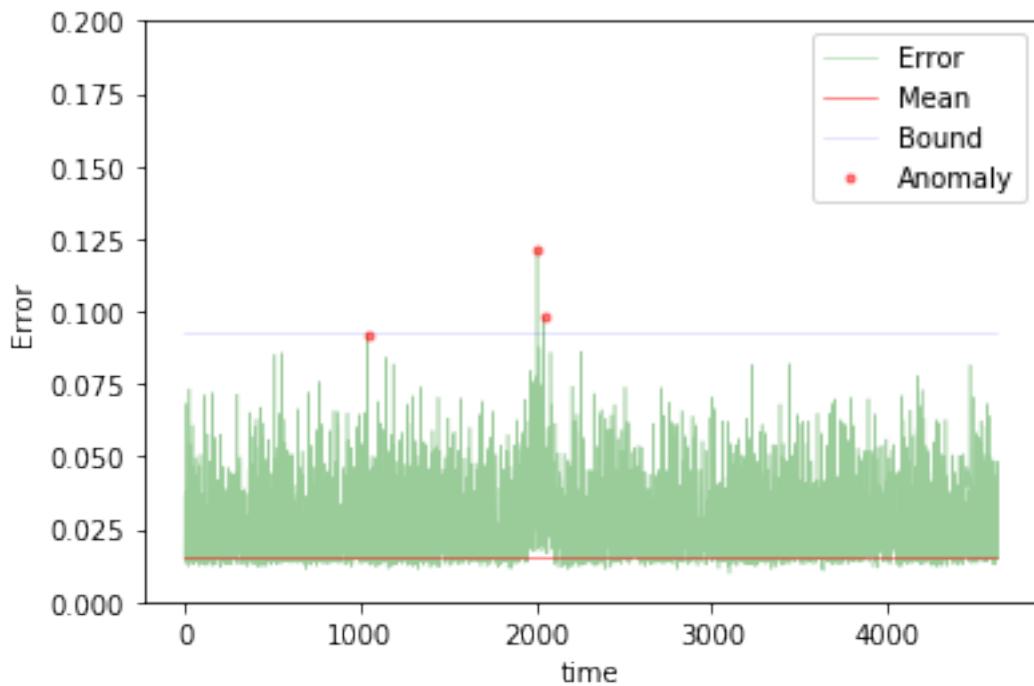
```
Training loss for final epoch is 0.014178670117864386
Validation loss for final epoch is 0.01498048027139157
----- Beginning tests for gru1_100 -----
Testing on normal data.
```



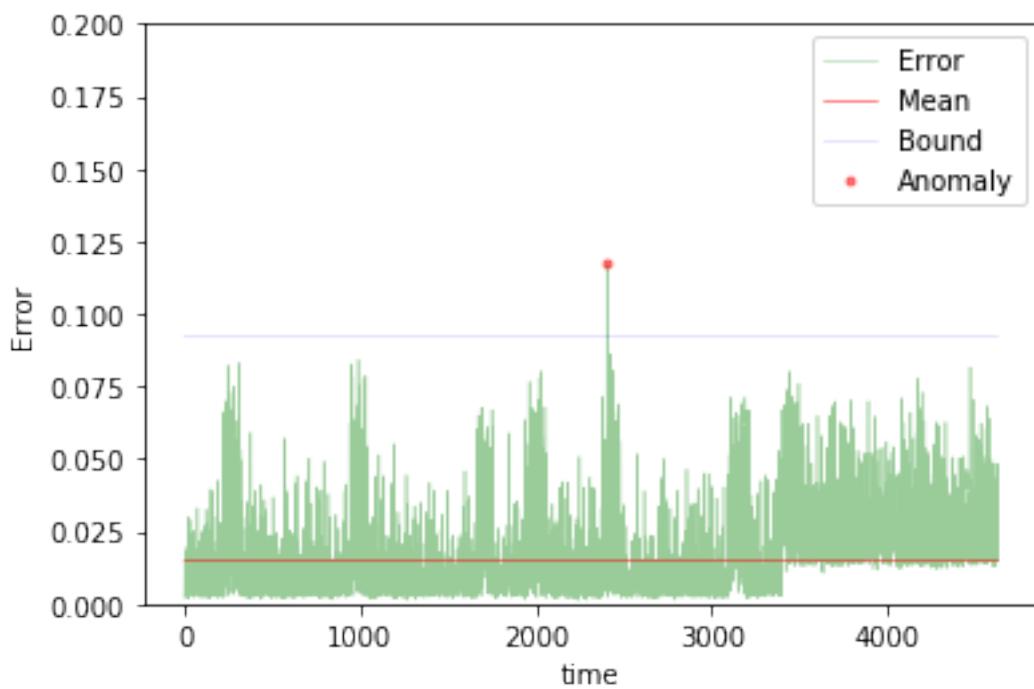
The mean error for gru1_100_normal_ is 0.014995388952710152 for length 4629
Testing on anomaly data.



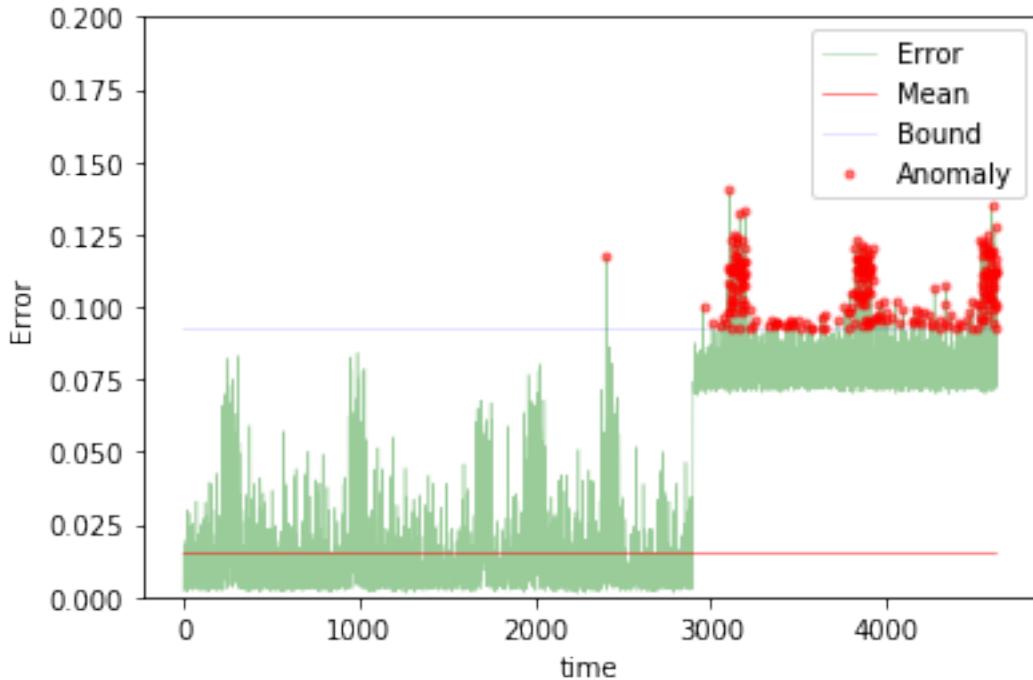
The mean error for gru1_100_anomaly_ is 0.019539647408847678 for length 4629
Testing on different app data.



The mean error for gru1_100_diff_app_ is 0.026093511491077394 for length 4629
Testing on App change synthetic data.



```
The mean error for gru1_100_app_change_ is 0.018336157995732186 for length 4629  
Testing on Net flood synthetic data.
```



```
The mean error for gru1_100_net_flood_ is 0.03987454424273587 for length 4629  
=====
```

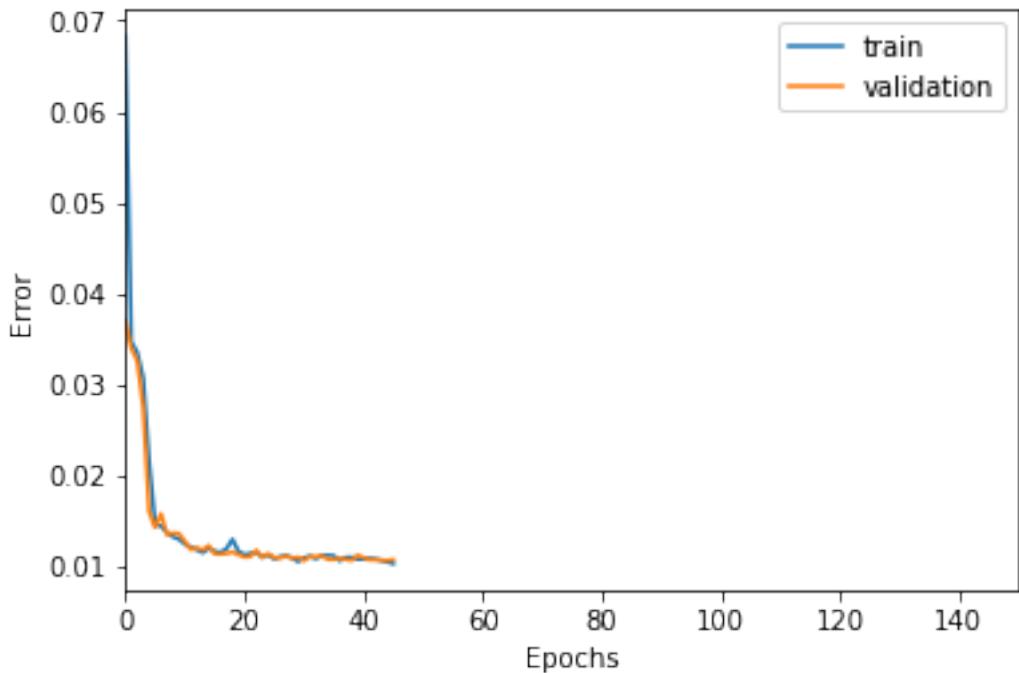
200 steps

```
In [172]: TIMESTEPS = 200  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru1_200"
```

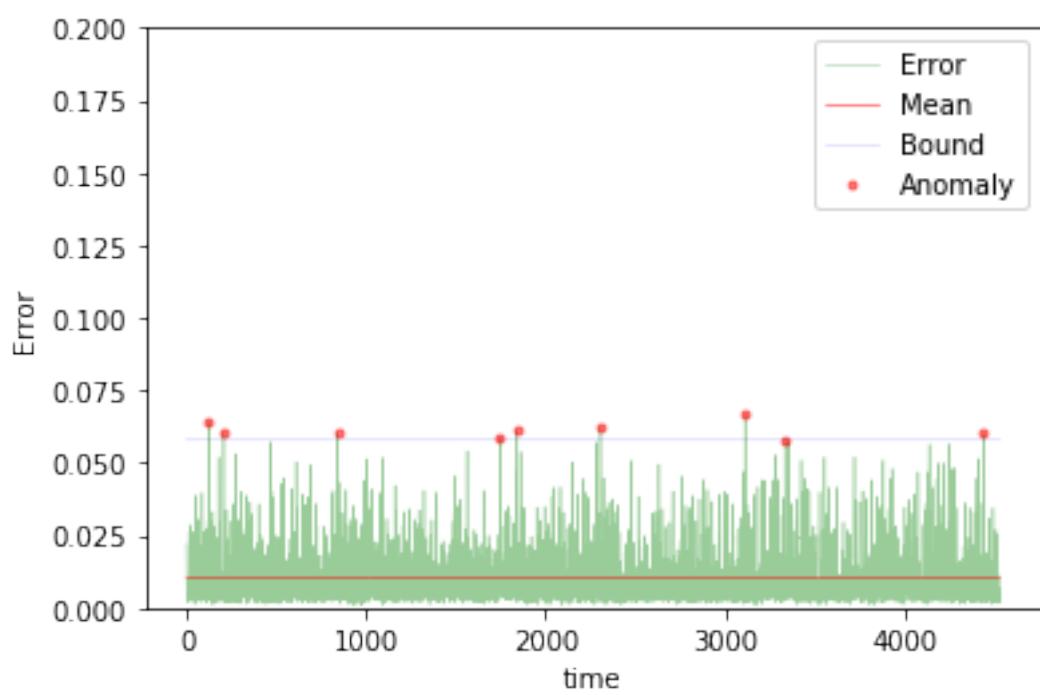
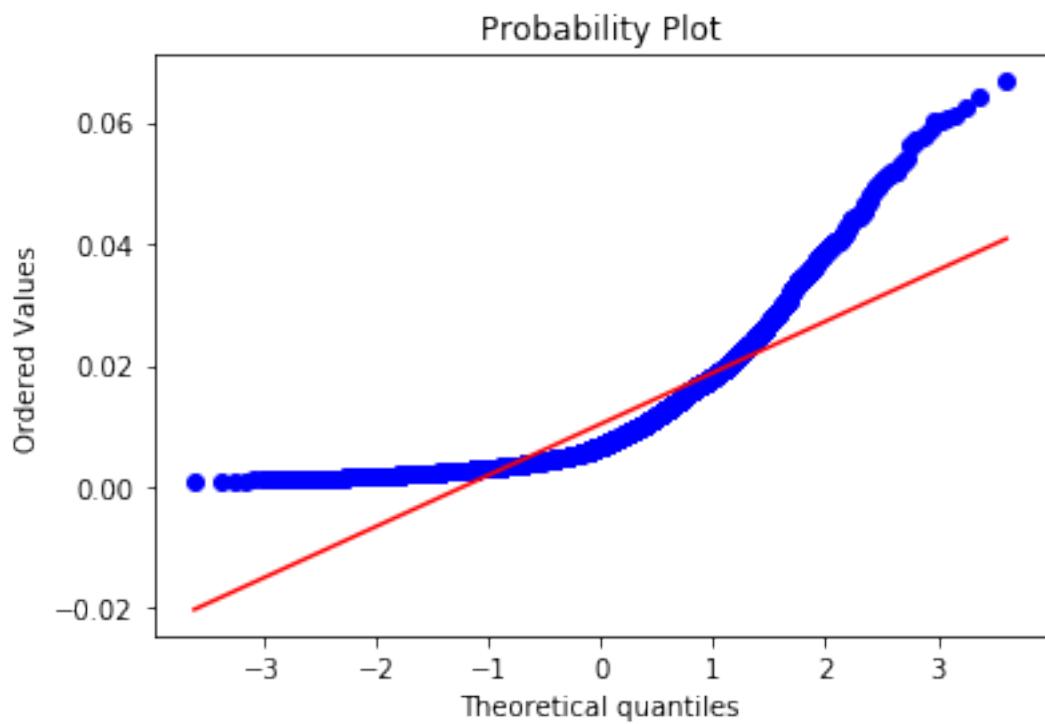
```
In [173]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu')(input_layer)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [174]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

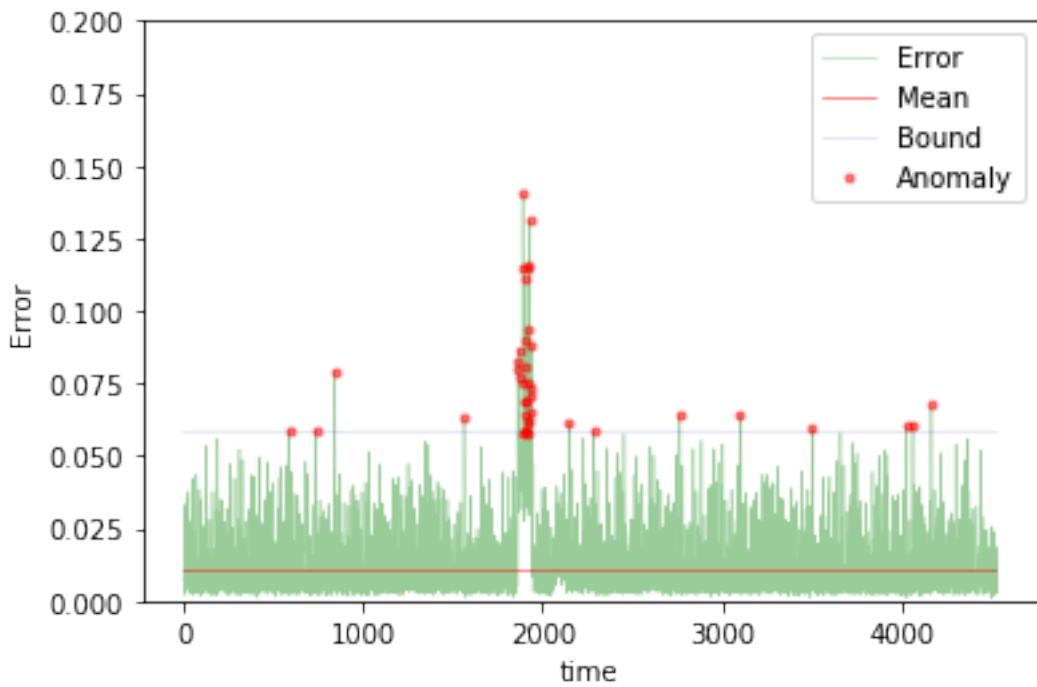
In [175]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



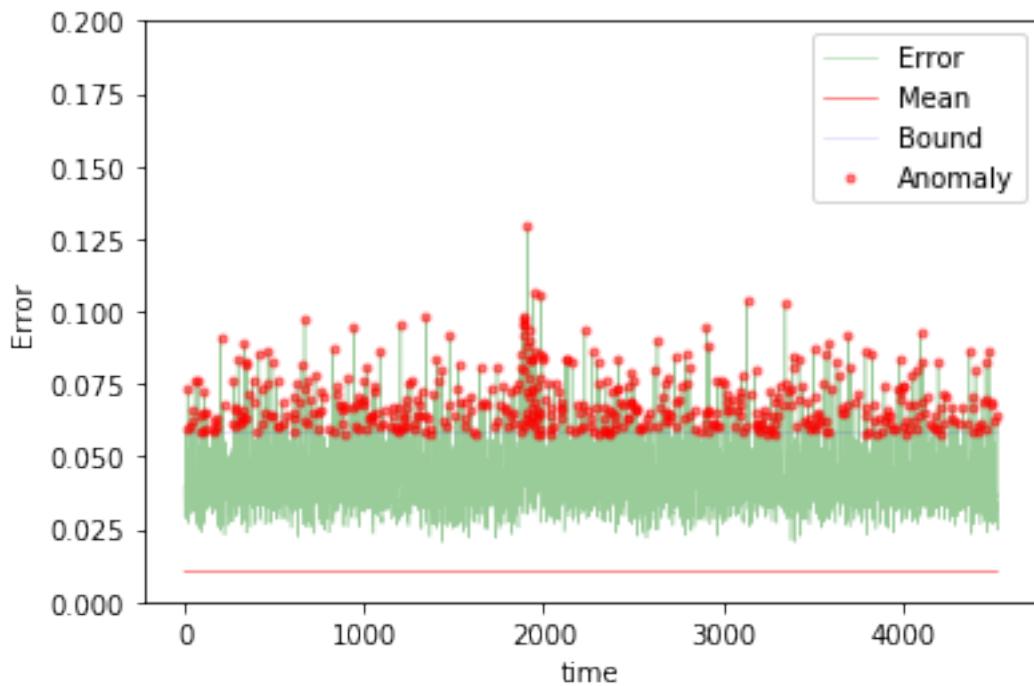
```
Training loss for final epoch is 0.010343692054739222
Validation loss for final epoch is 0.010782045458909125
----- Beginning tests for gru1_200 -----
Testing on normal data.
```



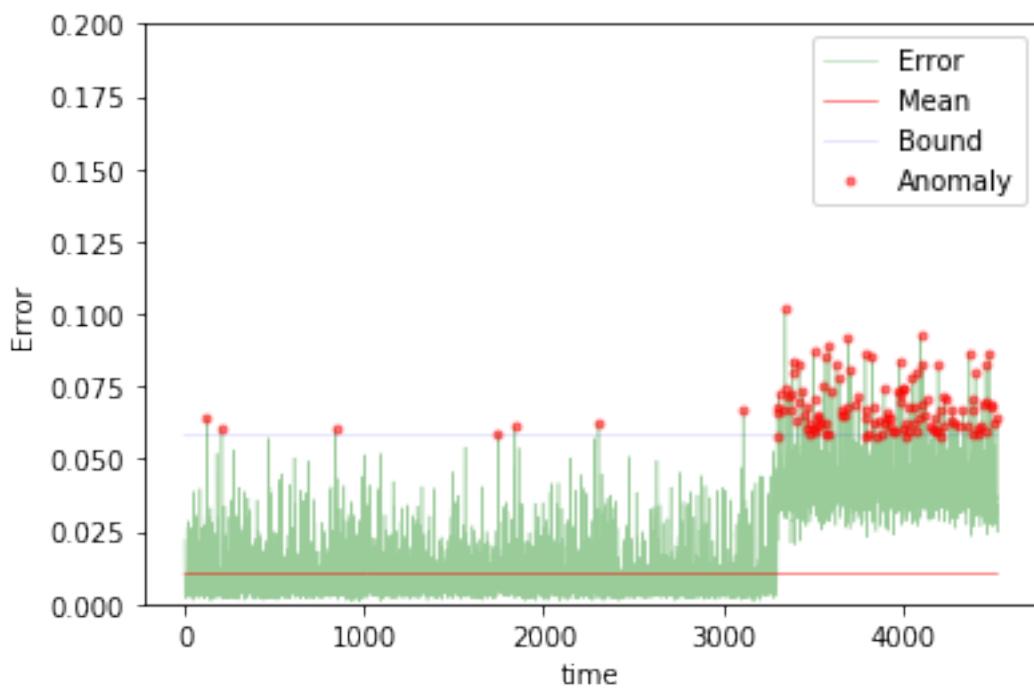
The mean error for gru1_200_normal_ is 0.010323449398808854 for length 4529
Testing on anomaly data.



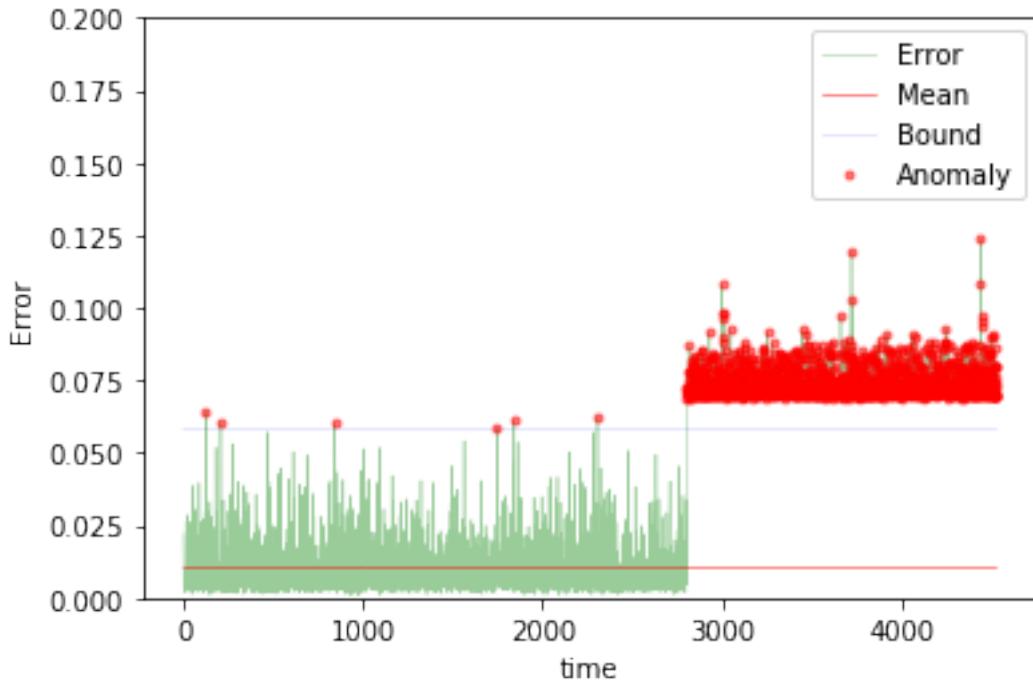
The mean error for gru1_200_anomaly_ is 0.01225523424502087 for length 4529
Testing on different app data.



The mean error for gru1_200_diff_app_ is 0.042366866472603515 for length 4529
Testing on App change synthetic data.



```
The mean error for gru1_200_app_change_ is 0.018859418166178848 for length 4529
Testing on Net flood synthetic data.
```



```
The mean error for gru1_200_net_flood_ is 0.03477269407686612 for length 4529
=====
=====
```

2.1.6 RNN with 2 GRU layers

2 steps

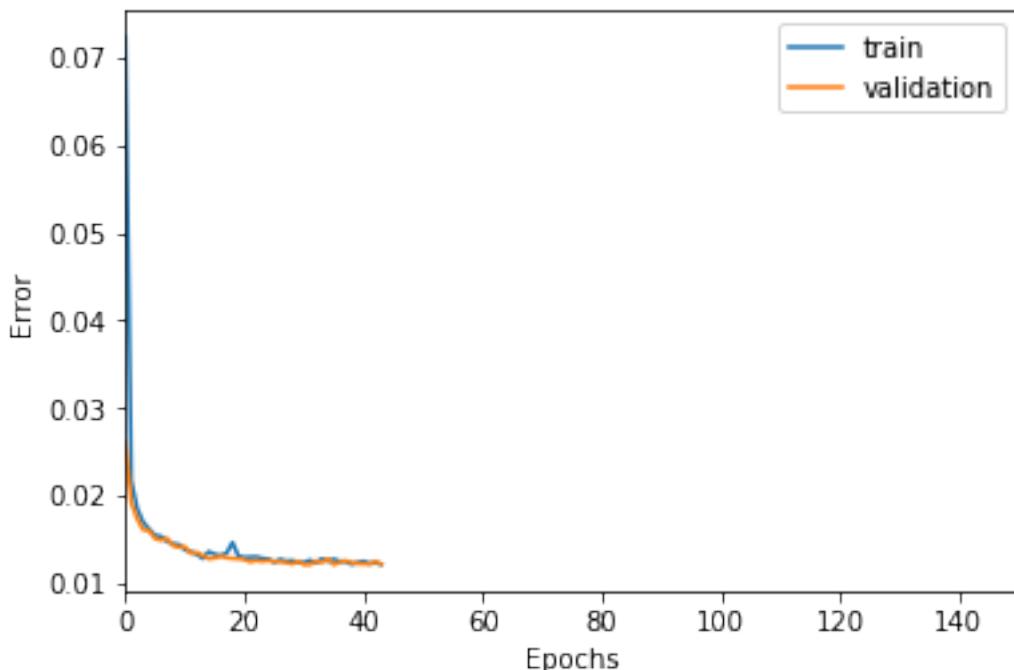
```
In [176]: TIMESTEPS = 2
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru2_2"
```

```
In [177]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
```

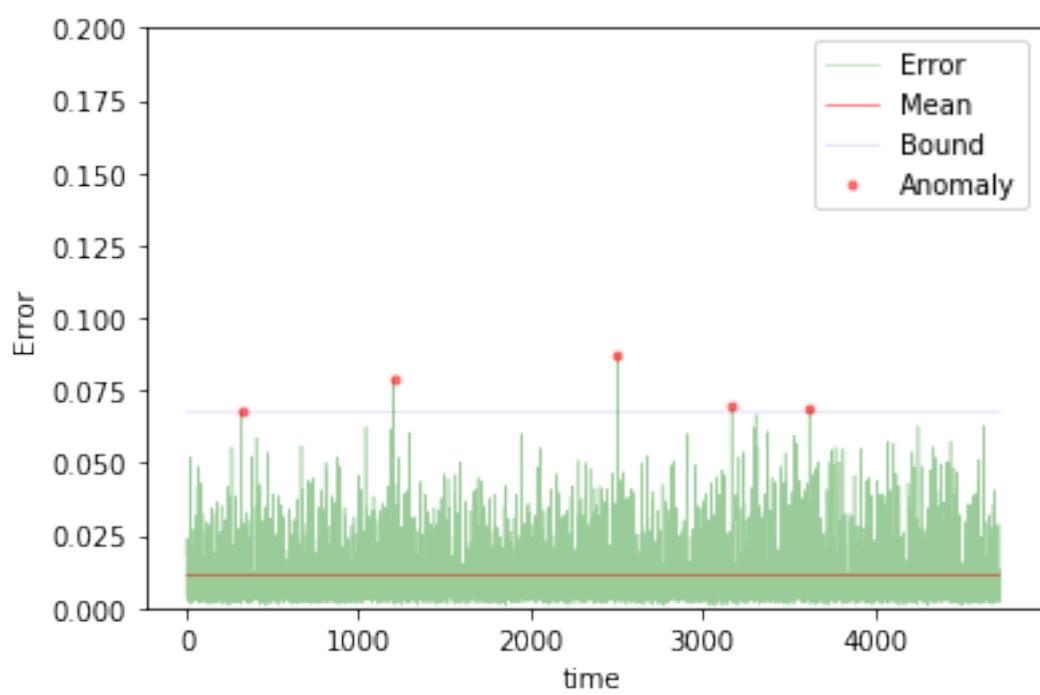
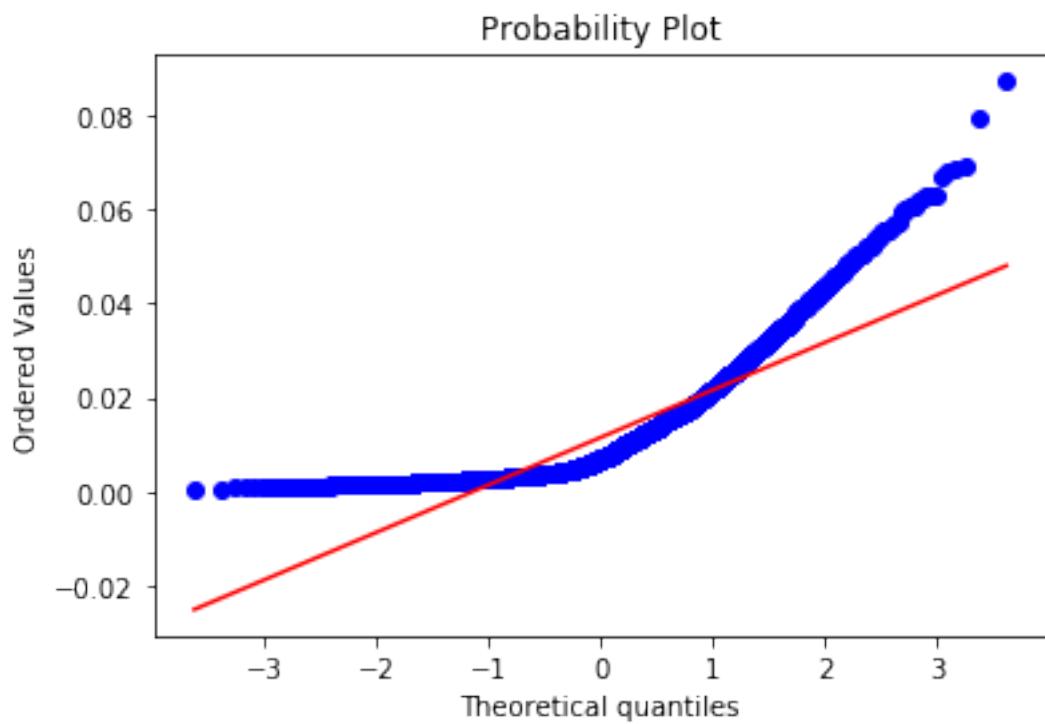
```
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [178]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

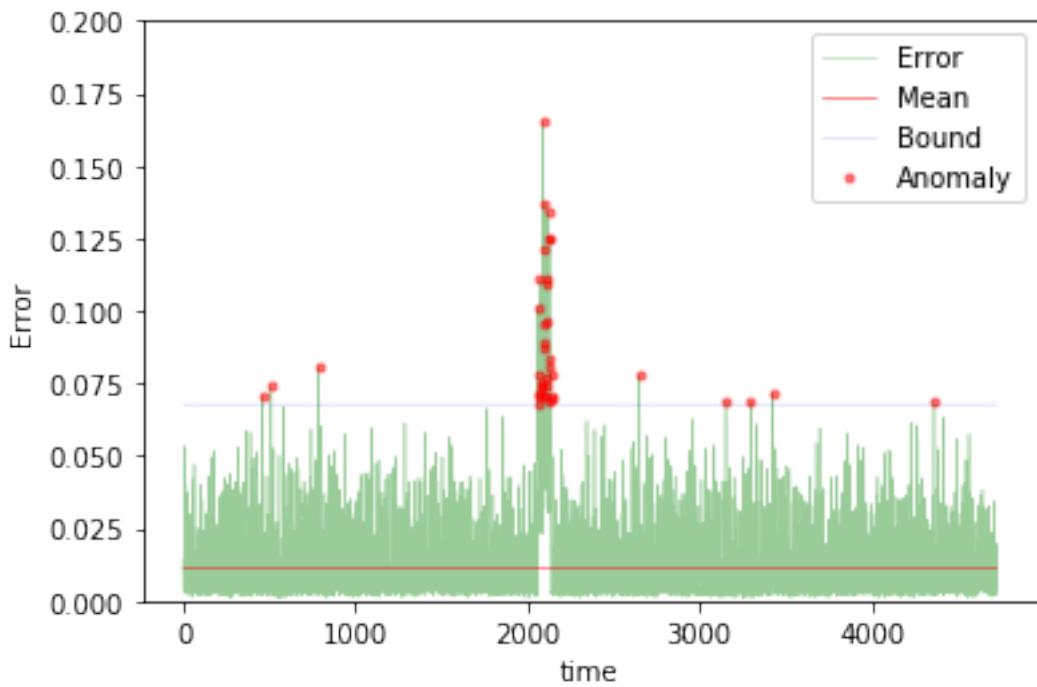
```
In [179]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



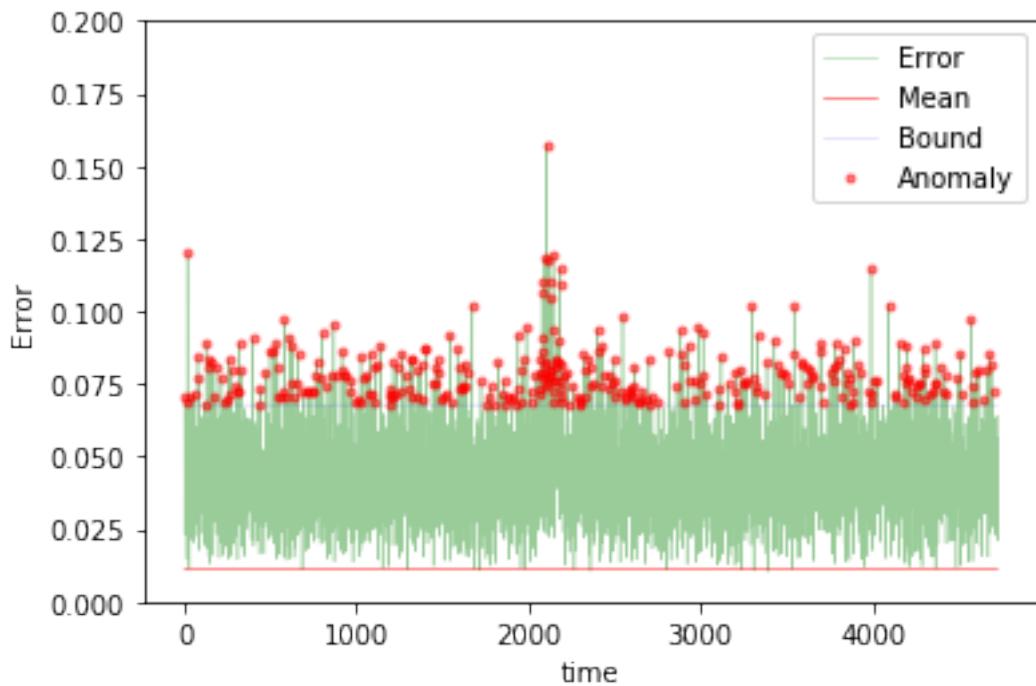
```
Training loss for final epoch is 0.012188010055338963
Validation loss for final epoch is 0.012179672989877871
----- Beginning tests for gru2_2 -----
Testing on normal data.
```



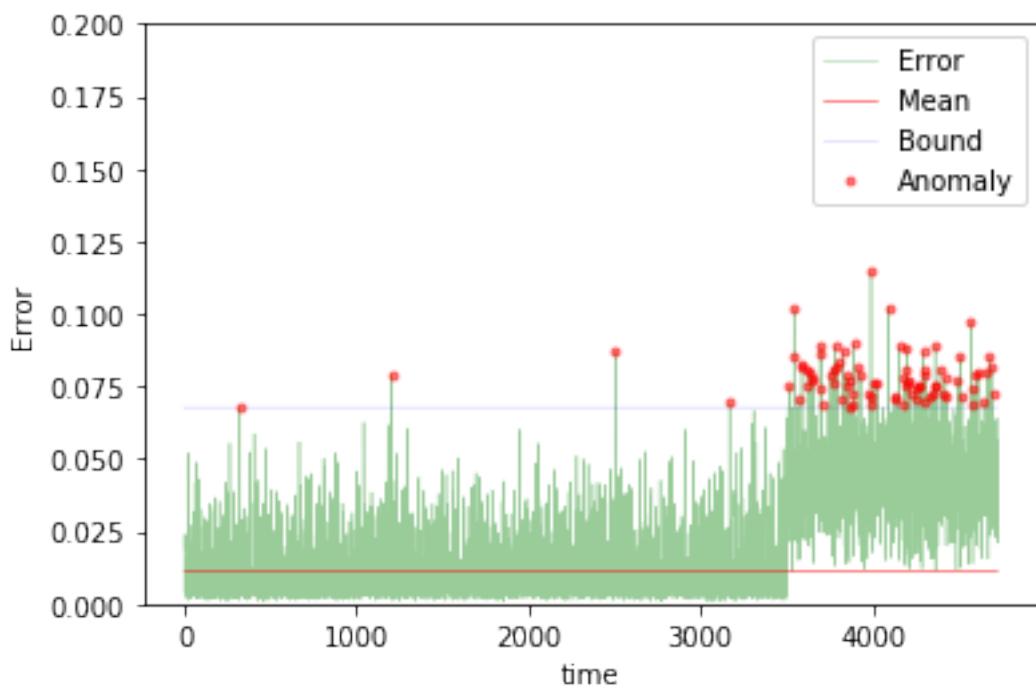
The mean error for gru2_2_normal_ is 0.011495802789756912 for length 4727
Testing on anomaly data.



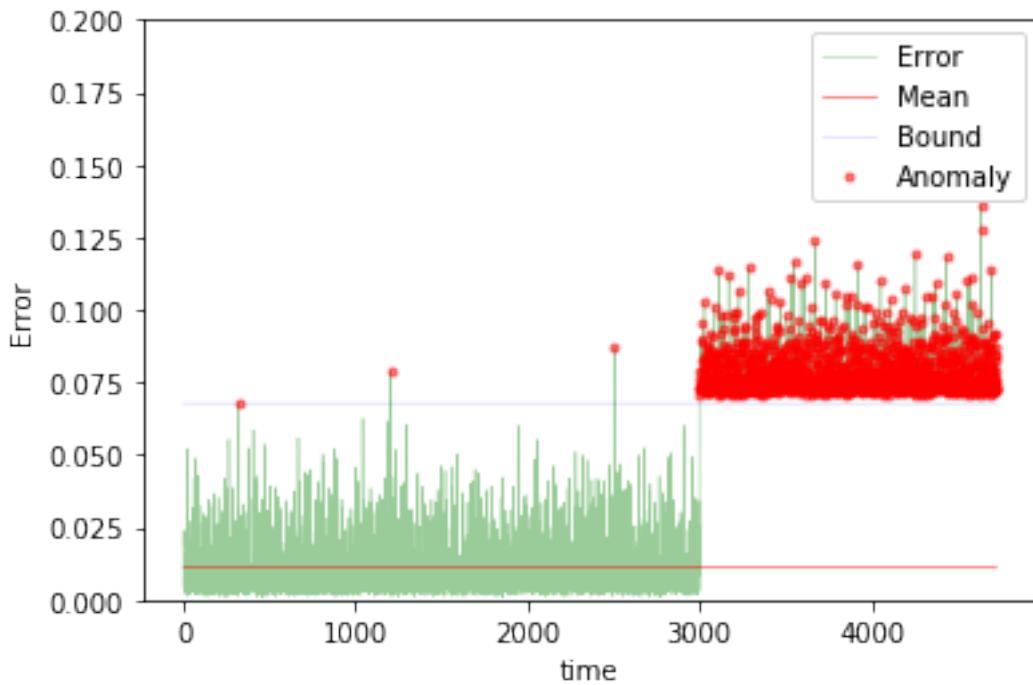
The mean error for gru2_2_anomaly_ is 0.013325474991064007 for length 4727
Testing on different app data.



The mean error for gru2_2_diff_app_ is 0.04264227123941146 for length 4727
Testing on App change synthetic data.



The mean error for gru2_2_app_change_ is 0.019453655265064766 for length 4727
Testing on Net flood synthetic data.



The mean error for gru2_2_net_flood_ is 0.03583564065489814 for length 4727
=====

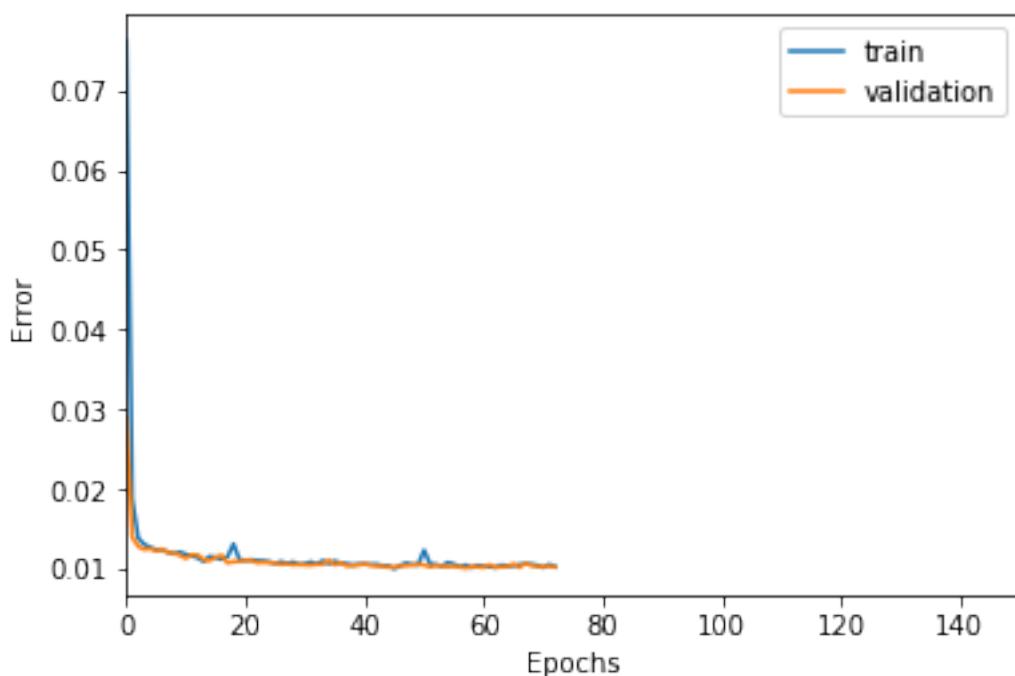
5 steps

```
In [180]: TIMESTEPS = 5
DIM = 29
tgen = flat_generator(X, TIMESTEPS, 0)
vgen = flat_generator(val_X, TIMESTEPS, 0)
name = "gru2_5"

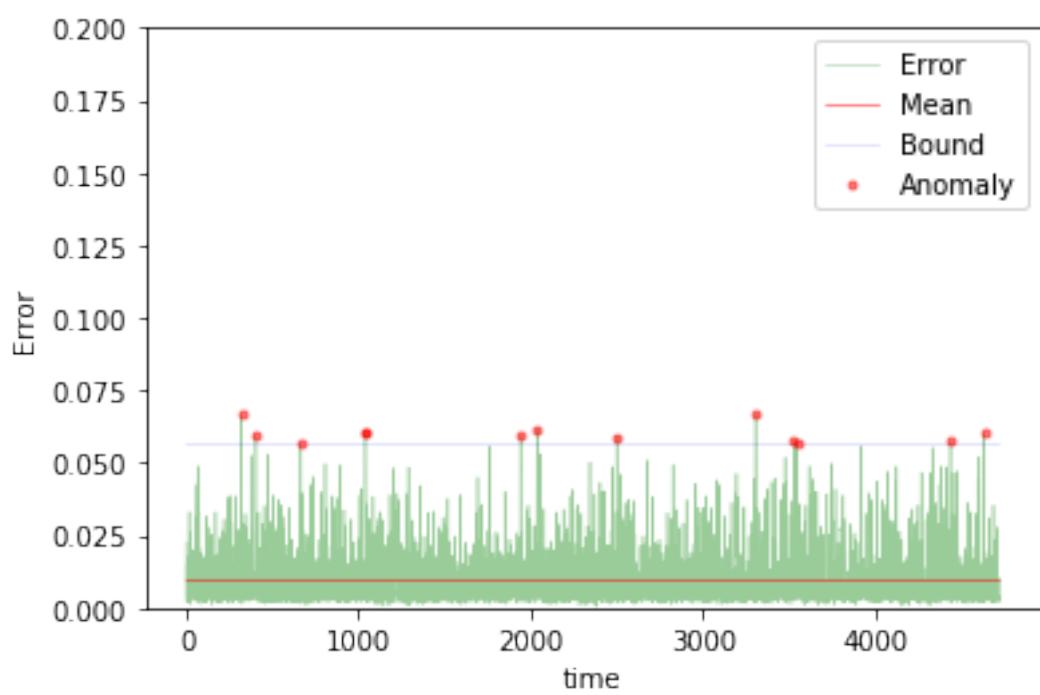
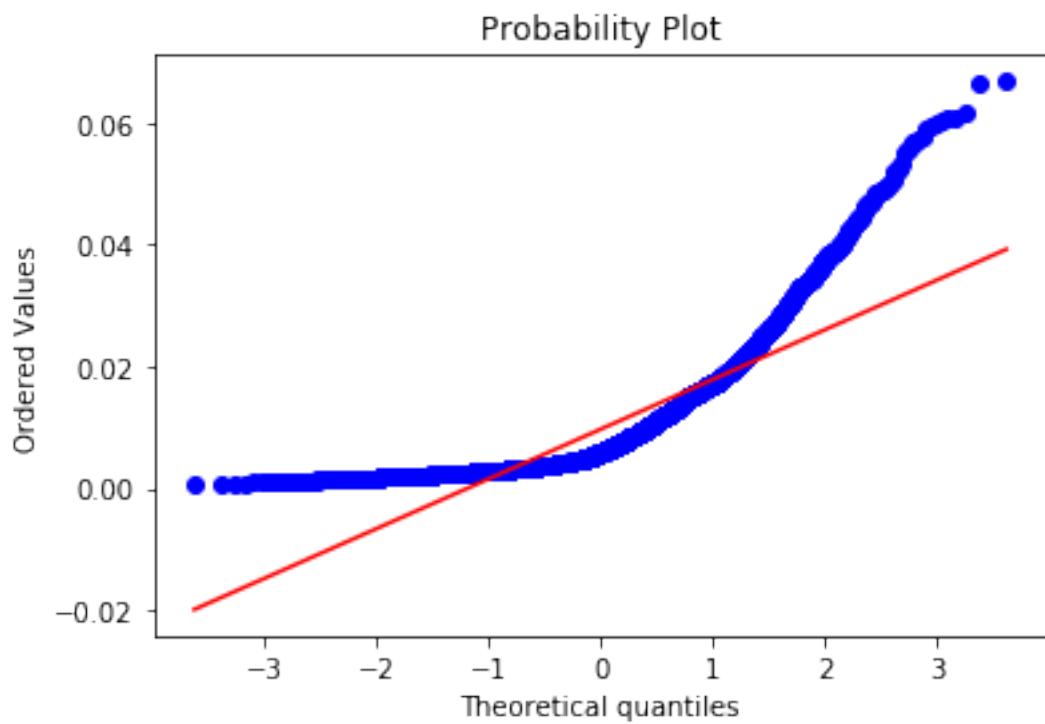
In [181]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [182]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

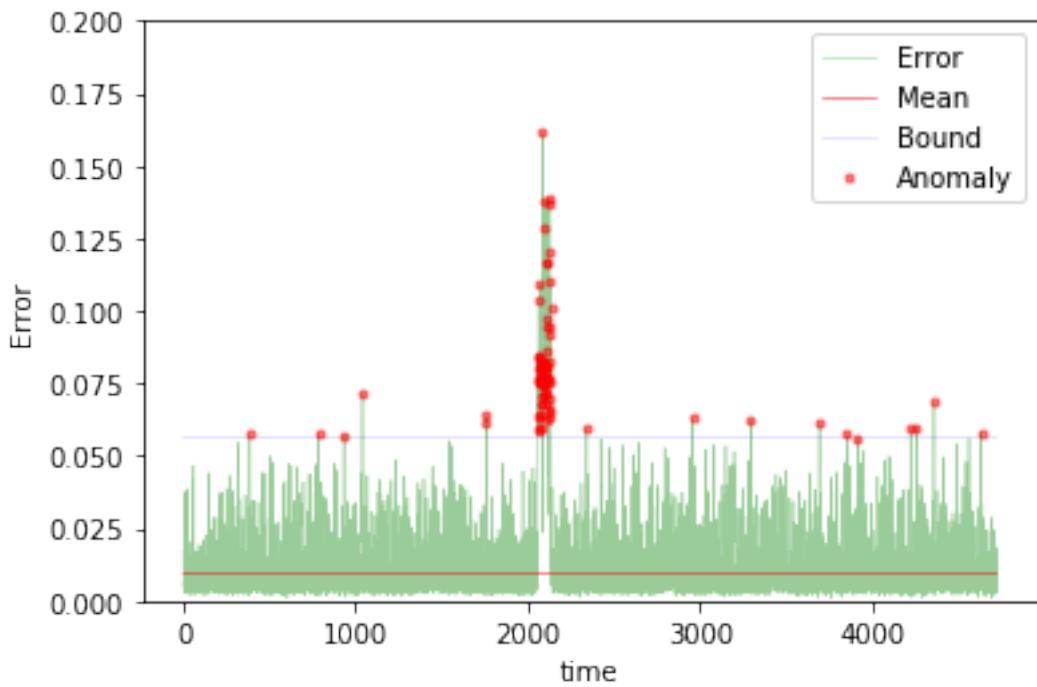
```
In [183]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



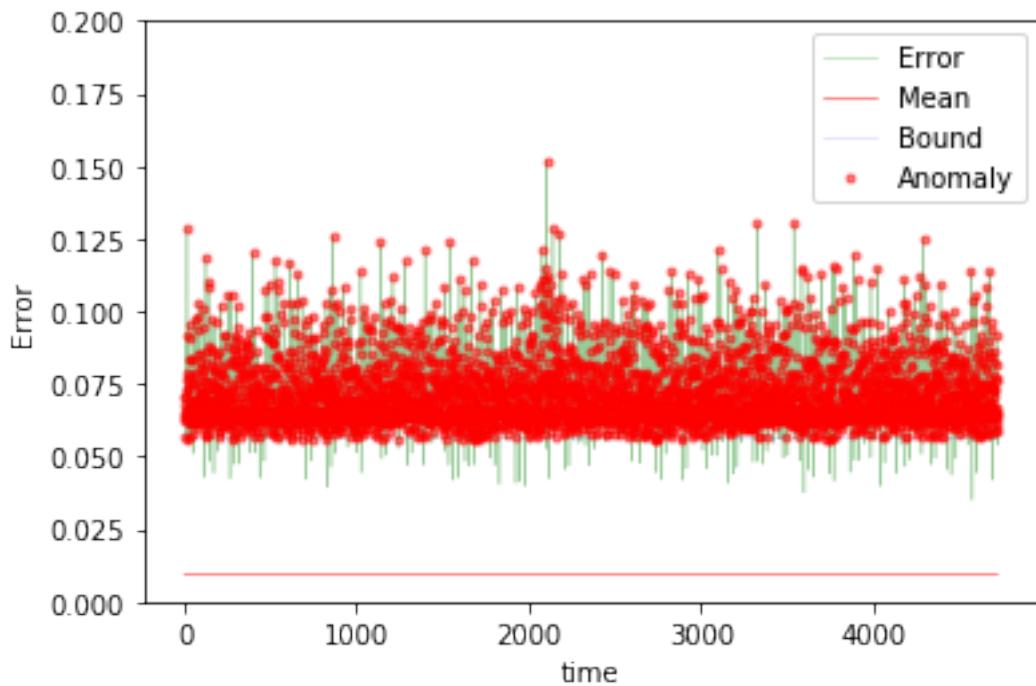
```
Training loss for final epoch is 0.010304250651388428
Validation loss for final epoch is 0.010259894917951897
----- Beginning tests for gru2_5 -----
Testing on normal data.
```



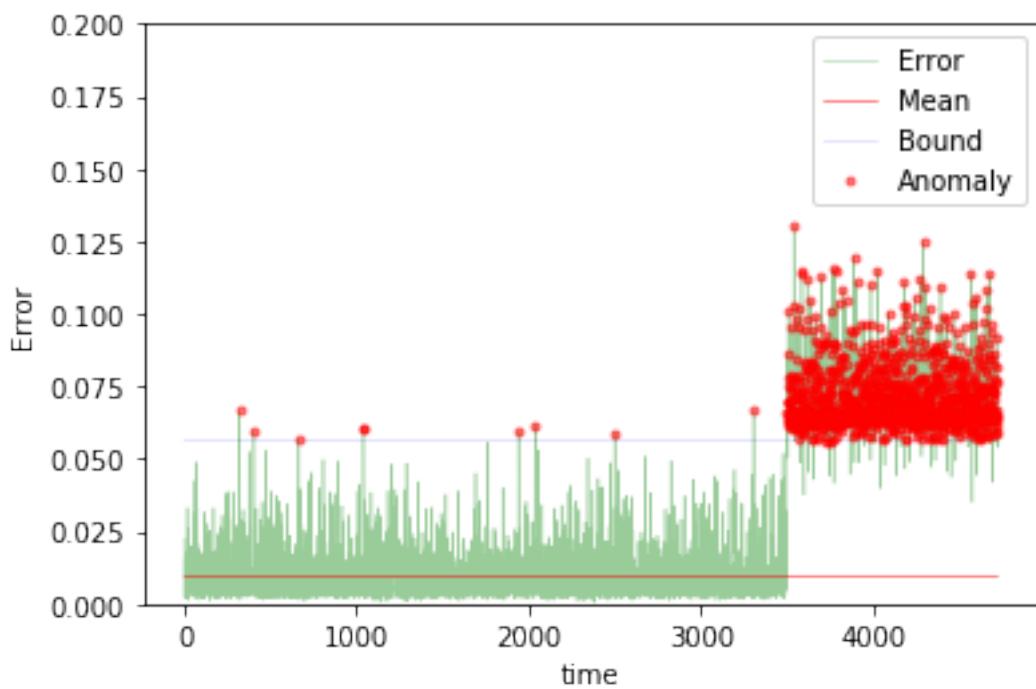
The mean error for gru2_5_normal_ is 0.009701978788339444 for length 4724
Testing on anomaly data.



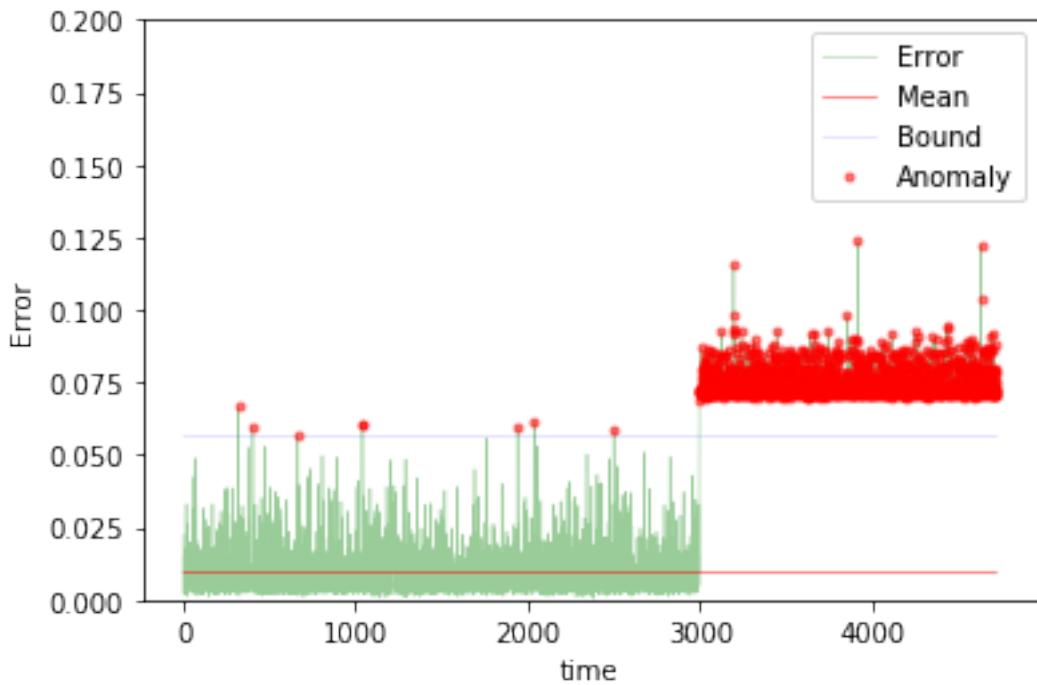
The mean error for gru2_5_anomaly_ is 0.01197314125335541 for length 4724
Testing on different app data.



The mean error for gru2_5_diff_app_ is 0.07082846543025689 for length 4724
Testing on App change synthetic data.



```
The mean error for gru2_5_app_change_ is 0.02546341419787154 for length 4724  
Testing on Net flood synthetic data.
```



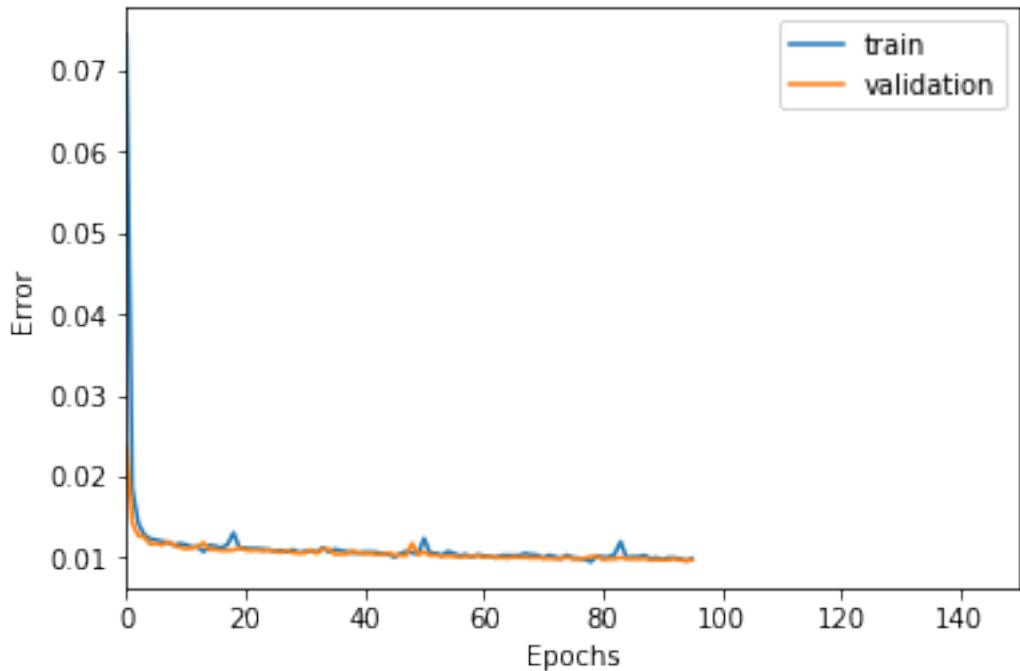
```
The mean error for gru2_5_net_flood_ is 0.033676633392838065 for length 4724  
=====
```

10 steps

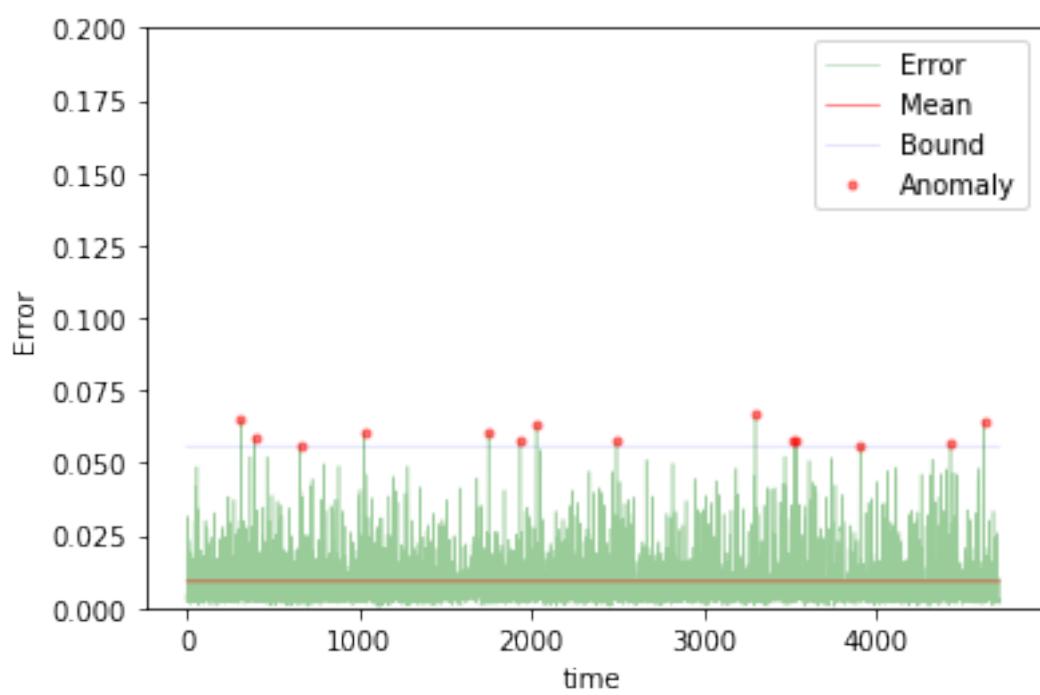
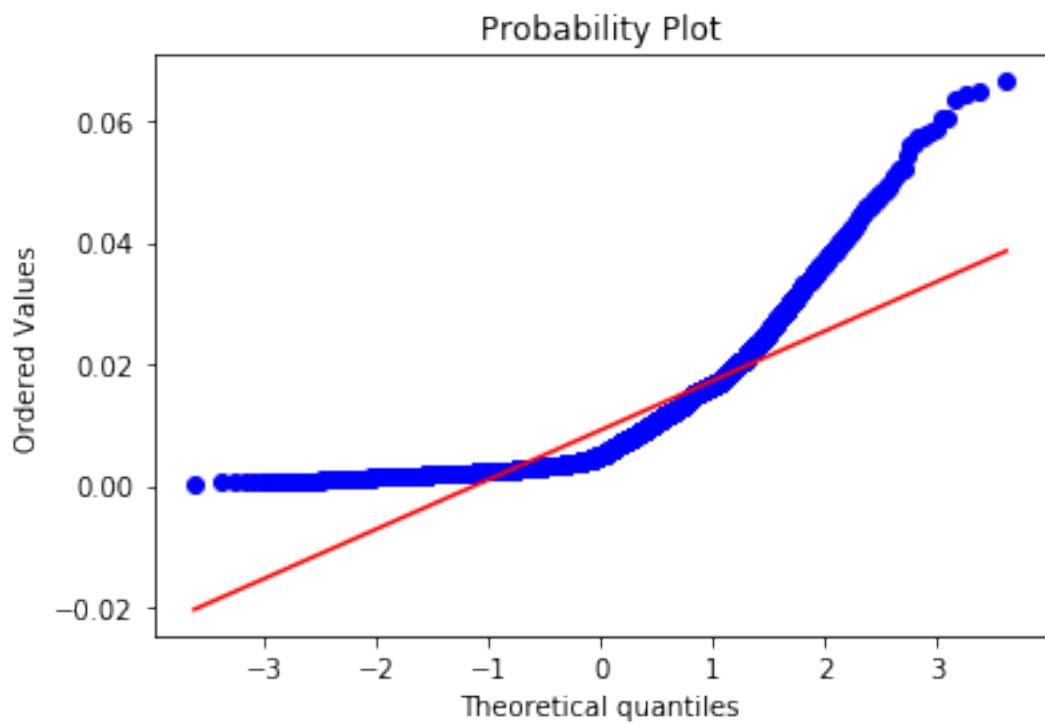
```
In [184]: TIMESTEPS = 10  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS, 0)  
vgen = flat_generator(val_X, TIMESTEPS, 0)  
name = "gru2_10"  
  
In [185]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(10, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [186]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

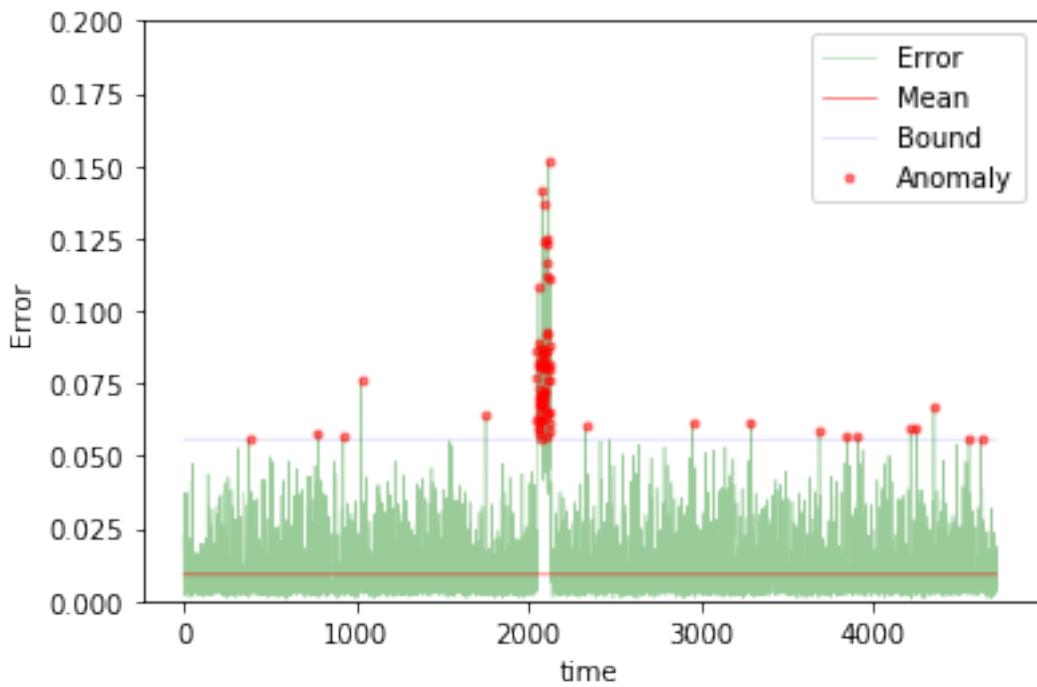
In [187]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



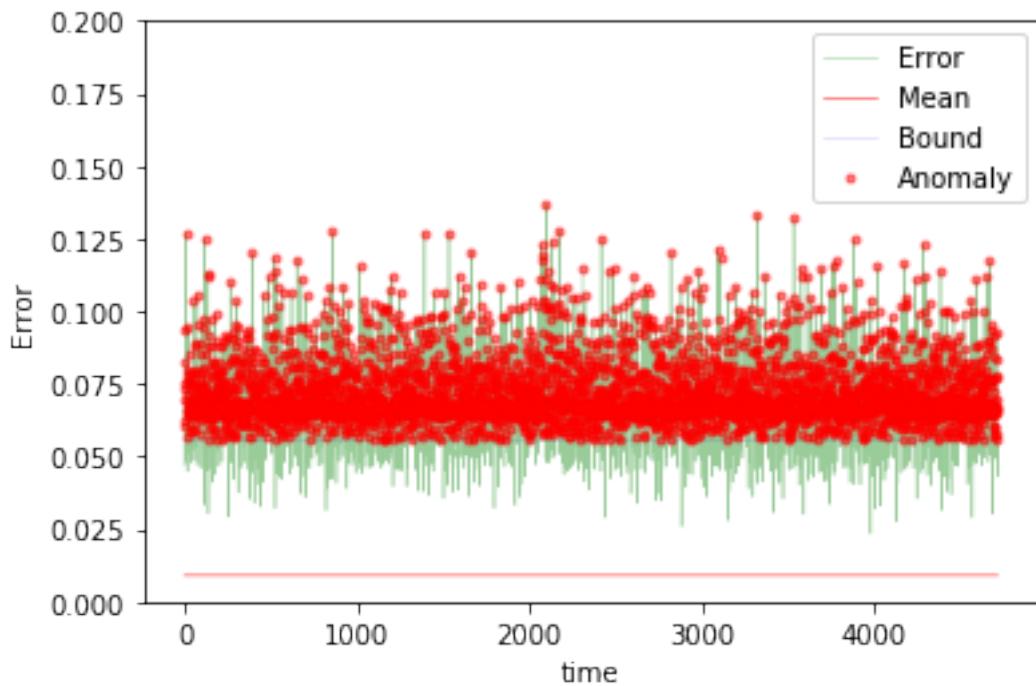
```
Training loss for final epoch is 0.009909274531761185
Validation loss for final epoch is 0.009752766541554592
----- Beginning tests for gru2_10 -----
Testing on normal data.
```



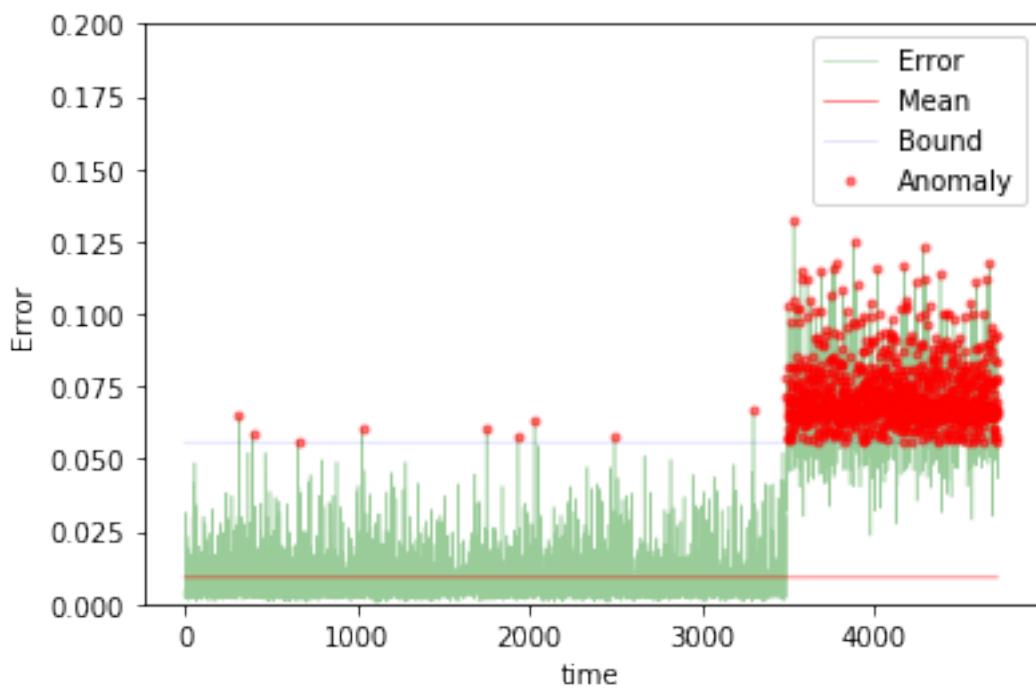
The mean error for gru2_10_normal_ is 0.009114843784203948 for length 4719
Testing on anomaly data.



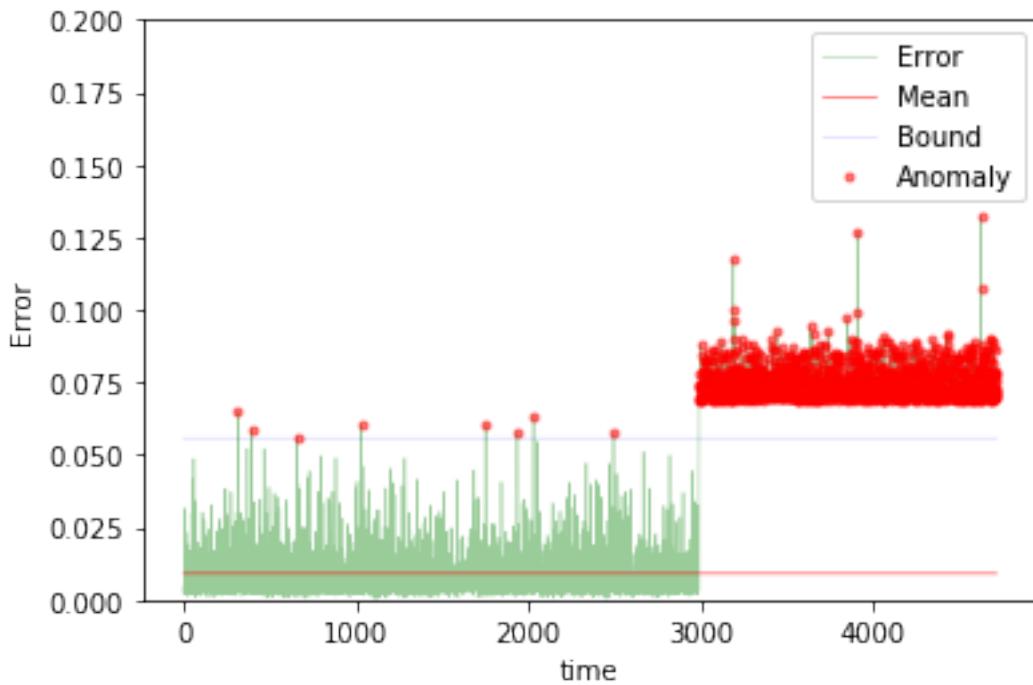
The mean error for gru2_10_anomaly_ is 0.011347613012379434 for length 4719
Testing on different app data.



The mean error for gru2_10_diff_app_ is 0.06901902862283589 for length 4719
Testing on App change synthetic data.



```
The mean error for gru2_10_app_change_ is 0.02456734822440574 for length 4719
Testing on Net flood synthetic data.
```



```
The mean error for gru2_10_net_flood_ is 0.03304917246386239 for length 4719
=====
=====
```

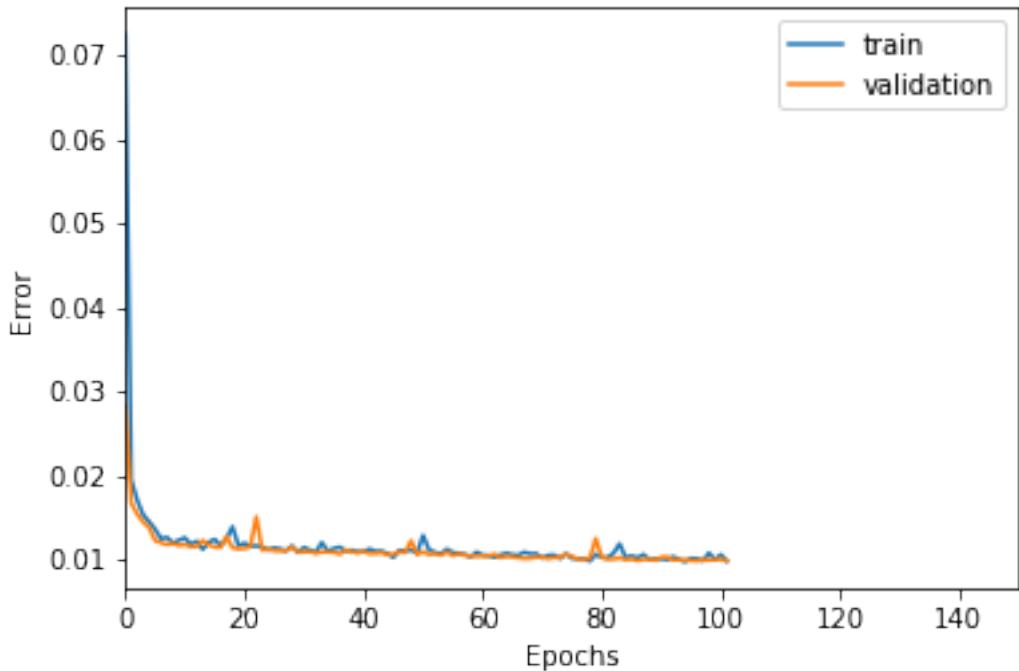
20 steps

```
In [188]: TIMESTEPS = 20
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru2_20"

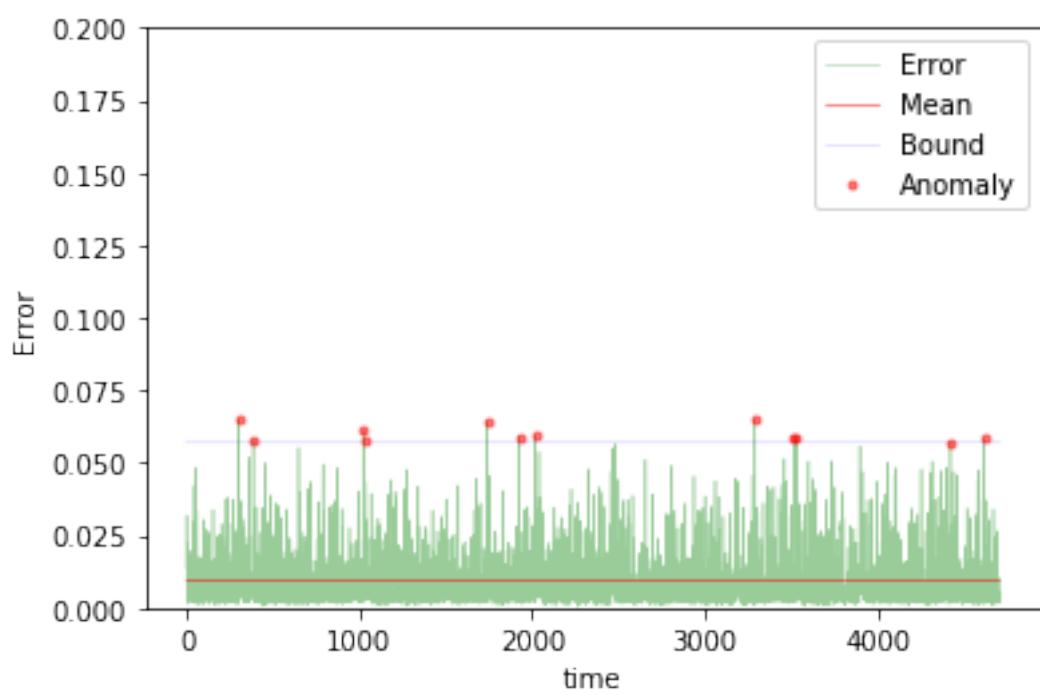
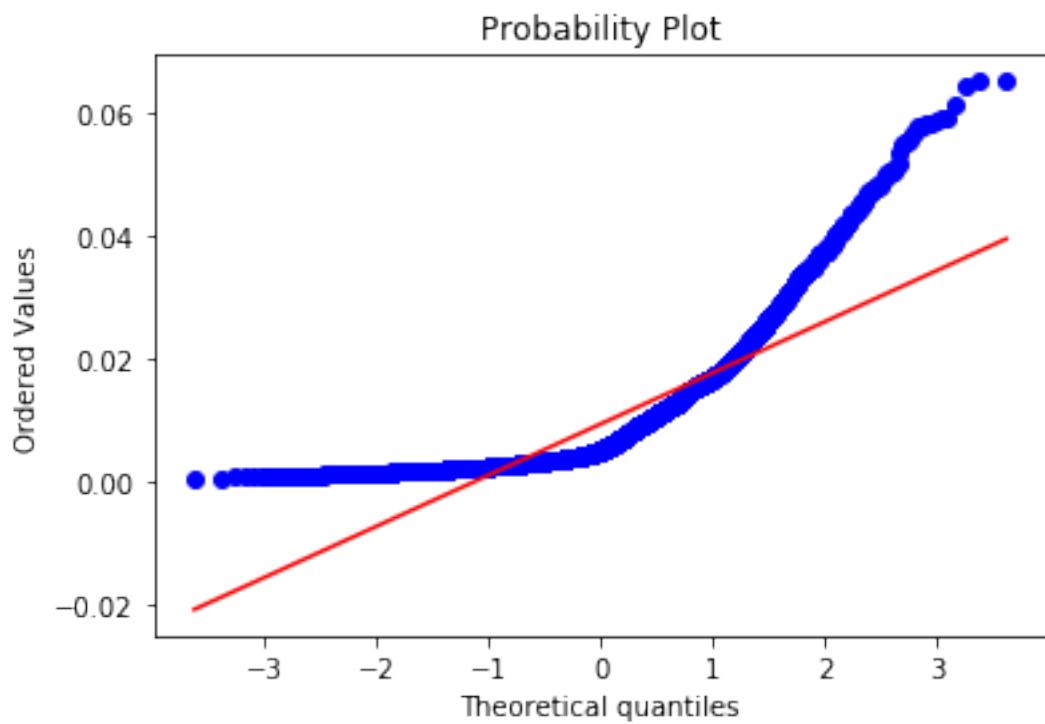
In [189]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [190]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

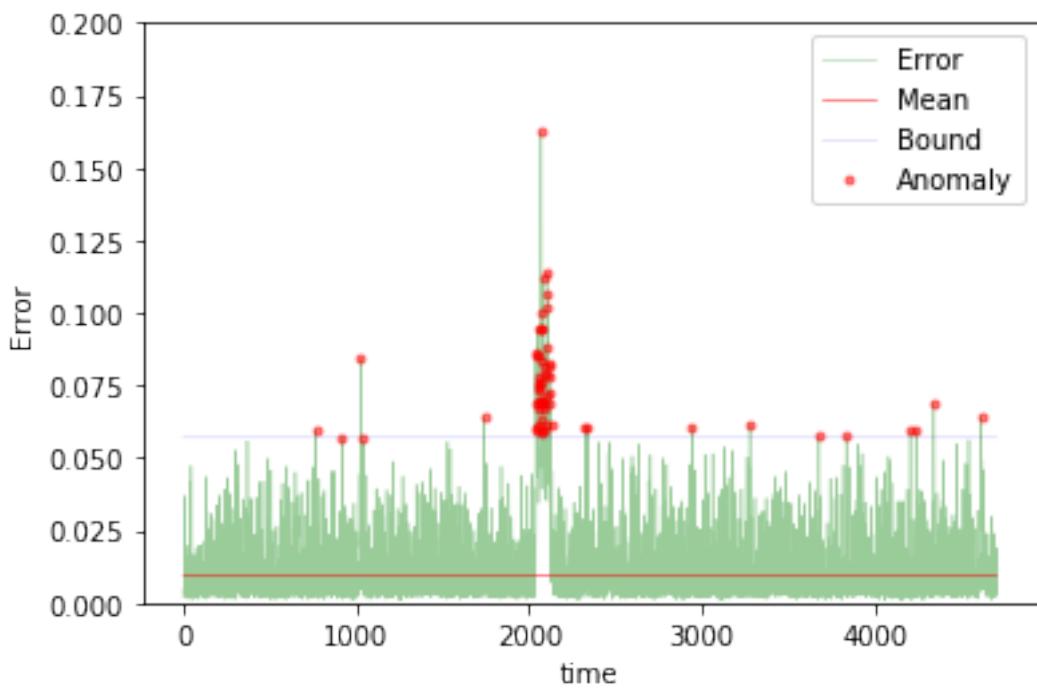
In [191]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



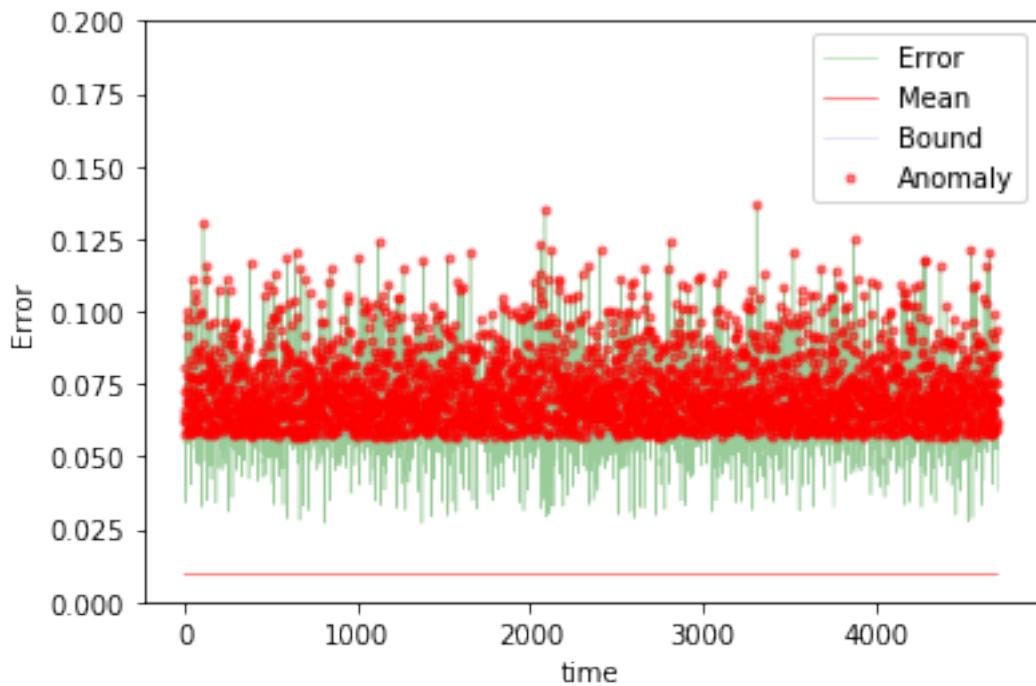
```
Training loss for final epoch is 0.009760482370620594
Validation loss for final epoch is 0.009830554115935228
----- Beginning tests for gru2_20 -----
Testing on normal data.
```



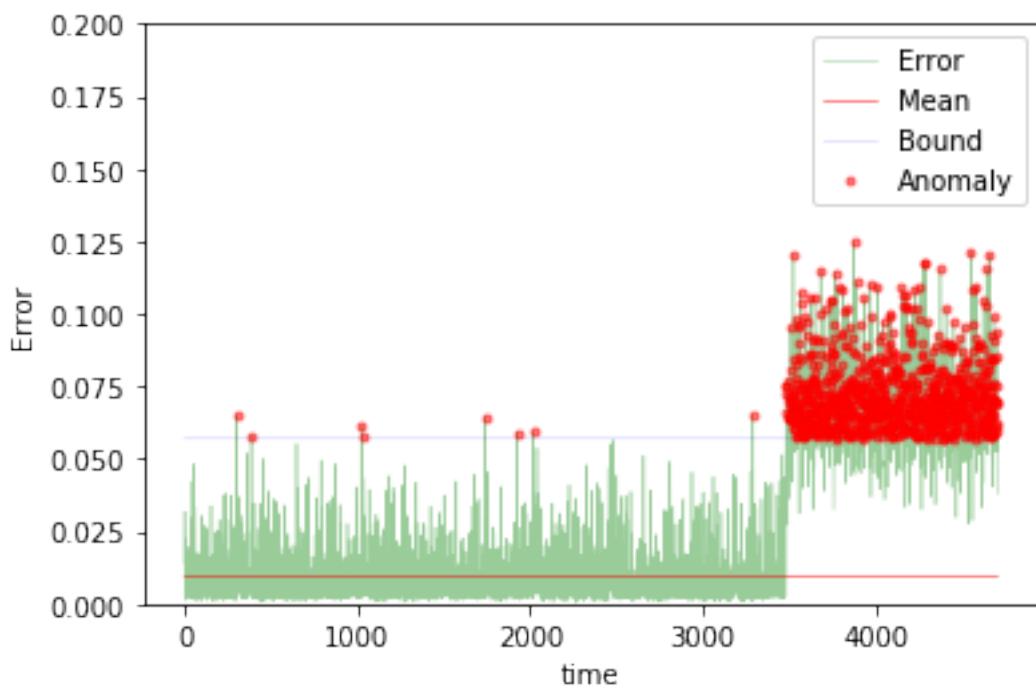
The mean error for gru2_20_normal_ is 0.009304953514968415 for length 4709
Testing on anomaly data.



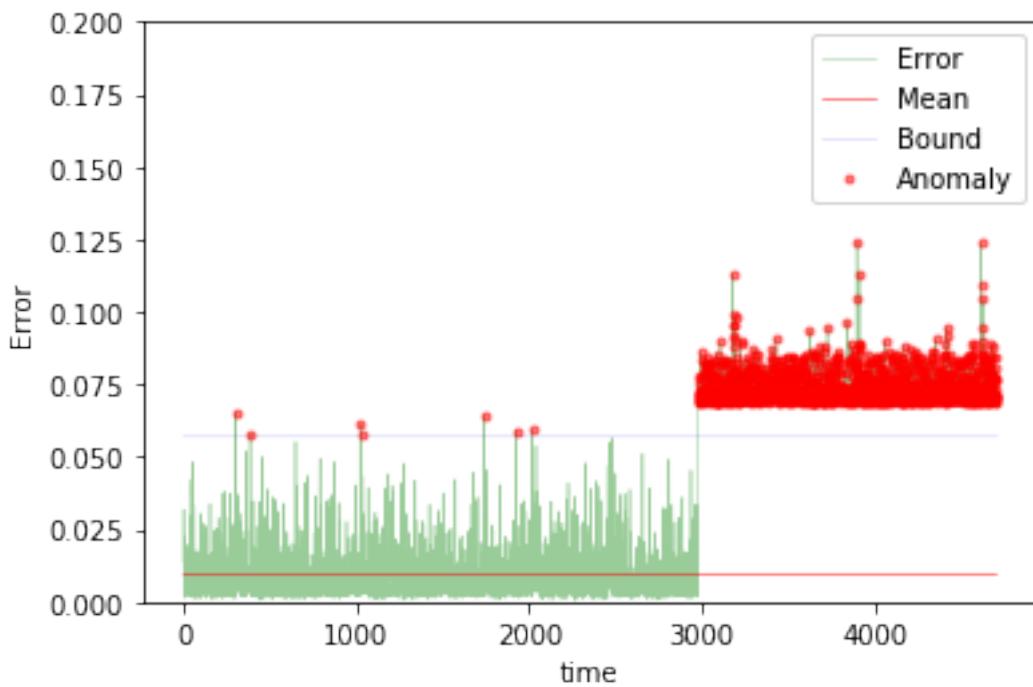
The mean error for gru2_20_anomaly_ is 0.011396194656773224 for length 4709
Testing on different app data.



The mean error for gru2_20_diff_app_ is 0.06683171400384218 for length 4709
Testing on App change synthetic data.



```
The mean error for gru2_20_app_change_ is 0.02421374972775486 for length 4709
Testing on Net flood synthetic data.
```



```
The mean error for gru2_20_net_flood_ is 0.03301289198327418 for length 4709
=====
=====
```

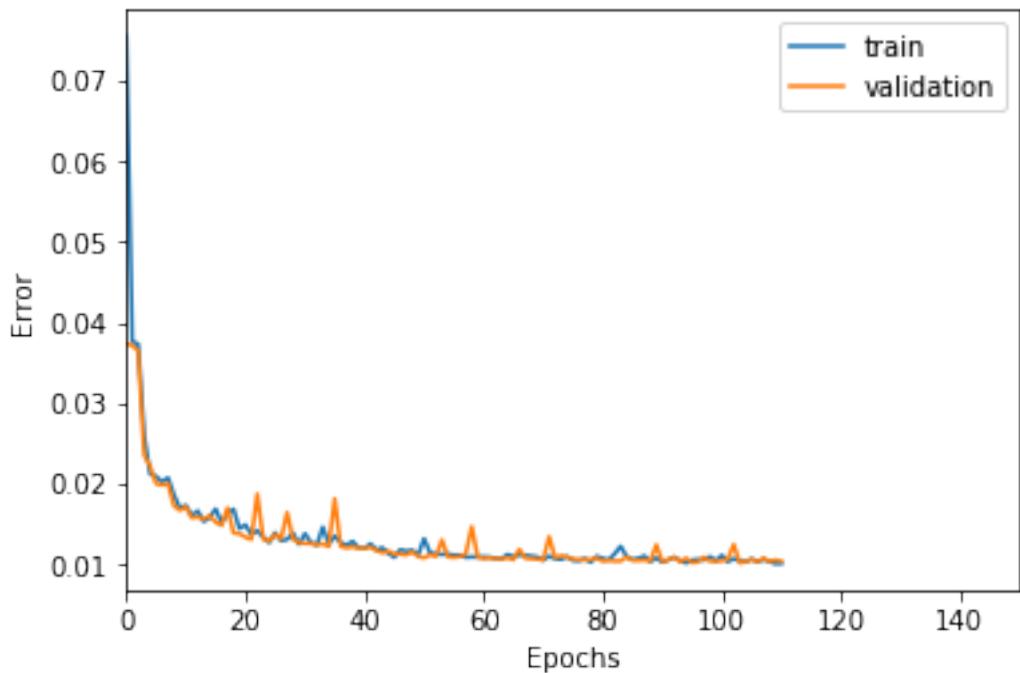
50 steps

```
In [192]: TIMESTEPS = 50
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru2_50"

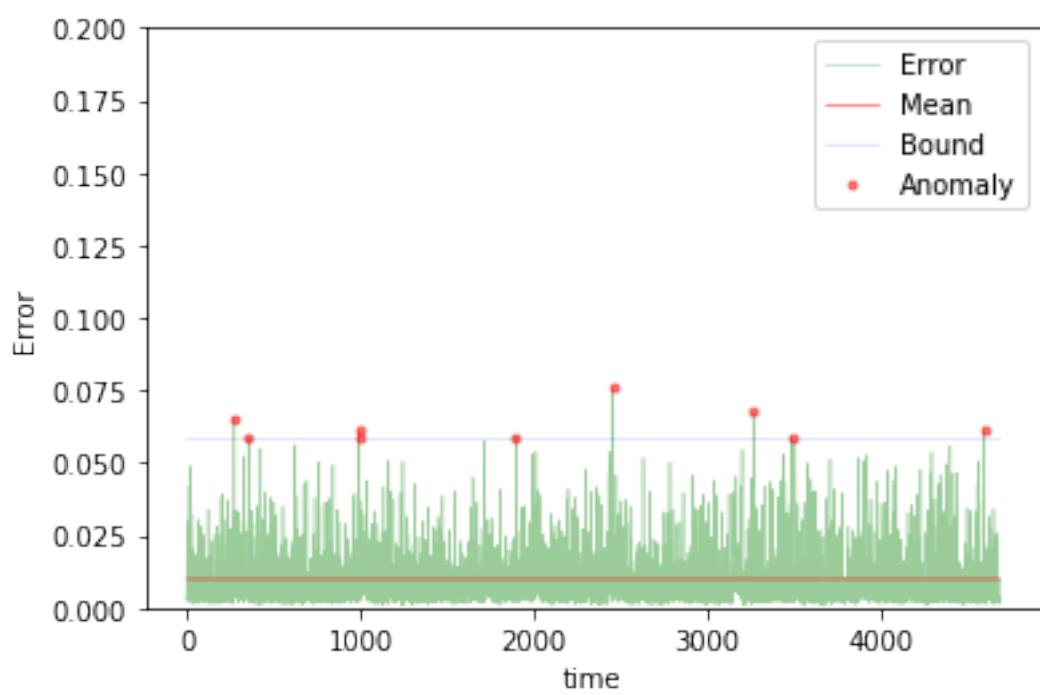
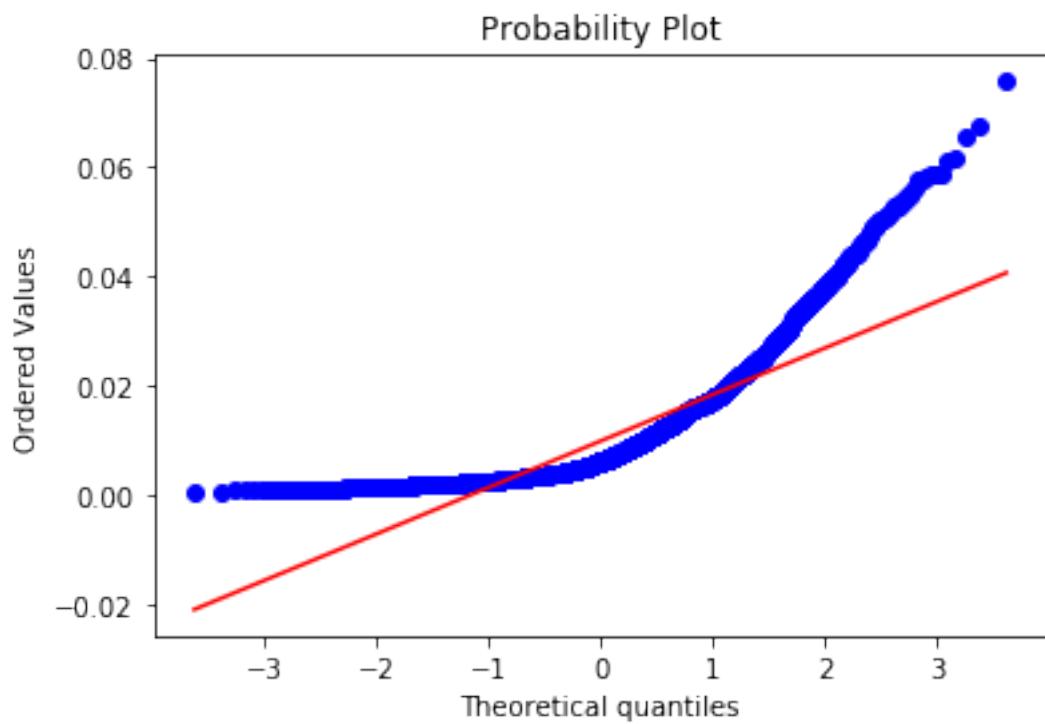
In [193]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [194]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

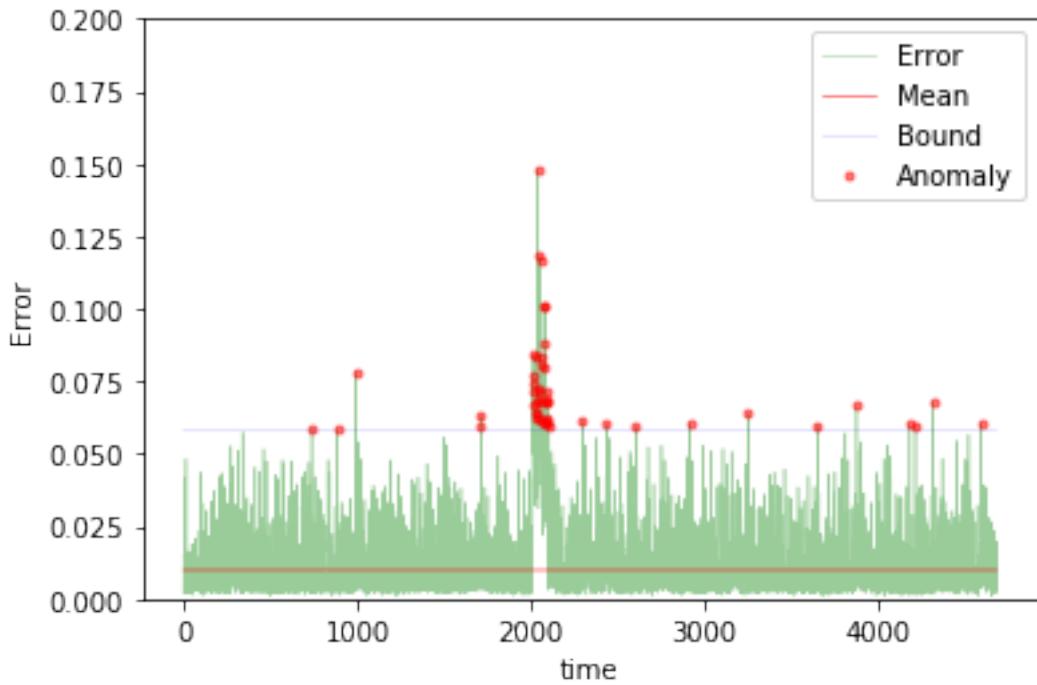
In [195]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



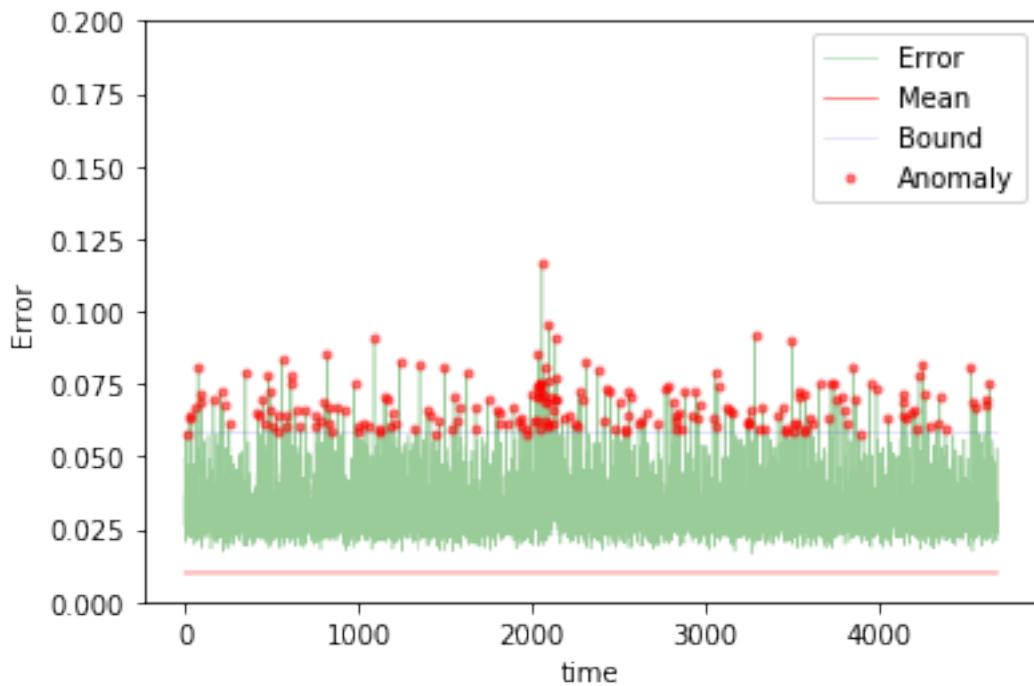
```
Training loss for final epoch is 0.010180886655114591
Validation loss for final epoch is 0.010442171080270781
----- Beginning tests for gru2_50 -----
Testing on normal data.
```



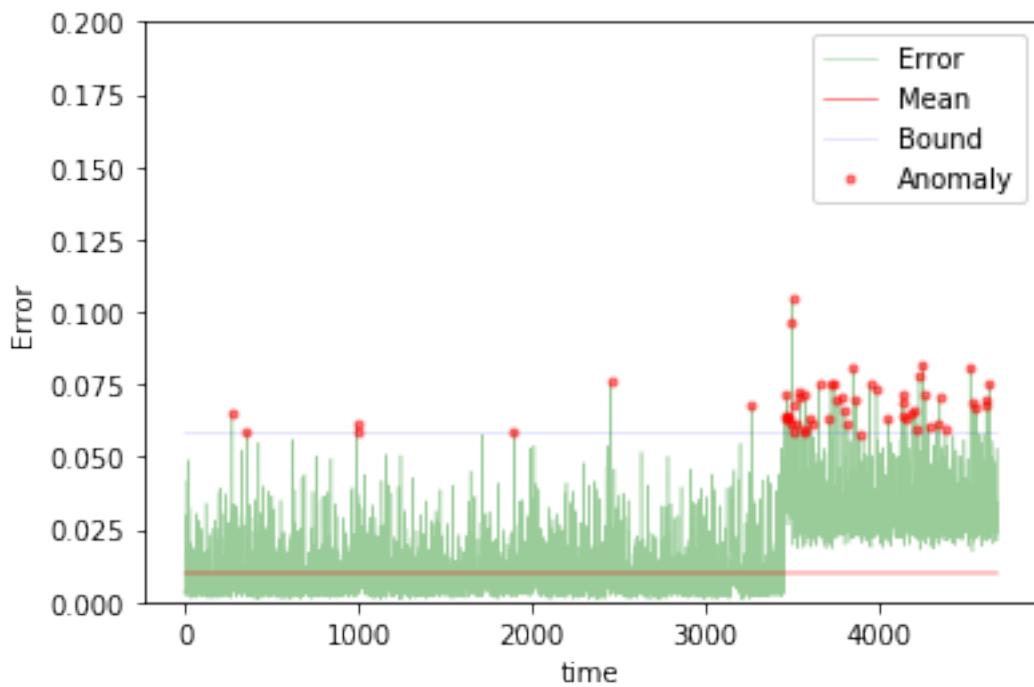
The mean error for gru2_50_normal_ is 0.009960870409586741 for length 4679
Testing on anomaly data.



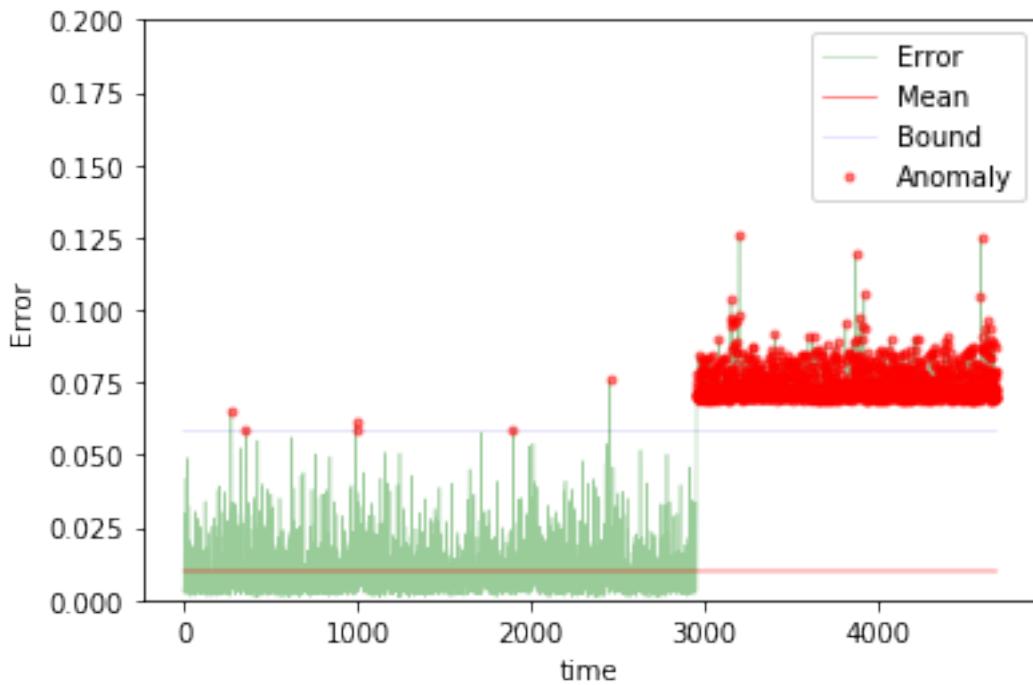
The mean error for gru2_50_anomaly_ is 0.011917344157904089 for length 4679
Testing on different app data.



The mean error for gru2_50_diff_app_ is 0.032547737952559115 for length 4679
Testing on App change synthetic data.



```
The mean error for gru2_50_app_change_ is 0.01593518848023431 for length 4679
Testing on Net flood synthetic data.
```



```
The mean error for gru2_50_net_flood_ is 0.033622448863746095 for length 4679
=====
=====
```

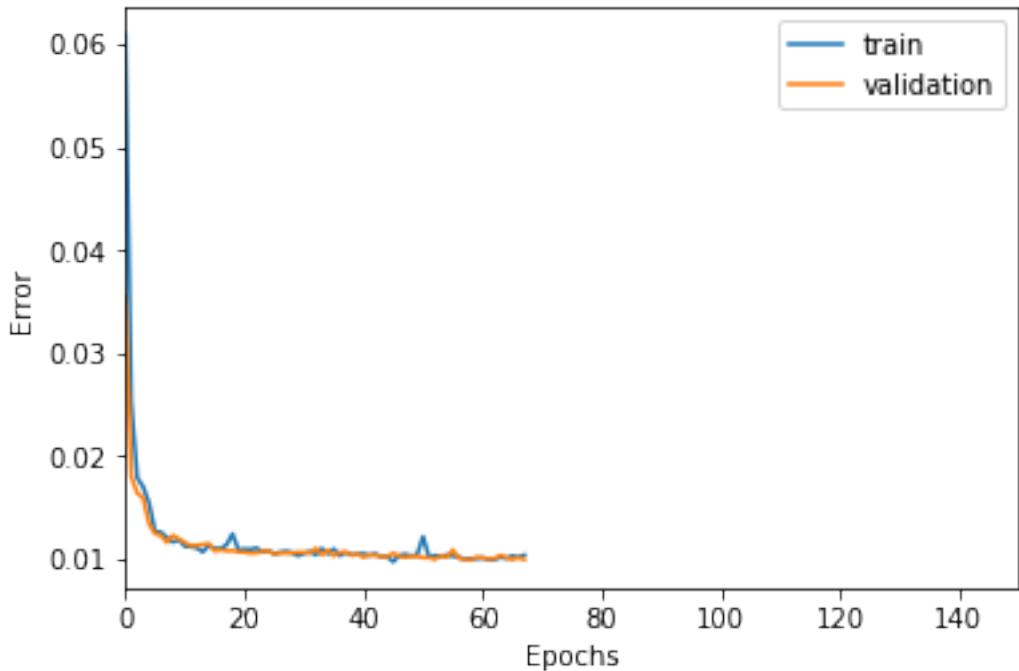
100 steps

```
In [196]: TIMESTEPS = 100
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru2_100"

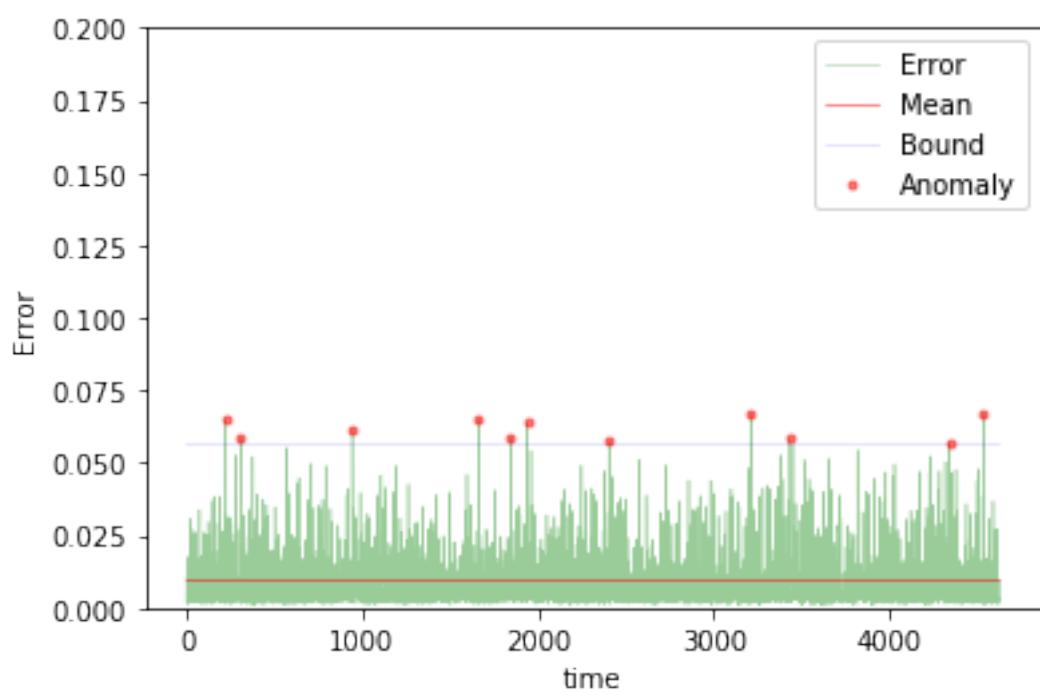
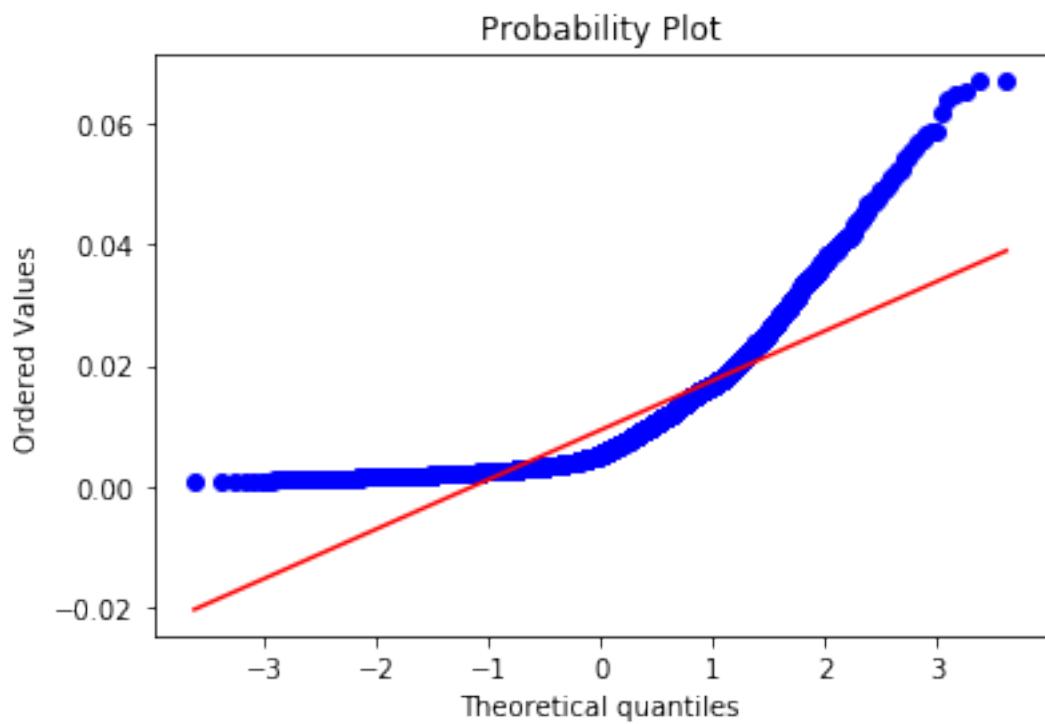
In [197]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [198]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

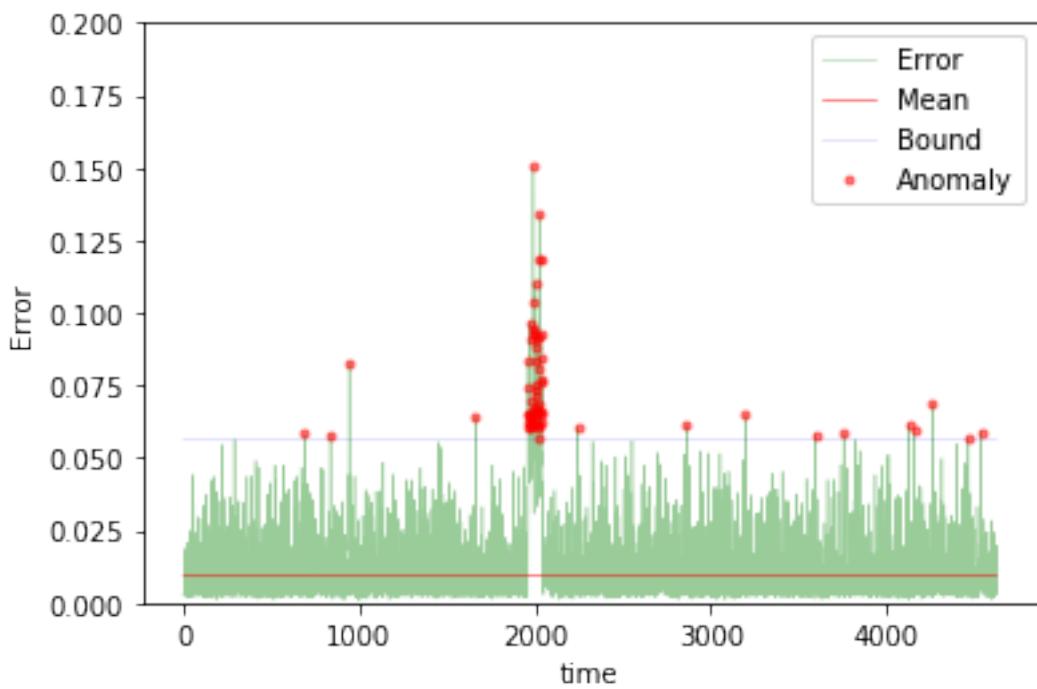
In [199]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



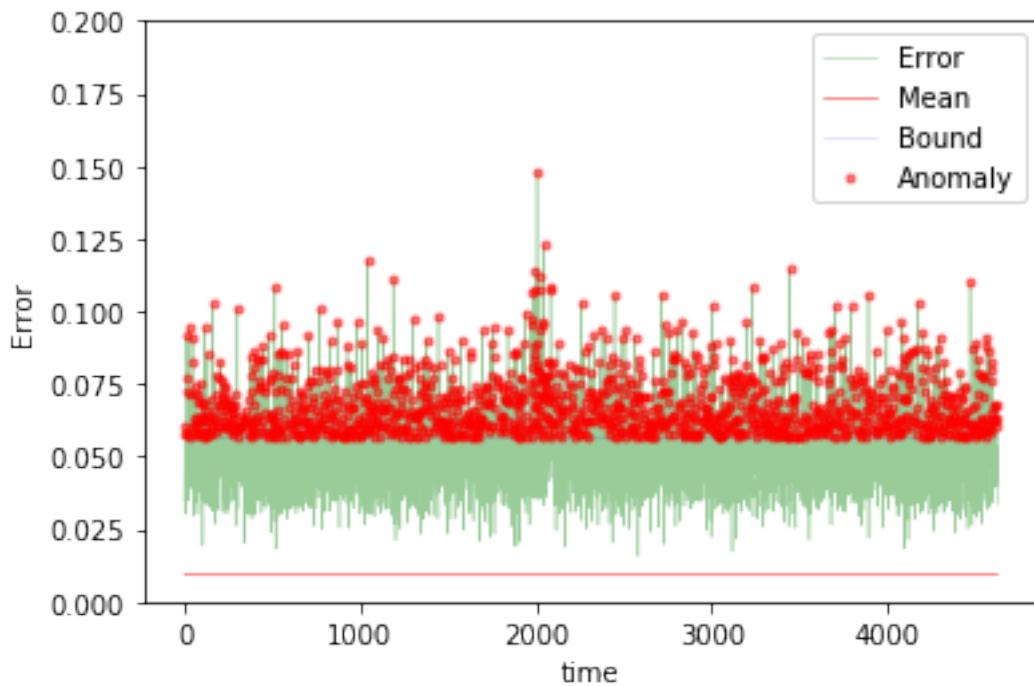
```
Training loss for final epoch is 0.01041349656577222
Validation loss for final epoch is 0.009982876165304333
----- Beginning tests for gru2_100 -----
Testing on normal data.
```



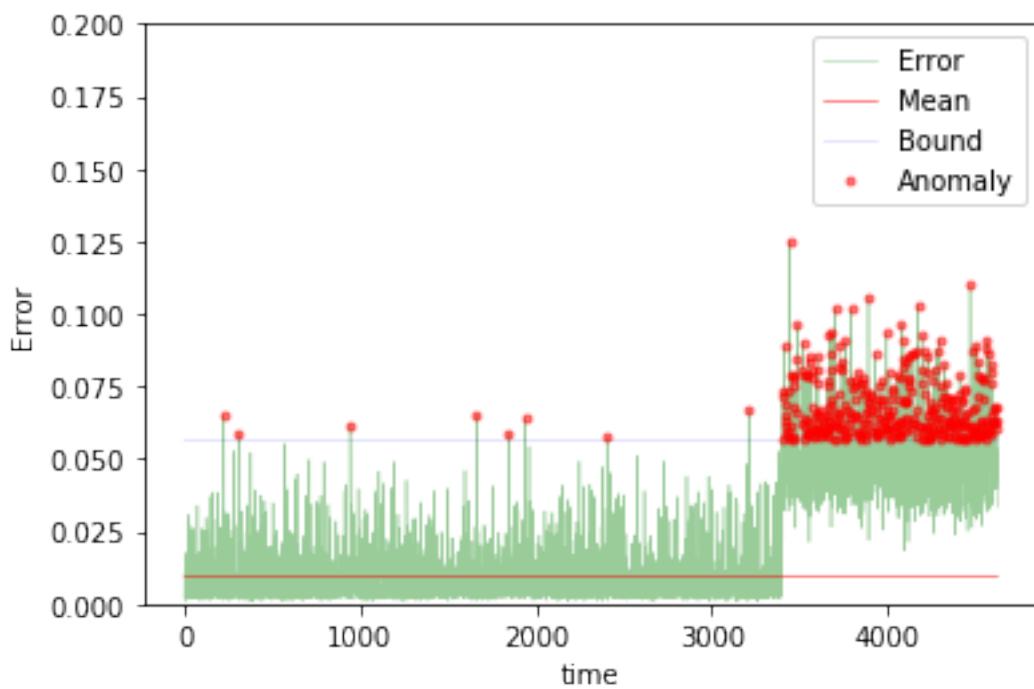
The mean error for gru2_100_normal_ is 0.009262515674569208 for length 4629
Testing on anomaly data.



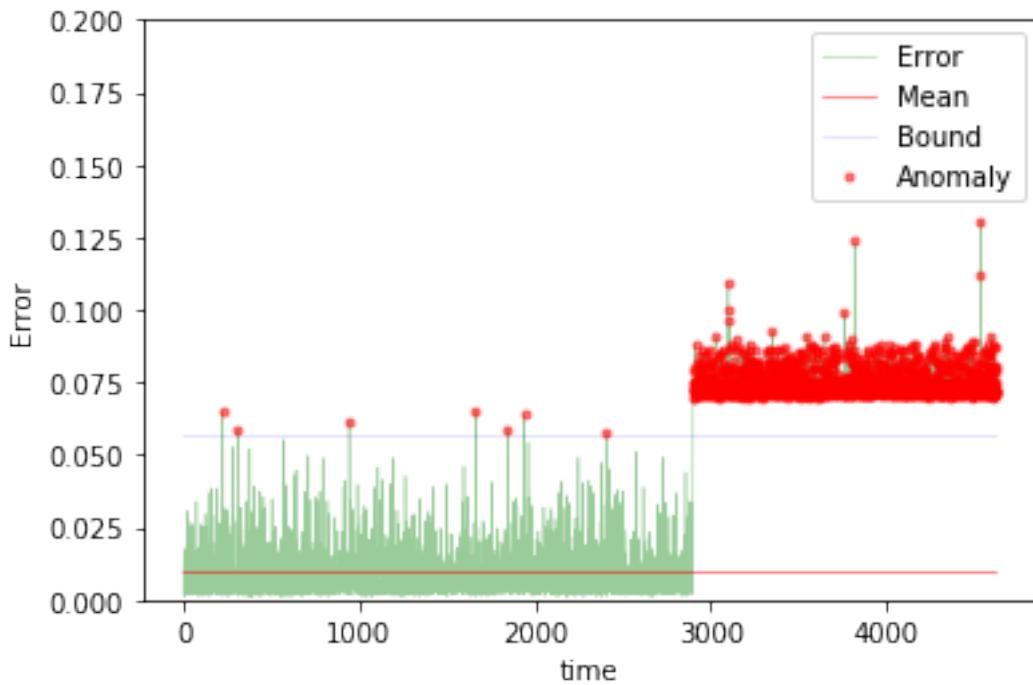
The mean error for gru2_100_anomaly_ is 0.01150897207799301 for length 4629
Testing on different app data.



The mean error for gru2_100_diff_app_ is 0.051726490313273005 for length 4629
Testing on App change synthetic data.



The mean error for gru2_100_app_change_ is 0.020358500221251803 for length 4629
Testing on Net flood synthetic data.



The mean error for gru2_100_net_flood_ is 0.03378854494726383 for length 4629
=====

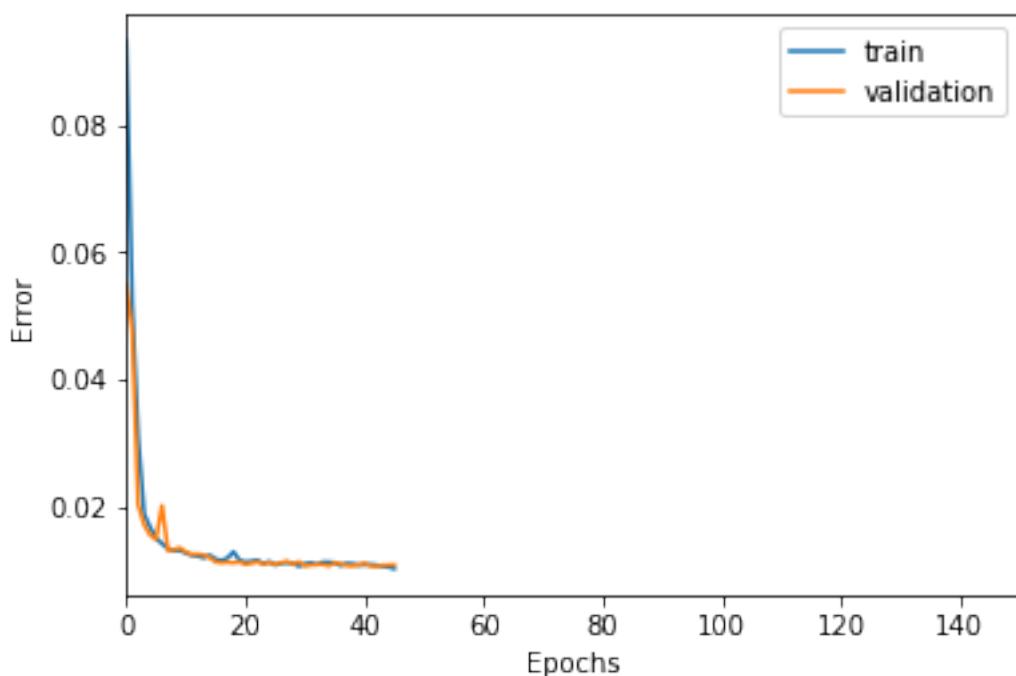
200 steps

```
In [200]: TIMESTEPS = 200
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru2_200"

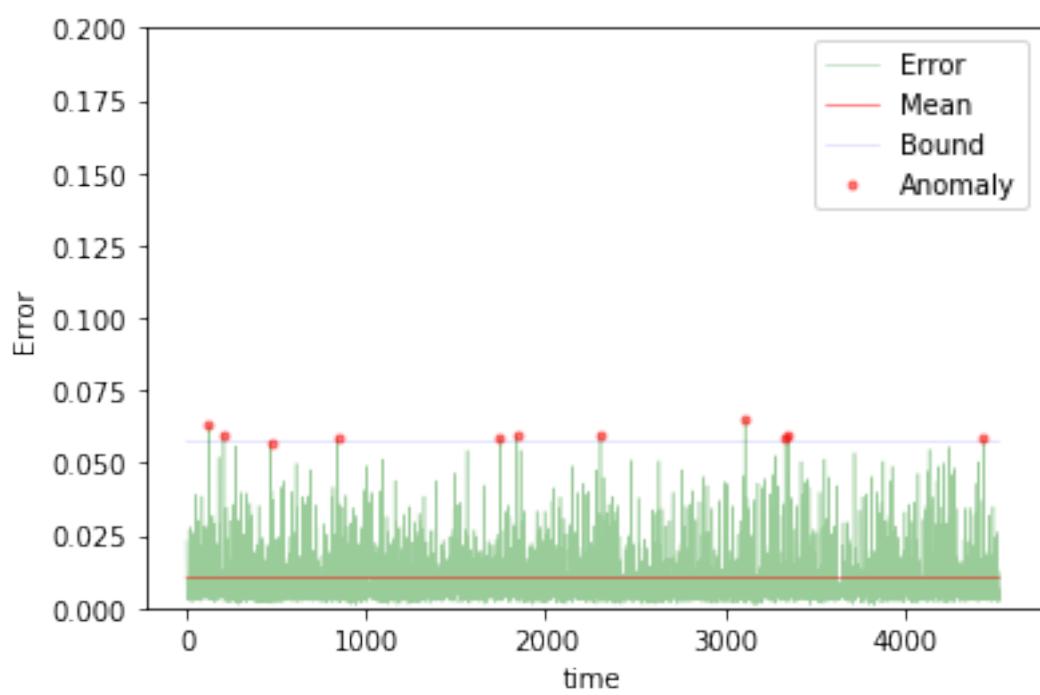
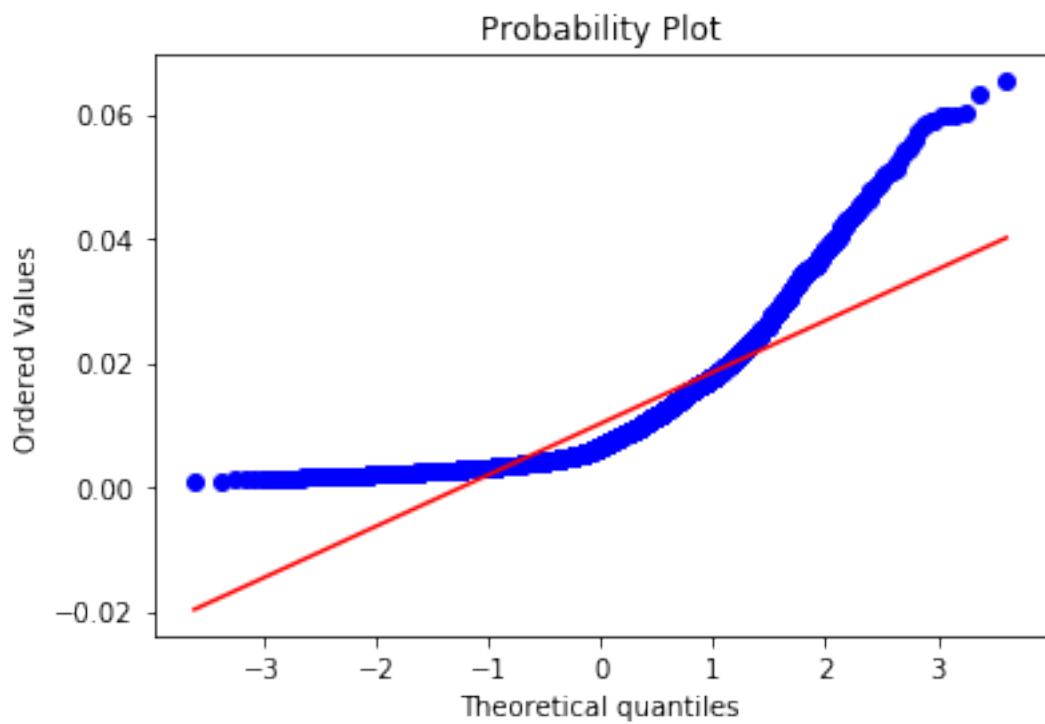
In [201]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [202]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

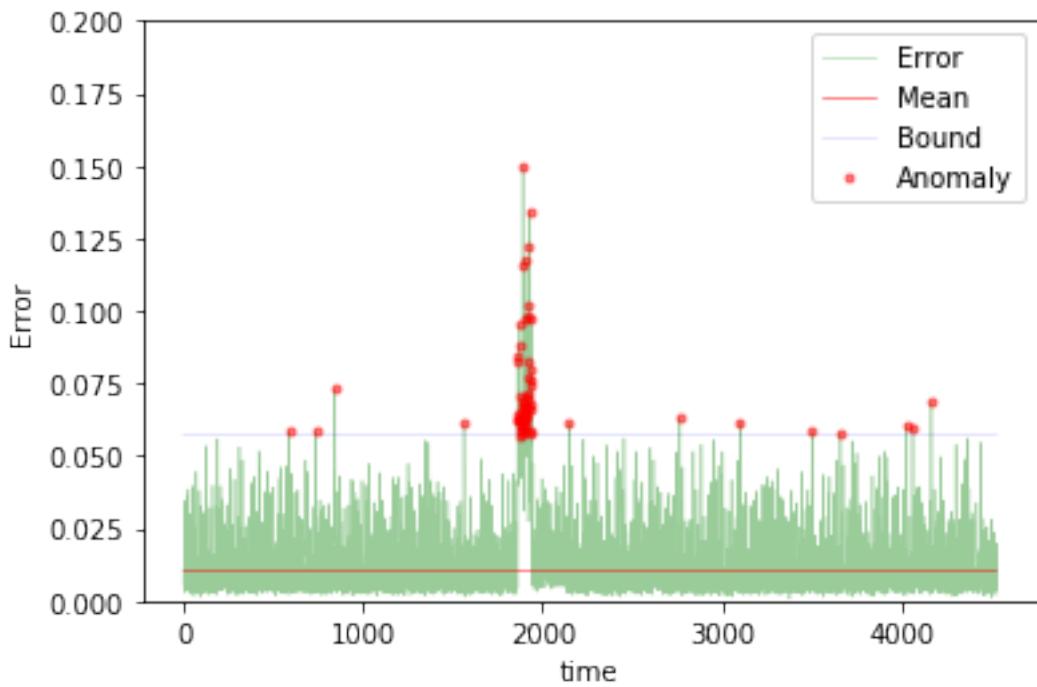
```
In [203]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



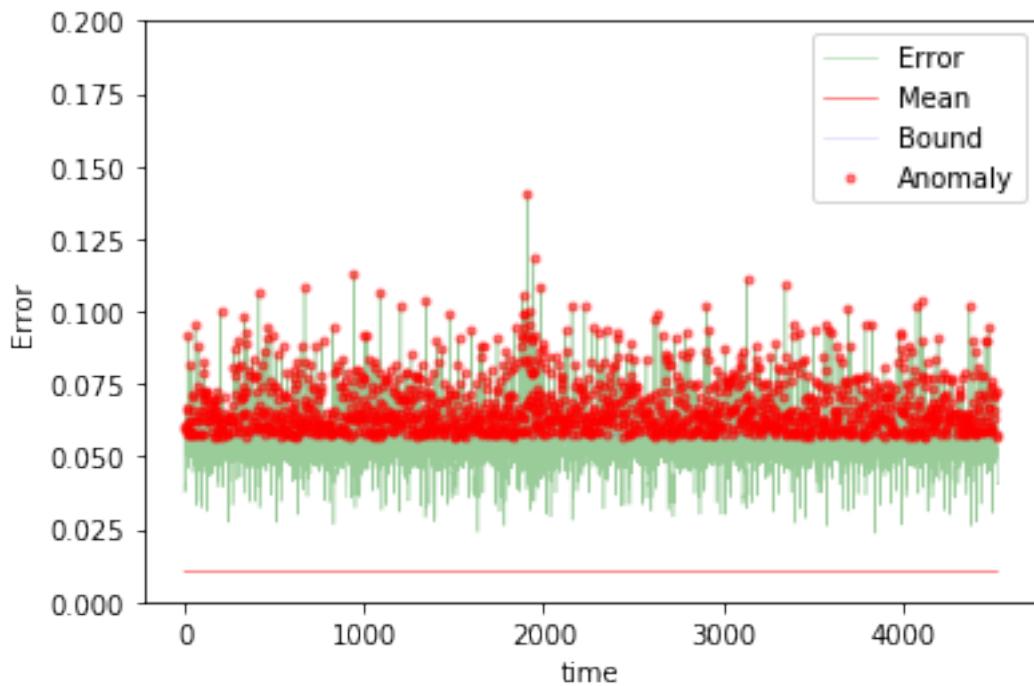
```
Training loss for final epoch is 0.01021584390150383
Validation loss for final epoch is 0.010816389935091137
----- Beginning tests for gru2_200 -----
Testing on normal data.
```



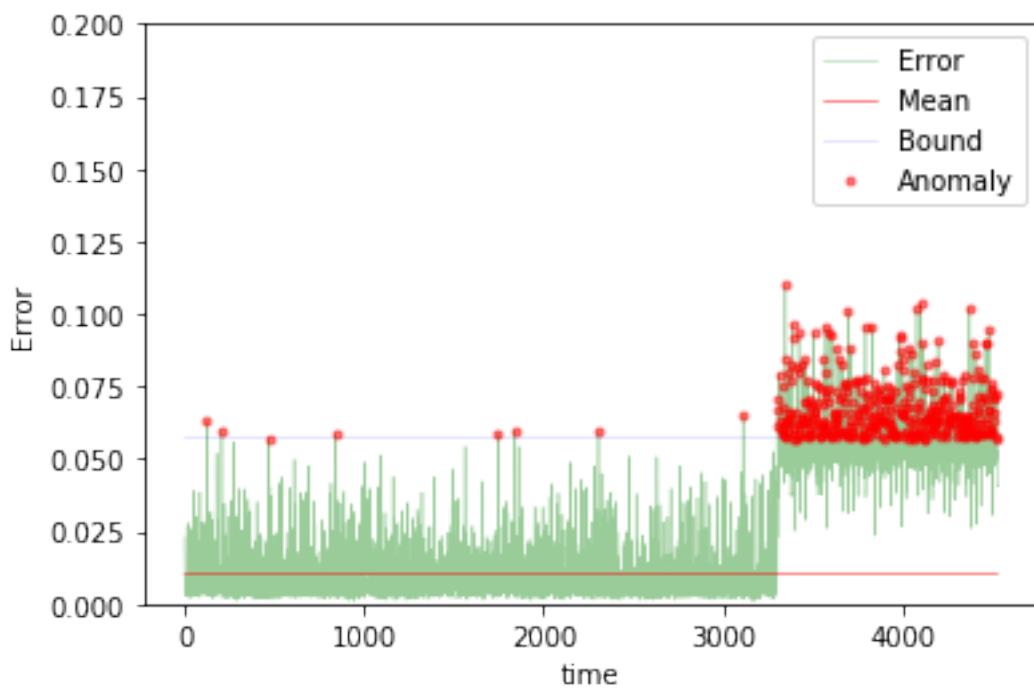
The mean error for gru2_200_normal_ is 0.010165218527114906 for length 4529
Testing on anomaly data.



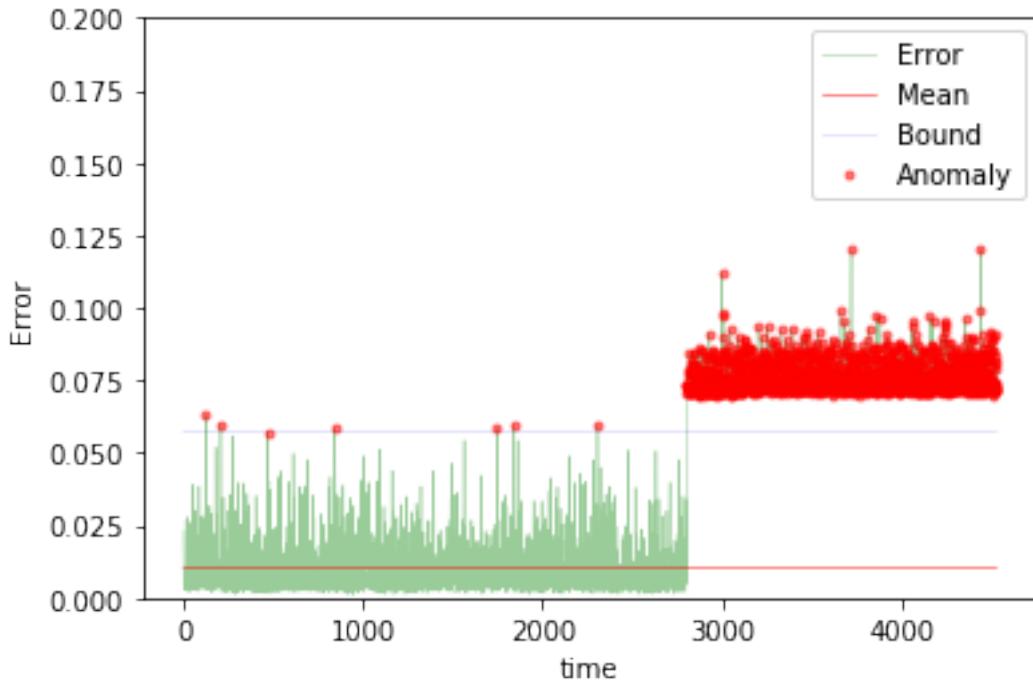
The mean error for gru2_200_anomaly_ is 0.012504780044425765 for length 4529
Testing on different app data.



The mean error for gru2_200_diff_app_ is 0.055712293578371715 for length 4529
Testing on App change synthetic data.



```
The mean error for gru2_200_app_change_ is 0.022383882290908333 for length 4529  
Testing on Net flood synthetic data.
```



```
The mean error for gru2_200_net_flood_ is 0.03537458963580255 for length 4529  
=====
```

2.1.7 RNN with 3 GRU layers

2 steps

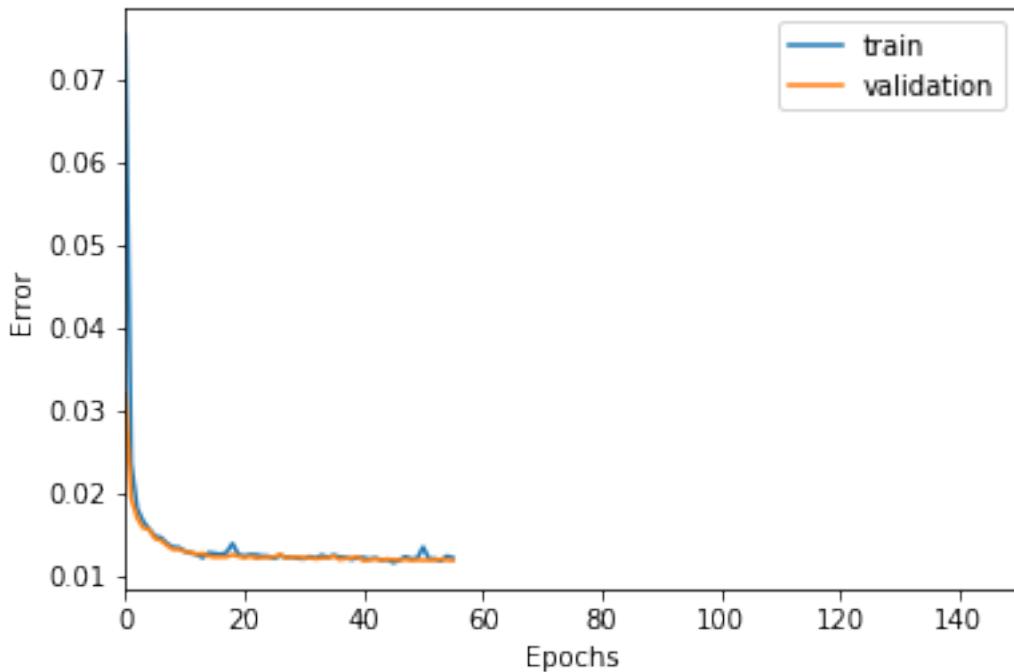
```
In [204]: TIMESTEPS = 2  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru3_2"
```

```
In [205]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
```

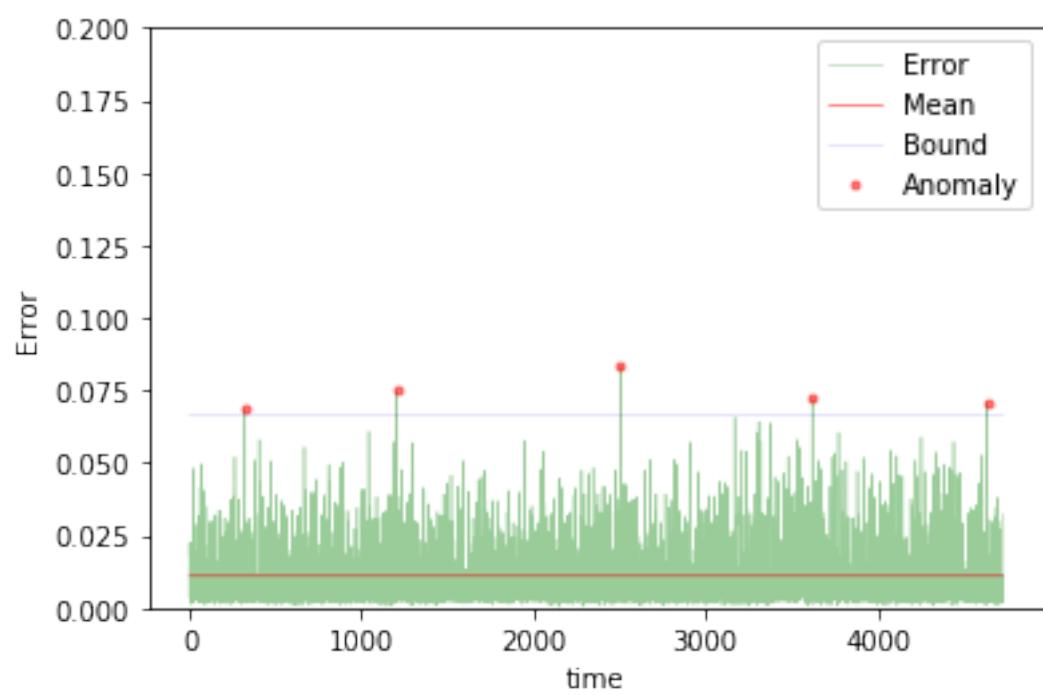
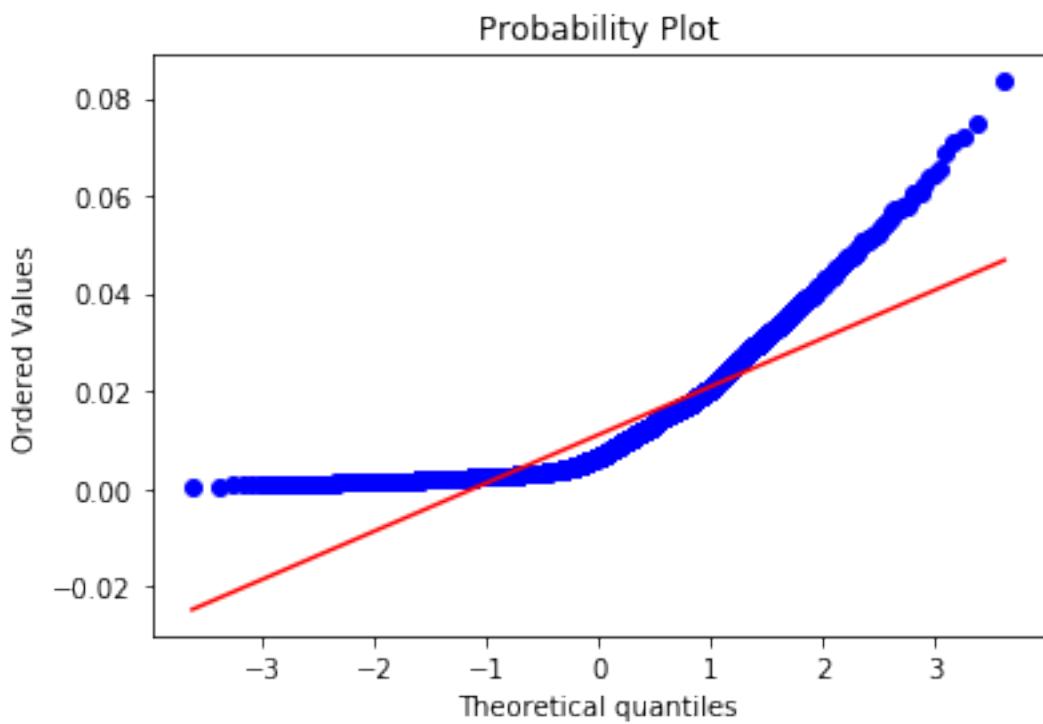
```
hidden = GRU(10, activation='relu', return_sequences=True)(hidden)
hidden = GRU(10, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [206]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

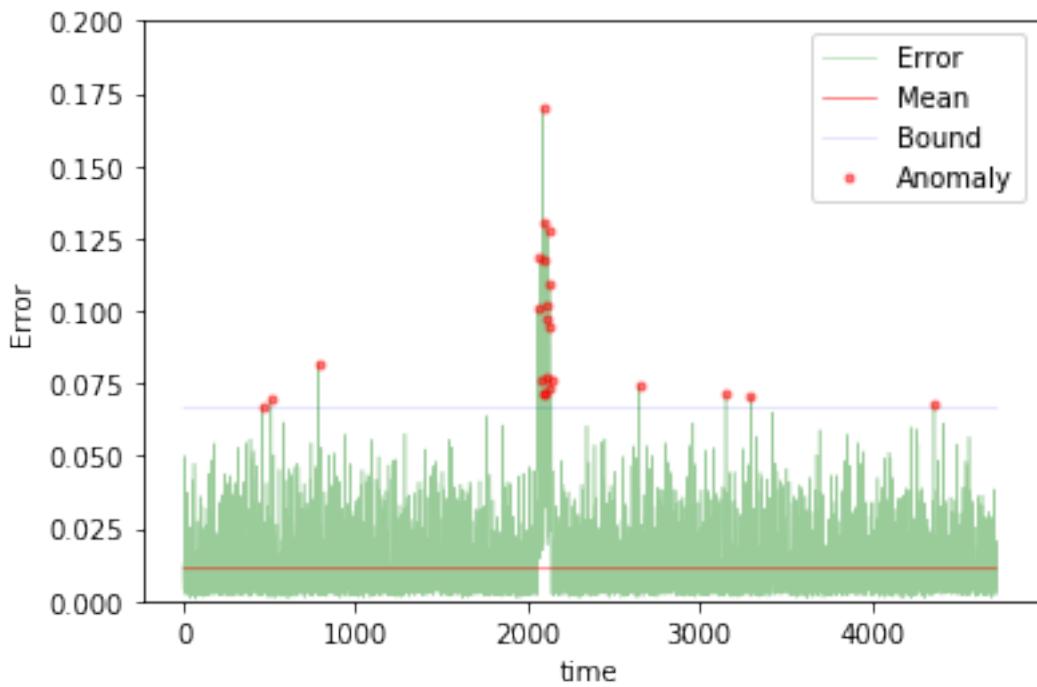
```
In [207]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



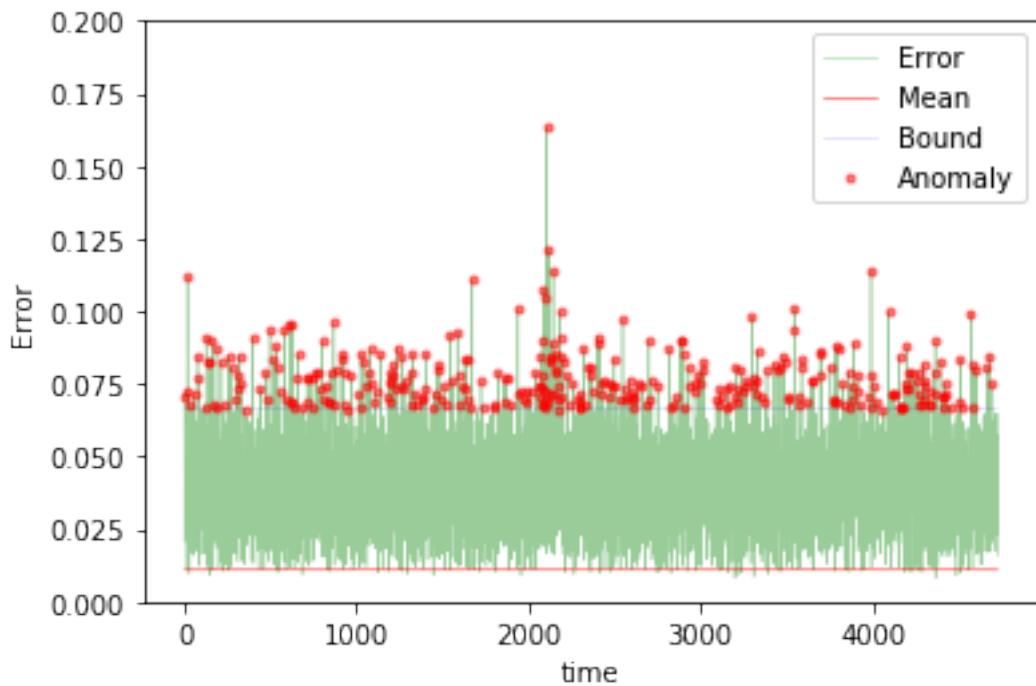
```
Training loss for final epoch is 0.01213966842647642
Validation loss for final epoch is 0.011815385169000365
----- Beginning tests for gru3_2 -----
Testing on normal data.
```



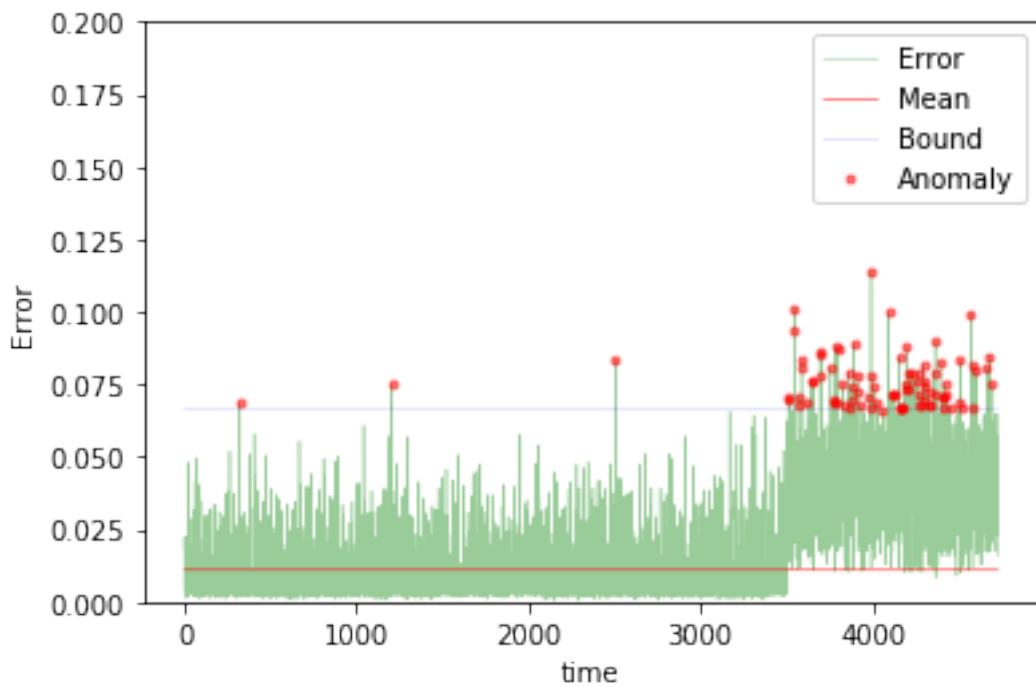
The mean error for gru3_2_normal_ is 0.011044816902553229 for length 4727
Testing on anomaly data.



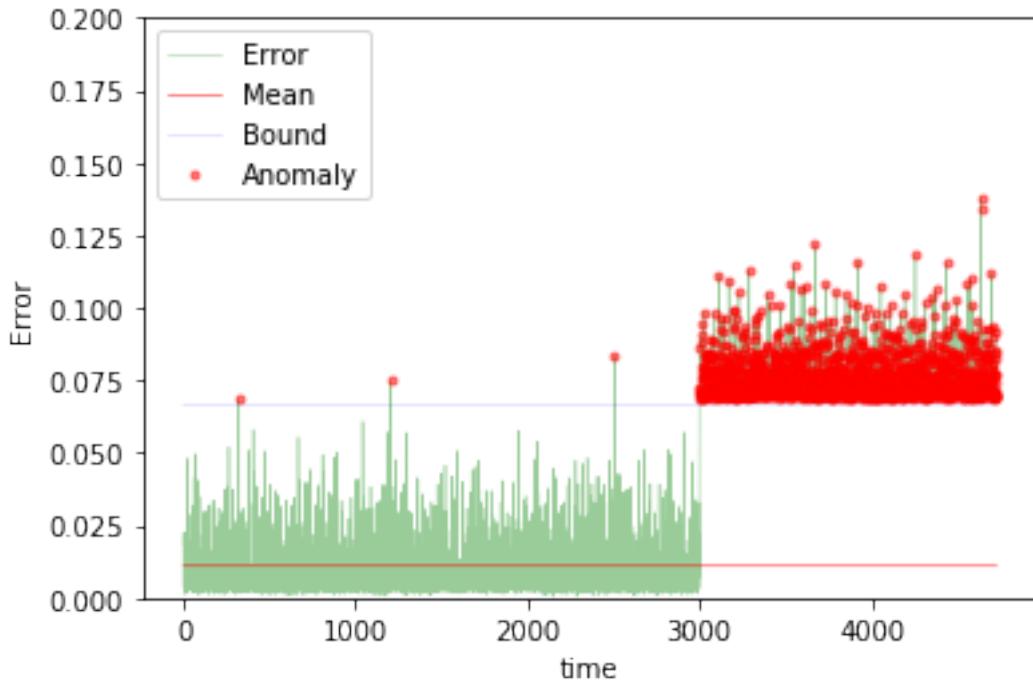
The mean error for gru3_2_anomaly_ is 0.012914008075283536 for length 4727
Testing on different app data.



The mean error for gru3_2_diff_app_ is 0.04032749563832749 for length 4727
Testing on App change synthetic data.



The mean error for gru3_2_app_change_ is 0.018584859249521166 for length 4727
Testing on Net flood synthetic data.



The mean error for gru3_2_net_flood_ is 0.03474057512970229 for length 4727
=====

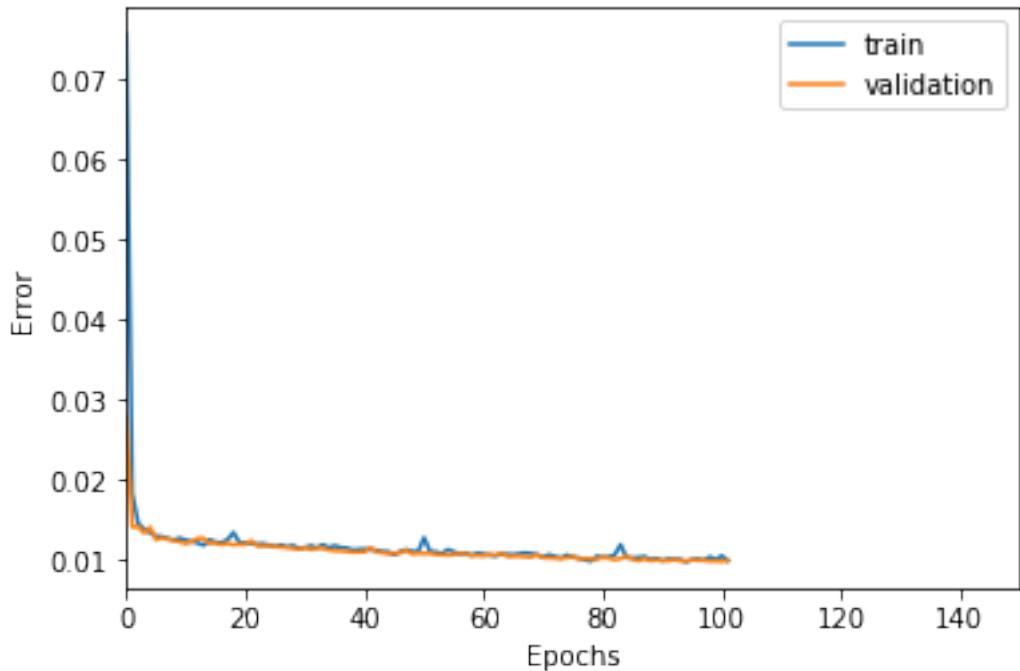
5 steps

```
In [208]: TIMESTEPS = 5
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS, 0)
          vgen = flat_generator(val_X, TIMESTEPS, 0)
          name = "gru3_5"

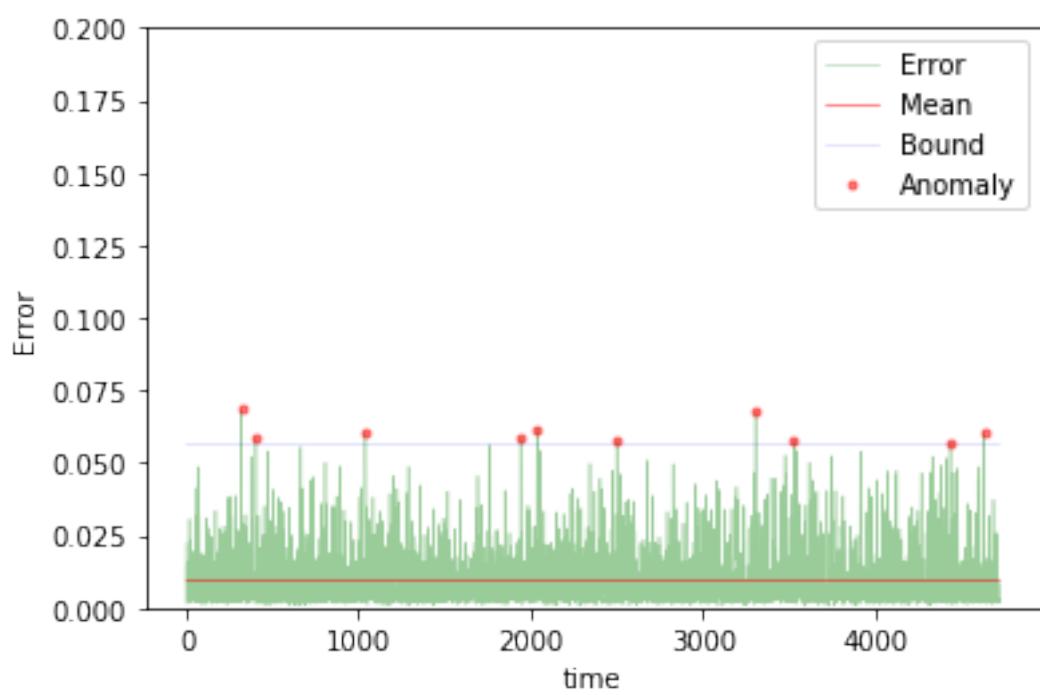
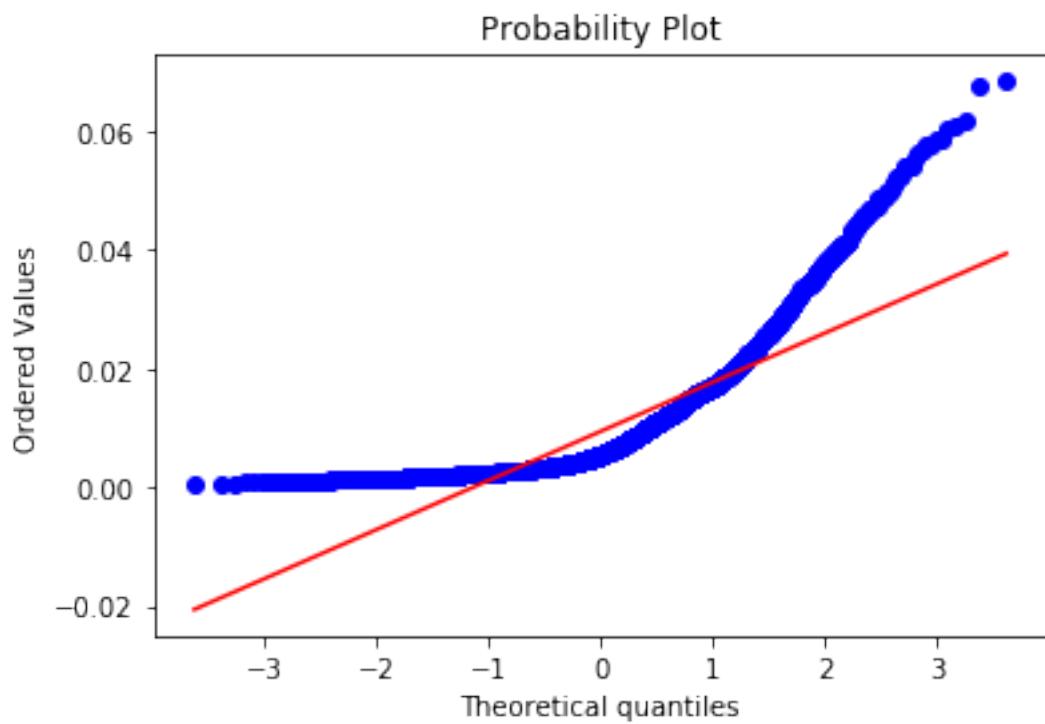
In [209]: input_layer = Input(shape=(TIMESTEPS,DIM))
          hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
          hidden = GRU(10, activation='relu', return_sequences=True)(hidden)
          hidden = GRU(10, activation='relu')(hidden)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [210]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

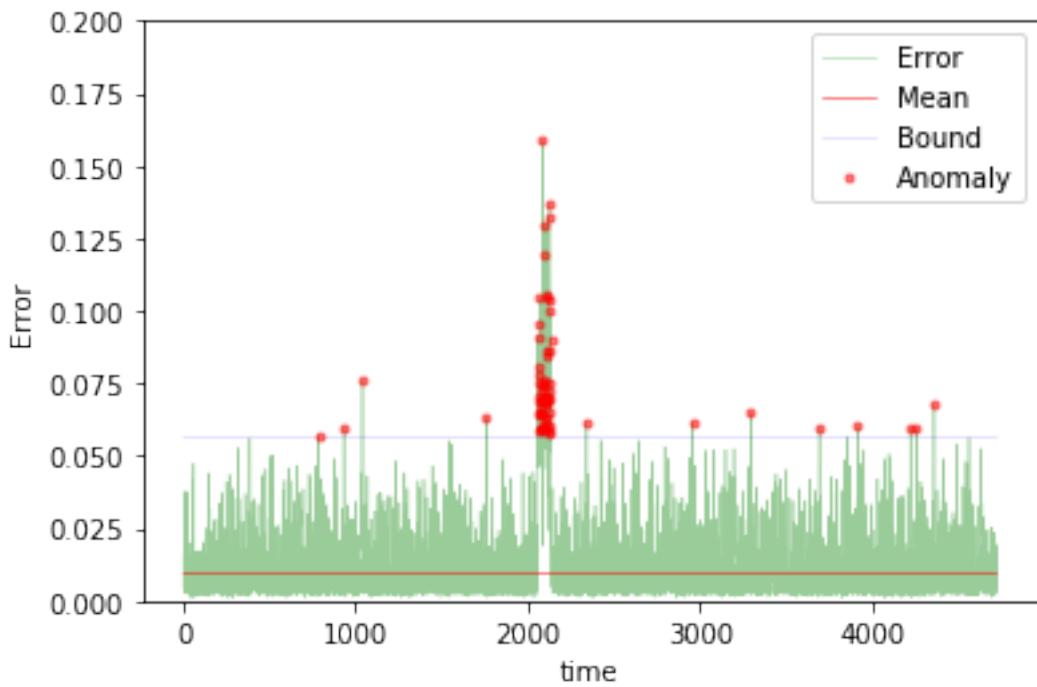
In [211]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



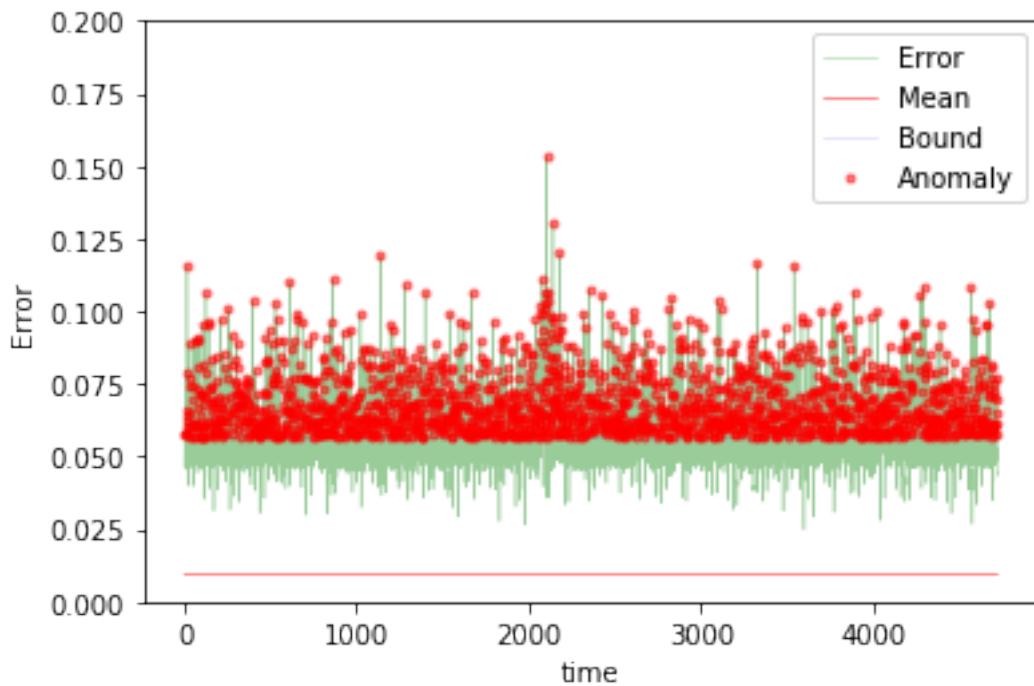
```
Training loss for final epoch is 0.009828221344621852
Validation loss for final epoch is 0.009945153599604965
----- Beginning tests for gru3_5 -----
Testing on normal data.
```



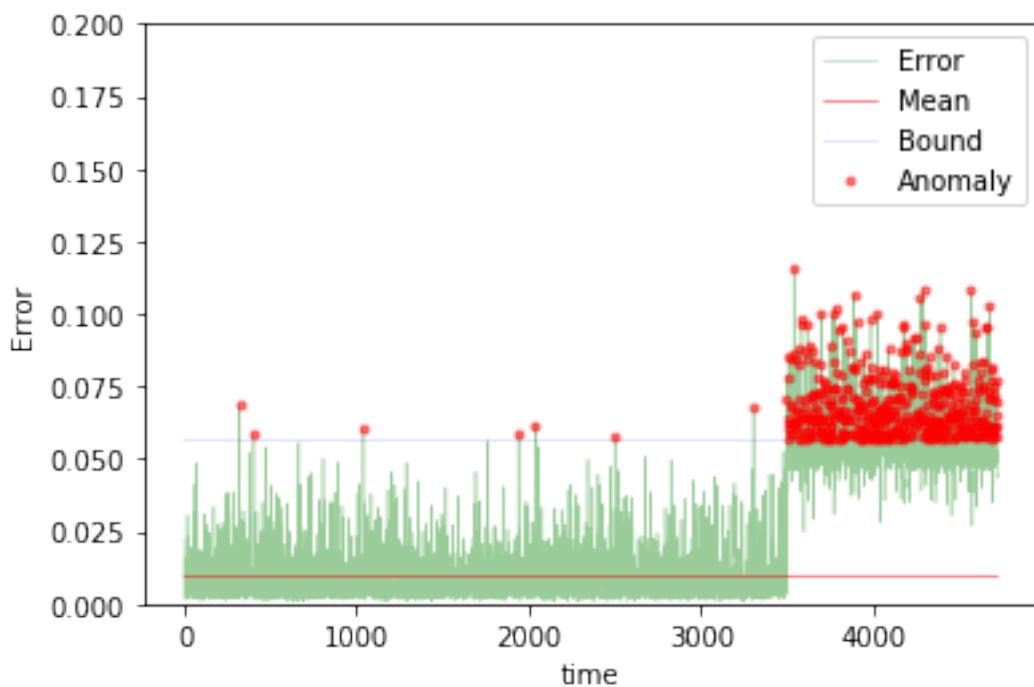
The mean error for gru3_5_normal_ is 0.009408932581983843 for length 4724
Testing on anomaly data.



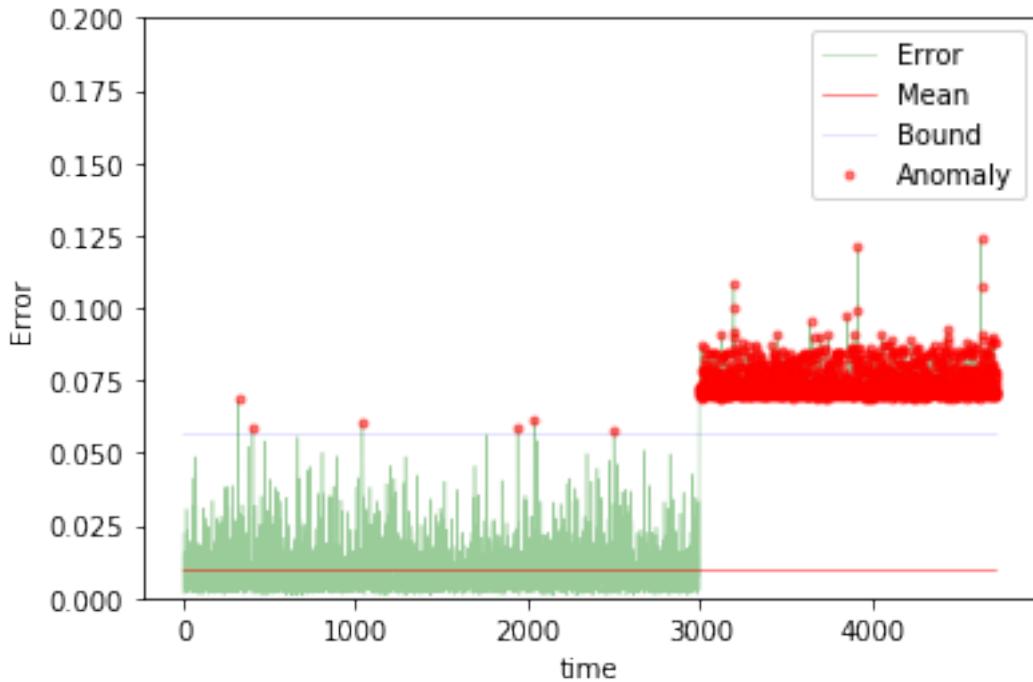
The mean error for gru3_5_anomaly_ is 0.011391927662230208 for length 4724
Testing on different app data.



The mean error for gru3_5_diff_app_ is 0.057104573140836314 for length 4724
Testing on App change synthetic data.



The mean error for gru3_5_app_change_ is 0.02166708748835923 for length 4724
Testing on Net flood synthetic data.



The mean error for gru3_5_net_flood_ is 0.03310203208648304 for length 4724
=====

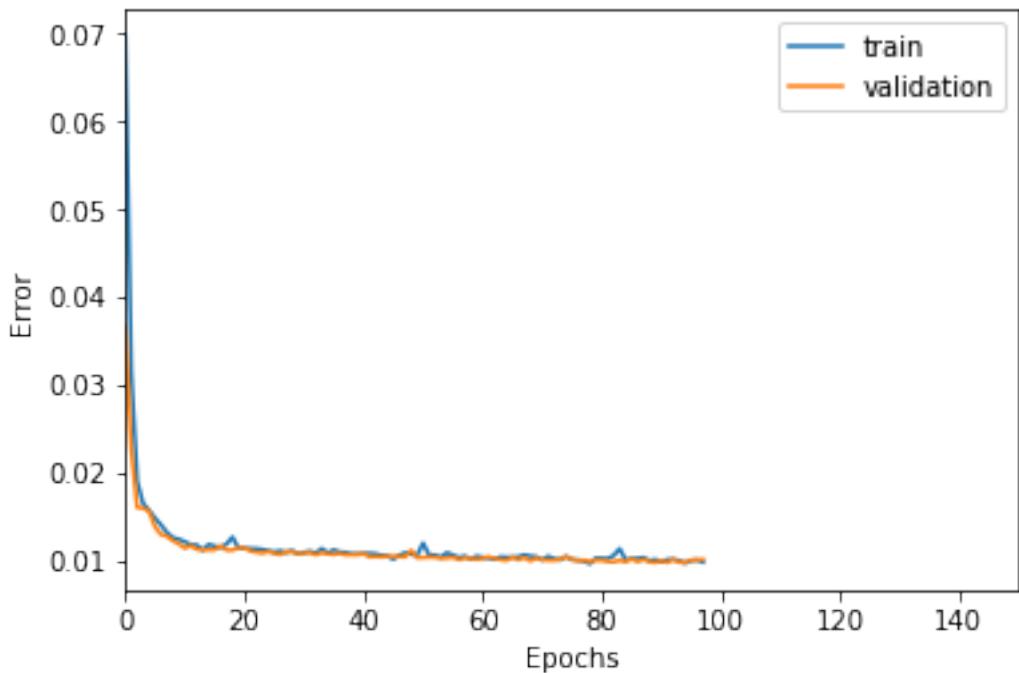
10 steps

```
In [212]: TIMESTEPS = 10
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS, 0)
          vgen = flat_generator(val_X, TIMESTEPS, 0)
          name = "gru3_10"

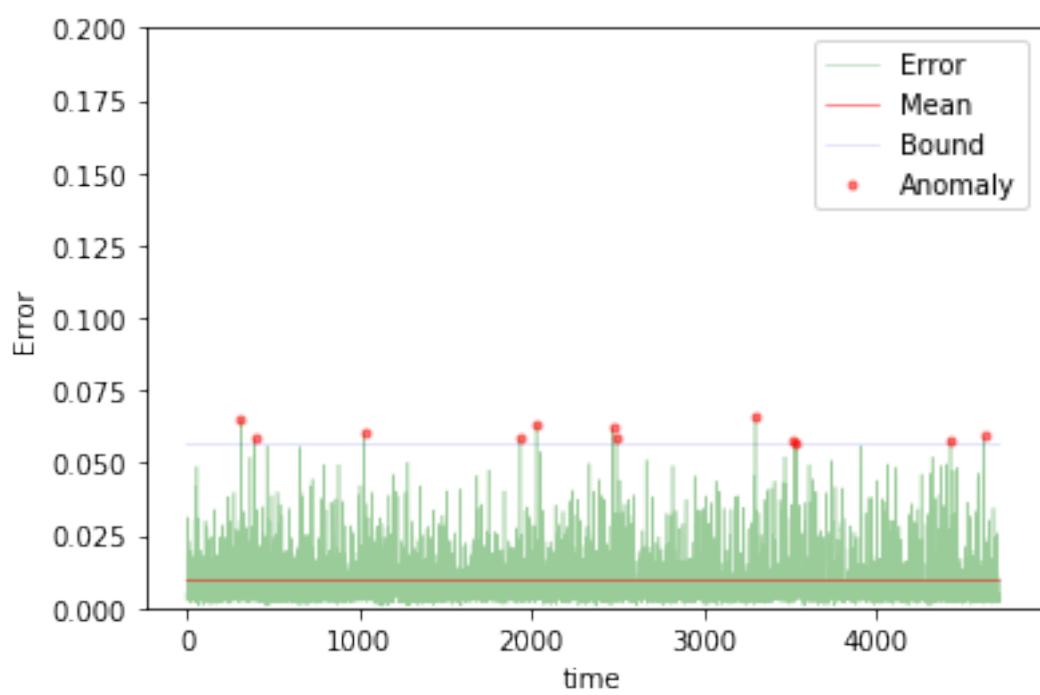
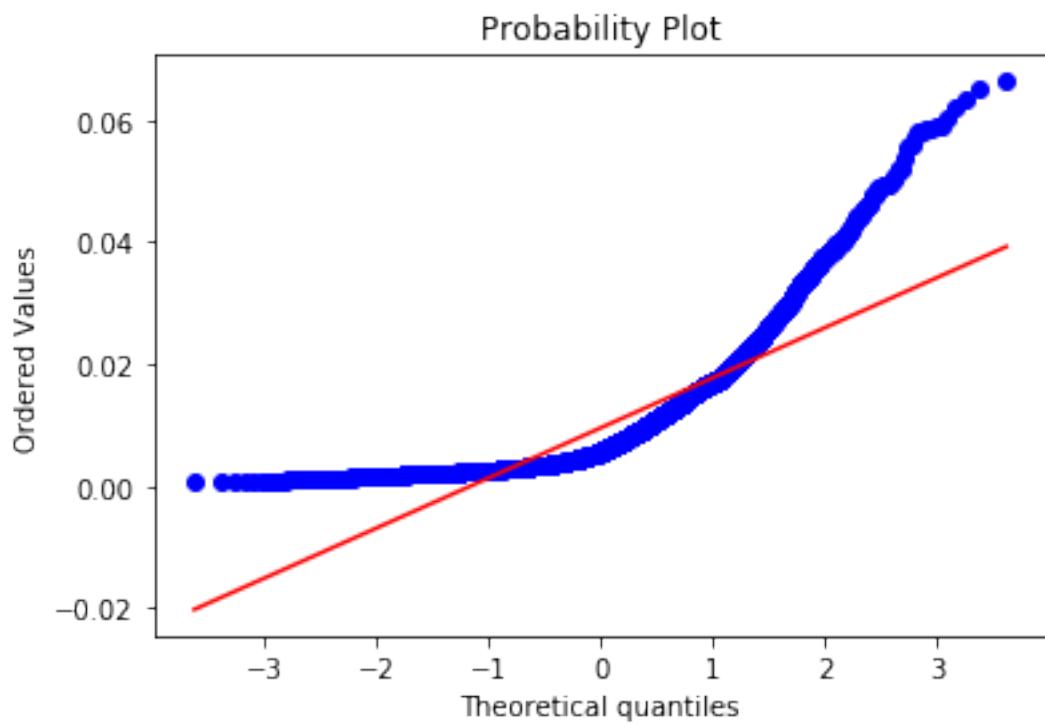
In [213]: input_layer = Input(shape=(TIMESTEPS,DIM))
          hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
          hidden = GRU(10, activation='relu', return_sequences=True)(hidden)
          hidden = GRU(10, activation='relu')(hidden)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [214]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

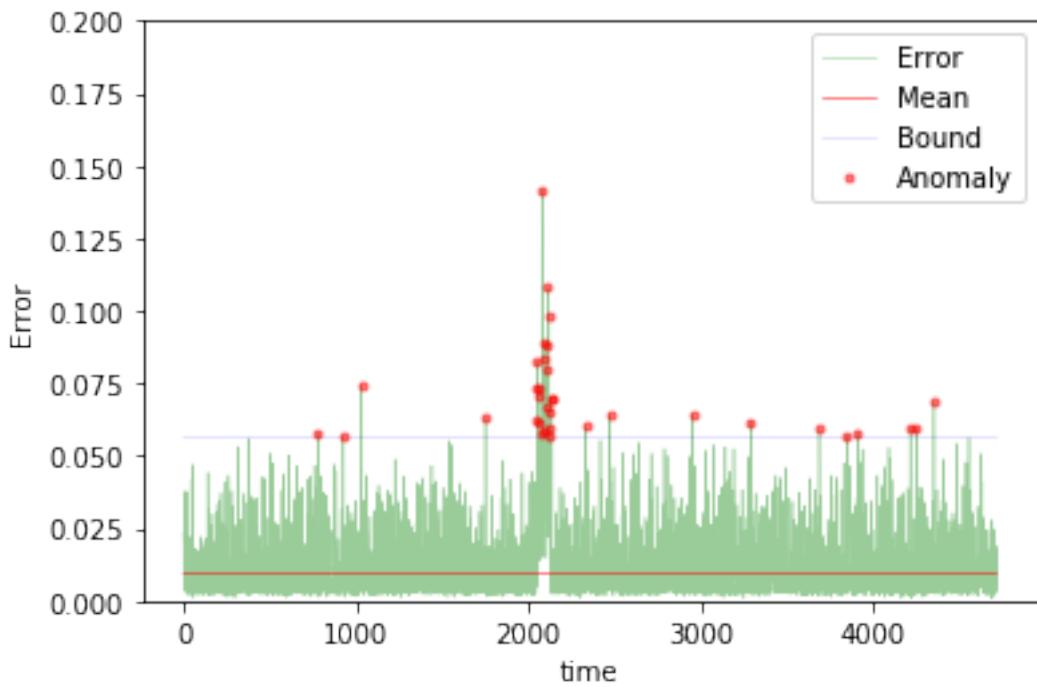
In [215]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



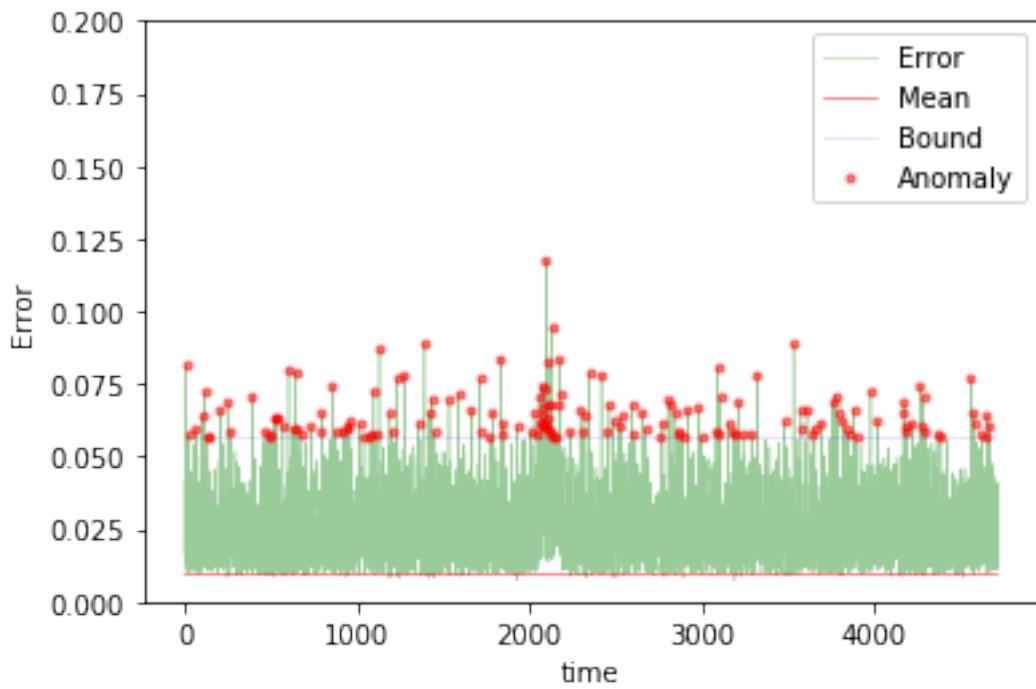
```
Training loss for final epoch is 0.009829794996301643
Validation loss for final epoch is 0.010128887355444021
----- Beginning tests for gru3_10 -----
Testing on normal data.
```



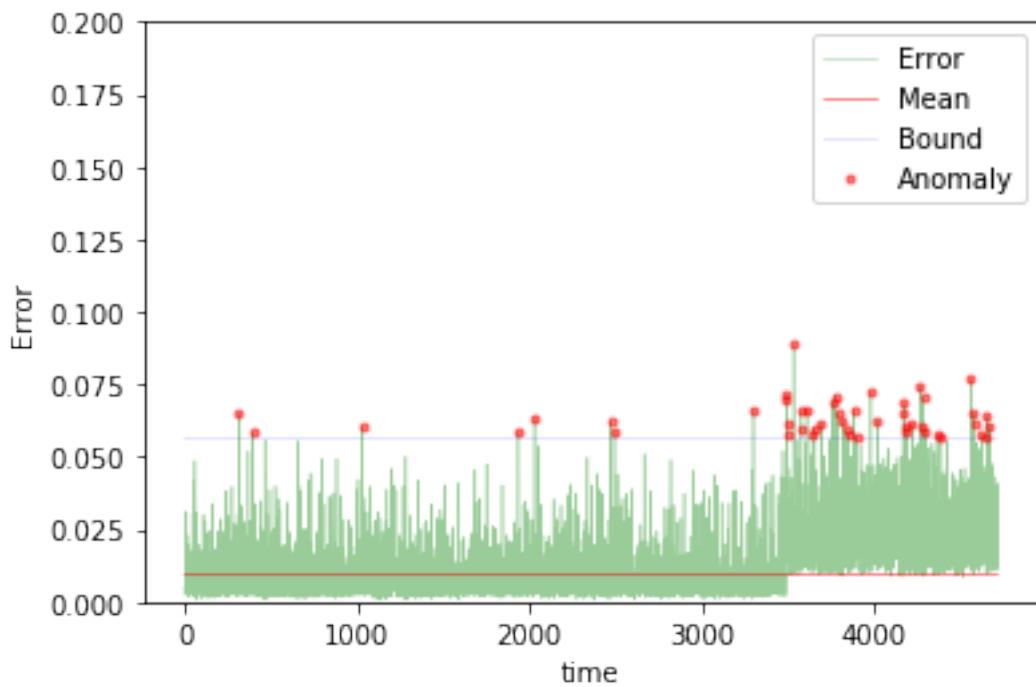
The mean error for gru3_10_normal_ is 0.00950133984586731 for length 4719
Testing on anomaly data.



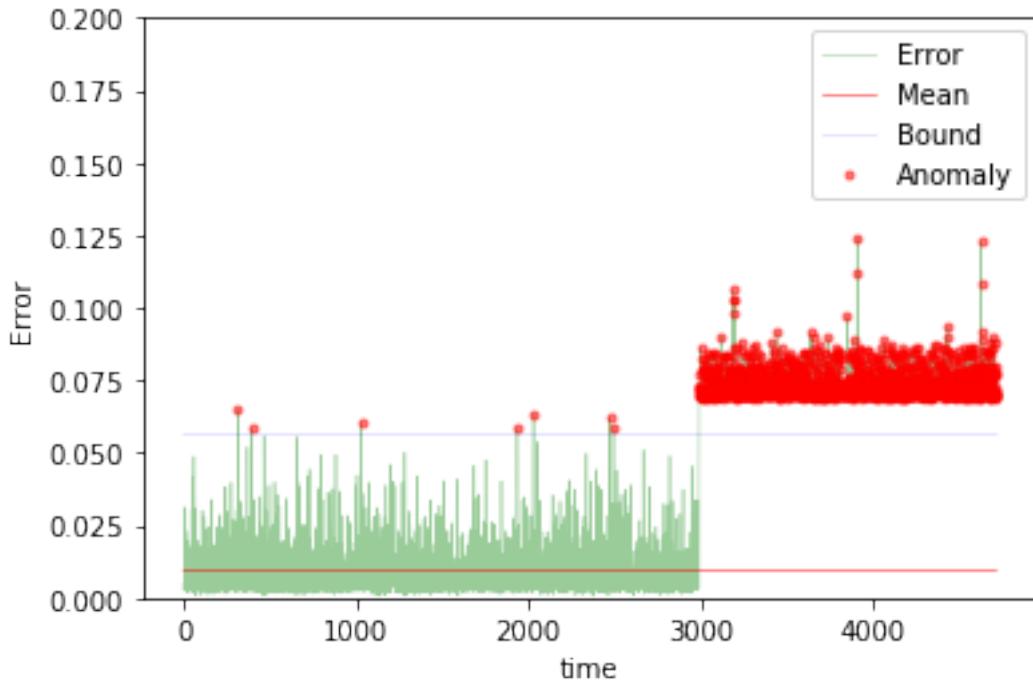
The mean error for gru3_10_anomaly_ is 0.011212639265534516 for length 4719
Testing on different app data.



The mean error for gru3_10_diff_app_ is 0.024535160484939377 for length 4719
Testing on App change synthetic data.



```
The mean error for gru3_10_app_change_ is 0.013324054042687192 for length 4719  
Testing on Net flood synthetic data.
```



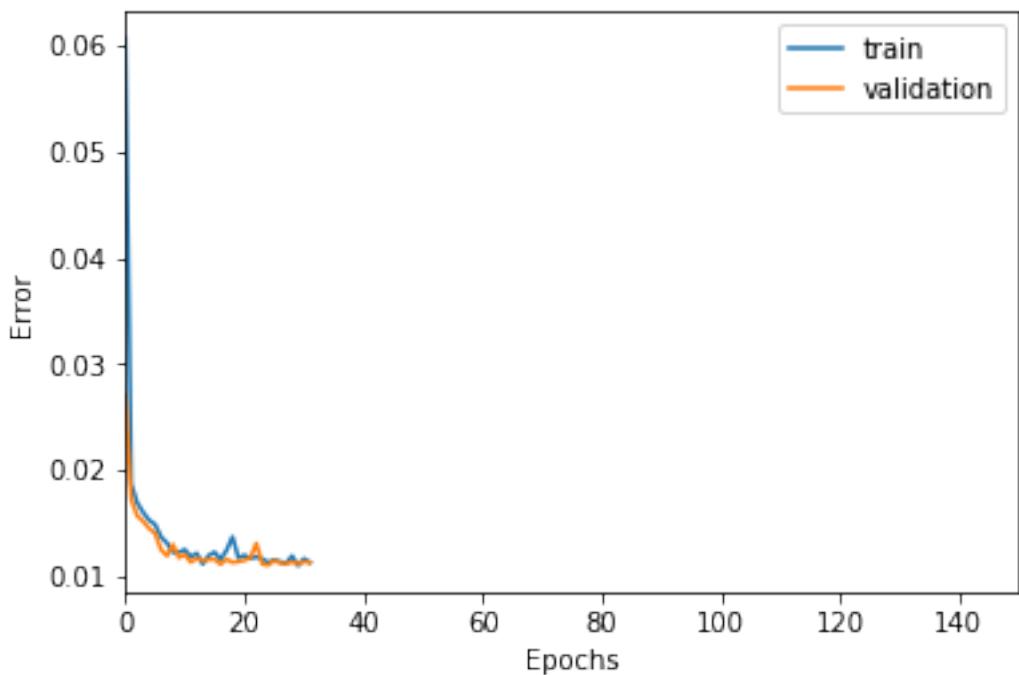
```
The mean error for gru3_10_net_flood_ is 0.033002550538112074 for length 4719  
=====
```

20 steps

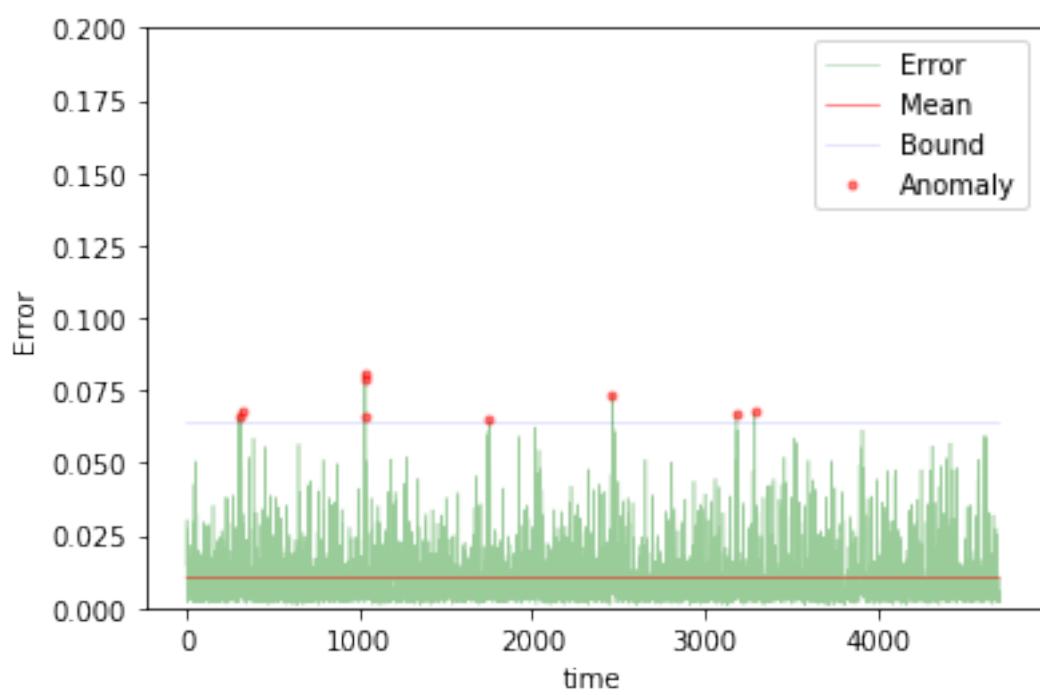
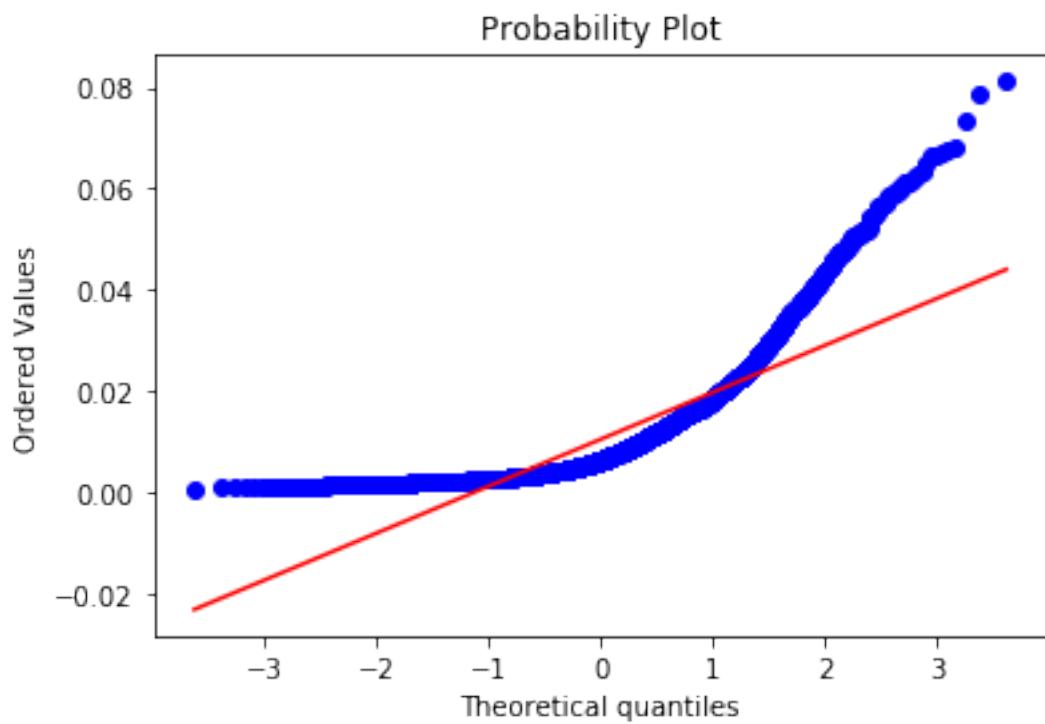
```
In [216]: TIMESTEPS = 20  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru3_20"  
  
In [217]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(10, activation='relu', return_sequences=True)(hidden)  
hidden = GRU(10, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [218]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

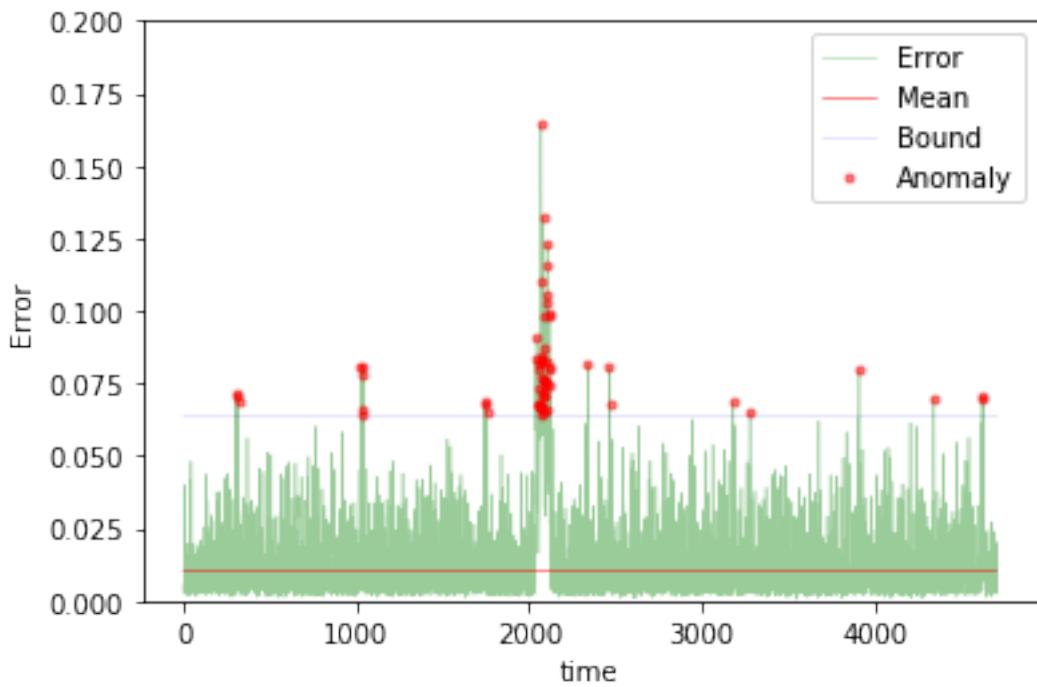
```
In [219]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



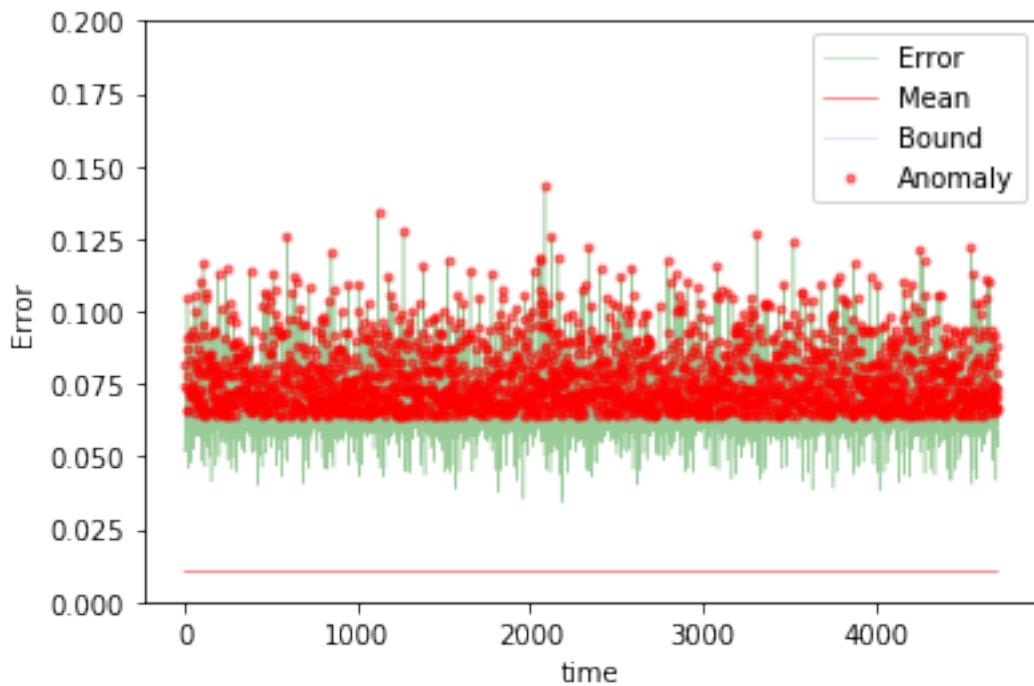
```
Training loss for final epoch is 0.011332053570076822
Validation loss for final epoch is 0.011292890631826594
----- Beginning tests for gru3_20 -----
Testing on normal data.
```



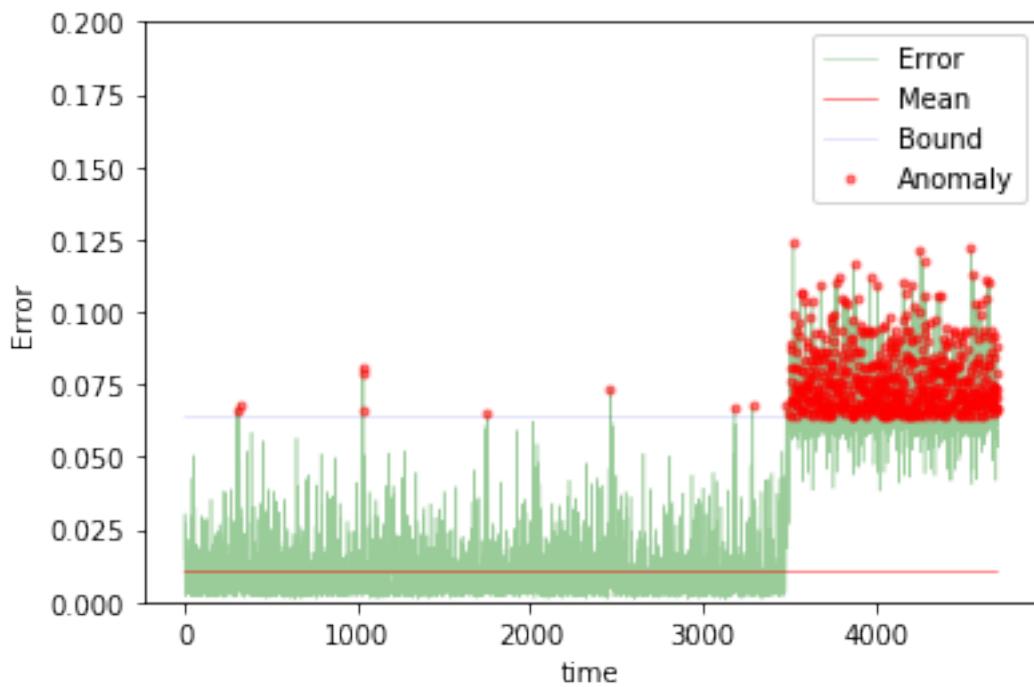
The mean error for gru3_20_normal_ is 0.010490038933699186 for length 4709
Testing on anomaly data.



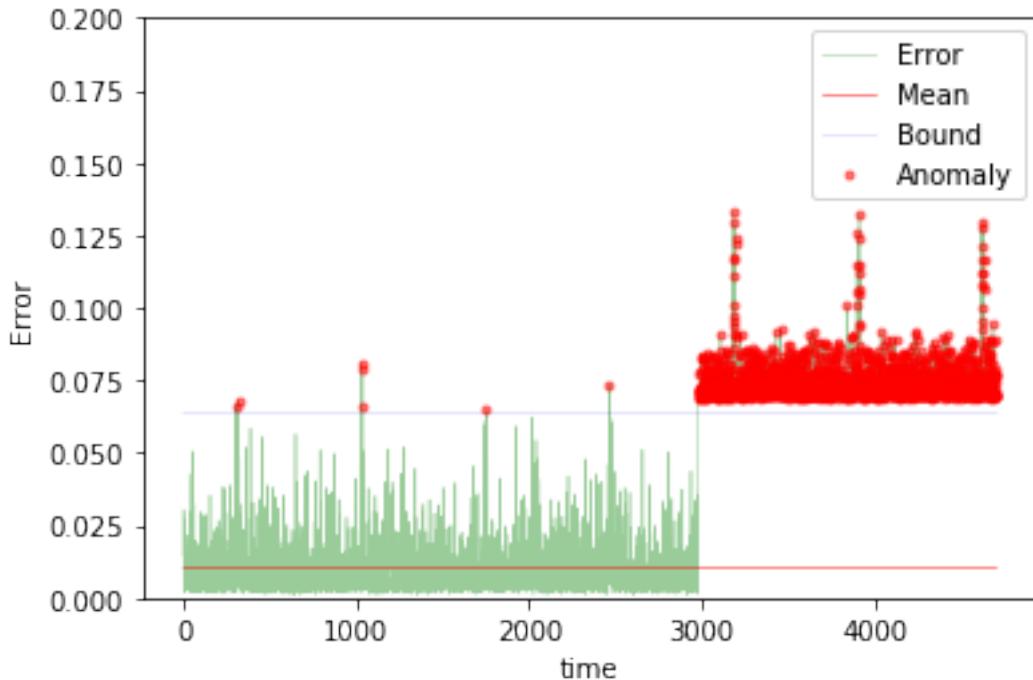
The mean error for gru3_20_anomaly_ is 0.013024925770367344 for length 4709
Testing on different app data.



The mean error for gru3_20_diff_app_ is 0.06797019771519455 for length 4709
Testing on App change synthetic data.



The mean error for gru3_20_app_change_ is 0.02530953111542536 for length 4709
Testing on Net flood synthetic data.



The mean error for gru3_20_net_flood_ is 0.034184410522945755 for length 4709
=====

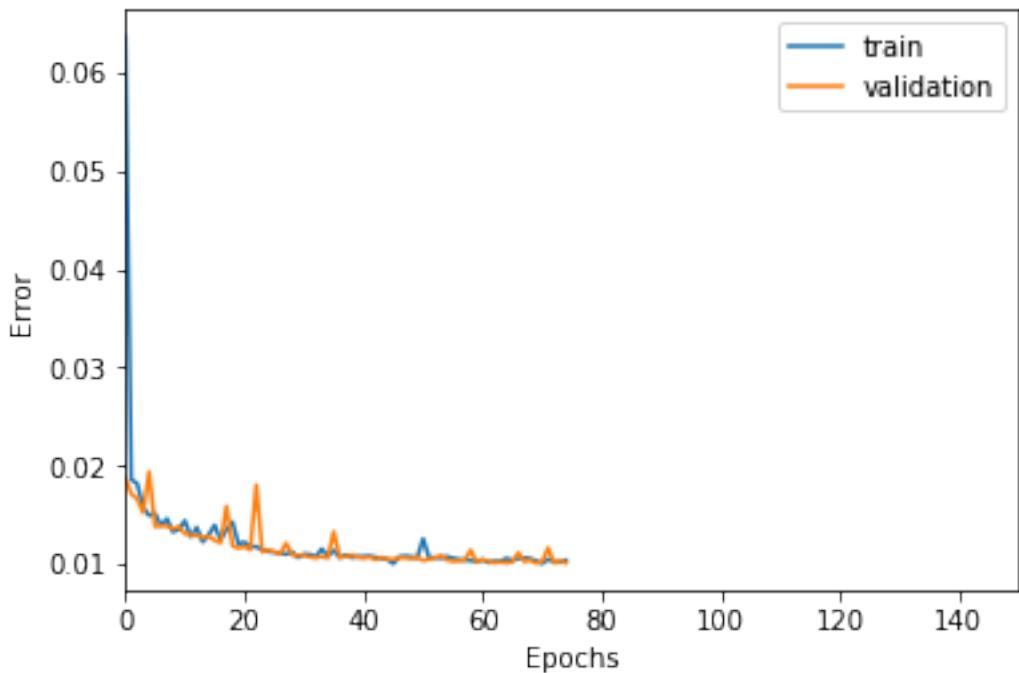
50 steps

```
In [220]: TIMESTEPS = 50
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS,0)
          vgen = flat_generator(val_X, TIMESTEPS,0)
          name = "gru3_50"

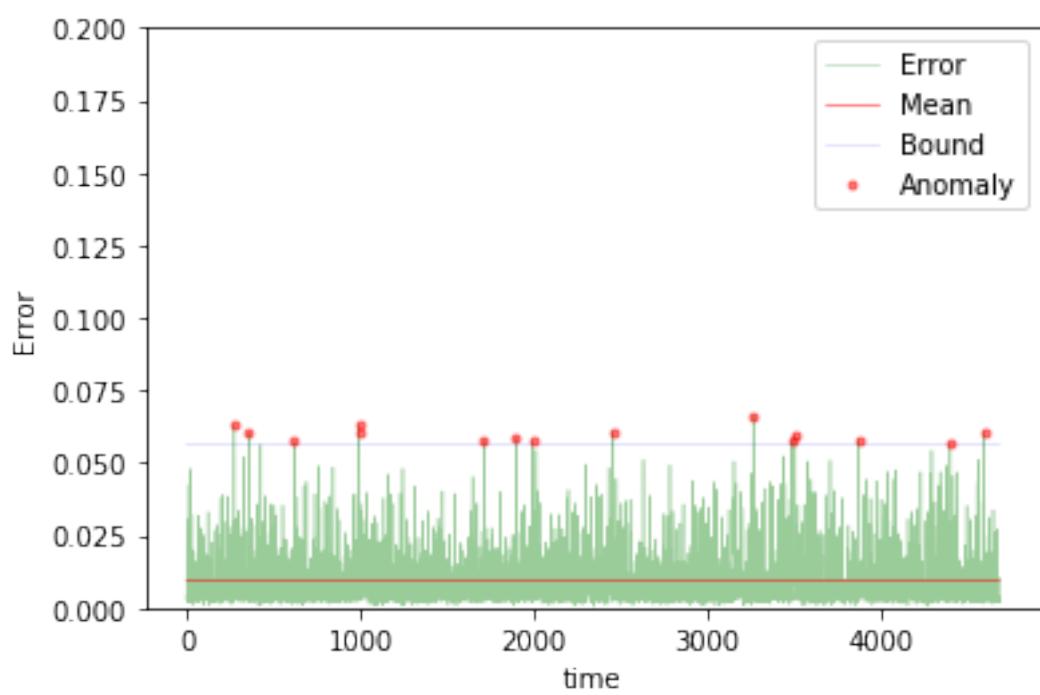
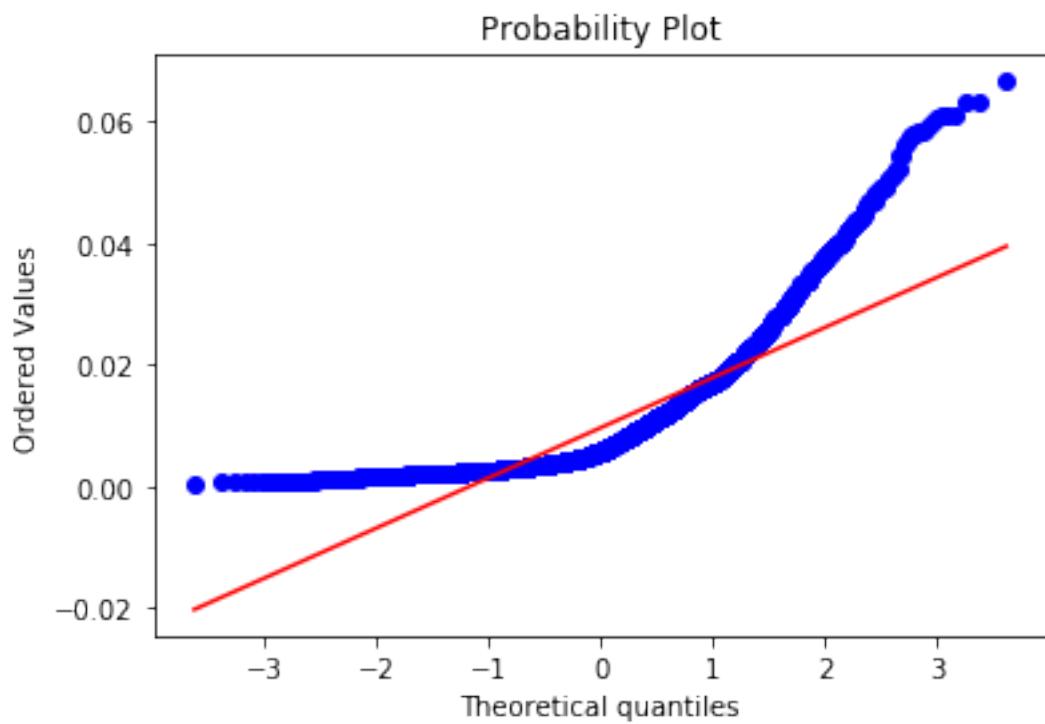
In [221]: input_layer = Input(shape=(TIMESTEPS,DIM))
          hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
          hidden = GRU(10, activation='relu', return_sequences=True)(hidden)
          hidden = GRU(10, activation='relu')(hidden)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [222]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

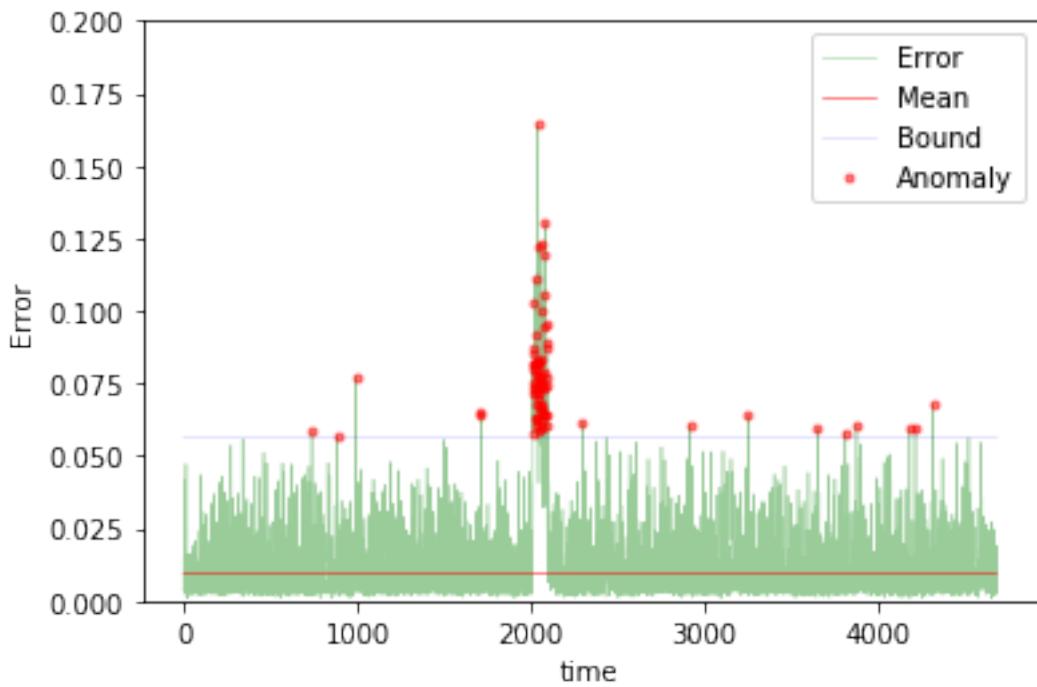
In [223]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)
```



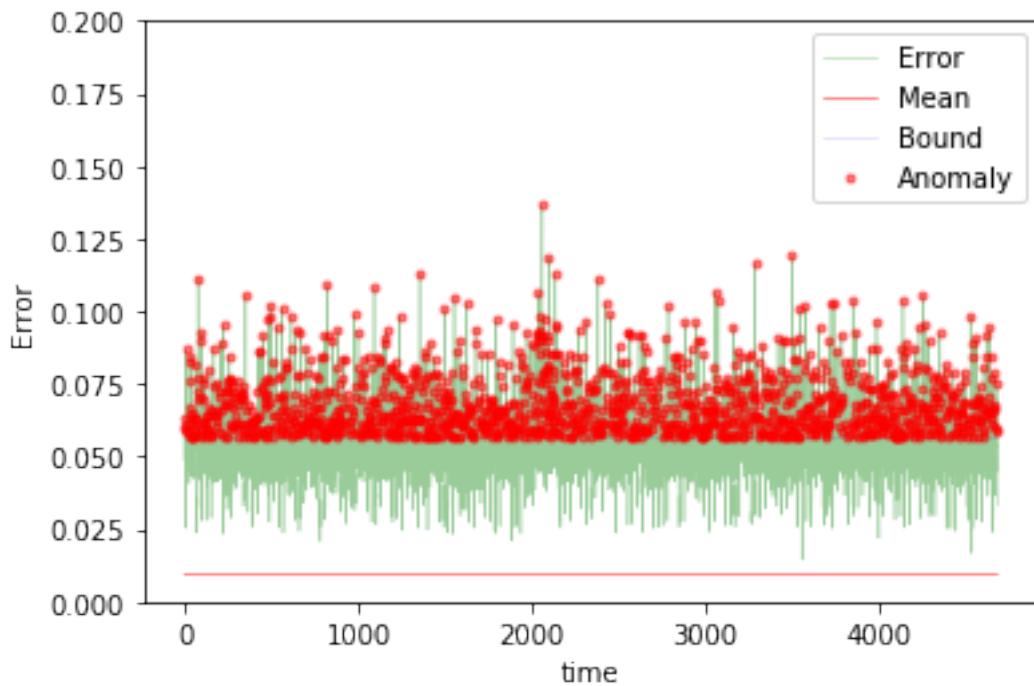
```
Training loss for final epoch is 0.01036520163156092
Validation loss for final epoch is 0.010102849345421418
----- Beginning tests for gru3_50 -----
Testing on normal data.
```



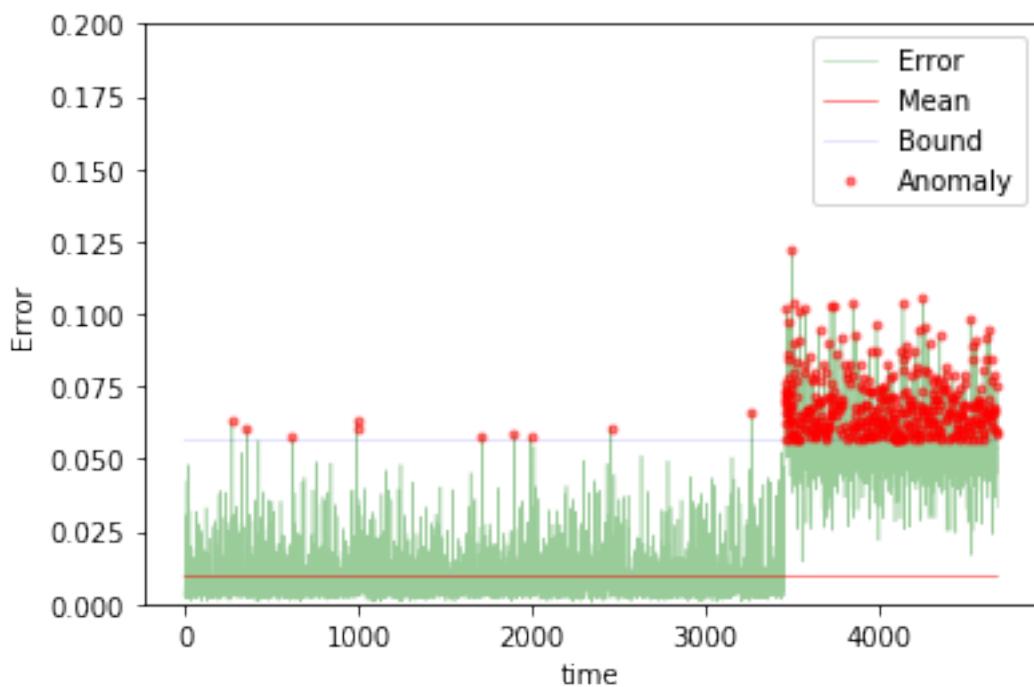
The mean error for gru3_50_normal_ is 0.009602063350246106 for length 4679
Testing on anomaly data.



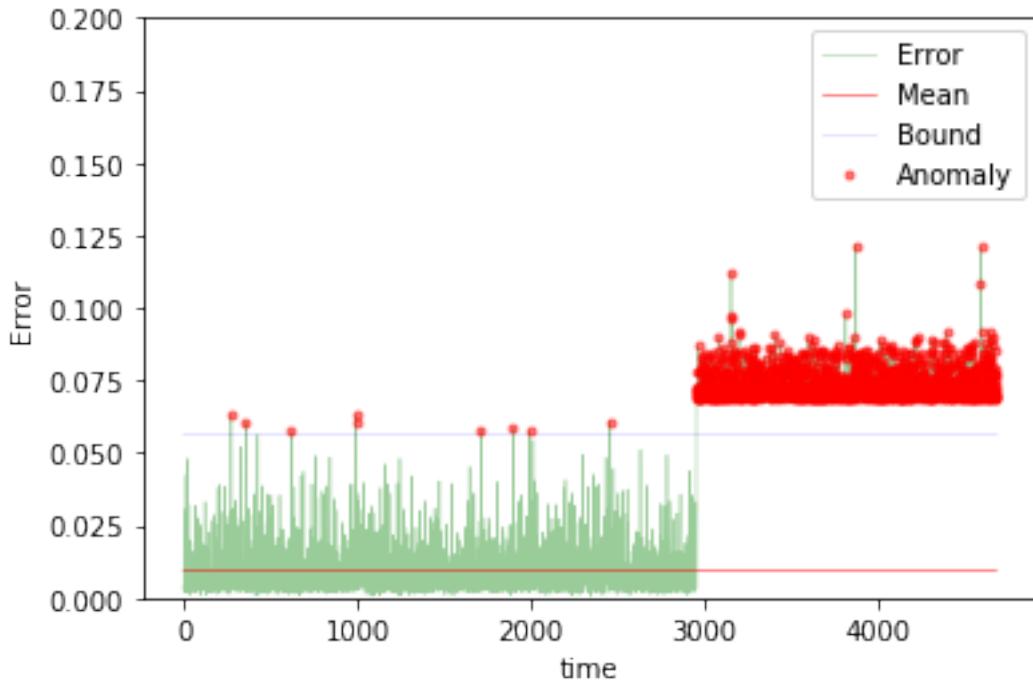
The mean error for gru3_50_anomaly_ is 0.01183514785569374 for length 4679
Testing on different app data.



The mean error for gru3_50_diff_app_ is 0.05384293626329634 for length 4679
Testing on App change synthetic data.



```
The mean error for gru3_50_app_change_ is 0.021227103982402816 for length 4679  
Testing on Net flood synthetic data.
```



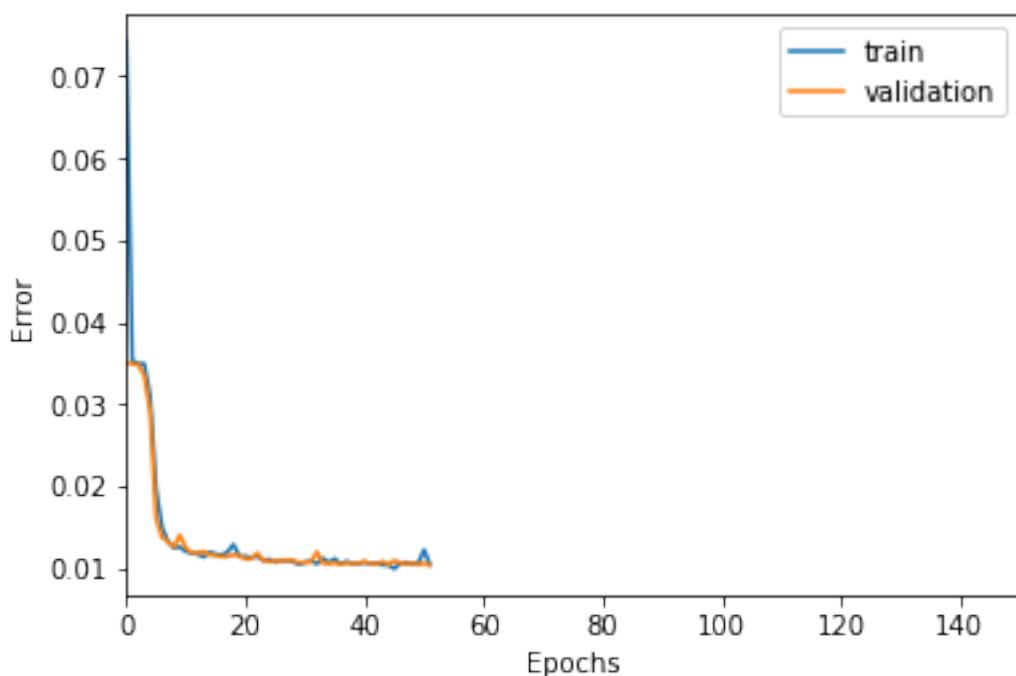
```
The mean error for gru3_50_net_flood_ is 0.0333183377411641 for length 4679  
=====
```

100 steps

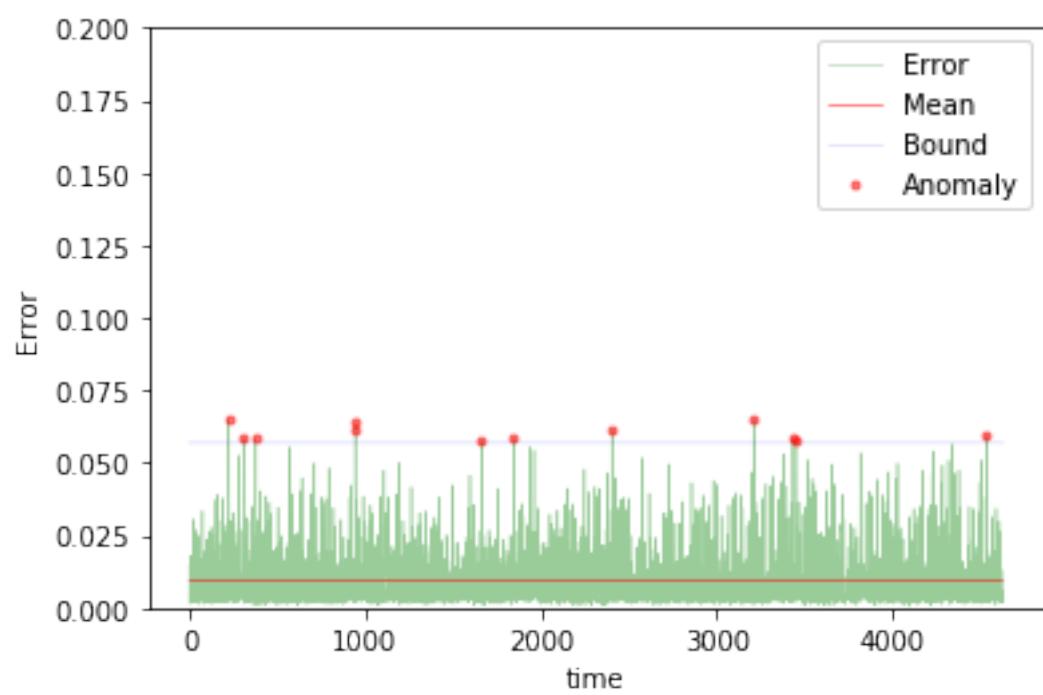
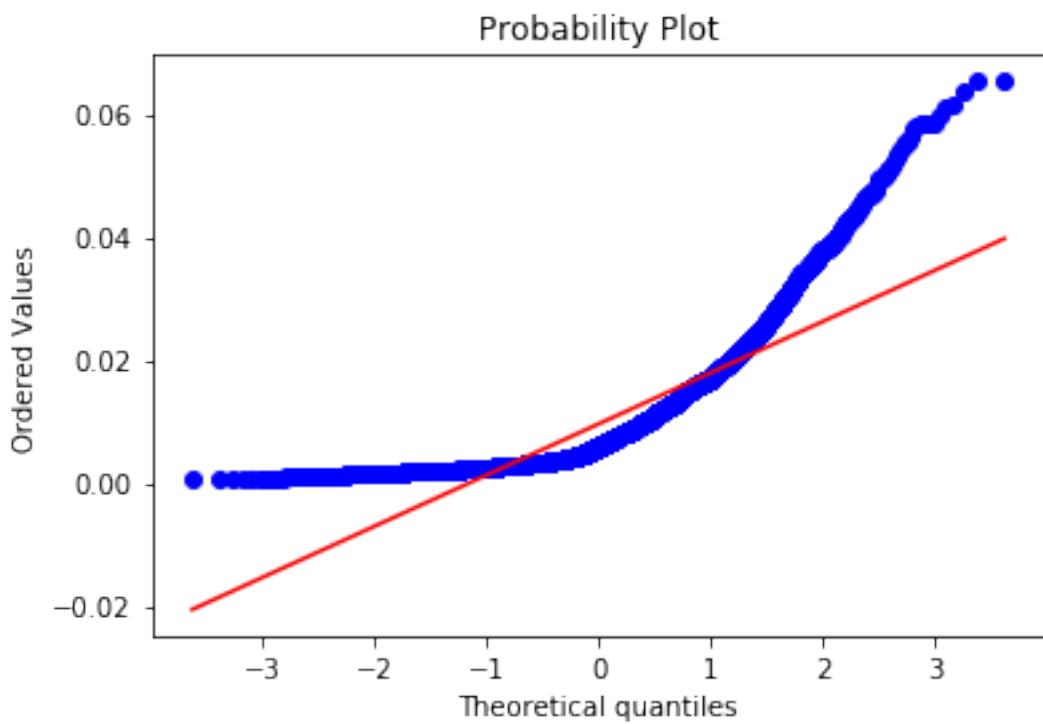
```
In [224]: TIMESTEPS = 100  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru3_100"  
  
In [225]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(10, activation='relu', return_sequences=True)(hidden)  
hidden = GRU(10, activation='relu')(hidden)  
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [226]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

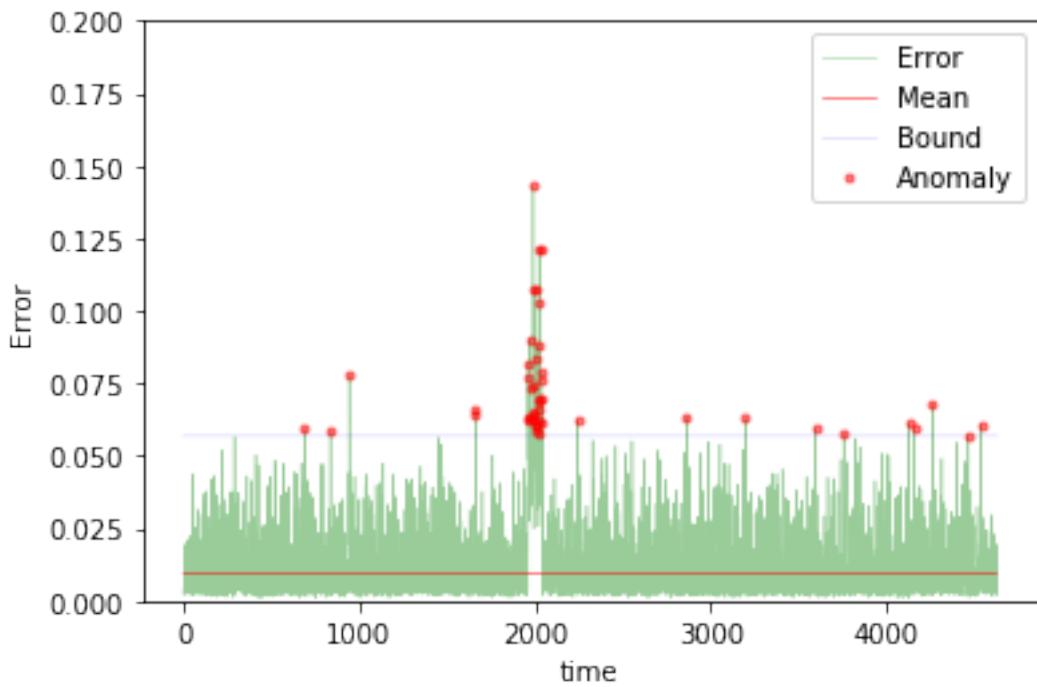
```
In [227]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



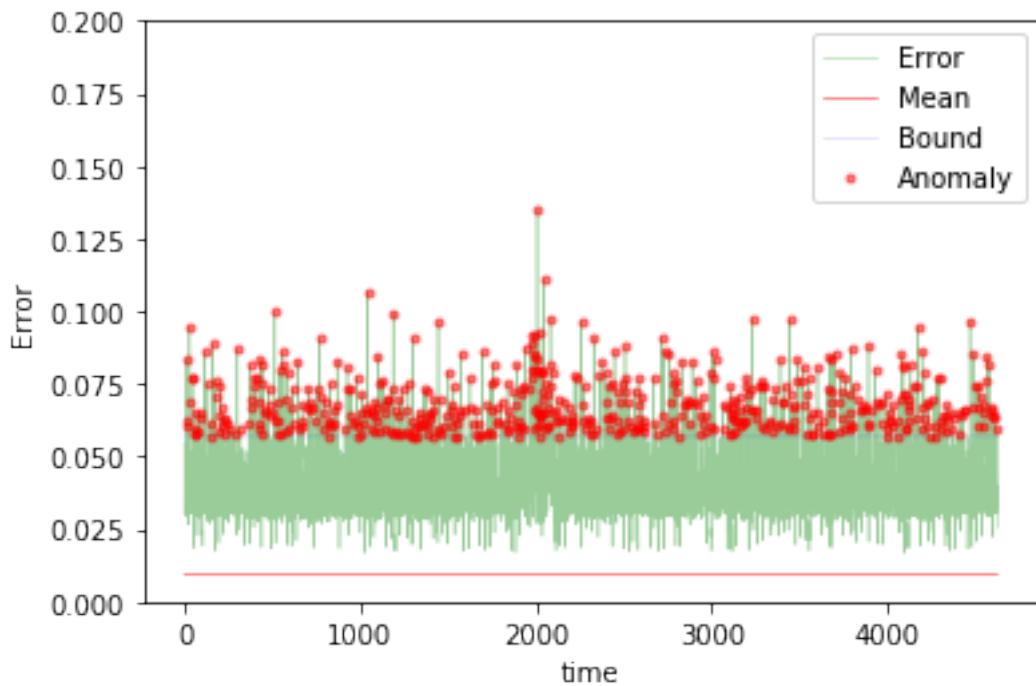
```
Training loss for final epoch is 0.010453580897767097
Validation loss for final epoch is 0.010389218746917322
----- Beginning tests for gru3_100 -----
Testing on normal data.
```



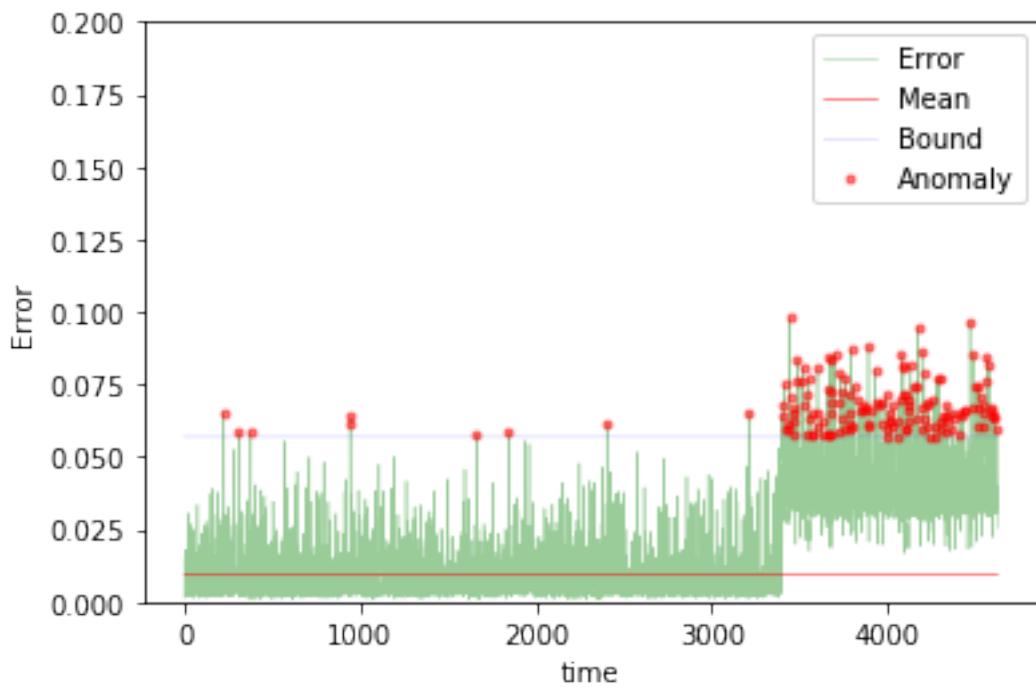
The mean error for gru3_100_normal_ is 0.009595710480588802 for length 4629
Testing on anomaly data.



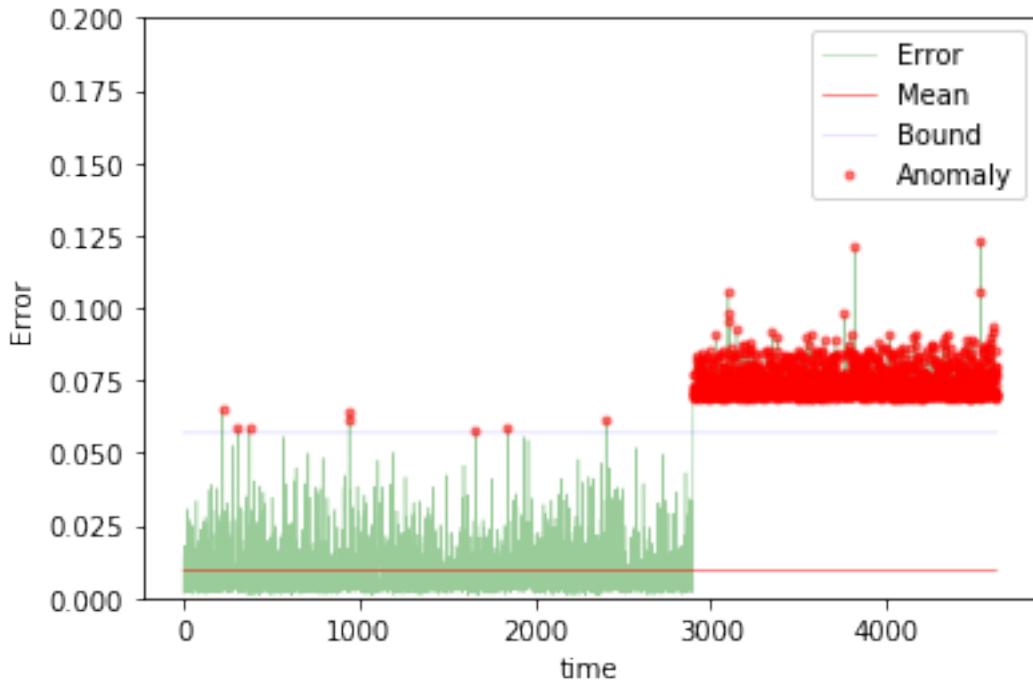
The mean error for gru3_100_anomaly_ is 0.011624636959128264 for length 4629
Testing on different app data.



The mean error for gru3_100_diff_app_ is 0.041160178237795 for length 4629
Testing on App change synthetic data.



The mean error for gru3_100_app_change_ is 0.017879266451288078 for length 4629
Testing on Net flood synthetic data.



The mean error for gru3_100_net_flood_ is 0.03366312245387238 for length 4629
=====

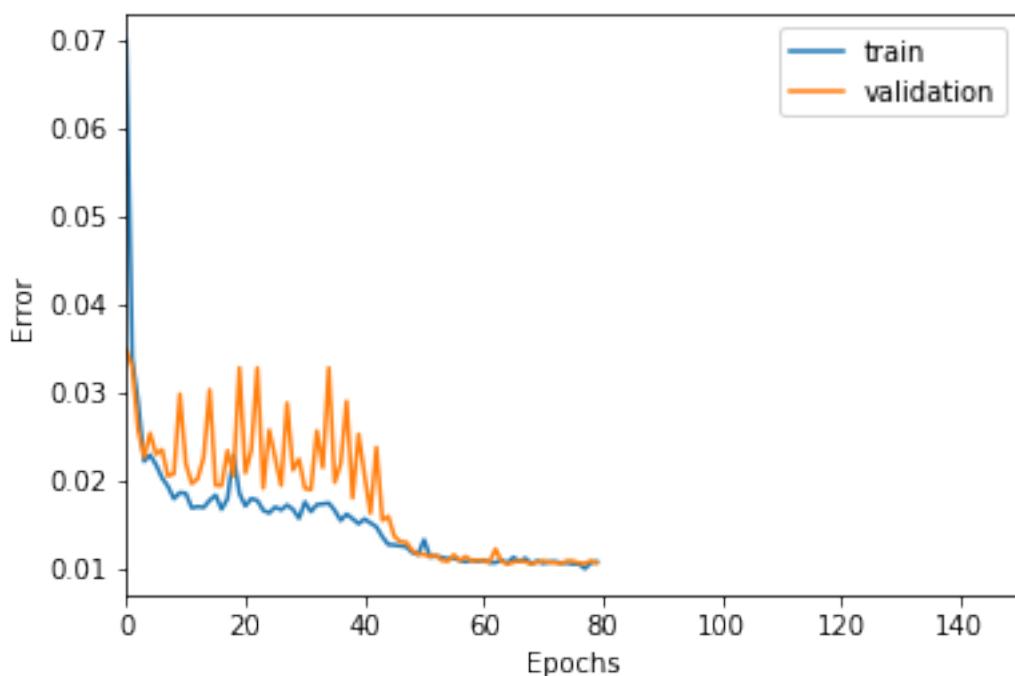
200 steps

```
In [228]: TIMESTEPS = 200
          DIM = 29
          tgen = flat_generator(X, TIMESTEPS,0)
          vgen = flat_generator(val_X, TIMESTEPS,0)
          name = "gru3_200"

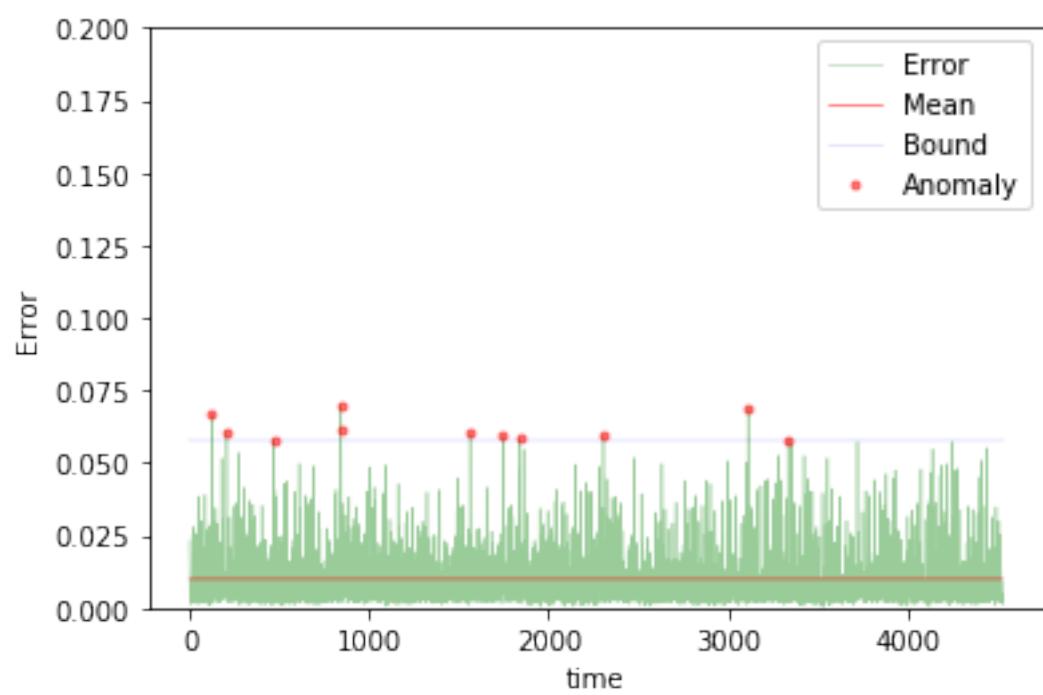
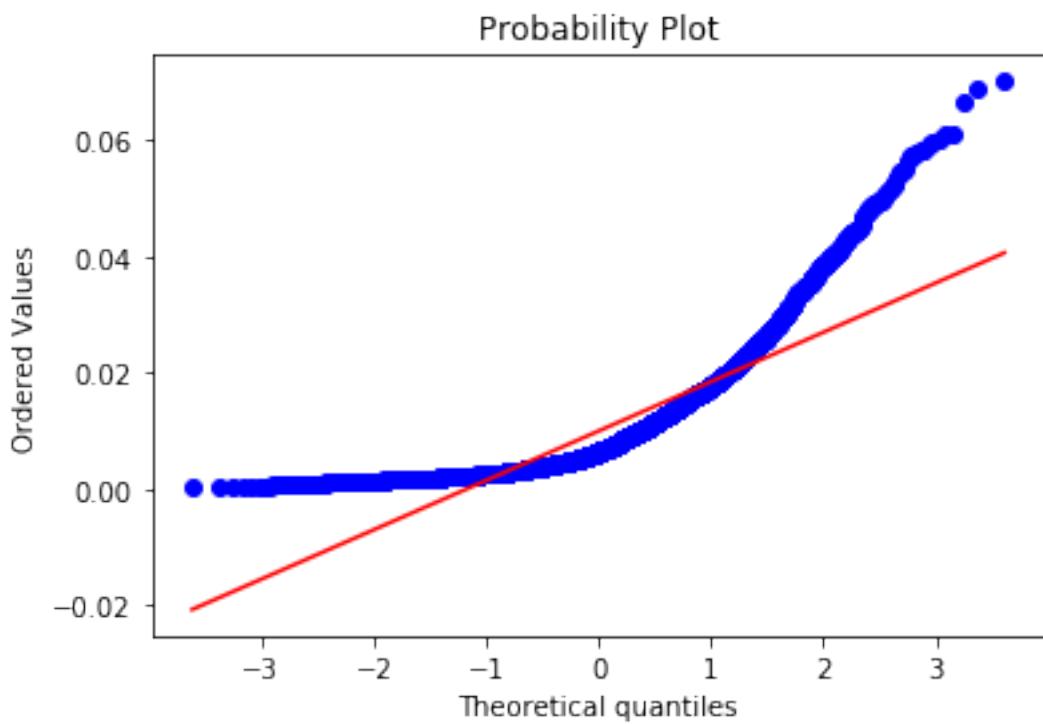
In [229]: input_layer = Input(shape=(TIMESTEPS,DIM))
          hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
          hidden = GRU(10, activation='relu', return_sequences=True)(hidden)
          hidden = GRU(10, activation='relu')(hidden)
          output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [230]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

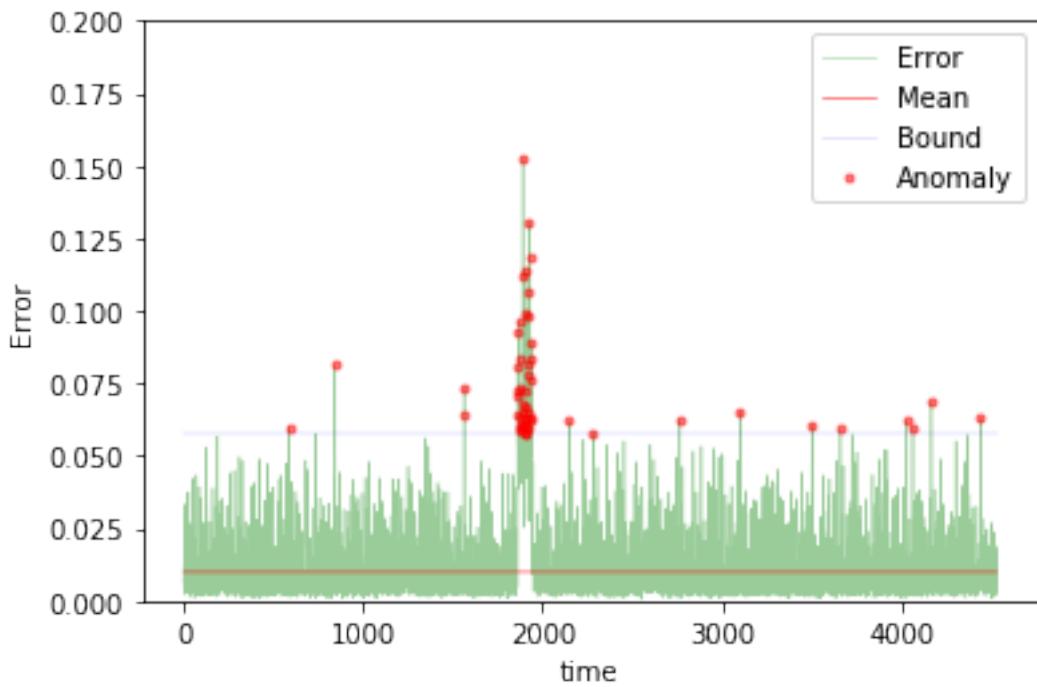
```
In [231]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



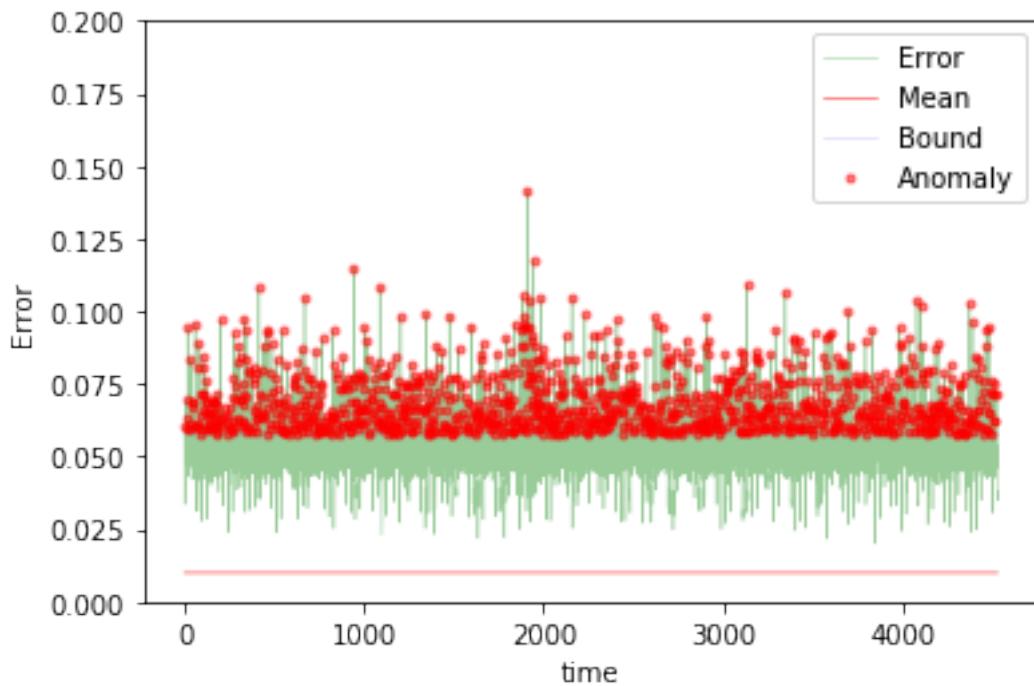
```
Training loss for final epoch is 0.010675009701400996
Validation loss for final epoch is 0.010507098317611962
----- Beginning tests for gru3_200 -----
Testing on normal data.
```



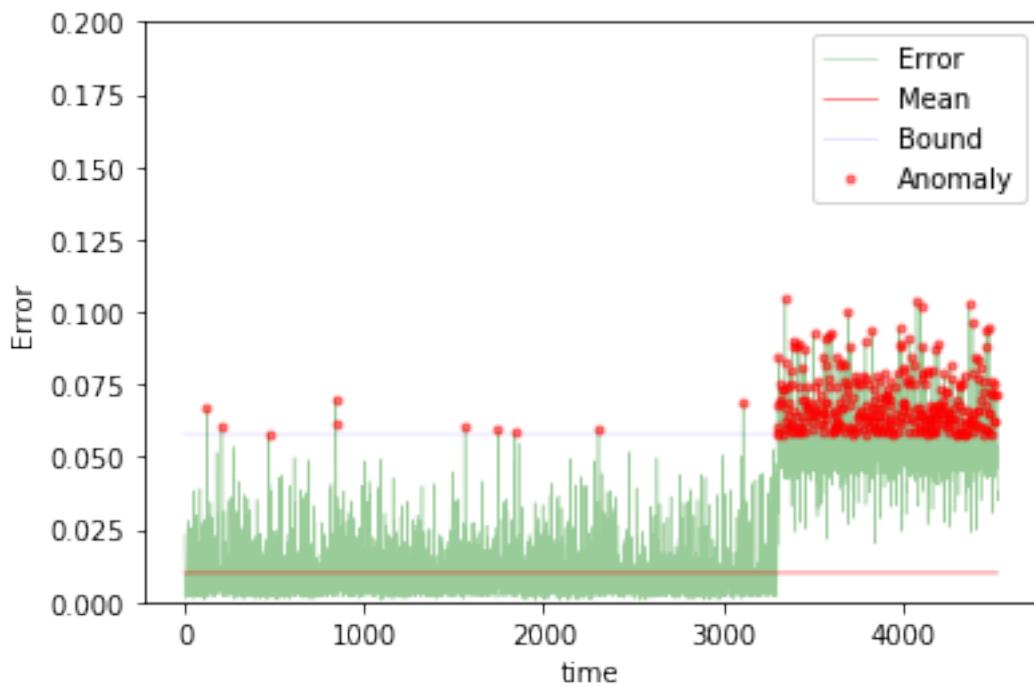
The mean error for gru3_200_normal_ is 0.00999448974774902 for length 4529
Testing on anomaly data.



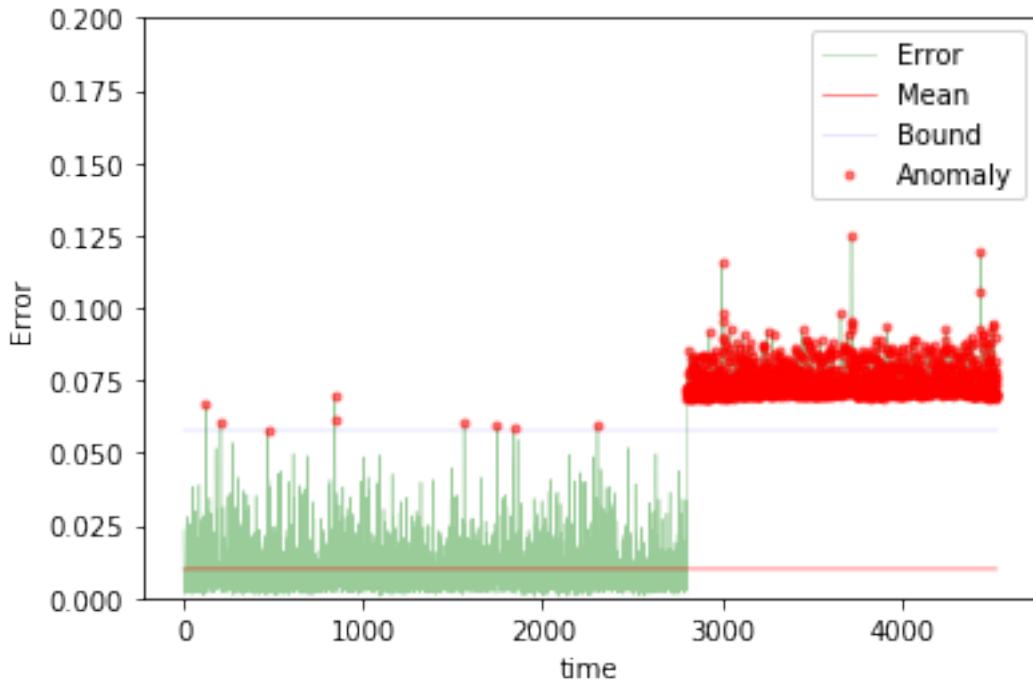
The mean error for gru3_200_anomaly_ is 0.011887932758283148 for length 4529
Testing on different app data.



The mean error for gru3_200_diff_app_ is 0.05361880374050776 for length 4529
Testing on App change synthetic data.



```
The mean error for gru3_200_app_change_ is 0.02171278353697239 for length 4529  
Testing on Net flood synthetic data.
```



```
The mean error for gru3_200_net_flood_ is 0.03459443306791885 for length 4529  
=====
```

2.1.8 RNN with 4 GRU layers dim compression.

2 steps

```
In [232]: TIMESTEPS = 2  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru4_2"
```

```
In [233]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
```

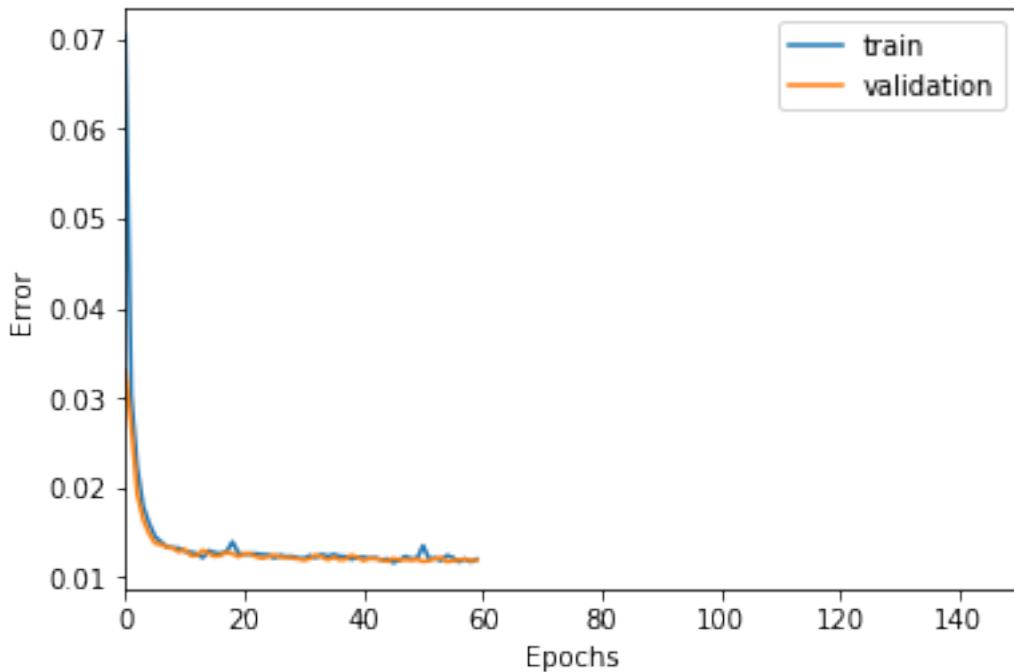
```

hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)

In [234]: model = Model(input_layer, output)
          model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])

In [235]: train(model, tgen, vgen, name=name)
          test(model, ravel=0, name=name, window=TIMESTEPS)

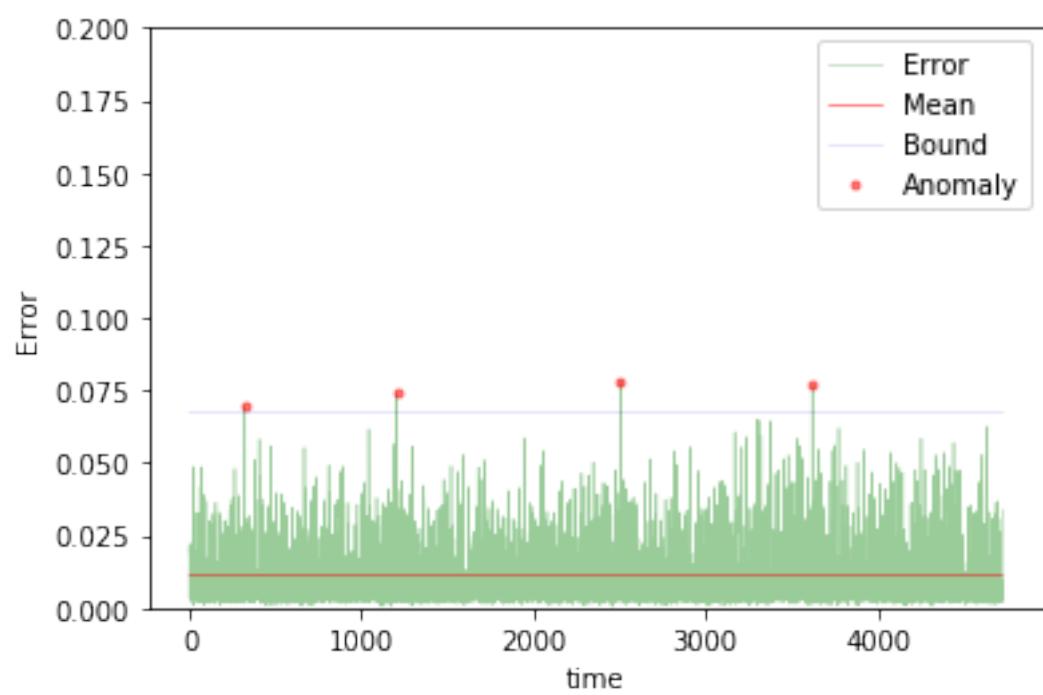
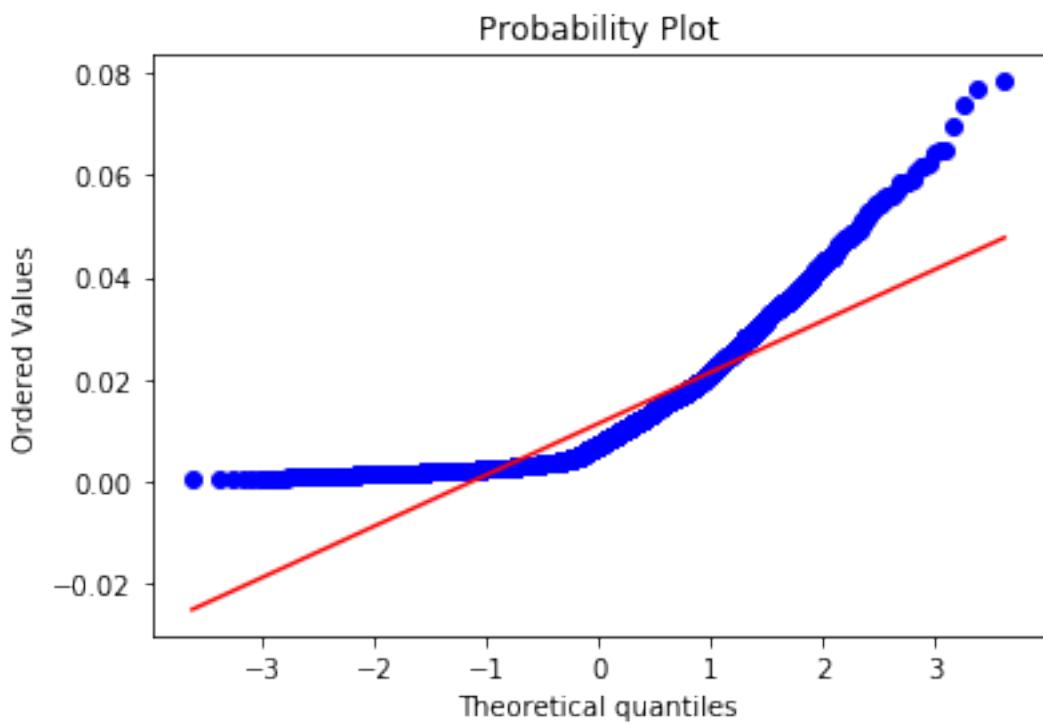
```



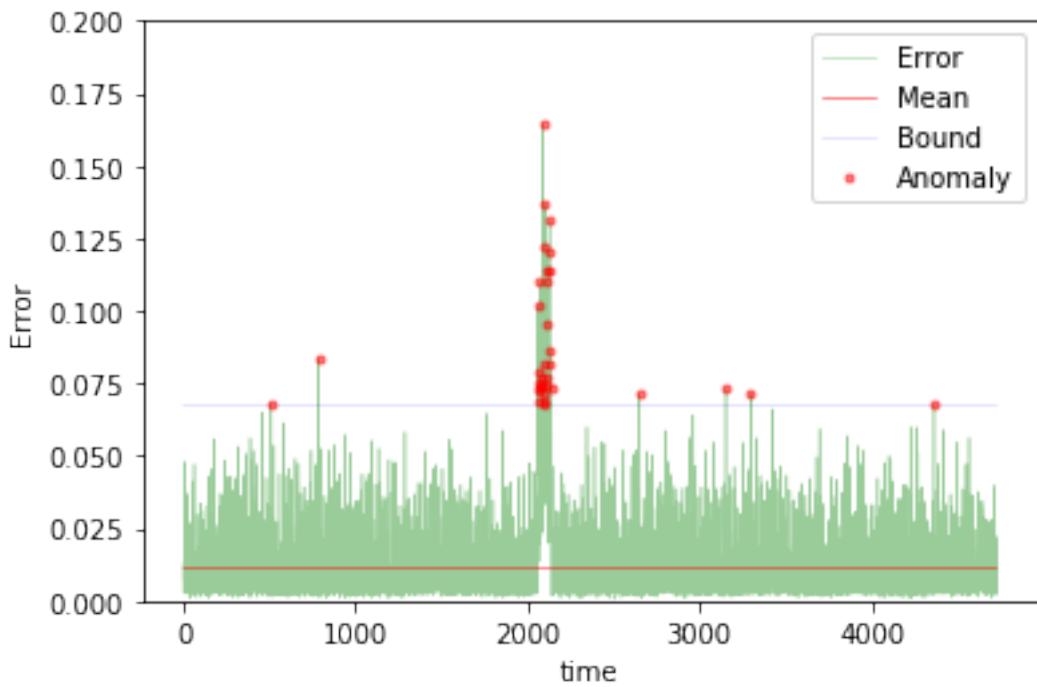
```

Training loss for final epoch is 0.012012882114038803
Validation loss for final epoch is 0.011900658656959422
----- Beginning tests for gru4_2 -----
Testing on normal data.

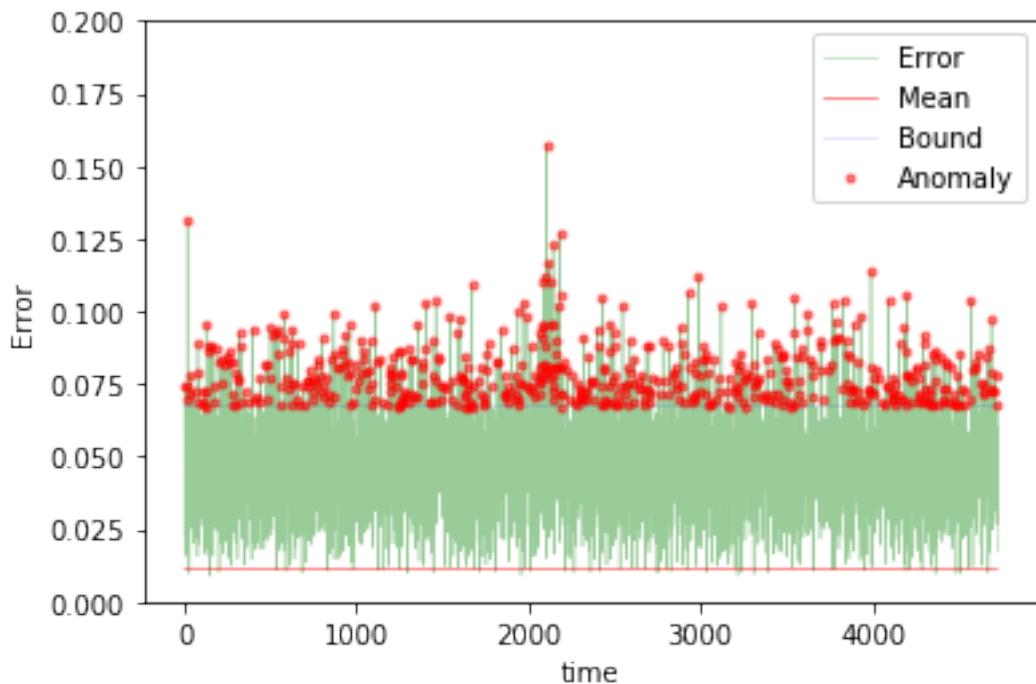
```



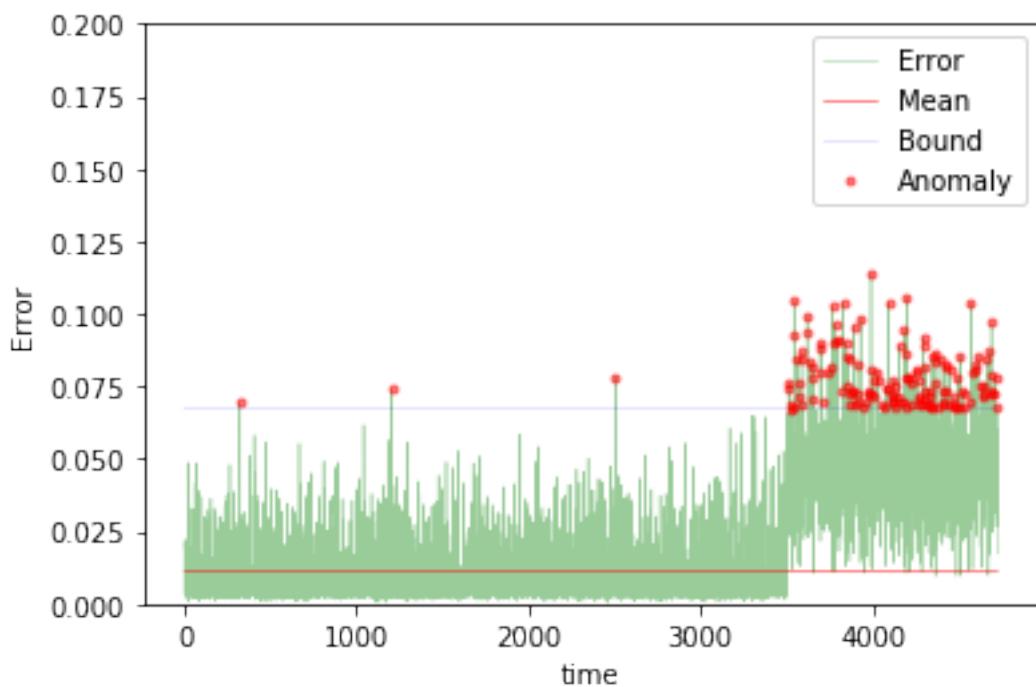
The mean error for gru4_2_normal_ is 0.011383797429090143 for length 4727
Testing on anomaly data.



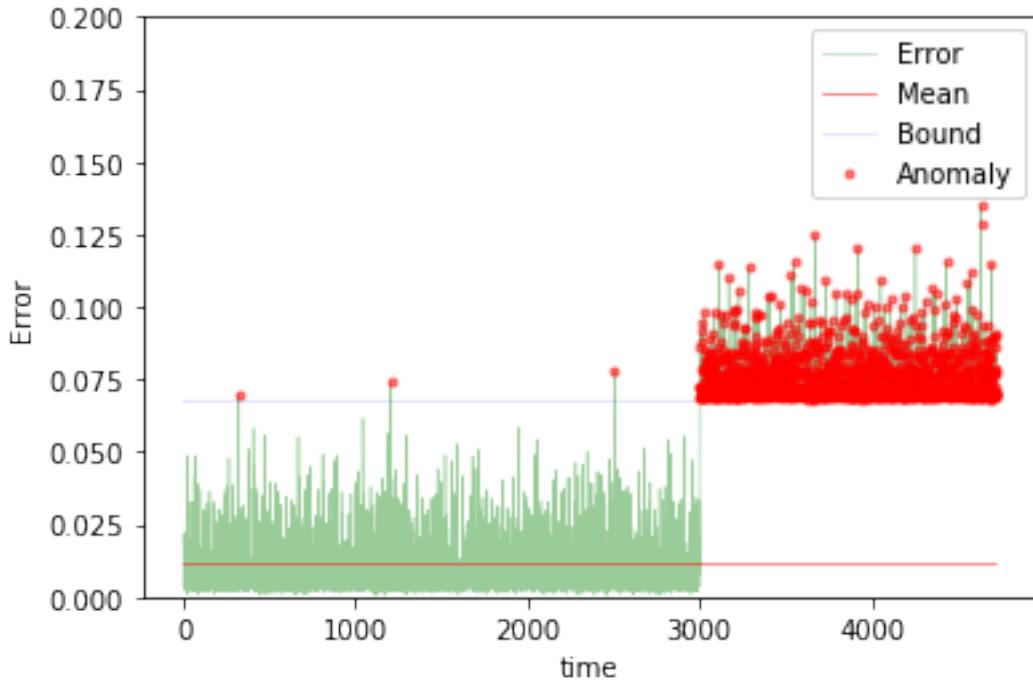
The mean error for gru4_2_anomaly_ is 0.012937666140441763 for length 4727
Testing on different app data.



The mean error for gru4_2_diff_app_ is 0.047490425401865086 for length 4727
Testing on App change synthetic data.



```
The mean error for gru4_2_app_change_ is 0.02066905083476427 for length 4727  
Testing on Net flood synthetic data.
```



```
The mean error for gru4_2_net_flood_ is 0.03483220124164683 for length 4727  
=====
```

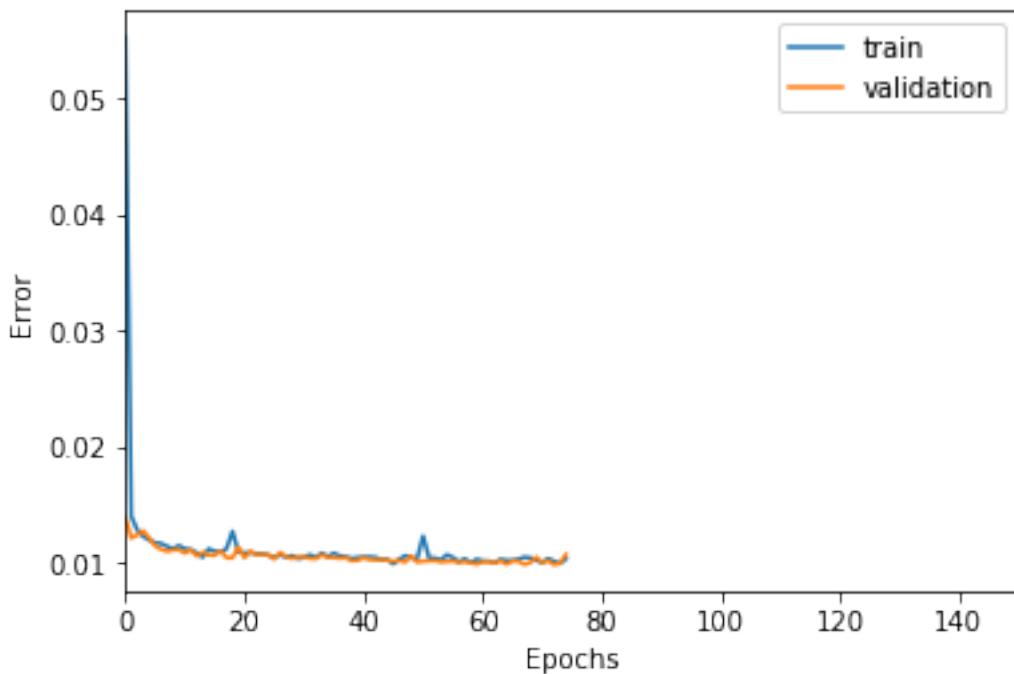
5 steps

```
In [236]: TIMESTEPS = 5  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS, 0)  
vgen = flat_generator(val_X, TIMESTEPS, 0)  
name = "gru4_5"  
  
In [237]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
```

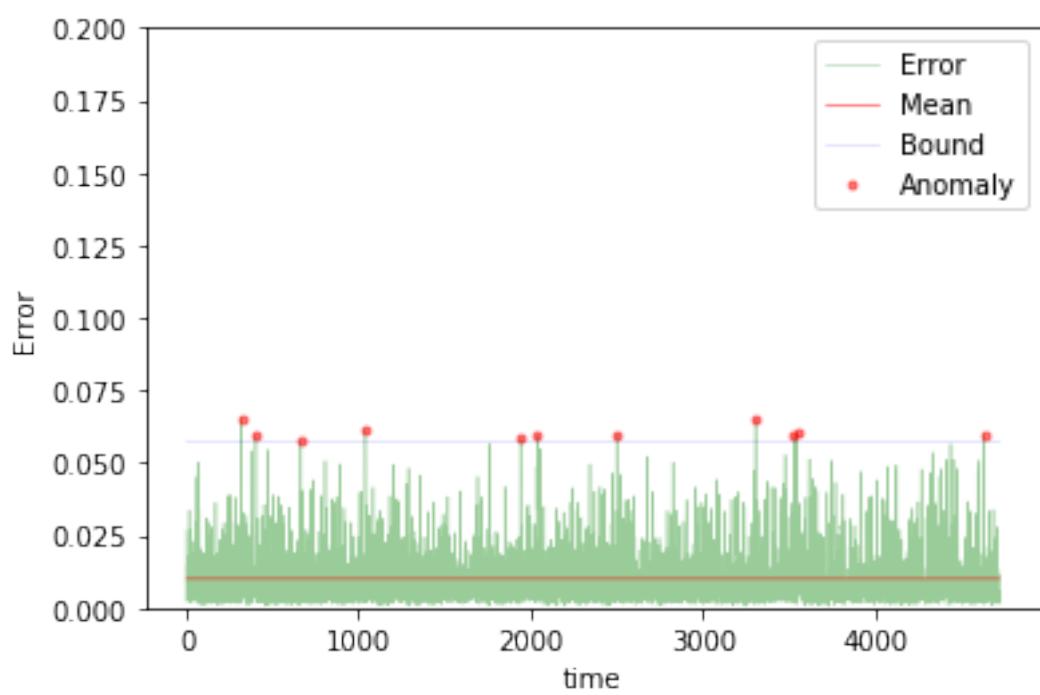
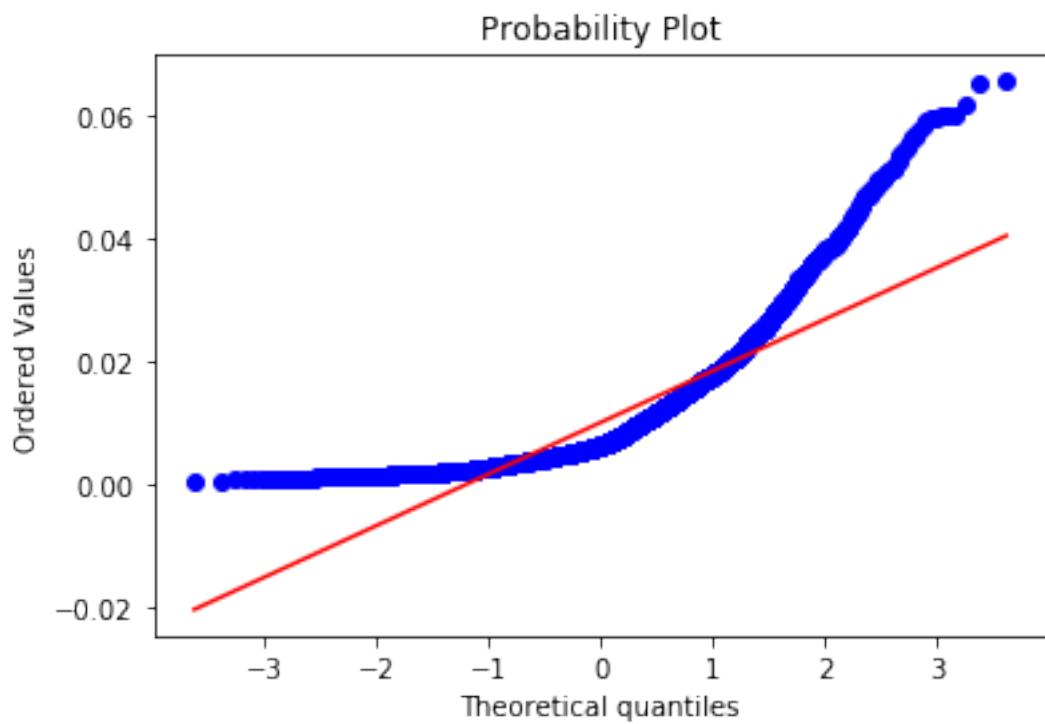
```
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [238]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

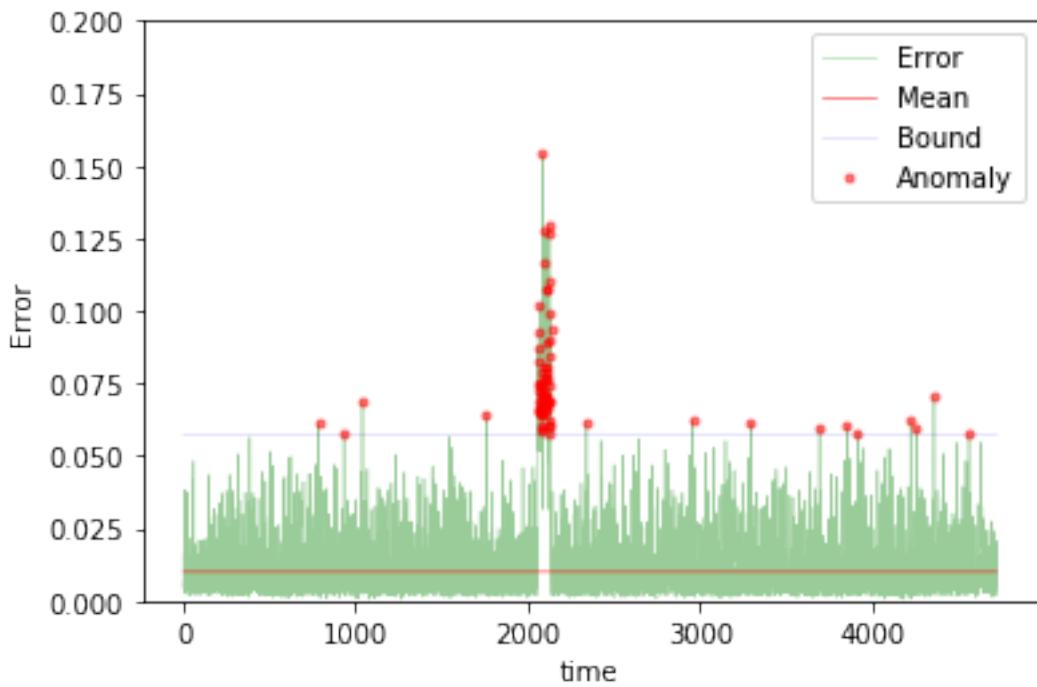
```
In [239]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



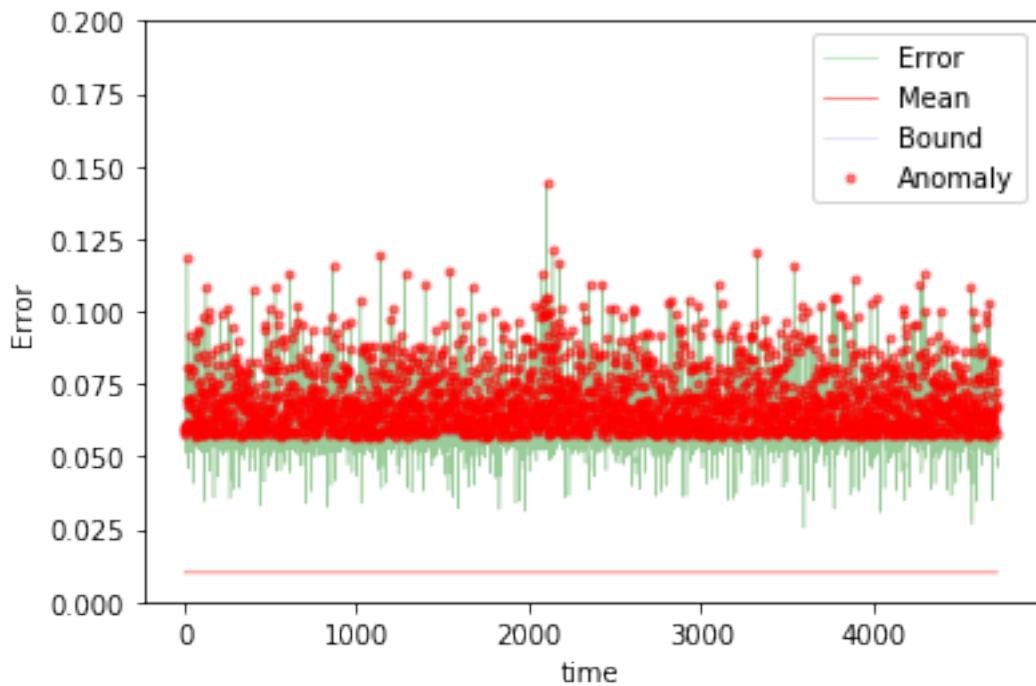
```
Training loss for final epoch is 0.010409382773330434
Validation loss for final epoch is 0.01077880627149716
----- Beginning tests for gru4_5 -----
Testing on normal data.
```



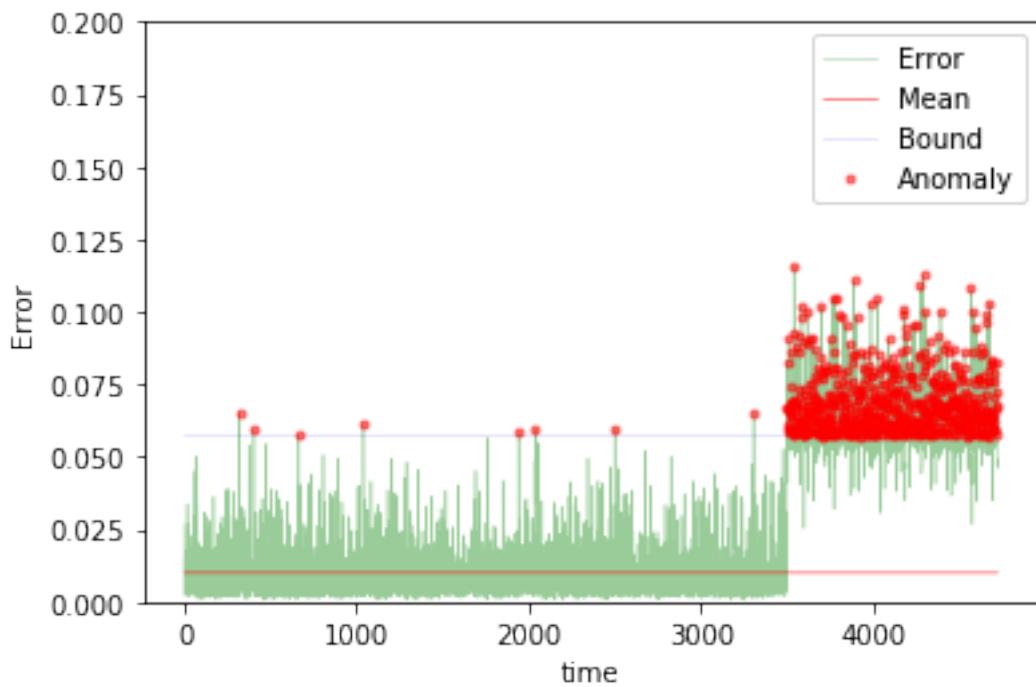
The mean error for gru4_5_normal_ is 0.01003813747334581 for length 4724
Testing on anomaly data.



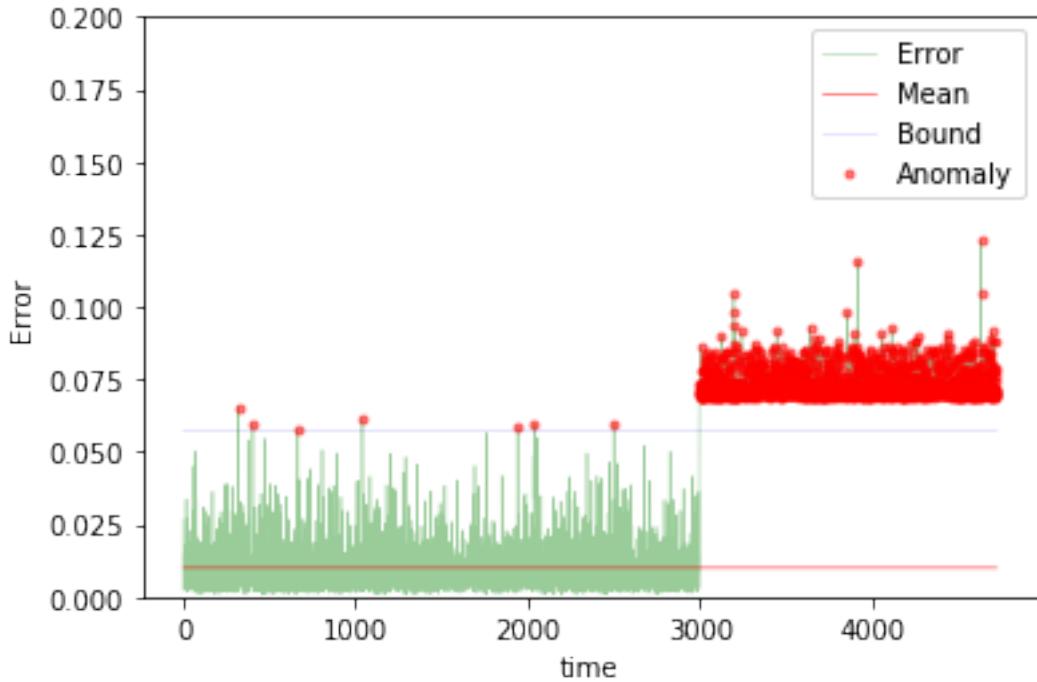
The mean error for gru4_5_anomaly_ is 0.012186676756417367 for length 4724
Testing on different app data.



The mean error for gru4_5_diff_app_ is 0.06196840630289877 for length 4724
Testing on App change synthetic data.



```
The mean error for gru4_5_app_change_ is 0.023411186556830804 for length 4724  
Testing on Net flood synthetic data.
```



```
The mean error for gru4_5_net_flood_ is 0.033312678610805765 for length 4724  
=====
```

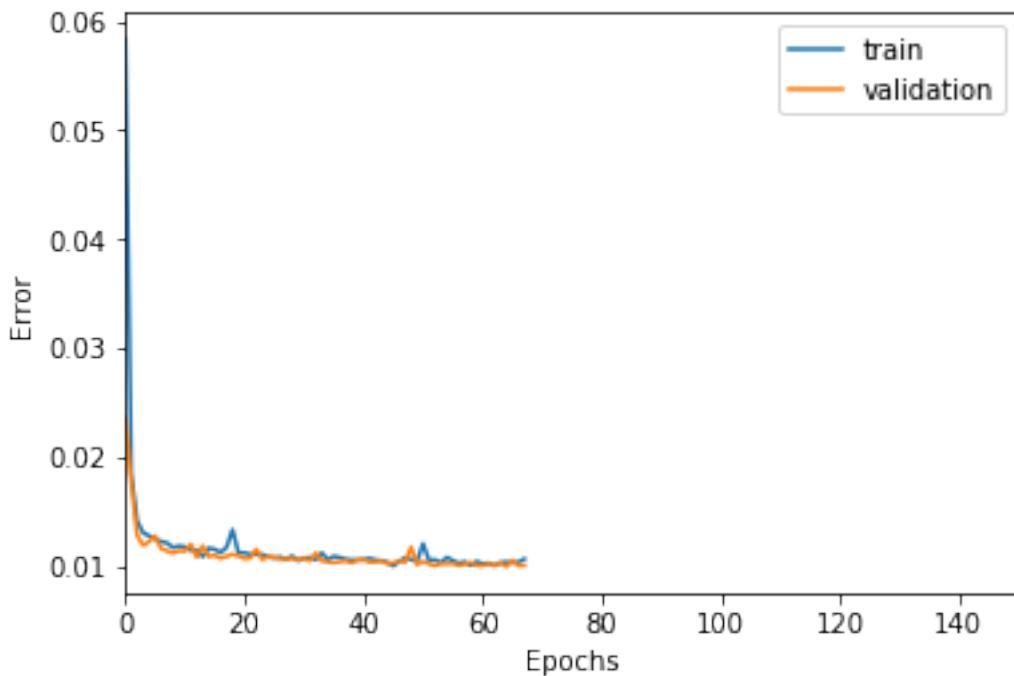
10 steps

```
In [240]: TIMESTEPS = 10  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS, 0)  
vgen = flat_generator(val_X, TIMESTEPS, 0)  
name = "gru4_10"  
  
In [241]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
```

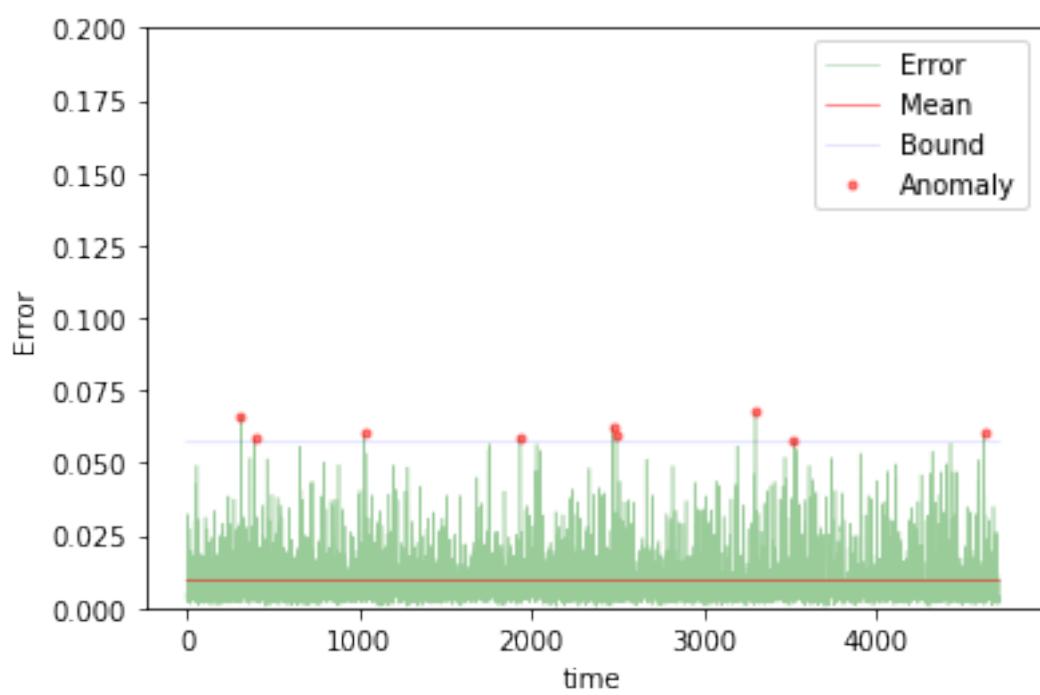
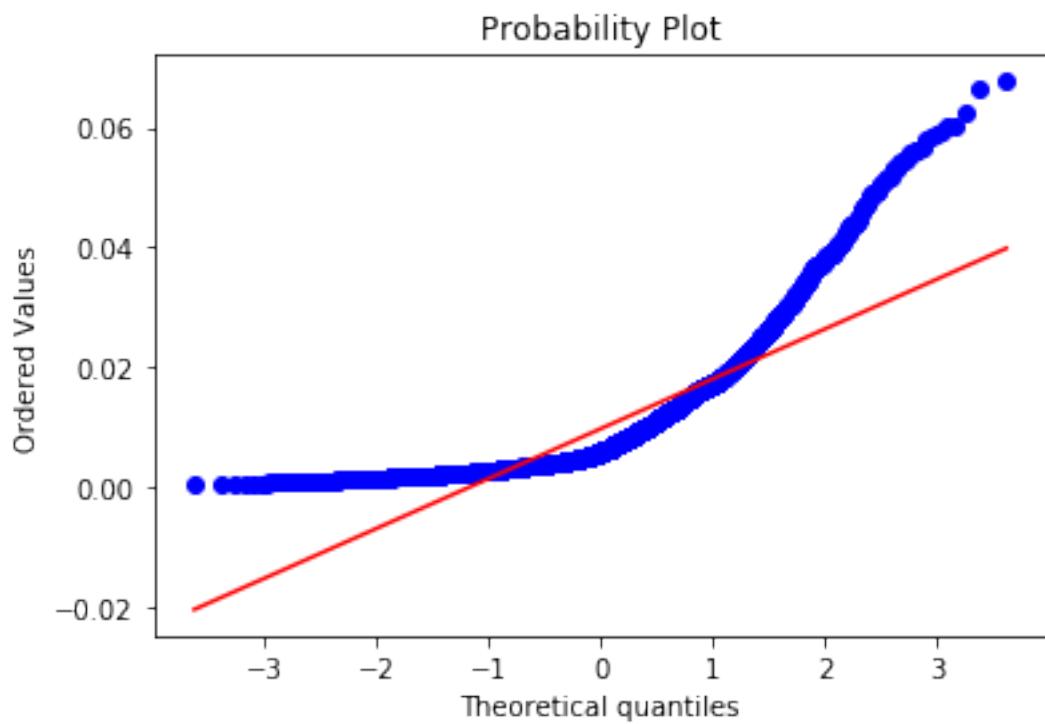
```
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [242]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

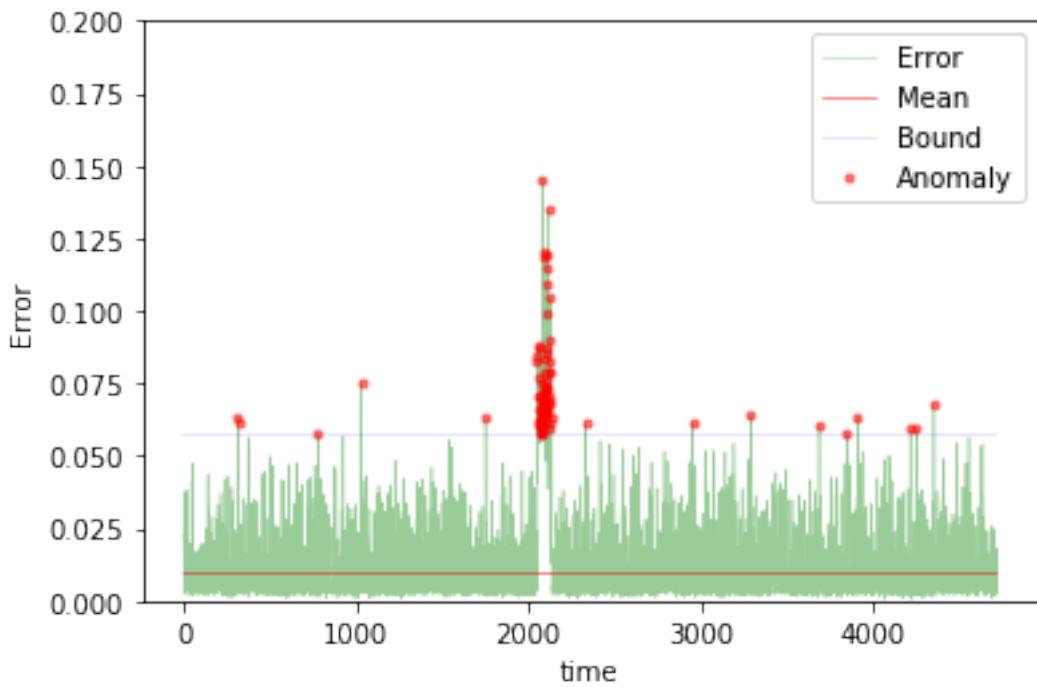
```
In [243]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



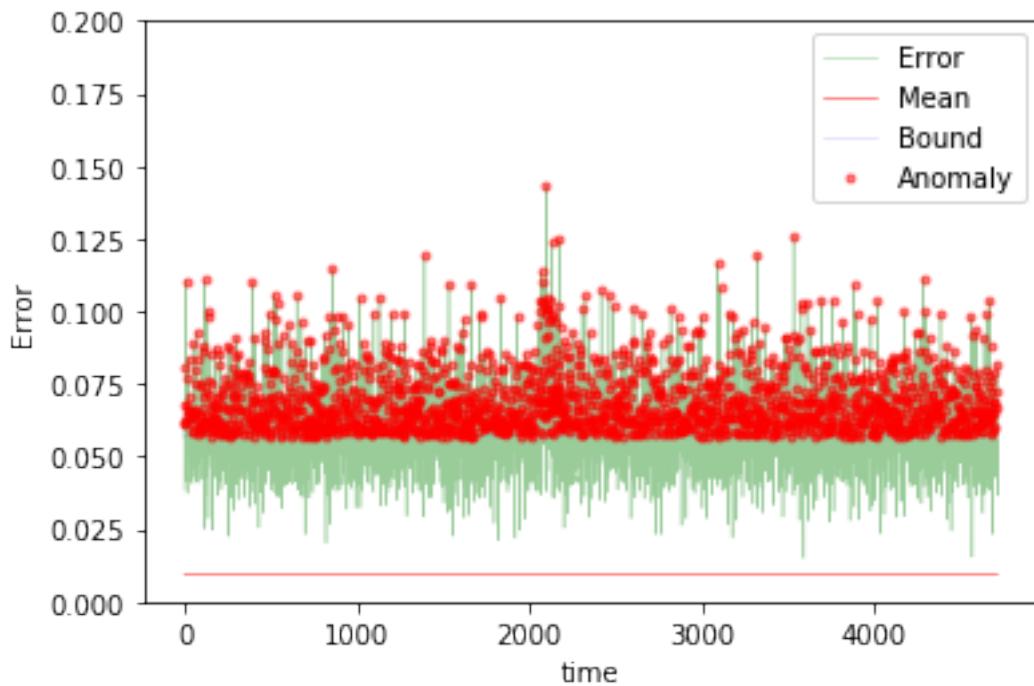
```
Training loss for final epoch is 0.010683064274257048
Validation loss for final epoch is 0.010086670933407732
----- Beginning tests for gru4_10 -----
Testing on normal data.
```



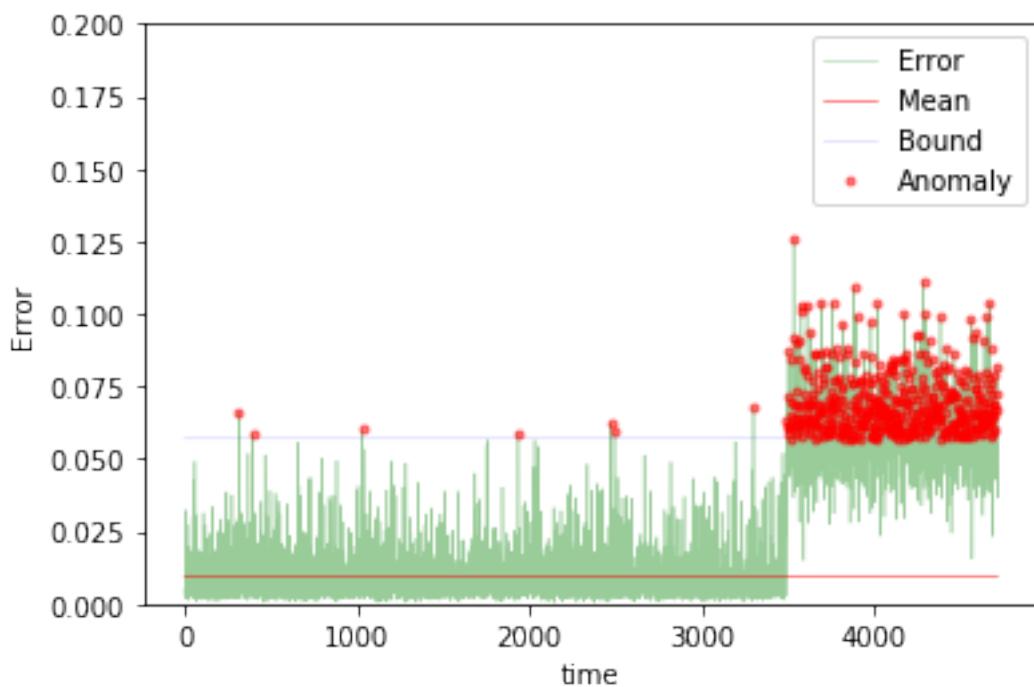
The mean error for gru4_10_normal_ is 0.009651217459220578 for length 4719
Testing on anomaly data.



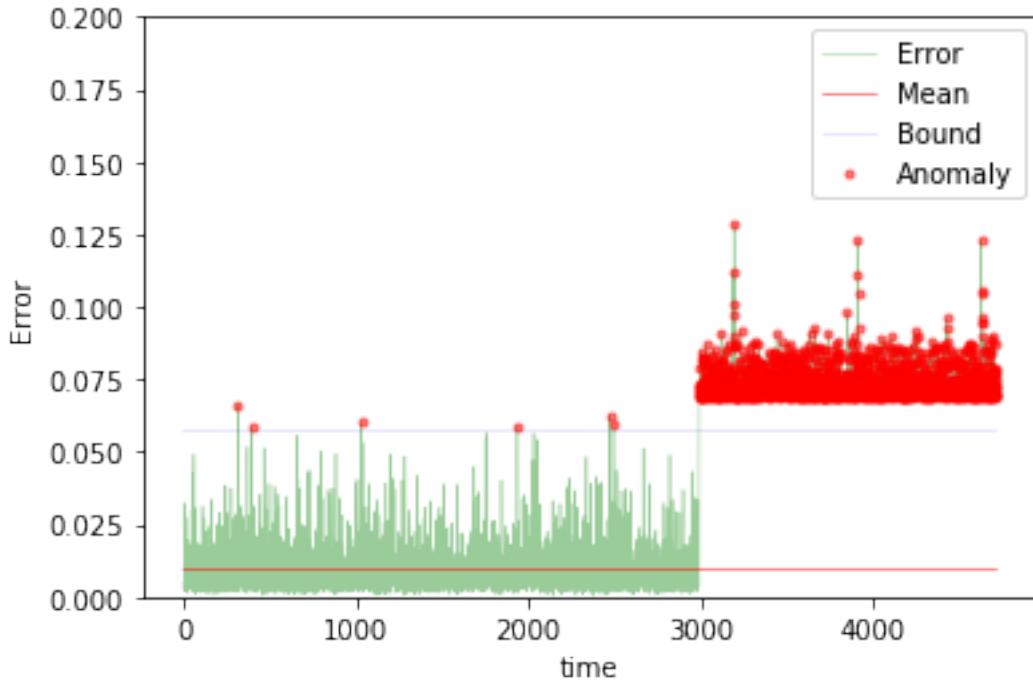
The mean error for gru4_10_anomaly_ is 0.011701236978110123 for length 4719
Testing on different app data.



The mean error for gru4_10_diff_app_ is 0.05776261756069235 for length 4719
Testing on App change synthetic data.



```
The mean error for gru4_10_app_change_ is 0.022013824380294315 for length 4719  
Testing on Net flood synthetic data.
```



```
The mean error for gru4_10_net_flood_ is 0.03332792971367762 for length 4719  
=====
```

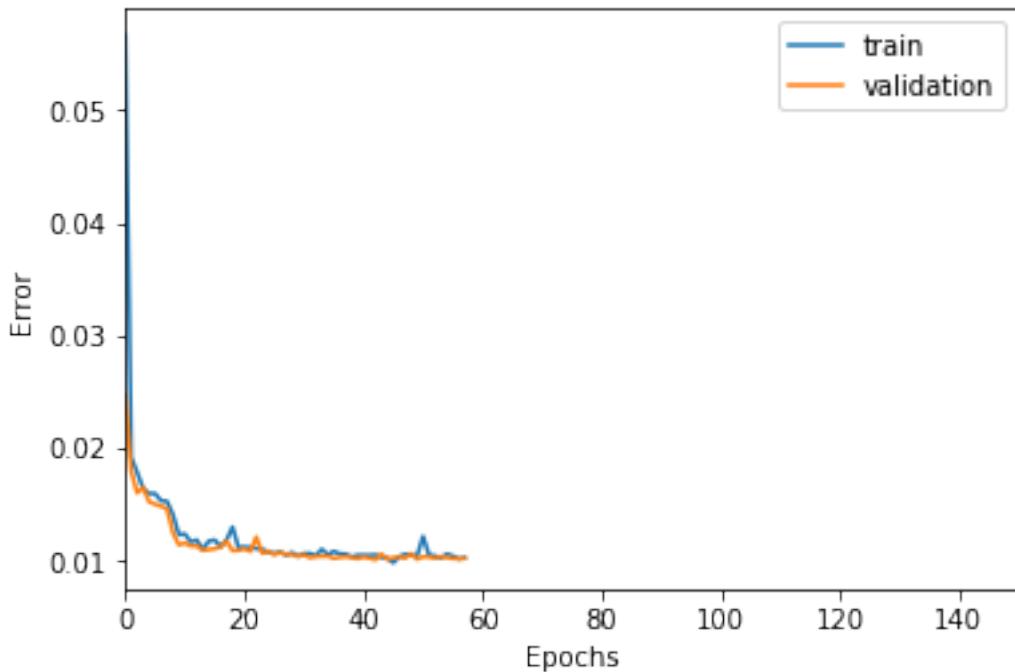
20 steps

```
In [244]: TIMESTEPS = 20  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru4_20"  
  
In [245]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
```

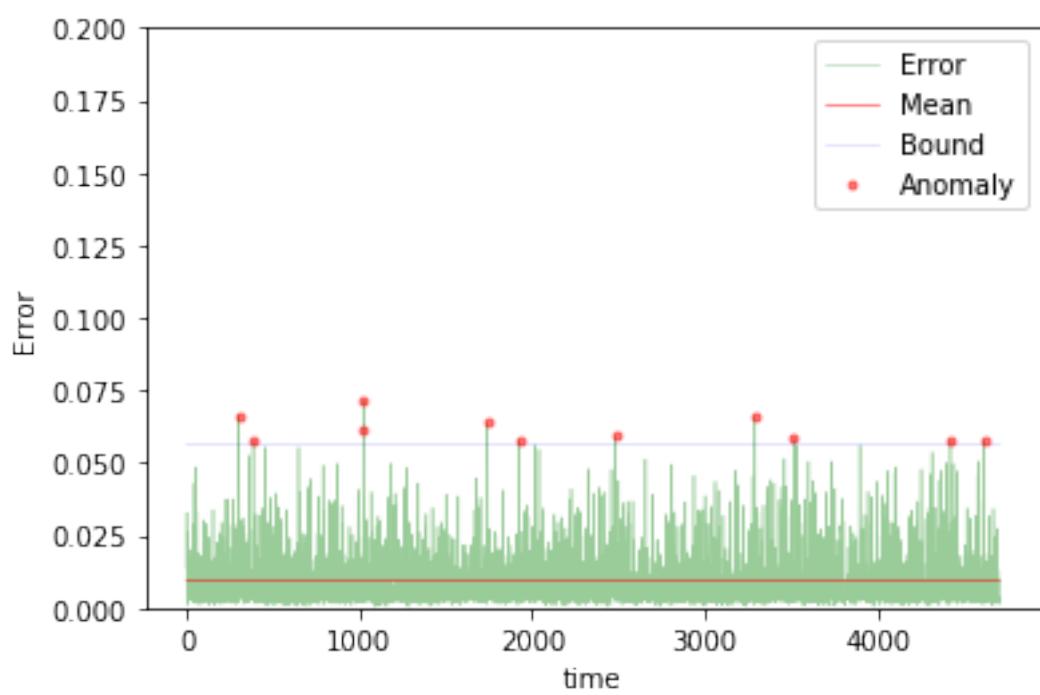
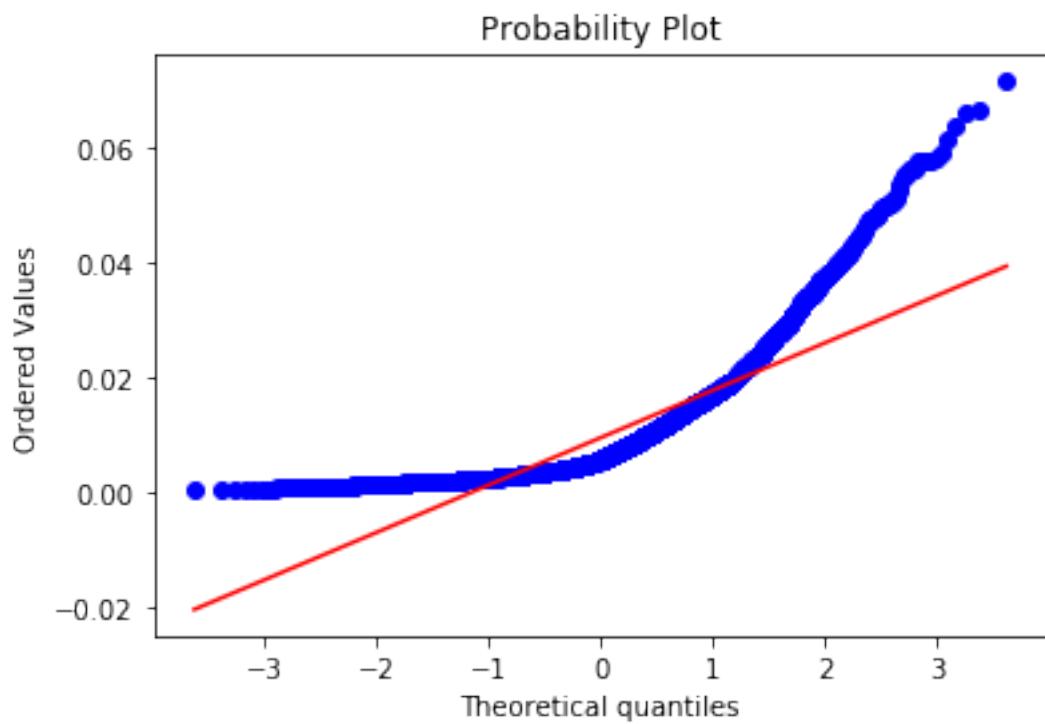
```
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [246]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

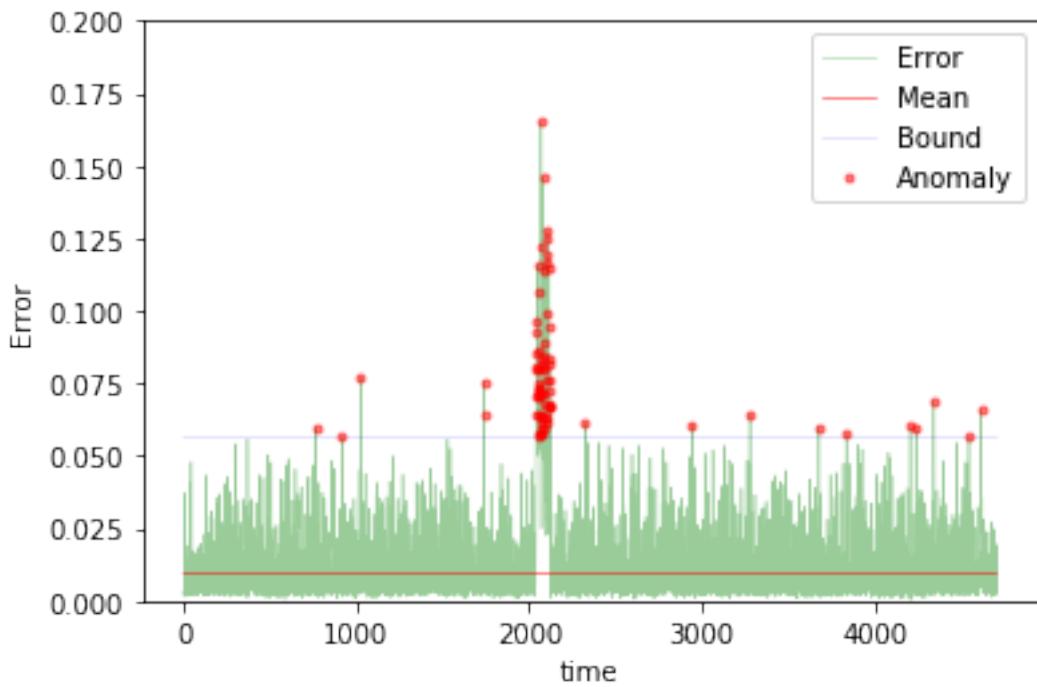
```
In [247]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



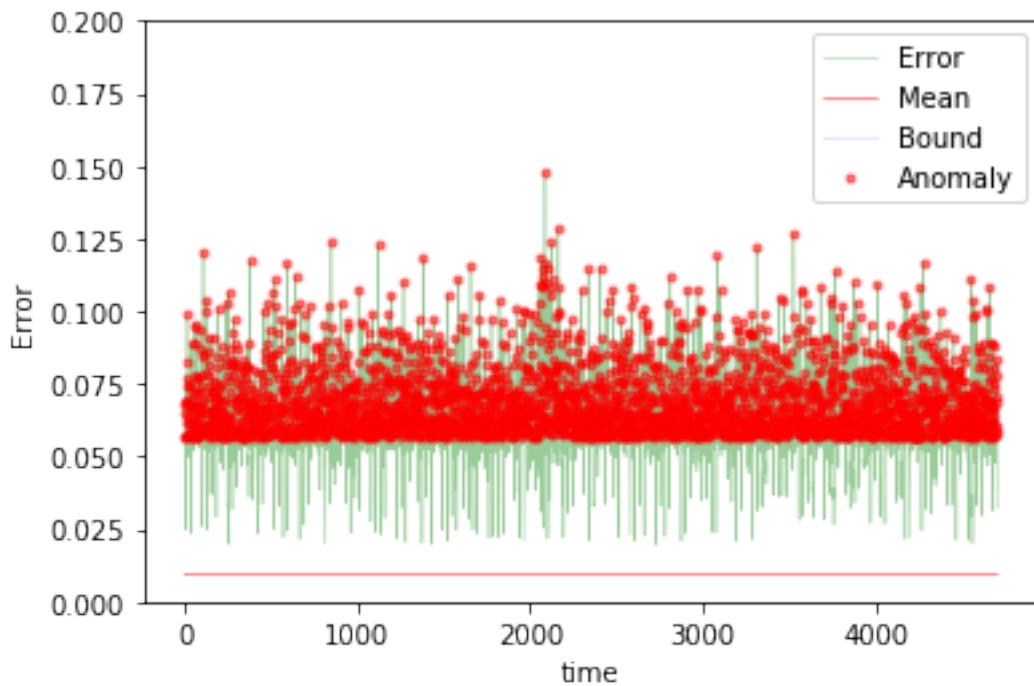
```
Training loss for final epoch is 0.010264529771753586
Validation loss for final epoch is 0.010206077374401502
----- Beginning tests for gru4_20 -----
Testing on normal data.
```



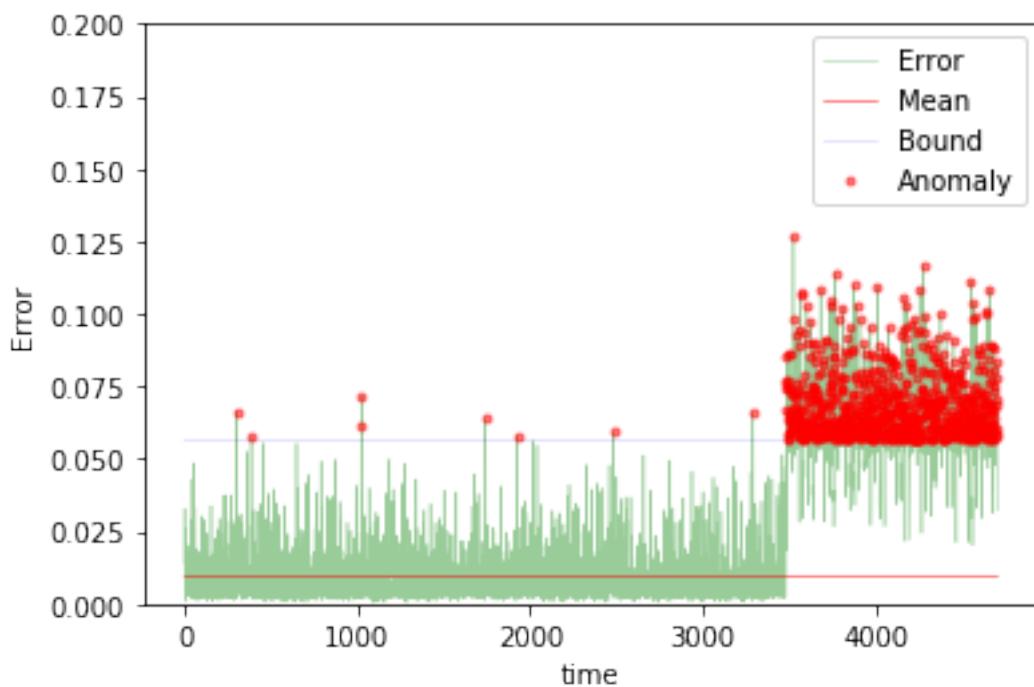
The mean error for gru4_20_normal_ is 0.009465857017430893 for length 4709
Testing on anomaly data.



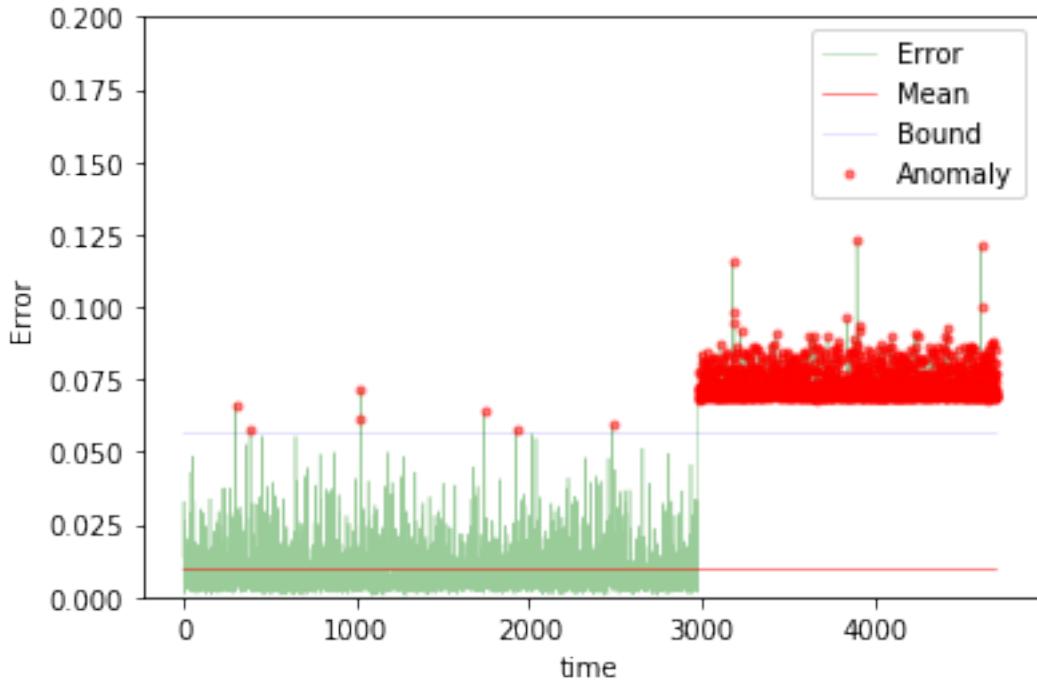
The mean error for gru4_20_anomaly_ is 0.011698425969181317 for length 4709
Testing on different app data.



The mean error for gru4_20_diff_app_ is 0.062241848138520454 for length 4709
Testing on App change synthetic data.



```
The mean error for gru4_20_app_change_ is 0.0231465008160676 for length 4709  
Testing on Net flood synthetic data.
```



```
The mean error for gru4_20_net_flood_ is 0.03291609516208223 for length 4709  
=====
```

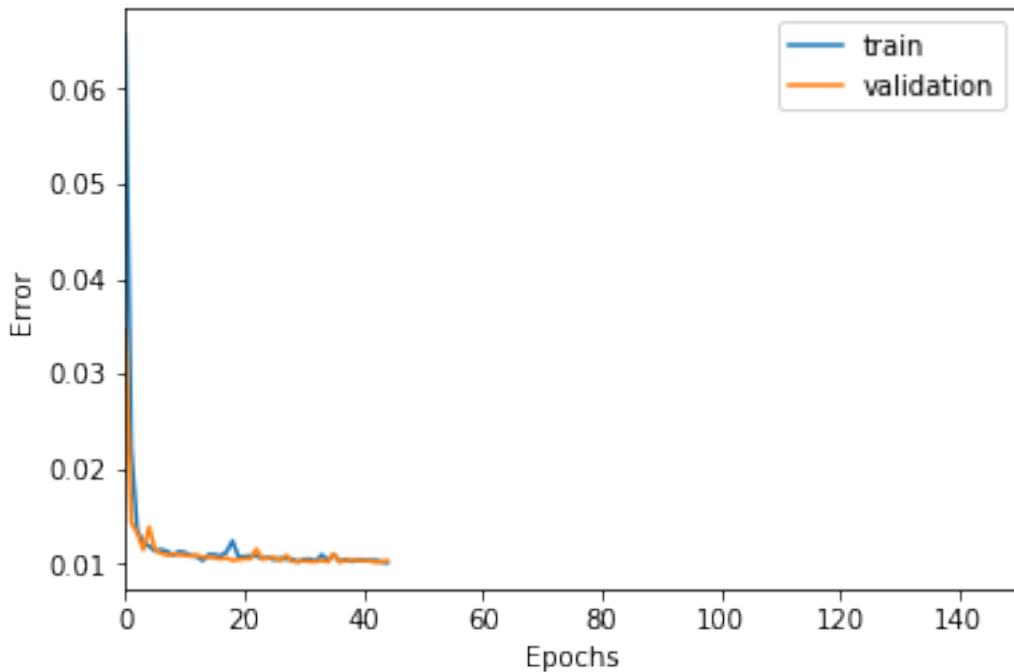
50 steps

```
In [248]: TIMESTEPS = 50  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru4_50"  
  
In [249]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
```

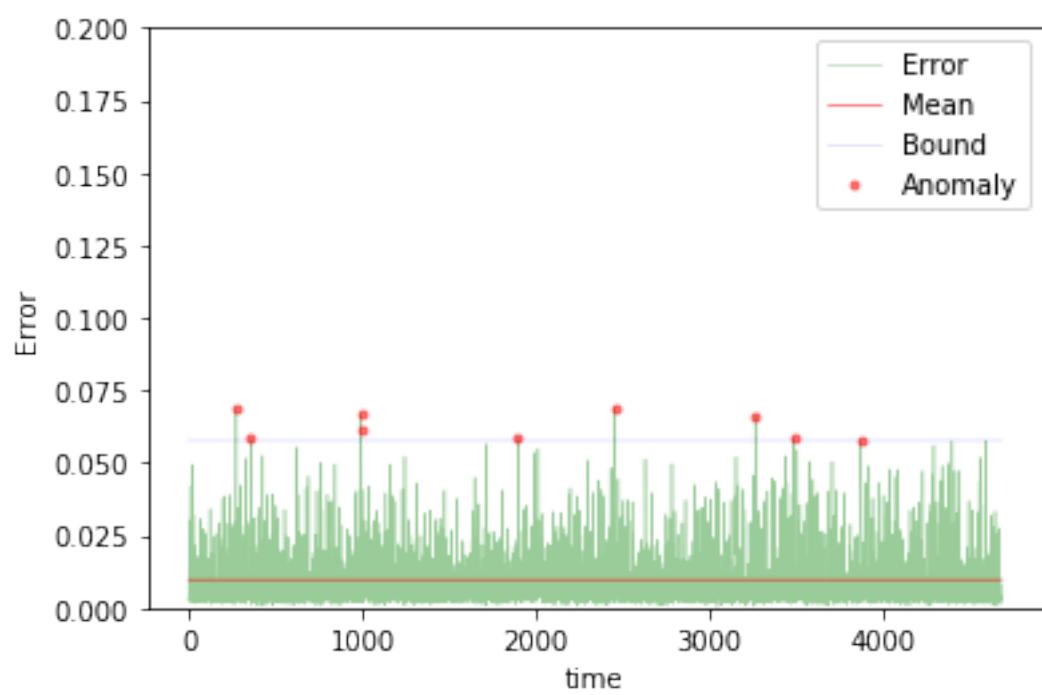
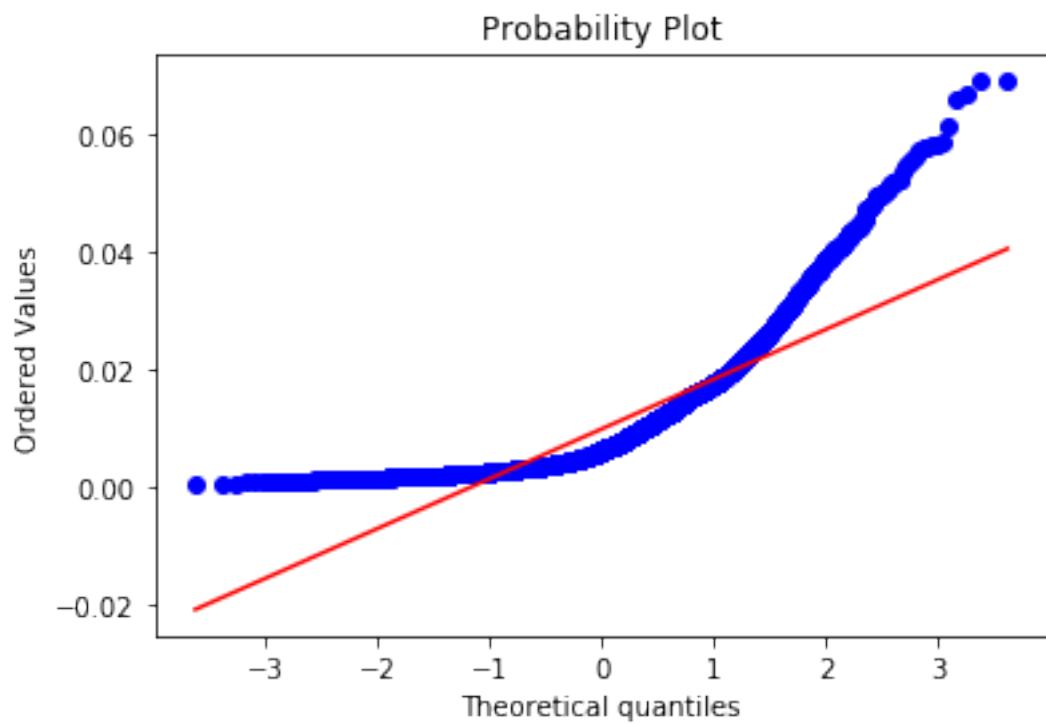
```
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [250]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

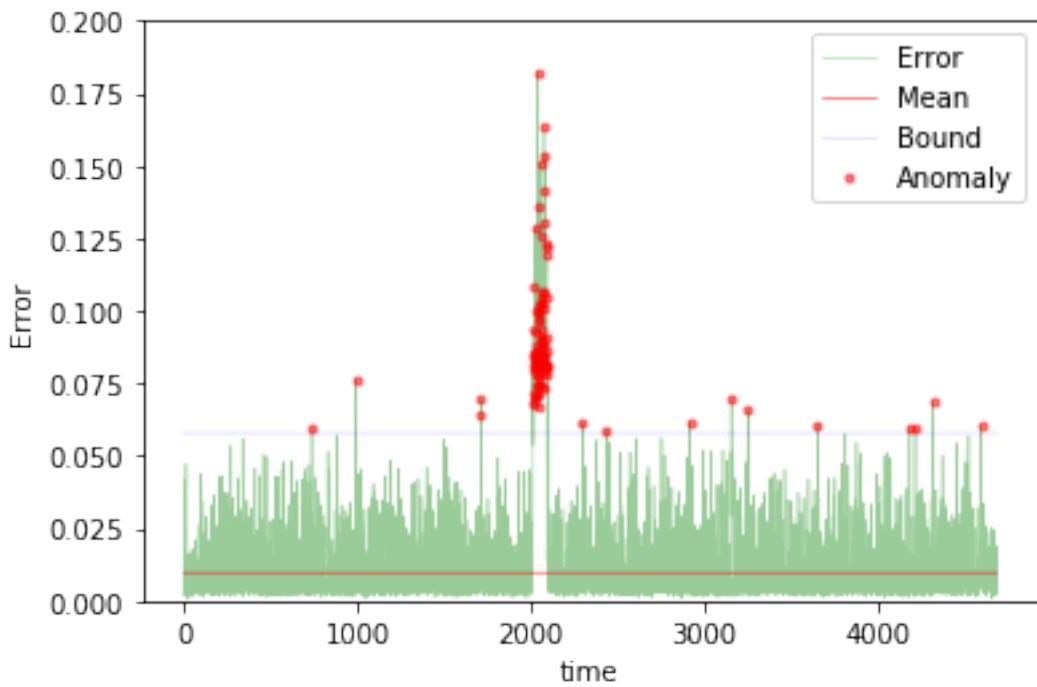
```
In [251]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



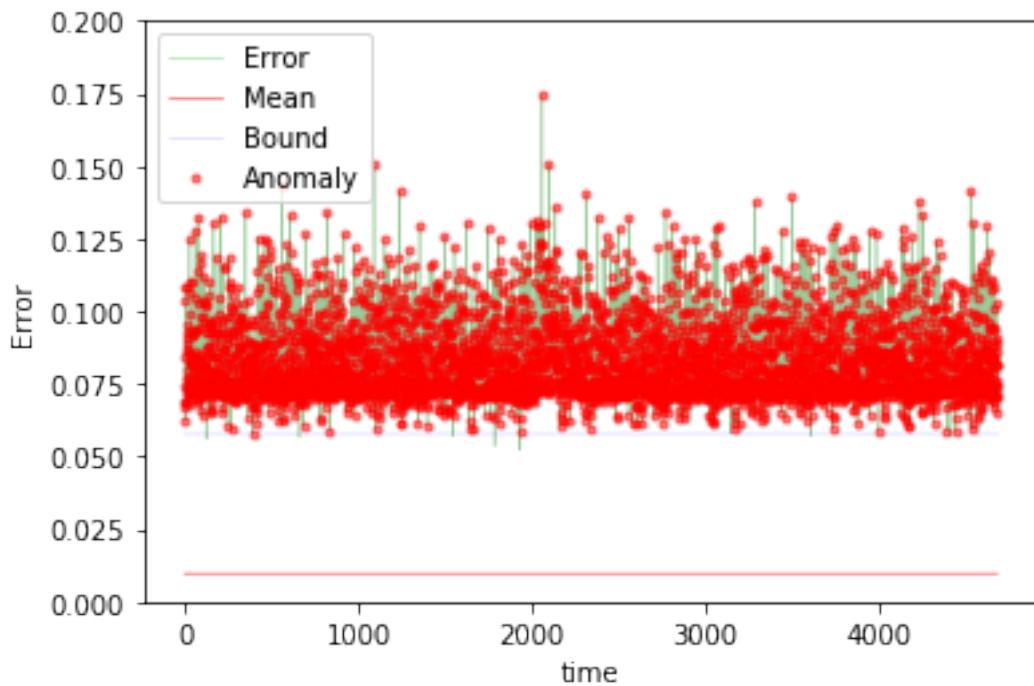
```
Training loss for final epoch is 0.010070379585493356
Validation loss for final epoch is 0.01029549436084926
----- Beginning tests for gru4_50 -----
Testing on normal data.
```



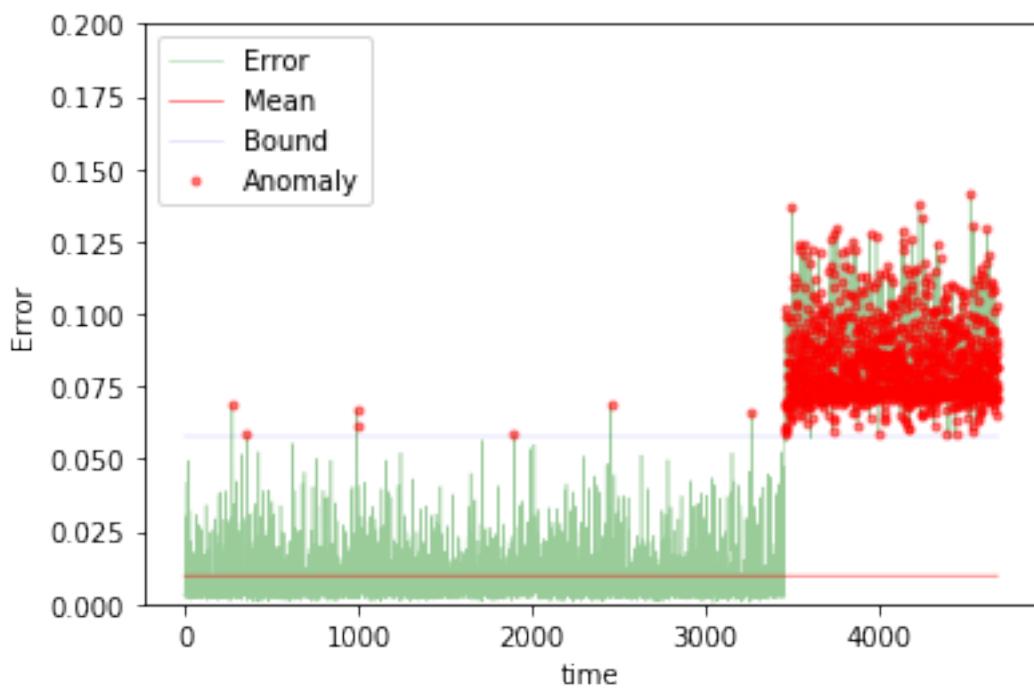
The mean error for gru4_50_normal_ is 0.009797088992185453 for length 4679
Testing on anomaly data.



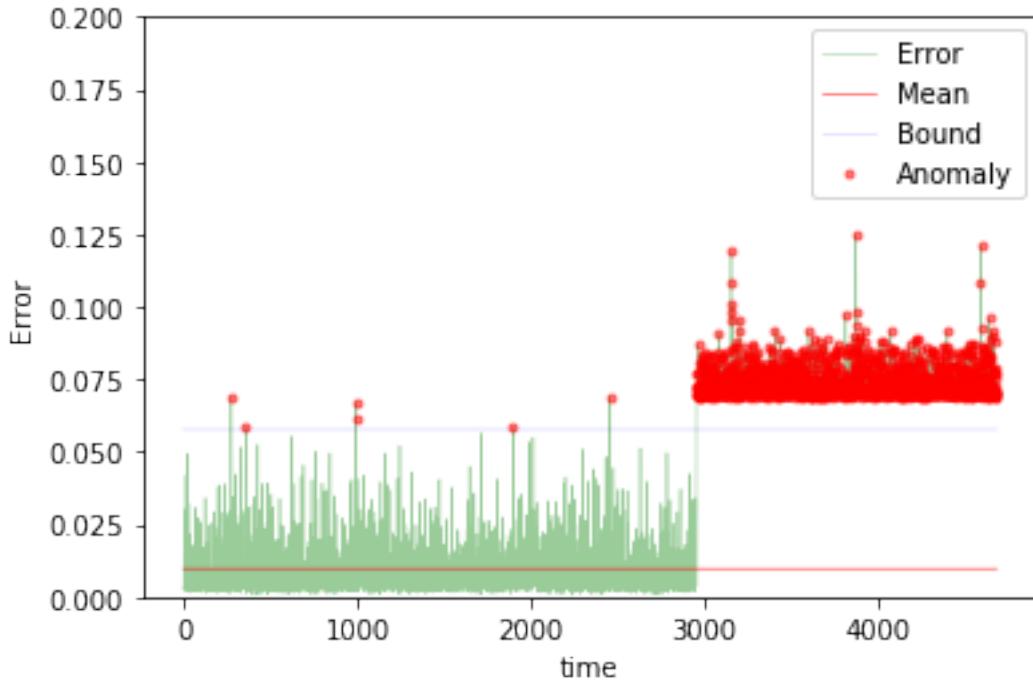
The mean error for gru4_50_anomaly_ is 0.012370336823299457 for length 4679
Testing on different app data.



The mean error for gru4_50_diff_app_ is 0.08308887190681004 for length 4679
Testing on App change synthetic data.



```
The mean error for gru4_50_app_change_ is 0.028834189293665154 for length 4679  
Testing on Net flood synthetic data.
```



```
The mean error for gru4_50_net_flood_ is 0.03345114819802997 for length 4679  
=====
```

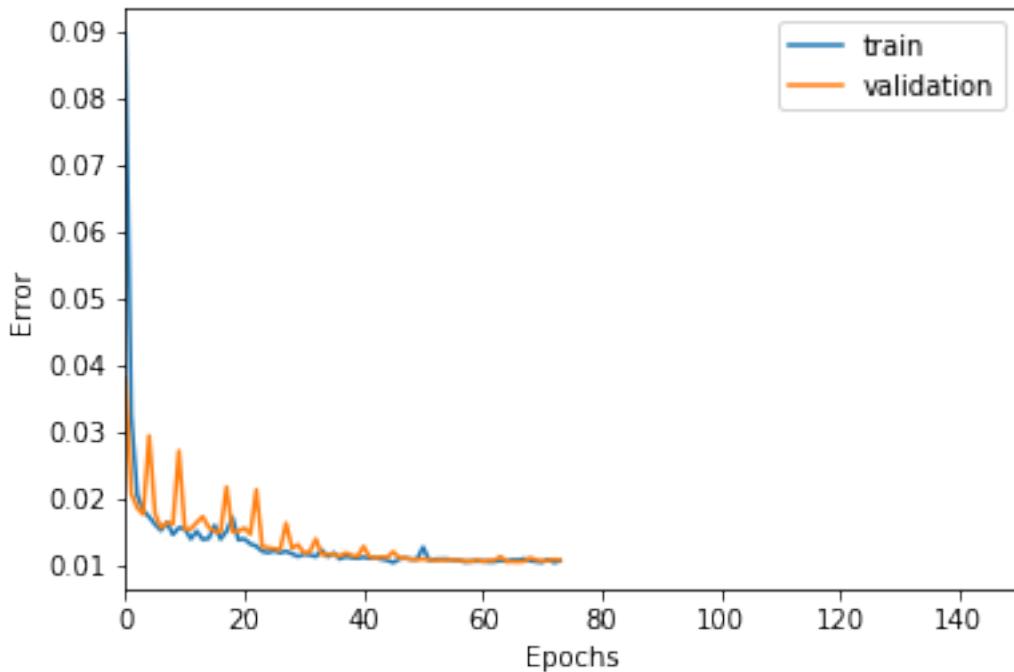
100 steps

```
In [252]: TIMESTEPS = 100  
DIM = 29  
tgen = flat_generator(X, TIMESTEPS,0)  
vgen = flat_generator(val_X, TIMESTEPS,0)  
name = "gru4_100"  
  
In [253]: input_layer = Input(shape=(TIMESTEPS,DIM))  
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)  
hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
```

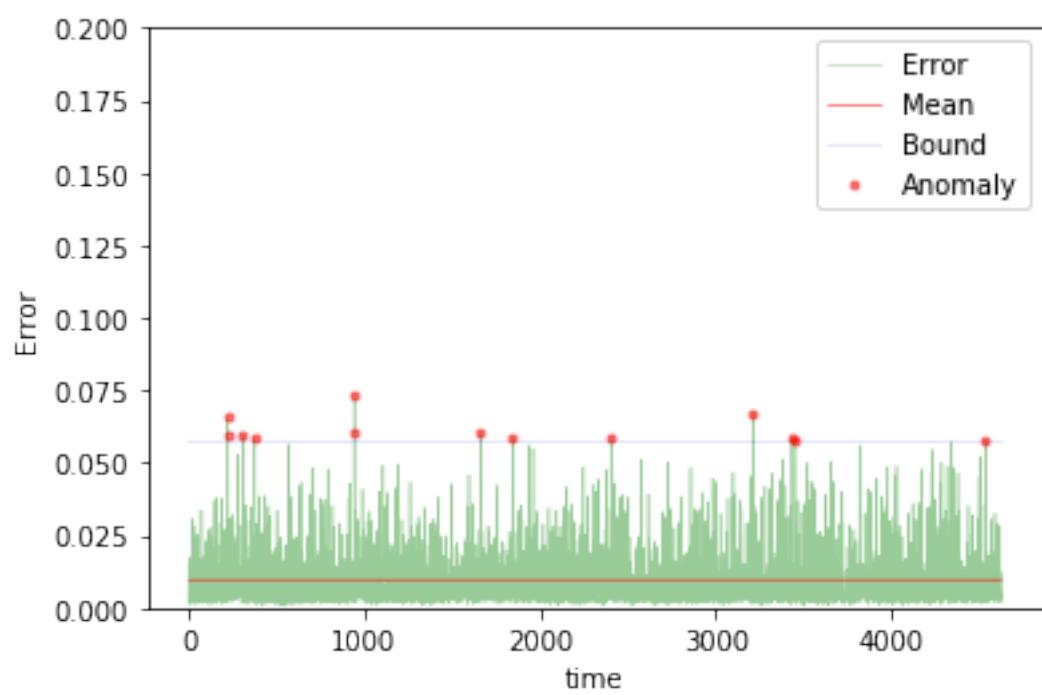
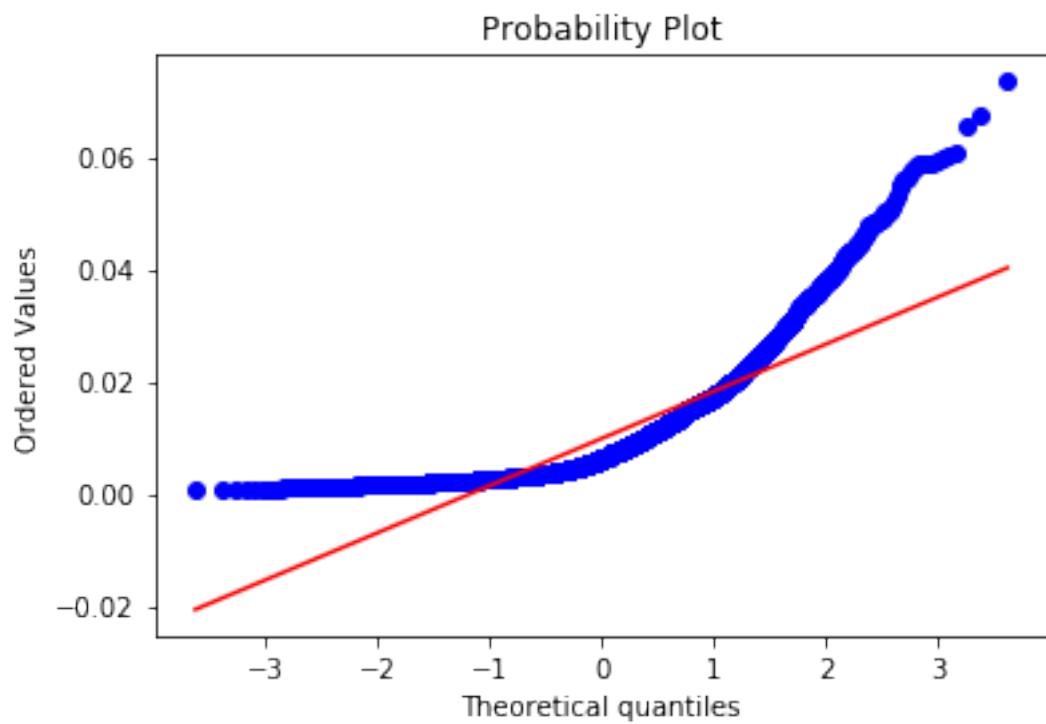
```
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [254]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

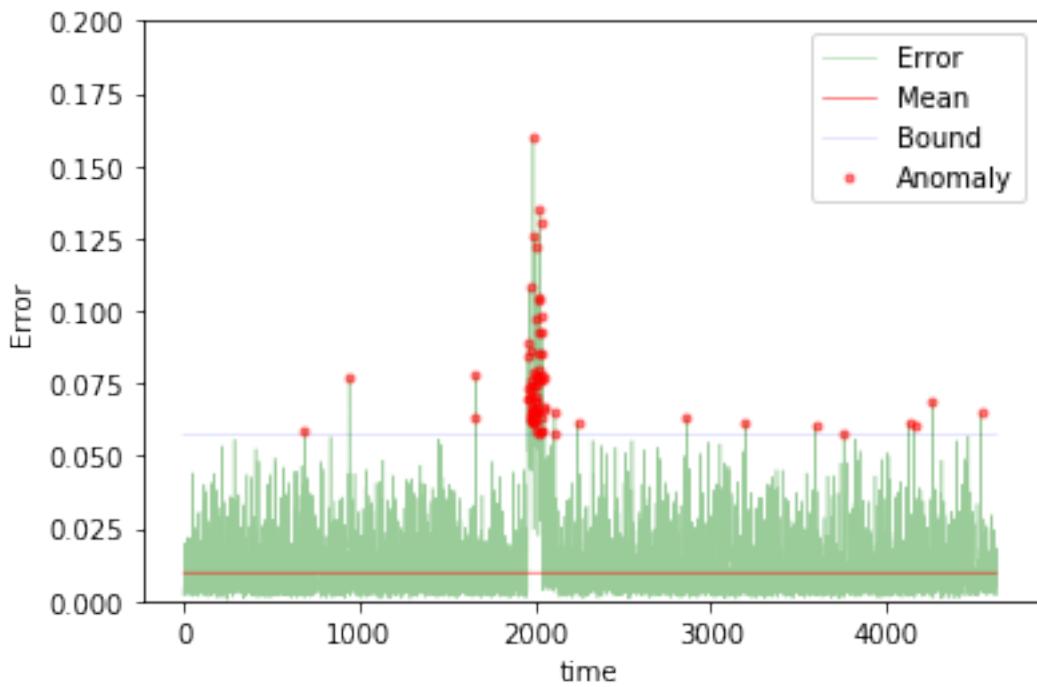
```
In [255]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



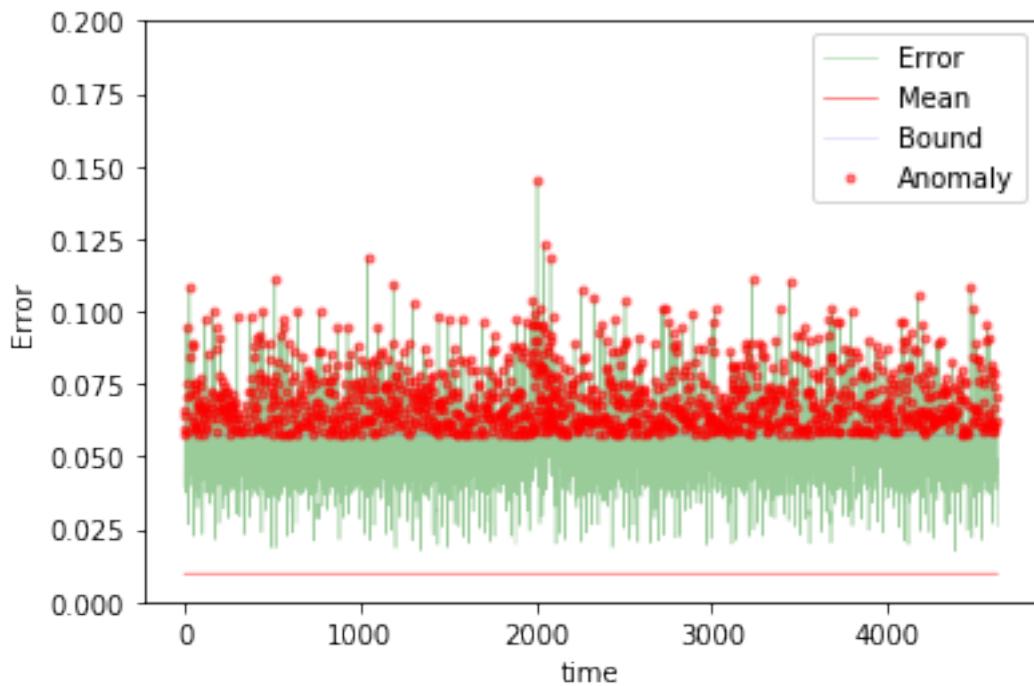
```
Training loss for final epoch is 0.010521025884663686
Validation loss for final epoch is 0.010639392620418221
----- Beginning tests for gru4_100 -----
Testing on normal data.
```



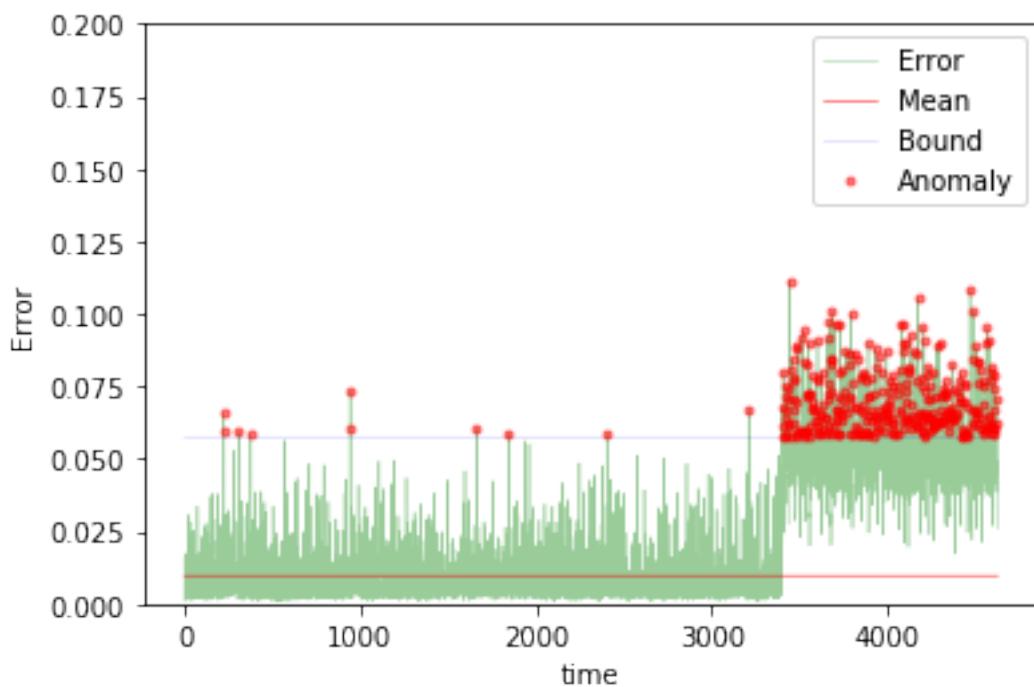
The mean error for gru4_100_normal_ is 0.009770706908408096 for length 4629
Testing on anomaly data.



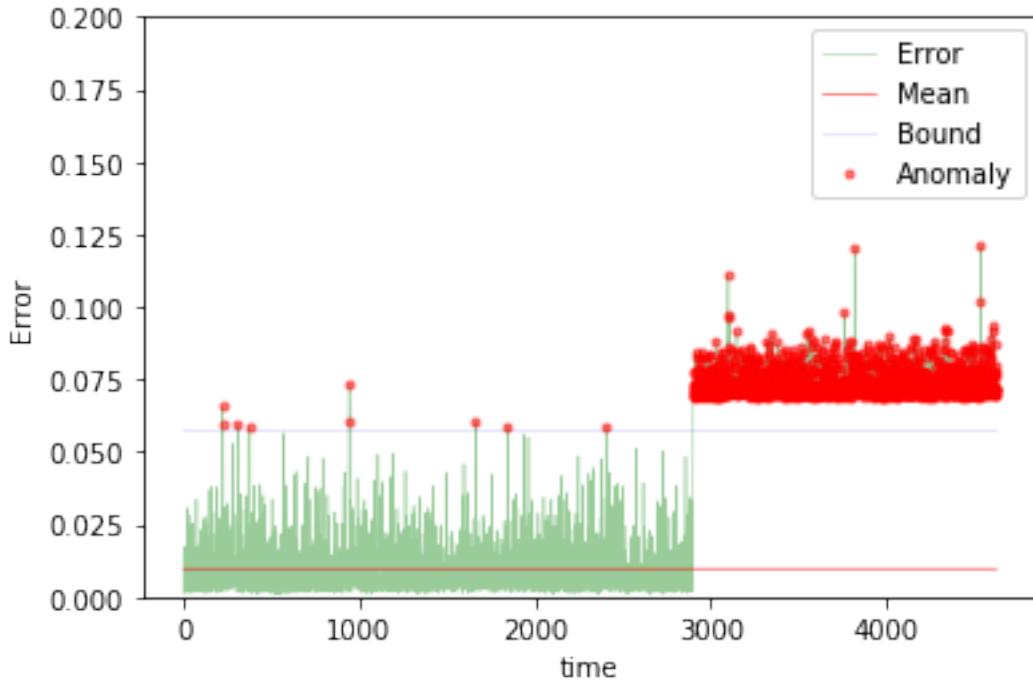
The mean error for gru4_100_anomaly_ is 0.012183110505949484 for length 4629
Testing on different app data.



The mean error for gru4_100_diff_app_ is 0.05211465311609195 for length 4629
Testing on App change synthetic data.



```
The mean error for gru4_100_app_change_ is 0.020882494135038846 for length 4629
Testing on Net flood synthetic data.
```



```
The mean error for gru4_100_net_flood_ is 0.03371087823435752 for length 4629
=====
=====
```

200 steps

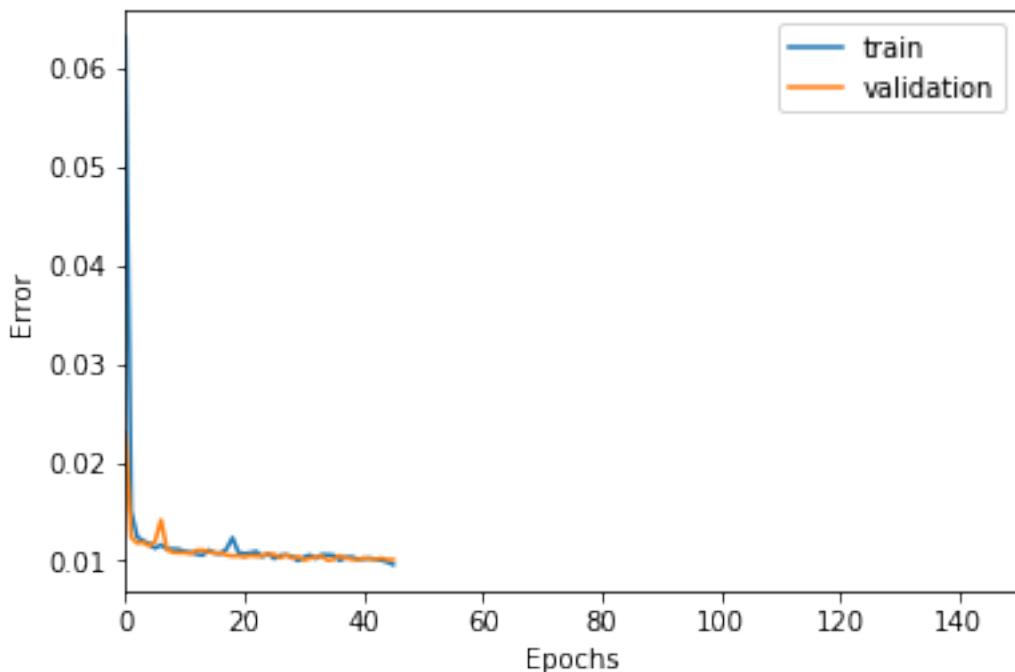
```
In [256]: TIMESTEPS = 200
DIM = 29
tgen = flat_generator(X, TIMESTEPS,0)
vgen = flat_generator(val_X, TIMESTEPS,0)
name = "gru4_200"

In [257]: input_layer = Input(shape=(TIMESTEPS,DIM))
hidden = GRU(10, activation='relu', return_sequences=True)(input_layer)
hidden = GRU(7, activation='relu', return_sequences=True)(hidden)
```

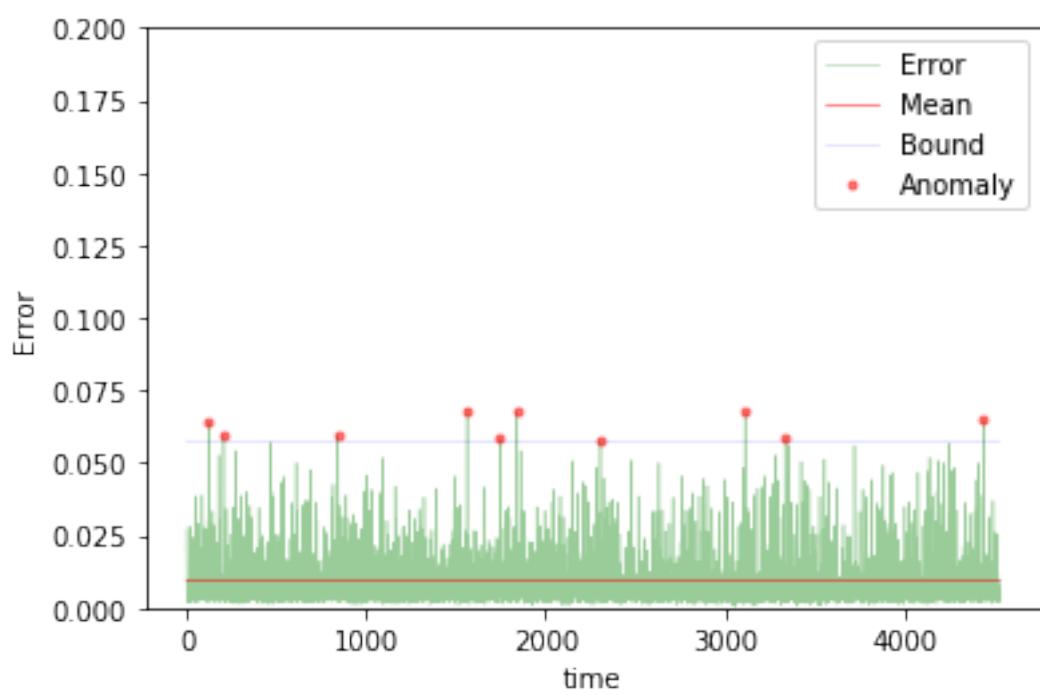
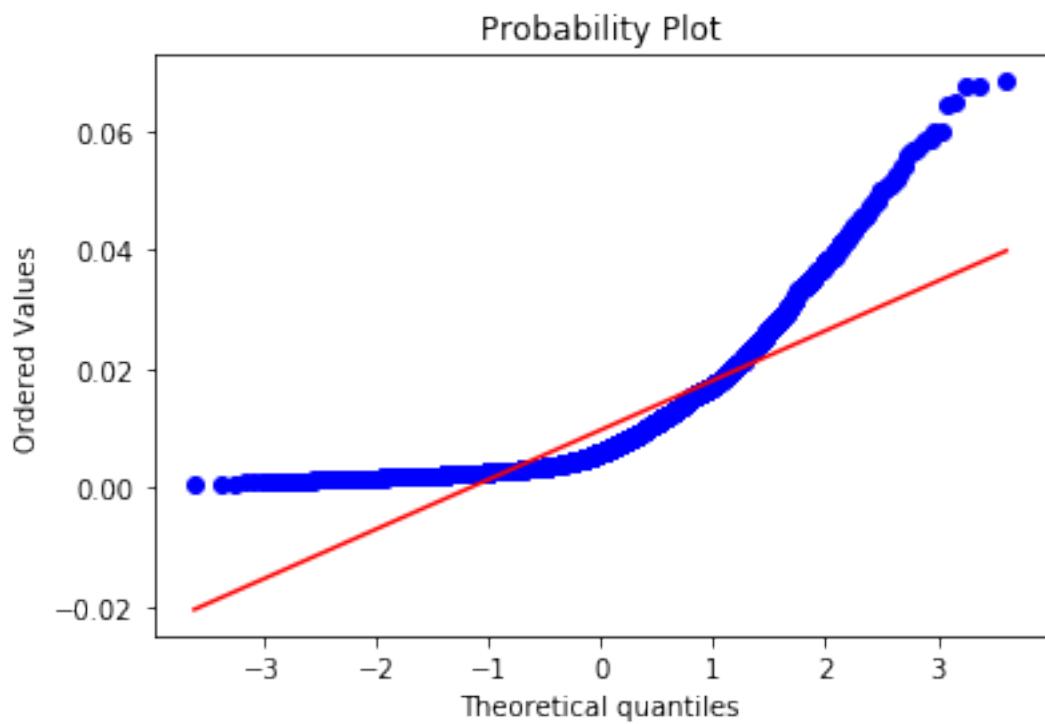
```
hidden = GRU(5, activation='relu', return_sequences=True)(hidden)
hidden = GRU(DIM, activation='relu')(hidden)
output = Dense(DIM, activation='sigmoid')(hidden)
```

```
In [258]: model = Model(input_layer, output)
model.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mae'])
```

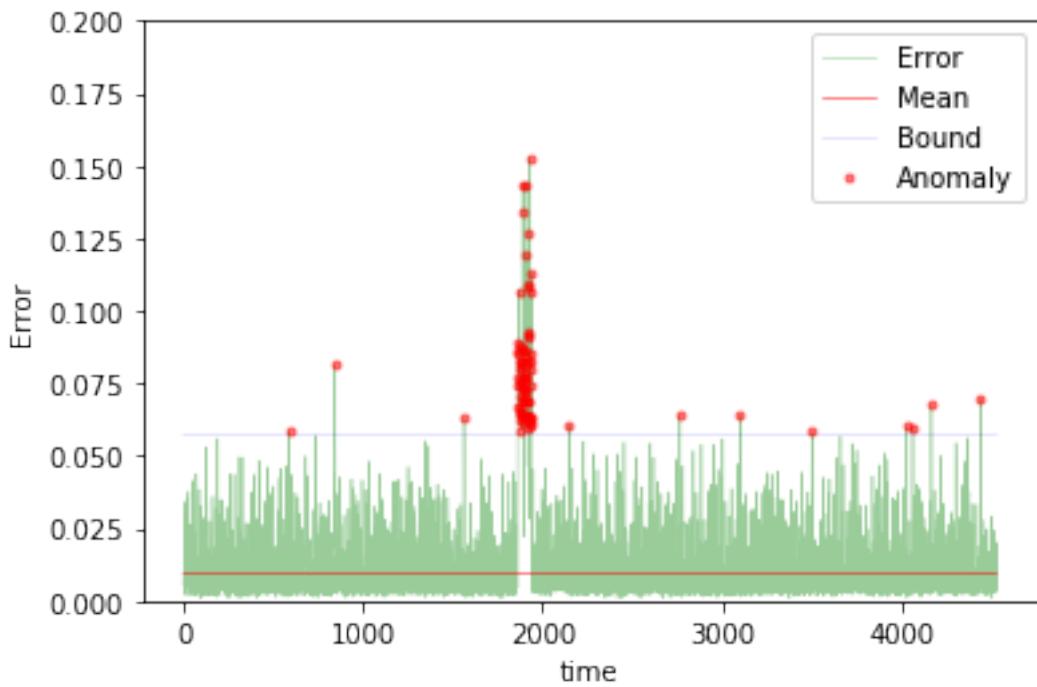
```
In [259]: train(model, tgen, vgen, name=name)
test(model, ravel=0, name=name, window=TIMESTEPS)
```



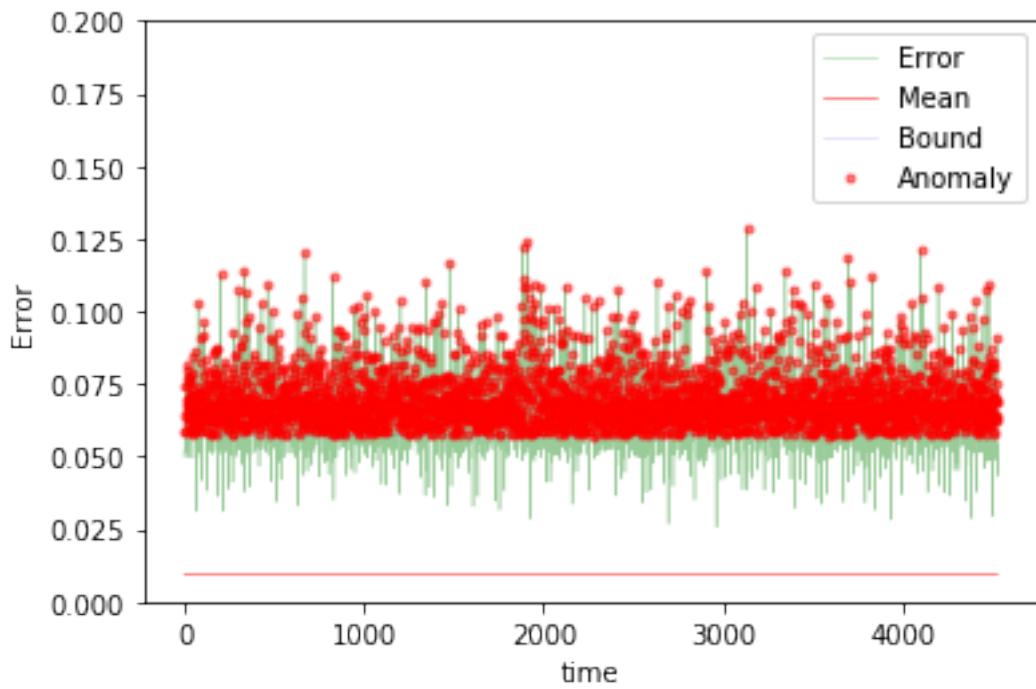
```
Training loss for final epoch is 0.009608839808148332
Validation loss for final epoch is 0.010123777986969798
----- Beginning tests for gru4_200 -----
Testing on normal data.
```



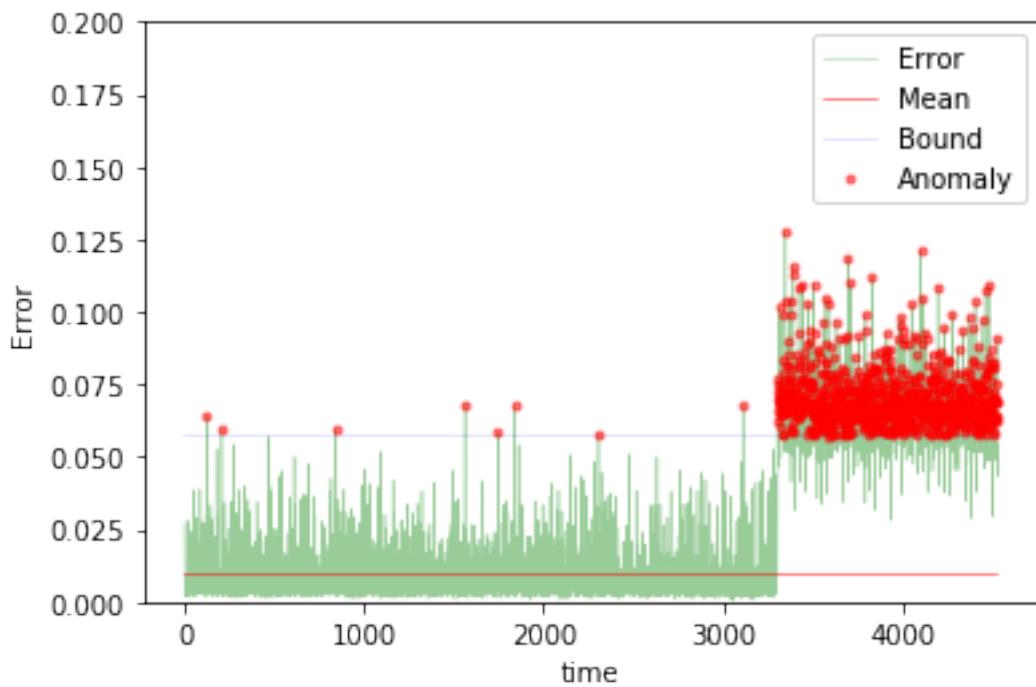
The mean error for gru4_200_normal_ is 0.009662958498385273 for length 4529
Testing on anomaly data.



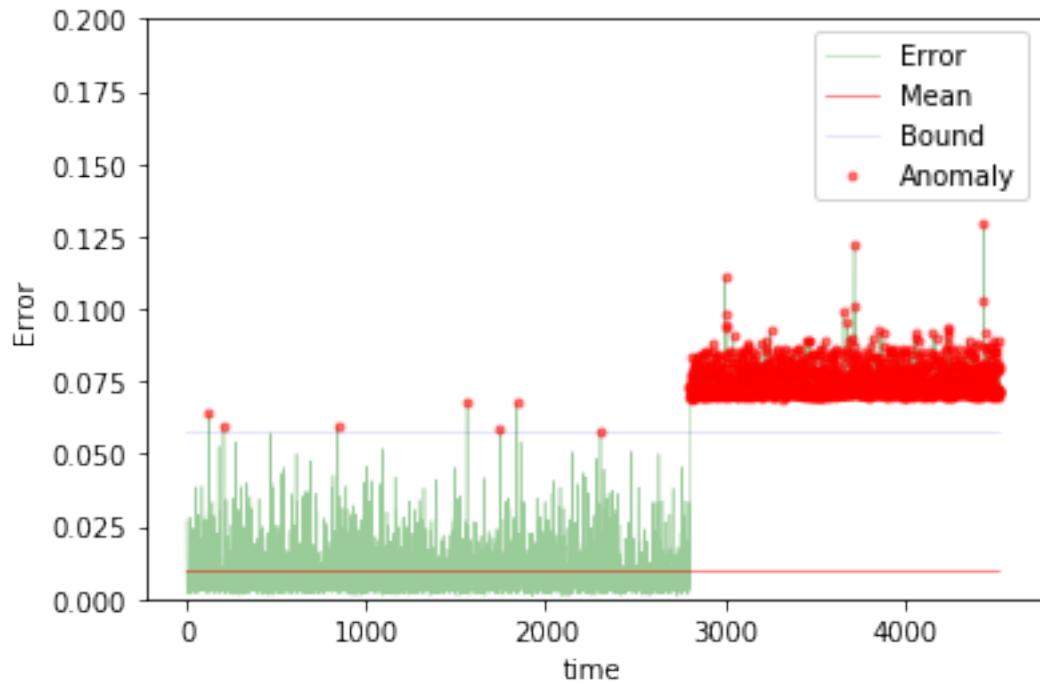
The mean error for gru4_200_anomaly_ is 0.011828274475735296 for length 4529
Testing on different app data.



The mean error for gru4_200_diff_app_ is 0.06594479441771446 for length 4529
Testing on App change synthetic data.



The mean error for gru4_200_app_change_ is 0.02492136200969809 for length 4529
Testing on Net flood synthetic data.



The mean error for gru4_200_net_flood_ is 0.03453525987174354 for length 4529
=====