Tim/AE_sym_TD

```
%sh

echo $PWD
rm TXLoader.py*
wget --no-check-certificate --no-cache --no-cookies https://raw.githubusercontent.com/tianle91
ls -l --block-size=M
```

FINISHED

```
--2019-03-01 22:39:48--  https://raw.githubusercontent.com/tianle91/forecastor/master/TXLoade
r.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... /home/hadoop
151.101.248.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.248.133|:443... co
nnected.
HTTP request sent, awaiting response... 200 OK
Length: 3820 (3.7K) [text/plain]
Saving to: ‘TXLoader.py’
     0K ...                                                   100% 62.7M=0s
2019-03-01 22:39:48 (62.7 MB/s) - ‘TXLoader.py’ saved [3820/3820]
total 2M
-rw-r--r-- 1 root   root   1M Mar  1 20:53 ae_1mo-1h_SYM:TD.h5
-rw-r--r-- 1 root   root   1M Mar  1 20:27 ae.h5
-rwxrwxr-x 1 hadoop hadoop 1M Mar  1 03:50 attach.sh
-rw-r--r-- 1 root   root   1M Mar  1 20:50 Book.py
-rw-rw-r-- 1 hadoop hadoop 1M Mar  1 03:50 complete.out
drwxr-xr-x 2 root   root   1M Mar  1 22:39 data
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:48 PM.

```
%sh

cd data
ls -ln --block-size=M
```

FINISHED

```
total 2M
-rw-r--r-- 1 0 0 1M Mar  1 22:36 1mo-1h_SYM:BMO_dates.pickle
-rw-r--r-- 1 0 0 1M Mar  1 20:59 1mo-1h_SYM:BMO_dt:2018-04-02 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:34 1mo-1h_SYM:BMO_dt:2018-04-02 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:04 1mo-1h_SYM:BMO_dt:2018-04-03 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:35 1mo-1h_SYM:BMO_dt:2018-04-03 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:10 1mo-1h_SYM:BMO_dt:2018-04-04 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:35 1mo-1h_SYM:BMO_dt:2018-04-04 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:14 1mo-1h_SYM:BMO_dt:2018-04-05 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:35 1mo-1h_SYM:BMO_dt:2018-04-05 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:20 1mo-1h_SYM:BMO_dt:2018-04-06 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:35 1mo-1h_SYM:BMO_dt:2018-04-06 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:25 1mo-1h_SYM:BMO_dt:2018-04-09 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:36 1mo-1h_SYM:BMO_dt:2018-04-09 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:29 1mo-1h_SYM:BMO_dt:2018-04-10 00:00:00_orders.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 22:36 1mo-1h_SYM:BMO_dt:2018-04-10 00:00:00_trades.pickle.gz
-rw-r--r-- 1 0 0 1M Mar  1 21:34 1mo-1h_SYM:BMO_dt:2018-04-11 00:00:00_orders.pickle.gz
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:48 PM.

FINISHED

```
%python

import os
import sys
import gzip
import pickle

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import preprocessing

#from TXLoader import TXLoader
exec(open(os.getcwd() + '/TXLoader.py').read())
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:48 PM.

FINISHED

```
%python

symbol = 'TD'
jobname = '1mo-1h'
dotraining = False
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:48 PM.

FINISHED

```
%python

xm = TXLoader(jobname=jobname, symbol=symbol).getxm(byday=False)
ntime, ncov = xm.shape
print ('xm.shape:', xm.shape)
print (xm[0, ...])
```

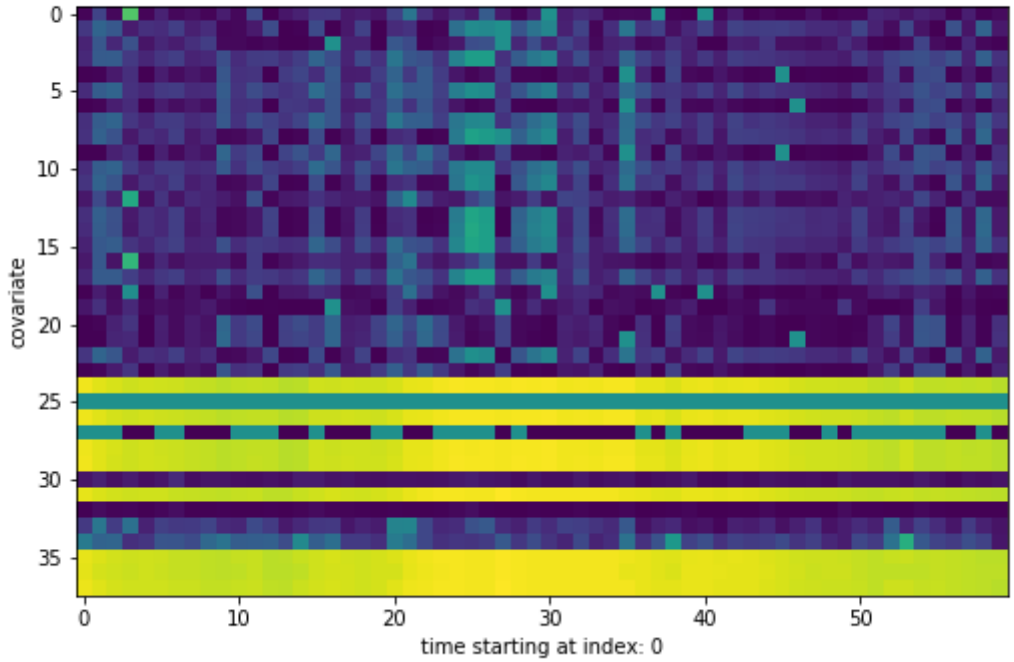
```
xm.shape: (1320, 38)
[400.0 258.0 10.0 57500.0 1800.0 229.0 2000.0 268.0 55700.0 10.0 509.0
 219.0 12.0 103500.0 45800.0 47800.0 2400.0 107700.0 2.0 2000.0 22.0 10.0
 487.0 4200.0 72.69 None 72.67 0.01999999999999602 72.67923076923077 72.68
 168.30107526881721 72.632043011 111.45527324274002 93.0
 0.02261728560699821 72.632563 72.6 72.68]
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:49 PM.

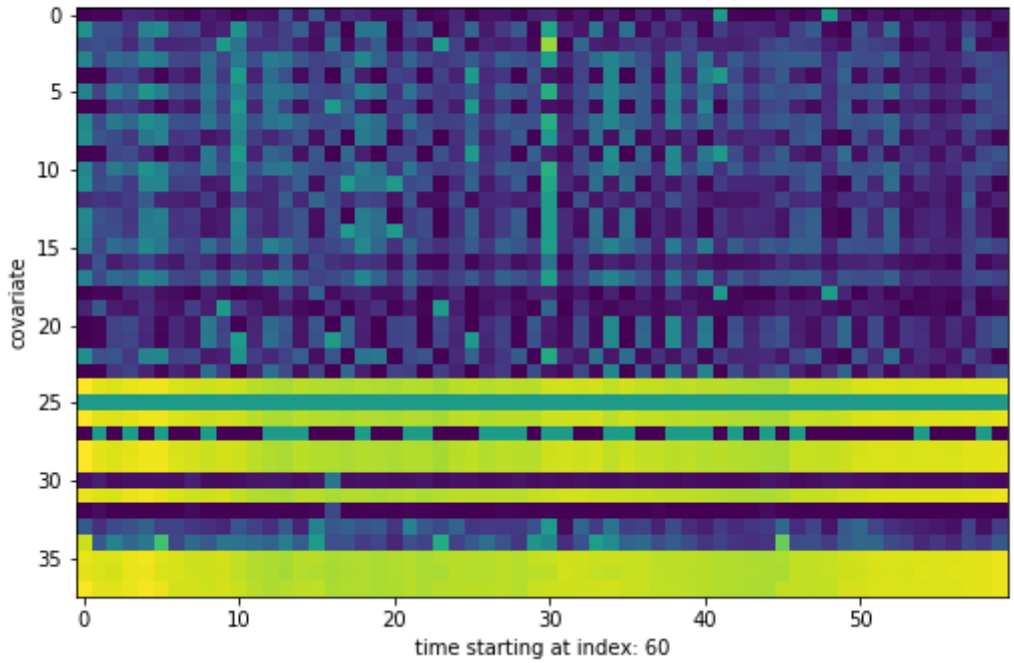
FINISHED

```
%python

scaler = preprocessing.MinMaxScaler((-1, 1))
scaler.fit(xm)

xmnormd = scaler.transform(xm)
xmnormd[np.isnan(xmnormd)] = 0

limmultiple = 5
for i in range(0, min(limmultiple*60, xm.shape[0]), 60):
    plt.imshow(np.transpose(xmnormd[i:(i+60), :]))
    plt.xlabel('time starting at index: %s' % (i))
    plt.ylabel('covariate')
    plt.show()
```

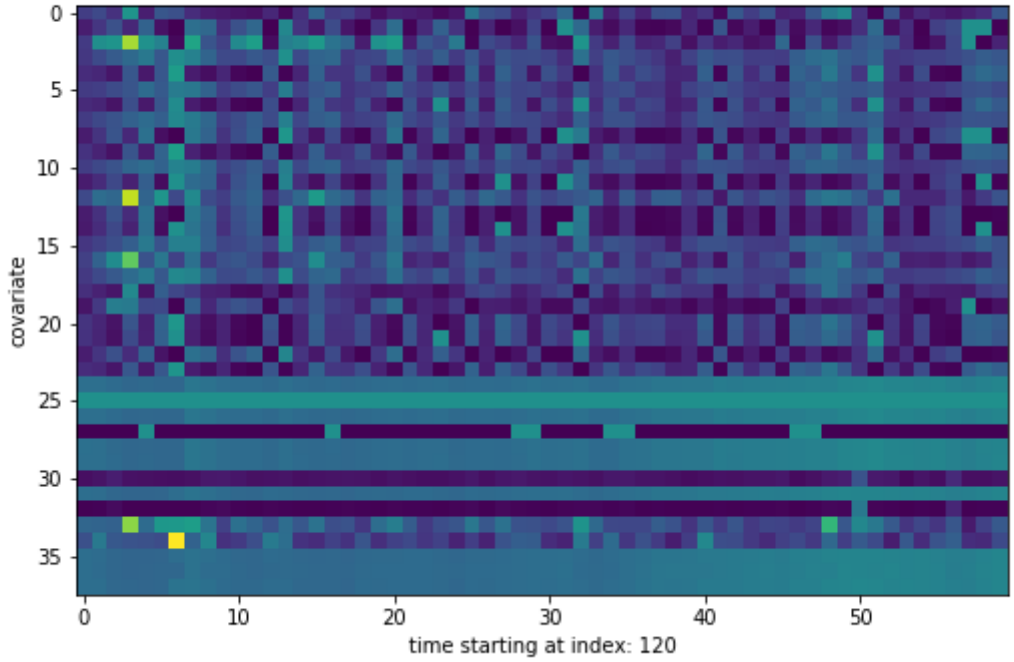
```
<matplotlib.image.AxesImage object at 0x7faf8ca1a518>
Text(0.5,29.75,'time starting at index: 0')
Text(51.25,0.5,'covariate')
```



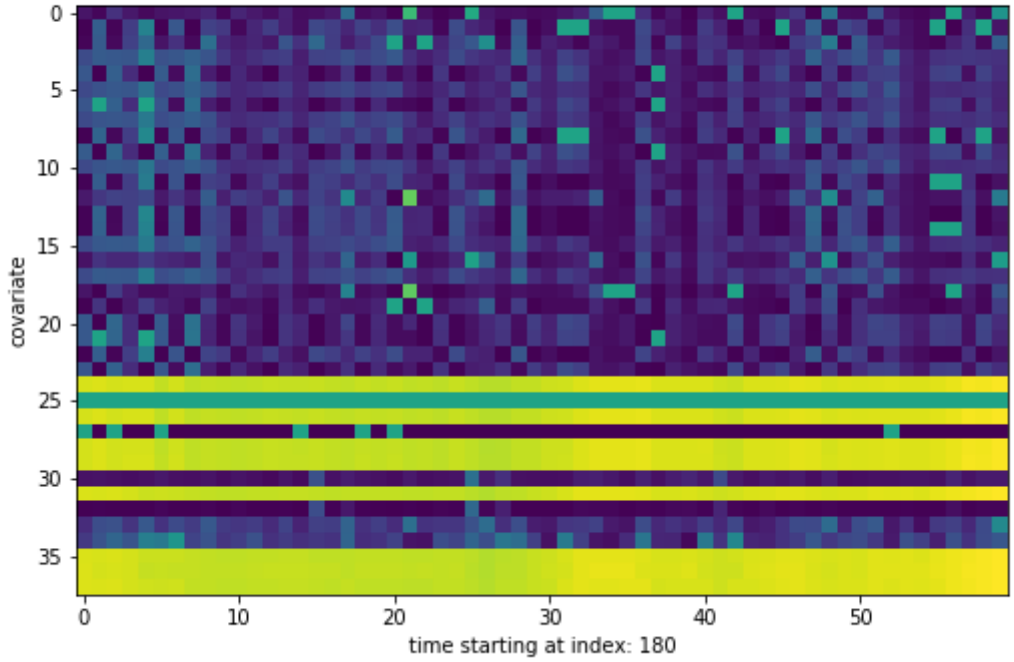
Text(0.5,29.75,'time starting at index: 60') Text(51.25,0.5,'covariate')



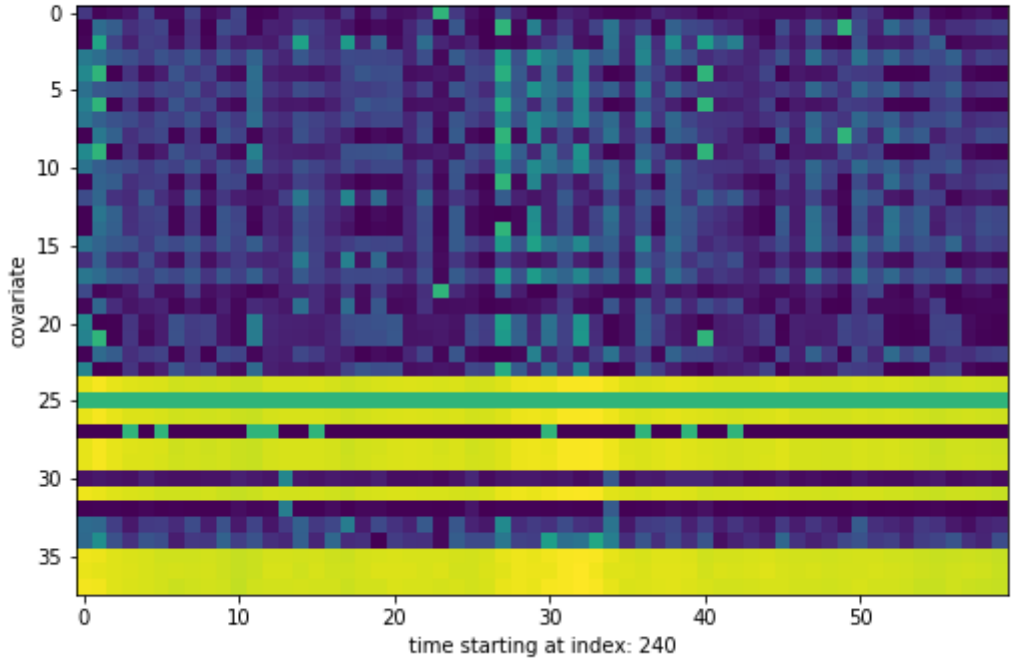
Text(0.5,29.75,'time starting at index: 120') Text(51.25,0.5,'covariate')



Text(0.5,29.75,'time starting at index: 180') Text(51.25,0.5,'covariate')



Text(0.5,29.75,'time starting at index: 240') Text(51.25,0.5,'covariate')



Took 3 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:52 PM.

```
%md

VAE from here:

[https://github.com/keras-team/keras/blob/master/examples/variational_autoencoder.py](https://

Modified to no variational
```

FINISHED

VAE from here:

https://github.com/keras-team/keras/blob/master/examples/variational_autoencoder.py
(https://github.com/keras-team/keras/blob/master/examples/variational_autoencoder.py)

Modified to no variational

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:49 PM.

```
%python

from tensorflow.keras.layers import Lambda, Input, Dense
from tensorflow.keras.models import Model
from tensorflow.keras.losses import mse
from tensorflow.keras import backend as K
from tensorflow.keras.optimizers import Adam
```

FINISHED

Took 3 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:52 PM.

```
%python

from tensorflow.python.client import device_lib
print(device_lib.list_local_devices())
```

FINISHED

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 6554164186288789723
, name: "/device:XLA_CPU:0"
device_type: "XLA_CPU"
memory_limit: 17179869184
locality {
}
incarnation: 3177413700698409481
physical_device_desc: "device: XLA_CPU device"
]
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:53 PM.

```
%python

input_shape = (ncov,)
interm_dim = 16
latent_dim = 8
```

FINISHED

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:53 PM.

```
%python

# VAE model = encoder + decoder
# build encoder model
inputs = Input(shape=input_shape, name='encoder_input')
x = Dense(interm_dim, activation='tanh')(inputs)
z = Dense(latent_dim, name='z')(x)

# instantiate encoder model
encoder = Model(inputs, z, name='encoder')
encoder.summary()
```

FINISHED

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
encoder_input (InputLayer)   (None, 38)                0
_________________________________________________________________
dense_15 (Dense)             (None, 16)                624
_________________________________________________________________
z (Dense)                    (None, 8)                 136
=================================================================
Total params: 760
Trainable params: 760
Non-trainable params: 0
_________________________________________________________________
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:53 PM.

```
%python

# build decoder model
latent_inputs = Input(shape=(latent_dim,), name='z')
```

FINISHED

```
x = Dense(interm_dim, activation='tanh')(latent_inputs)
outputs = Dense(ncov, activation='tanh')(x)

# instantiate decoder model
decoder = Model(latent_inputs, outputs, name='decoder')
decoder.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
z (InputLayer)               (None, 8)                 0
_________________________________________________________________
dense_16 (Dense)             (None, 16)                144
_________________________________________________________________
dense_17 (Dense)             (None, 38)                646
=================================================================
Total params: 790
Trainable params: 790
Non-trainable params: 0
_________________________________________________________________
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:53 PM.

```
%python

# instantiate AE model
outputs = decoder(encoder(inputs))
ae = Model(inputs, outputs, name='ae')
ae.add_loss(K.mean(ncov*mse(inputs, outputs)))
```

FINISHED

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:53 PM.

```
%python

ae.compile(optimizer=Adam(lr=0.00001))
ae.summary()
```

FINISHED

```
/usr/local/lib64/python3.4/site-packages/sklearn/model_selection/_split.py:2053: FutureWarnin
g: You should specify a value for 'cv' instead of relying on the default value. The default va
lue will change from 3 to 5 in version 0.22.
  warnings.warn(CV_WARNING, FutureWarning)
/usr/local/lib64/python3.4/site-packages/numpy/lib/function_base.py:3826: RuntimeWarning: Inva
lid value encountered in percentile
  interpolation=interpolation)
/usr/local/lib64/python3.4/site-packages/numpy/lib/function_base.py:3826: RuntimeWarning: Inva
lid value encountered in percentile
  interpolation=interpolation)
/usr/local/lib64/python3.4/site-packages/numpy/lib/function_base.py:3826: RuntimeWarning: Inva
lid value encountered in percentile
  interpolation=interpolation)
/usr/local/lib64/python3.4/site-packages/numpy/lib/function_base.py:3826: RuntimeWarning: Inva
lid value encountered in percentile
  interpolation=interpolation)
/usr/local/lib64/python3.4/site-packages/numpy/lib/function_base.py:3826: RuntimeWarning: Inva
lid value encountered in percentile
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:54 PM.

```
%md

# Run AE
```

FINISHED

Run AE

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:50 PM.

```
%python

ntest = 60
# test on last day
xtrain, xtest = xmnormd[:-ntest, :], xmnormd[-ntest:, :]
```

FINISHED

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:54 PM.

```
%python

weightsfname = 'ae_%s_SYM:%s.h5' % (jobname, symbol)

if dotraining:
    history = ae.fit(xtrain, epochs=5000, batch_size=64, verbose=2, validation_data=(xtest, Nc
    ae.save_weights(weightsfname)

    plt.plot(history.history['loss'], color='green', label='training_loss')
    plt.plot(history.history['val_loss'], color='red', label='validation_loss')
    plt.xlabel('iters')
    plt.ylabel('loss')
    plt.legend()
    plt.show()

else:
    ae.load_weights(weightsfname)
```

FINISHED

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:54 PM.

```
%python

z = encoder.predict(xmnormd)
```

FINISHED

Took 1 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:55 PM.

```
%python

from sklearn.manifold import TSNE

# find 2-dim embeddings of zmean
zmeanembd = TSNE(n_components=2).fit_transform(z)
```

FINISHED

Took 11 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:40:06 PM.

```
%md

GMM provided by [https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMix
#sklearn.mixture.GaussianMixture)
```

FINISHED

```
Tuning for number of components by [https://scikit-learn.org/stable/modules/generated/sklearn.
.model_selection.GridSearchCV.html#sklearn.model_selection.GridSearchCV)

Using `CV+loglike` instead of `AIC` mostly because not implemented in `GridSearchCV` and also
```

GMM provided by https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html#sklearn.mixture.GaussianMixture (https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html#sklearn.mixture.GaussianMixture)

Tuning for number of components by https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#sklearn.model_selection.Gri (https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#sklearn.model_selection.Gri

Using `CV+loglike` instead of `AIC` mostly because not implemented in `GridSearchCV` and also because we are already doing `CV`

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:51 PM.

```
%python

from sklearn.mixture import GaussianMixture
from sklearn.model_selection import GridSearchCV

param_grid = {
    'n_components': np.arange(1, latent_dim),
    'covariance_type': ['full', 'tied', 'diag', 'spherical']
    #'covariance_type': ['diag']
}

cv = GridSearchCV(GaussianMixture(random_state=0), param_grid=param_grid)
cv.fit(z)
```

FINISHED

```
GridSearchCV(cv='warn', error_score='raise-deprecating',
       estimator=GaussianMixture(covariance_type='full', init_params='kmeans', max_iter=100,
        means_init=None, n_components=1, n_init=1, precisions_init=None,
        random_state=0, reg_covar=1e-06, tol=0.001, verbose=0,
        verbose_interval=10, warm_start=False, weights_init=None),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'covariance_type': ['full', 'tied', 'diag', 'spherical'], 'n_components': a
rray([1, 2, 3, 4, 5, 6, 7])},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
```

Took 1 min 41 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:36 PM.

```
%python

print (cv.best_params_)
GMMo = cv.best_estimator_
```

FINISHED

```
{'covariance_type': 'full', 'n_components': 7}
```

Took 1 min 29 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:36 PM.

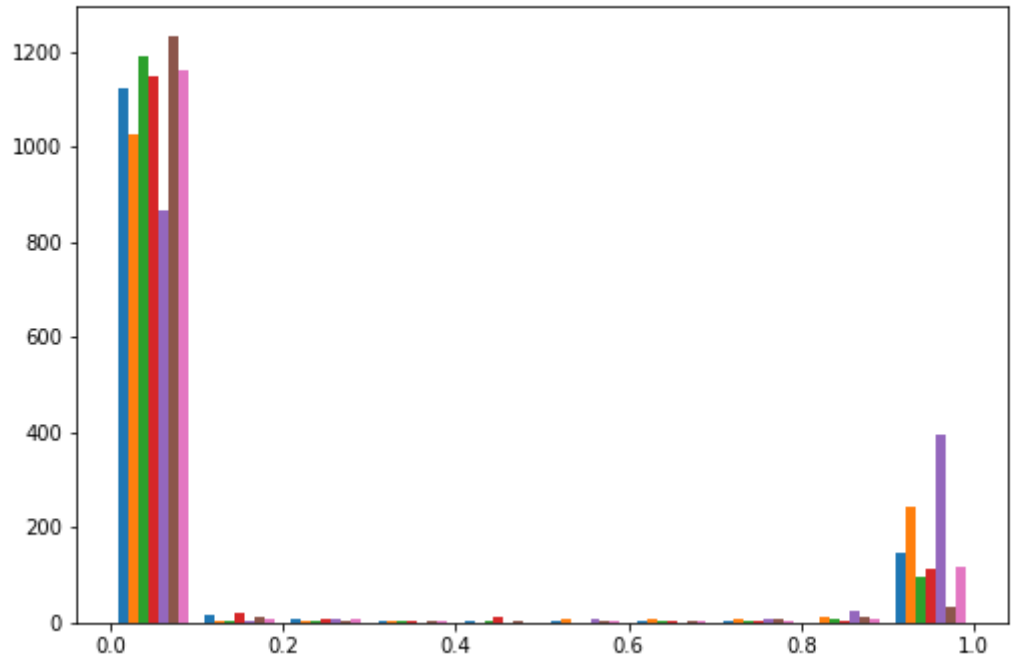
```
%python

gmmlabel = GMMo.fit_predict(z)
```

FINISHED

Took 6 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:42 PM.

```
%python

gmmproba = GMMo.predict_proba(z)
plt.hist(gmmproba)
plt.show()
```

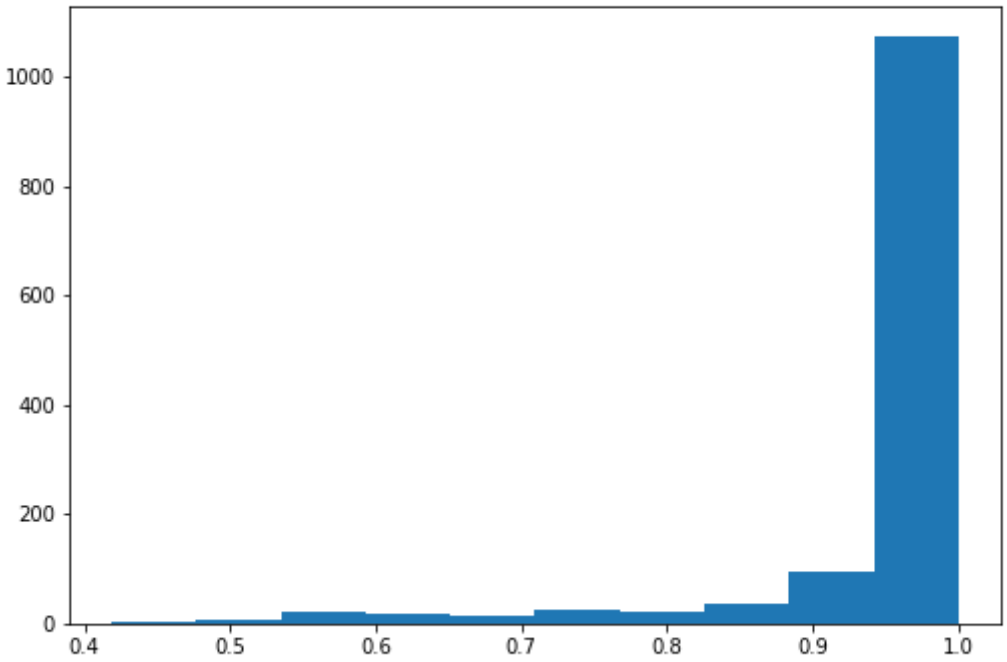
FINISHED



Took 12 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:48 PM.

```
%python

# max prob over all classes
gmmprobamax = np.max(gmmproba, axis=1)
plt.hist(gmmprobamax)
plt.show()
```

FINISHED



Took 6 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:49 PM.

FINISHED

```
%python

# define abnormal condition as: unlikely to be any class
pcutoff = .1
isabnormal = np.all(gmmproba <= pcutoff, axis=1)
np.sum(isabnormal)
```

```
0
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:49 PM.

FINISHED

```
%python

np.unique(gmmlabel, return_counts=True)
```

```
(array([0, 1, 2, 3, 4, 5, 6]), array([161, 279, 114, 130, 438,  62, 136]))
```

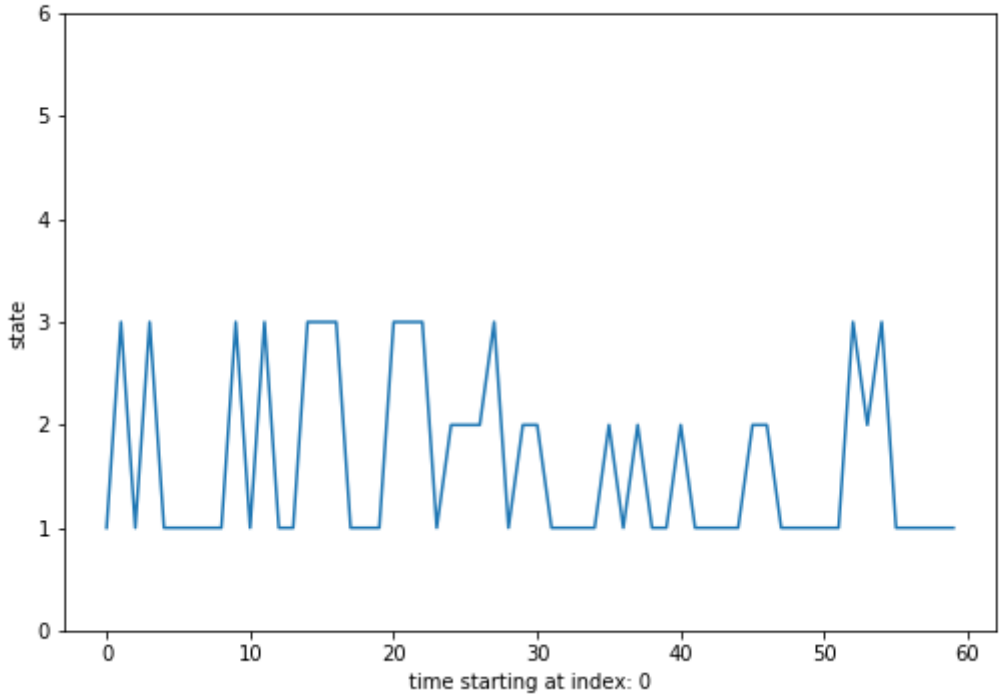
Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:50 PM.

FINISHED

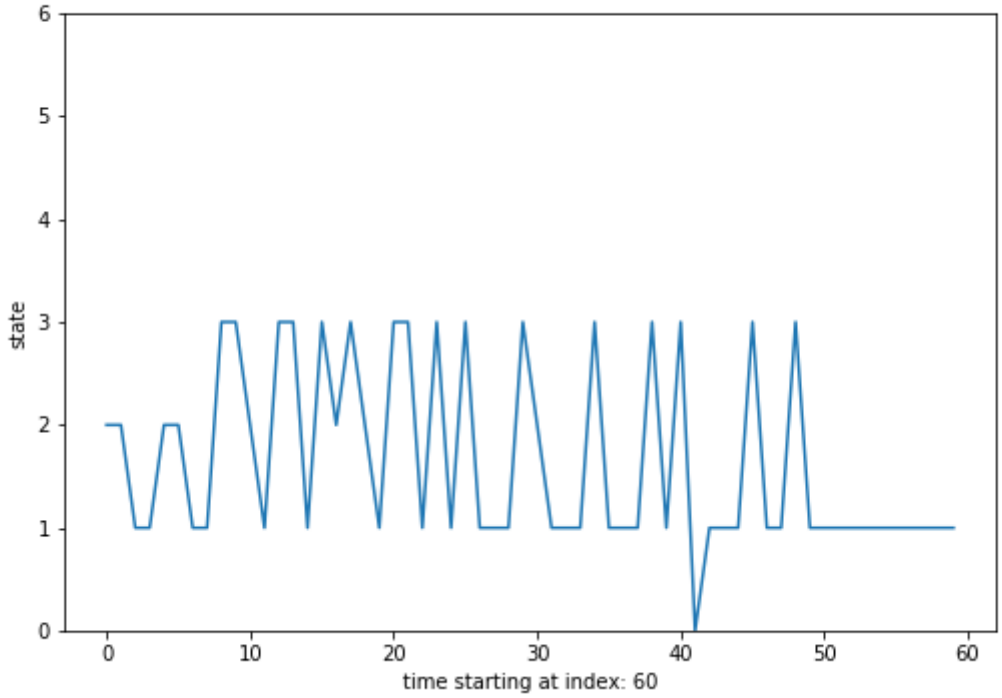
```
%python

limmultiple = 5
for i in range(0, min(limmultiple*60, ntime), 60):
    plt.plot(gmmlabel[i:(i+60)])
    plt.xlabel('time starting at index: %s' % (i))
    plt.ylabel('state')
    plt.ylim((0, max(np.unique(gmmlabel))))
    plt.show()
```

```
[<matplotlib.lines.Line2D object at 0x7faf8d212978>]
Text(0.5,23,'time starting at index: 0')
```

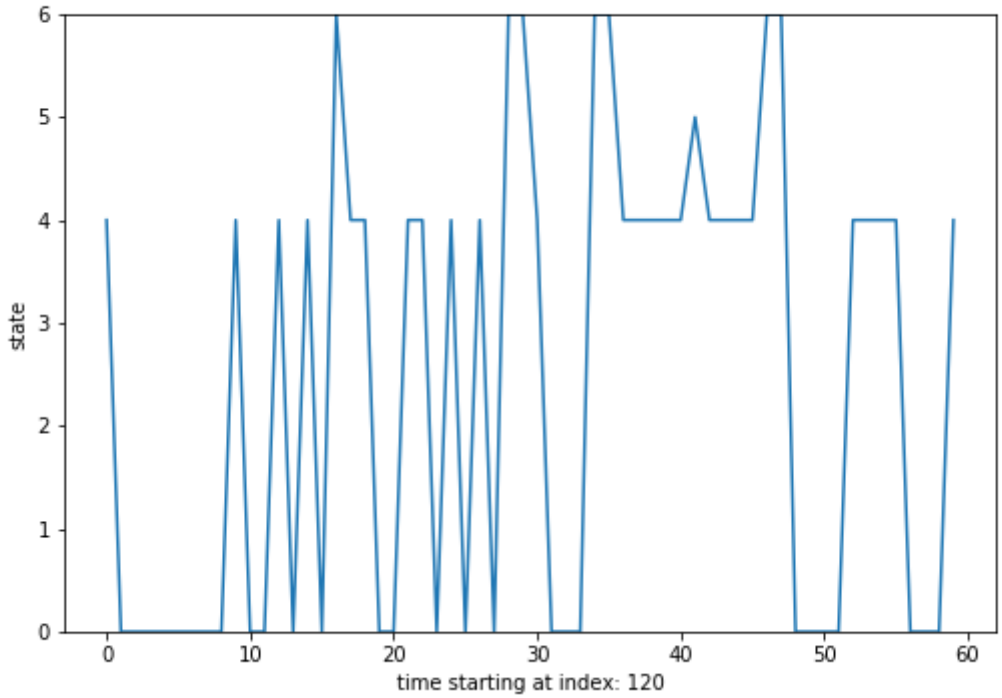
```
Text(41.625,0.5,'state')
(0, 6)
```



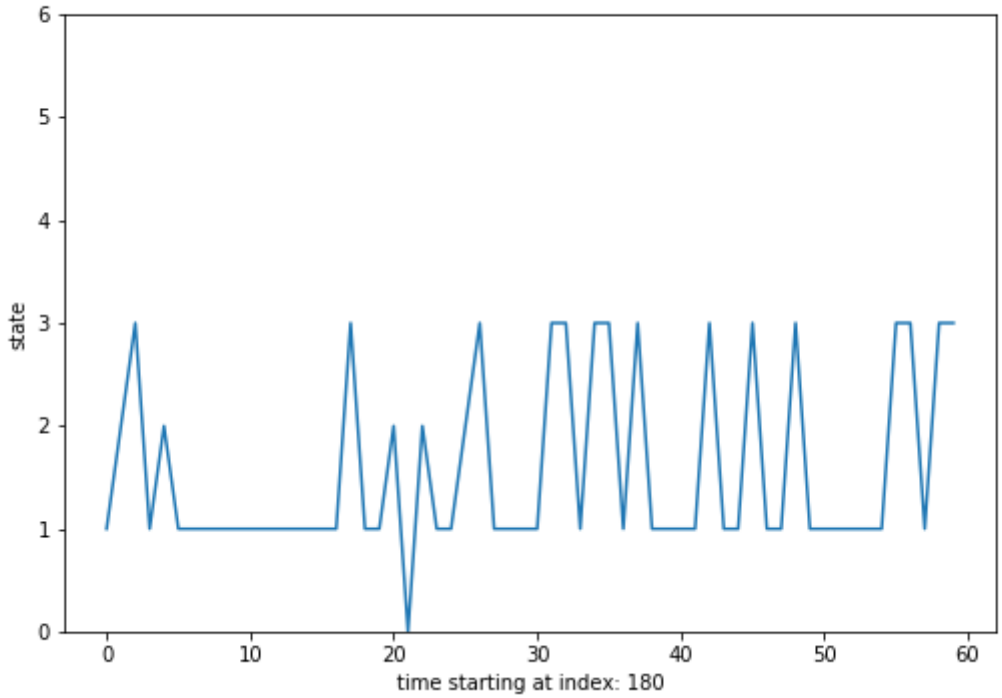
[] Text(0.5,23,'time starting at index: 60') Text(48,0.5,'state') (0, 6)



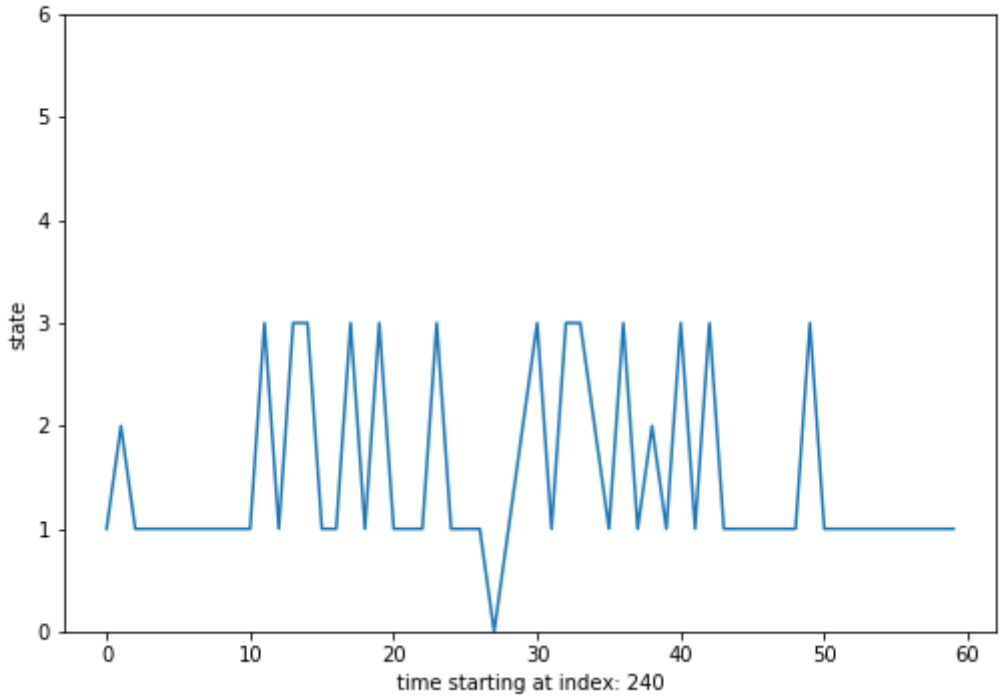
[] Text(0.5,23,'time starting at index: 120') Text(57.625,0.5,'state') (0, 6)



[] Text(0.5,23,'time starting at index: 180') Text(48,0.5,'state') (0, 6)



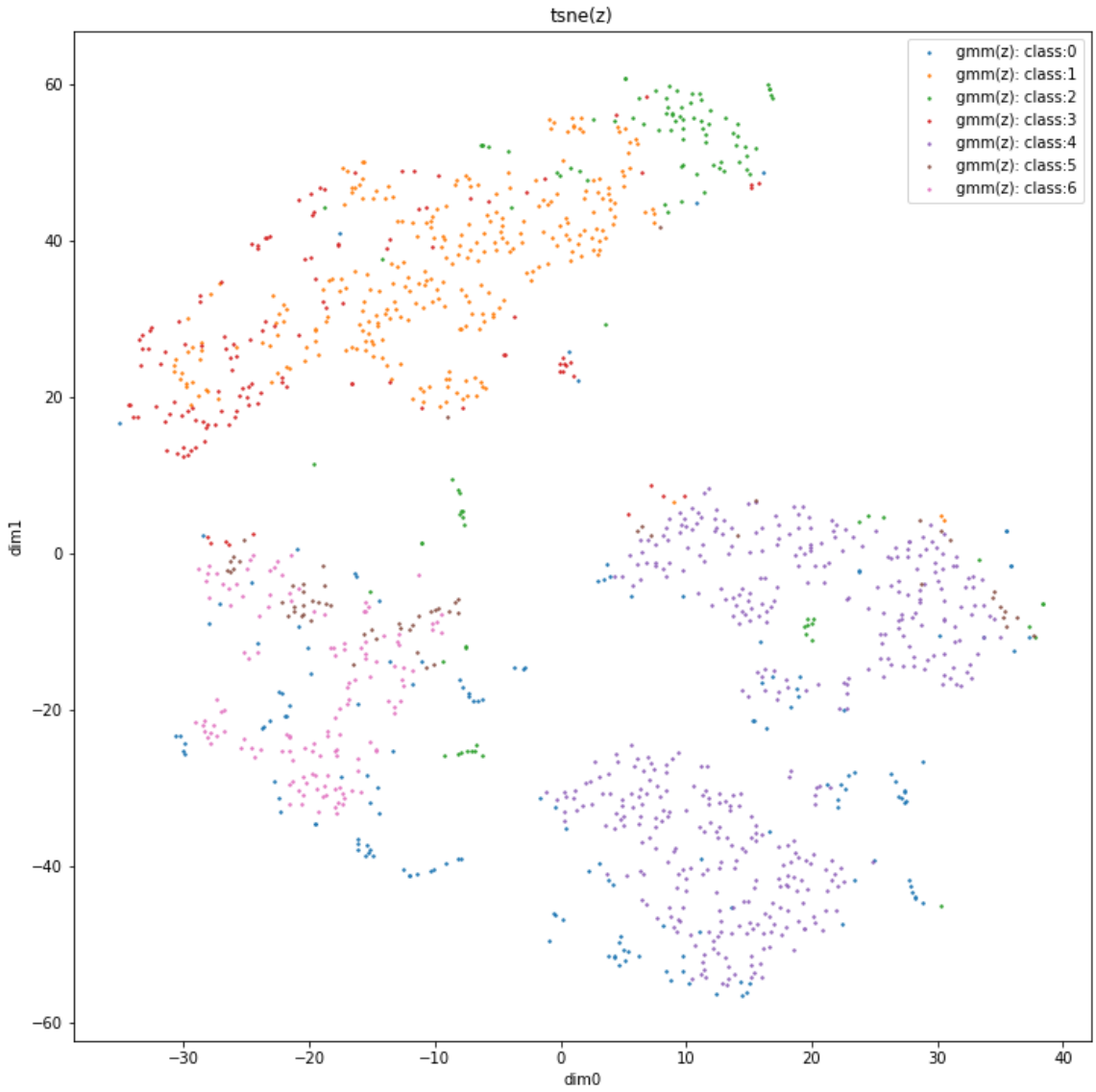
[] Text(0.5,23,'time starting at index: 240') Text(48,0.5,'state') (0, 6)



Took 2 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:52 PM.

FINISHED

```
%python

plt.figure(figsize=(12, 12))

for classdex in np.unique(gmmlabel):
    isclass = gmmlabel == classdex
    plt.scatter(zmeanembd[isclass, 0], zmeanembd[isclass, 1], s=2, label='gmm(z): class:%s' %

plt.title('tsne(z)')
plt.xlabel('dim0')
plt.ylabel('dim1')
plt.legend()
plt.show()
```



Took 4 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:54 PM.

FINISHED

Evaluation

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:39:52 PM.

FINISHED

```
%python

TXLoader(jobname=jobname, symbol=symbol).getcovnames()
```

```
['orders:sum(ABS(book_change))_for_type:Executed_side:Buy_orders_at_touch', 'orders:count(*)_f
or_type:New_side:Sell_orders_at_touch', 'orders:count(*)_for_type:Executed_side:Sell_orders_at
_touch', 'orders:sum(ABS(book_change))_for_type:New_side:All_orders_at_touch', 'orders:sum(ABS
```

```
(book_change))_for_type:New_side:Buy_orders_at_touch', 'orders:count(*)_for_type:Cancelled_sid
e:All_orders_at_touch', 'orders:sum(ABS(book_change))_for_type:Cancelled_side:Buy_orders_at_to
uch', 'orders:count(*)_for_type:New_side:All_orders_at_touch', 'orders:sum(ABS(book_change))_f
or_type:New_side:Sell_orders_at_touch', 'orders:count(*)_for_type:New_side:Buy_orders_at_touc
h', 'orders:count(*)_for_type:All_side:All_orders_at_touch', 'orders:count(*)_for_type:Cancell
ed_side:Sell_orders_at_touch', 'orders:count(*)_for_type:Executed_side:All_orders_at_touch',
 'orders:sum(ABS(book_change))_for_type:All_side:Sell_orders_at_touch', 'orders:sum(ABS(book_c
hange))_for_type:Cancelled_side:Sell_orders_at_touch', 'orders:sum(ABS(book_change))_for_type:
Cancelled_side:All_orders_at_touch', 'orders:sum(ABS(book_change))_for_type:Executed_side:All_
orders_at_touch', 'orders:sum(ABS(book_change))_for_type:All_side:All_orders_at_touch', 'order
s:count(*)_for_type:Executed_side:Buy_orders_at_touch', 'orders:sum(ABS(book_change))_for_typ
e:Executed_side:Sell_orders_at_touch', 'orders:count(*)_for_type:All_side:Buy_orders_at_touc
h', 'orders:count(*)_for_type:Cancelled_side:Buy_orders_at_touch', 'orders:count(*)_for_type:A
ll_side:Sell_orders_at_touch', 'orders:sum(ABS(book_change))_for_type:All_side:Buy_orders_at_t
```

Took 2 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:54 PM.

FINISHED

```
%python

def evaluate(xm, covname, xlim=None, plot=True):

    covnames = TXLoader(jobname=jobname, symbol=symbol).getcovnames()
    iscov = np.array(covnames)==covname

    pred = scaler.inverse_transform(ae.predict(xm))[:, iscov]
    true = scaler.inverse_transform(xm)[:, iscov]
    print ('mape: %.6f' % (np.mean(np.abs(pred-true)/true)))

    if plot:

        predplot = pred
        trueplot = true
        if xlim is not None:
            predplot = predplot[xlim[0]:xlim[1]]
            trueplot = trueplot[xlim[0]:xlim[1]]

        plt.plot(predplot, color='blue', label='reconstructed')
        plt.plot(trueplot, color='red', label='true')
        plt.title('covname: %s' % (covname))
        plt.legend()
        plt.show()

    return pred, true
```

Took 0 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:54 PM.

FINISHED

```
%python

evaluate(xtrain, 'prices:mid', xlim=(0, 60))
```

```
mape: 0.004656
```

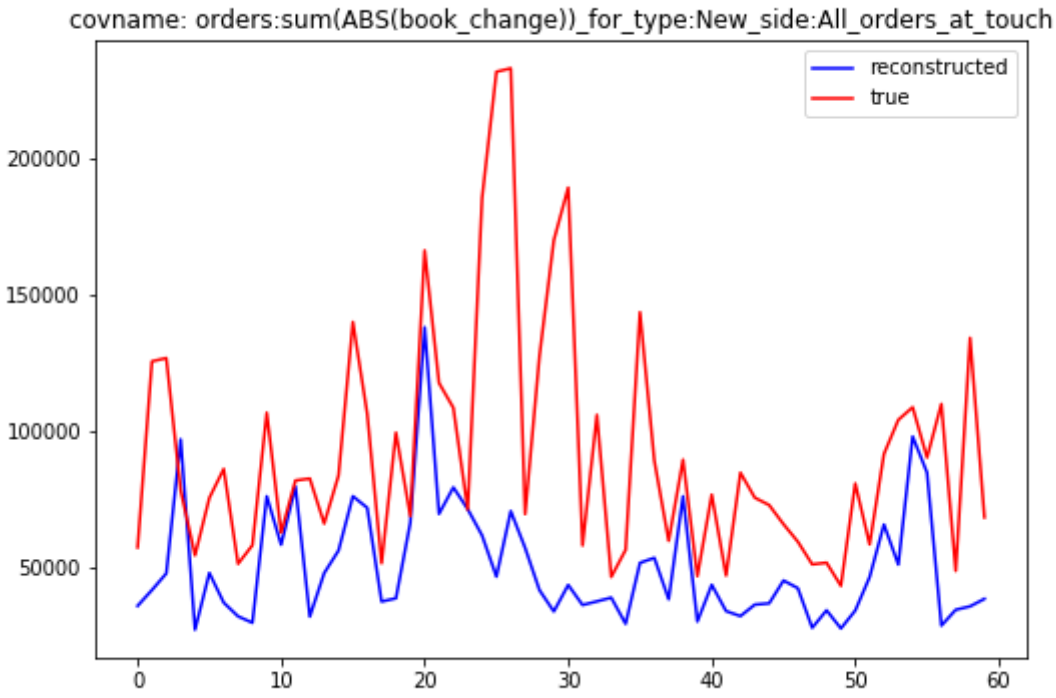
(array([[71.88826], [72.2005], [71.91081], ..., [71.88144], [71.85561], [72.06629]], dtype=float32), array([[72.68], [72.61], [72.55], ..., [72.53], [72.515], [72.53]]))

Took 2 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:56 PM.

```
%python

evaluate(xtrain, 'orders:sum(ABS(book_change))_for_type:New_side:All_orders_at_touch', xlim=(0
```

FINISHED

```
mape: 0.424471
```



(array([[36104.094], [41895.656], [47969.324], ..., [168982.11], [165326.83], [76757.07]], dtype=float32), array([[57500.], [125600.], [126800.], ..., [283270.], [264800.], [342000.]]))

Took 2 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:57 PM.

```
%python

evaluate(xtest, 'prices:mid')
```

FINISHED

```
mape: 0.006807
```

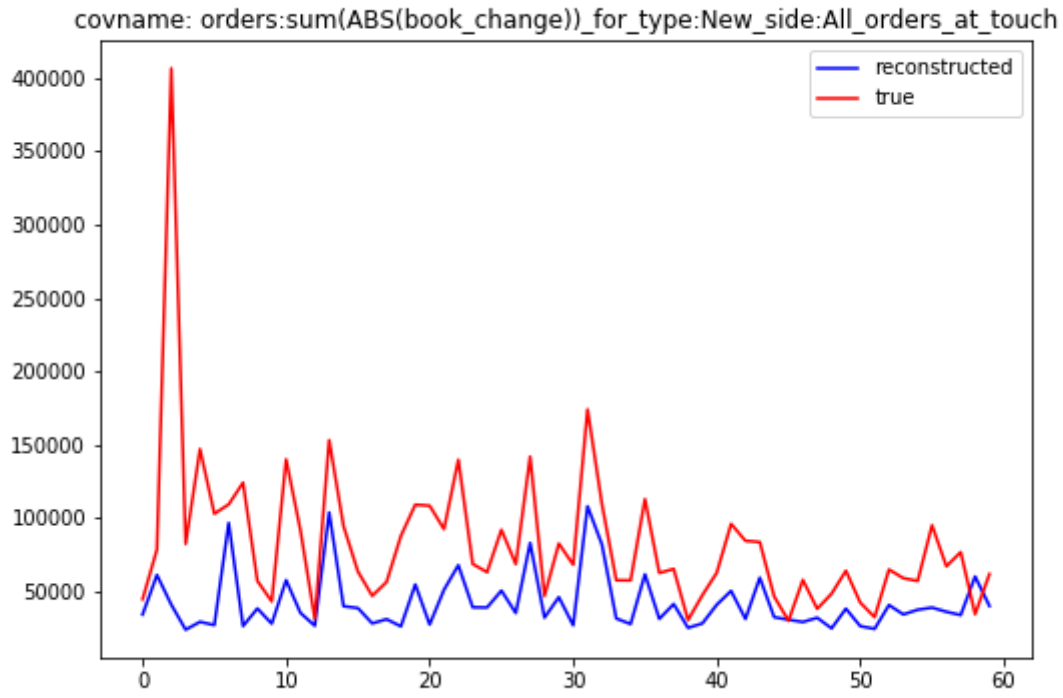
(array([[71.81533], [71.39098], [71.93295], [71.59298], [72.03303], [71.622604], [71.54255], [71.73599], [71.68296], [71.65289], [71.55546], [71.401695], [71.38647], [71.69308], [71.75703], [71.34175], [71.59983], [71.65374], [71.57617], [71.38849], [71.40623], [71.55602], [71.31284], [71.312], [71.41648], [71.63819], [71.56112], [71.56465], [71.468575], [71.69443], [71.38262], [71.61253], [71.63515], [71.46811], [71.46615], [71.507835], [71.522675], [71.48913], [71.55858], [71.4317], [71.52772], [71.671585], [71.5387], [71.398766], [71.69512], [71.556885], [71.700935], [71.466545], [71.5914], [71.52398], [71.50948], [71.58481], [71.48252], [71.52813], [71.47326], [71.63903], [71.59797], [71.535126], [71.66323], [71.8927]], dtype=float32), array([[72.045], [72.085], [72.125], [72.055], [72.035], [71.985], [71.97], [71.995], [71.995], [72.005], [72.005], [71.99], [71.965], [71.965], [71.995], [71.975], [71.985], [71.985], [71.975], [71.945], [71.975], [71.935], [71.935], [71.97], [71.955], [71.965], [72.005], [72.005], [72.005], [72.025], [72.055], [72.035], [72.1], [72.115], [72.115], [72.095], [72.105], [72.085], [72.105], [72.105], [72.105], [72.135], [72.155], [72.09], [72.095], [72.115], [72.125], [72.125], [72.145], [72.105], [72.125], [72.115], [72.105], [72.155], [72.125], [72.155], [72.145], [72.165], [72.15], [72.165]]))

Took 3 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:41:59 PM.

```
%python

evaluate(xtest, 'orders:sum(ABS(book_change))_for_type:New_side:All_orders_at_touch')
```

FINISHED

```
mape: 0.448511
```



(array([[33767.543], [60788.402], [40499.555], [23403.307], [28768.736], [26554.879], [96295.71], [25918.15], [37893.934], [27515.441], [57246.04], [34860.734], [26108.066], [103481.805], [39455.562], [38170.707], [27704.22], [30568.955], [25688.85], [54300.11], [26944.646], [50661.895], [67470.17], [38698.293], [38557.176], [50113.387], [34788.18], [82645.19], [31650.568], [45658.887], [26392.648], [107706.41], [81649.35], [31065.389], [27264.549], [61254.832], [30769.652], [40944.17], [24665.49], [27708.4], [40464.13], [49907.406], [30756.748], [58903.344], [31961.336], [30239.13], [28652.49], [31562.984], [24321.445], [37752.97], [25838.281], [23979.164], [40242.695], [33839.805], [37024.49], [38502.41], [35639.363], [33407.27], [59742.684], [39637.273]], dtype=float32), array([[44200.], [78200.], [406800.], [82000.], [146800.], [102600.], [109000.], [123800.], [56719.], [42700.], [139775.], [91200.], [30947.], [152900.], [94200.], [63200.], [46500.], [56000.], [87500.], [108800.], [108100.], [92100.], [139500.], [68300.], [62700.], [91658.], [68200.], [141600.], [46510.], [82229.], [67800.], [173900.], [109300.], [57200.], [57100.], [112600.], [62150.], [64900.], [30000.], [47000.], [62300.], [95730.], [84100.], [83300.], [46100.], [29500.], [57400.], [37700.], [48000.], [63655.], [41900.], [32000.], [64500.], [58600.], [56700.], [94750.], [66700.], [76400.], [34000.], [61400.]]))

Took 2 sec. Last updated by tianlechen@gmail.com at March 01 2019, 5:42:00 PM.

```
%python
```

FINISHED