Kaspersky hackathon

Anomalies detection/classification

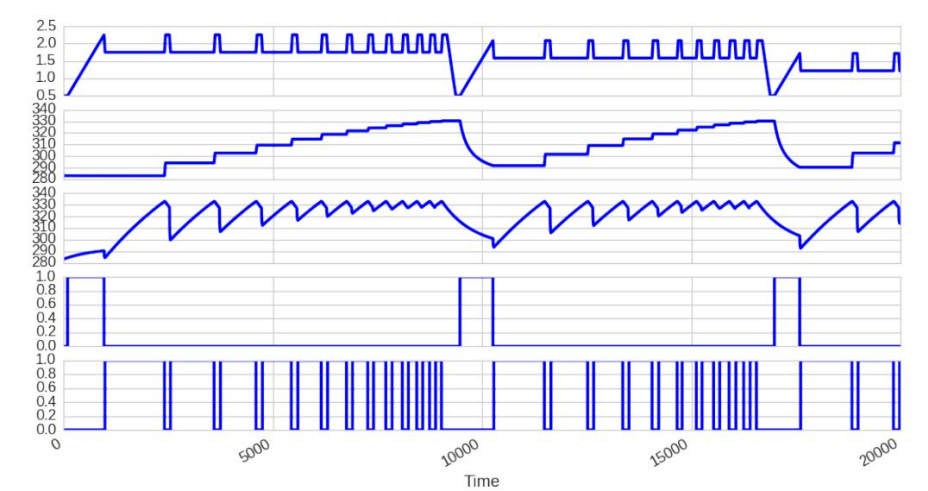
Team

DMIA - 1th place

Alexander Guschin
Roman Ushakov
Daniil Okhlopkov

Online stage

unsupervised anomalies detection in multivatiate time series



Online stage

Results from paper
https://arxiv.org/abs/1612.06676

https://www.youtube.com/watch?v=J9YMdOd4kFY

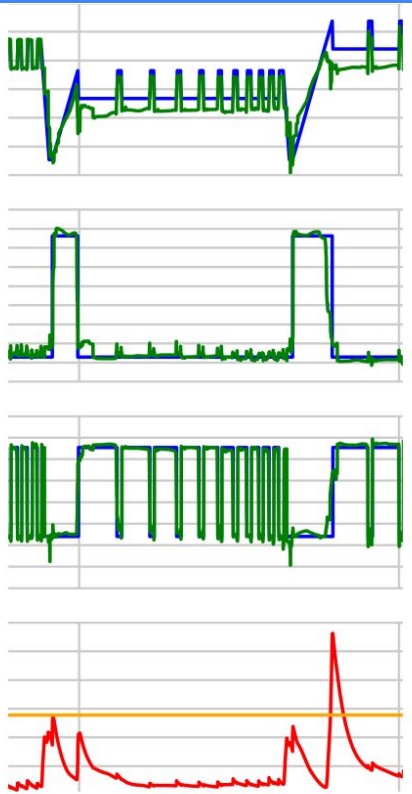


Figure 6: Example of the forecast, averaged MSE and fault detection threshold

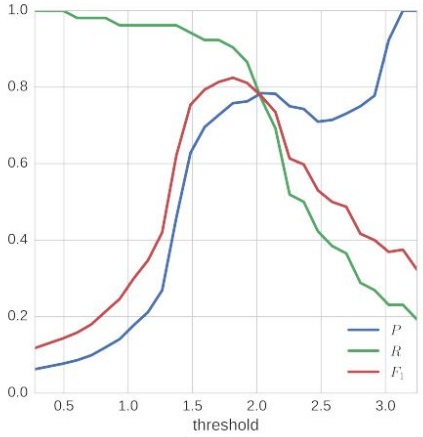


Figure 7: Precision, recall and F_1 score for different threshold levels

Offline stage

Multivariate timeseries classification based on Tennessee Eastman Problem

Xtrain: 500 files, each 96k*56 features
Ytrain: 0 if (file has no anomaly) else 1

Xtest: another 500 files

Metric: ROC AUC

Additional info

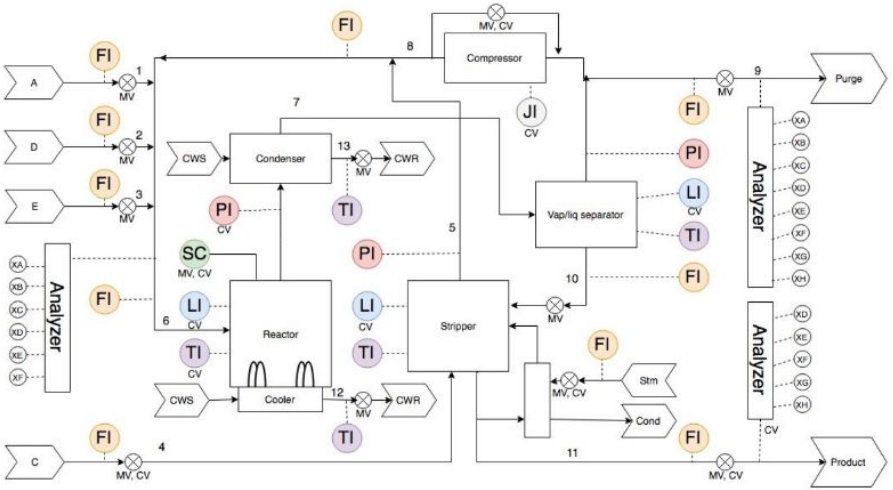
Task and features description

Дополнительная информация о заводе

В папках train и test лежат данные, полученные с модели химического завода «Tennessee Eastman Process». Вследствие реакции неизвестных химических компонент A, C, D, E получаются новые соединения (“g”-gas, “liq”-liquid) G и F:

A(g) + C(g) + D(g) -> G(liq), Product 1
A(g) + C(g) + E(g) -> H(liq), Product 2
A(g) + G(g) -> F(liq), Byproduct
3D(g) -> 2F(liq), Byproduct

Схема промышленного процесса имеет вид:

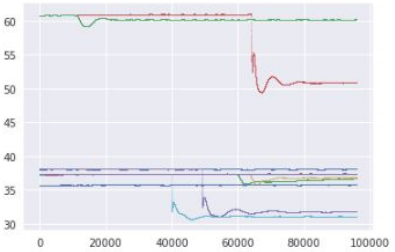


Example

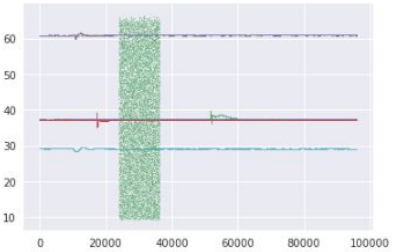
Notes:

1. No easy discoverable true positive patterns
2. Many false positive patterns
3. Little time to solve the task

In [67]:
```
for i in range(30, 40):
    plt.scatter(range(dfs_col_0.shape[0]), dfs_col_0[:,i], s=0.1)
```



In [69]:
```
for i in range(25, 35):
    plt.scatter(range(dfs_col_1.shape[0]), dfs_col_1[:,i], s=0.1)
```



Base solution

1. Feature generation
   a. Simple statistics for each feature (==column): min, max, std, median, mean
   b. The same stats for derivatives
   c. Stats on results of window functions

```
T = data.rolling(window=100).max() - data.rolling(window=100).min()
New_features = [func(T) for func in (np.max, np.min, np.mean, np.std)]
```

2. Models
   a.

```
param = {}
param['objective'] = 'binary:logistic'
param['booster'] = 'gblinear'
param['eta'] = 0.01
param['alpha'] = 0.00
param['lambda_bias'] = 0.1
```

   b. `ExtraTreesClassifier(n_estimators=1000, n_jobs=16, criterion='entropy', max_features=0.15, min_samples_split=5)`
   c. `KNeighborsClassifier(n_neighbors=400, weights='distance', n_jobs=20, p=20)`

Mix

```
from scipy.stats import rankdata

knn_ranks = rankdata(knn_preds)
xgb_ranks = rankdata(xgb_preds)
et_ranks = rankdata(et_preds)

final_preds = (knn_ranks + xgb_ranks + et_ranks) / 1500
```

LSTM part

1. Train LSTM to predict values N steps ahead on normal data (no anomalies)
2. Predict on all files
3. Use prediction error to make new features

Potential overfit in 1st-2nd steps

https://github.com/aguschin/kaspersky_hackathon