Dataswati

AutoEncoder / tsfresh / Clustering / Anomaly Detection

 ideas  what I've done

Anh-Thi Dinh | 02 – Mar – 2020

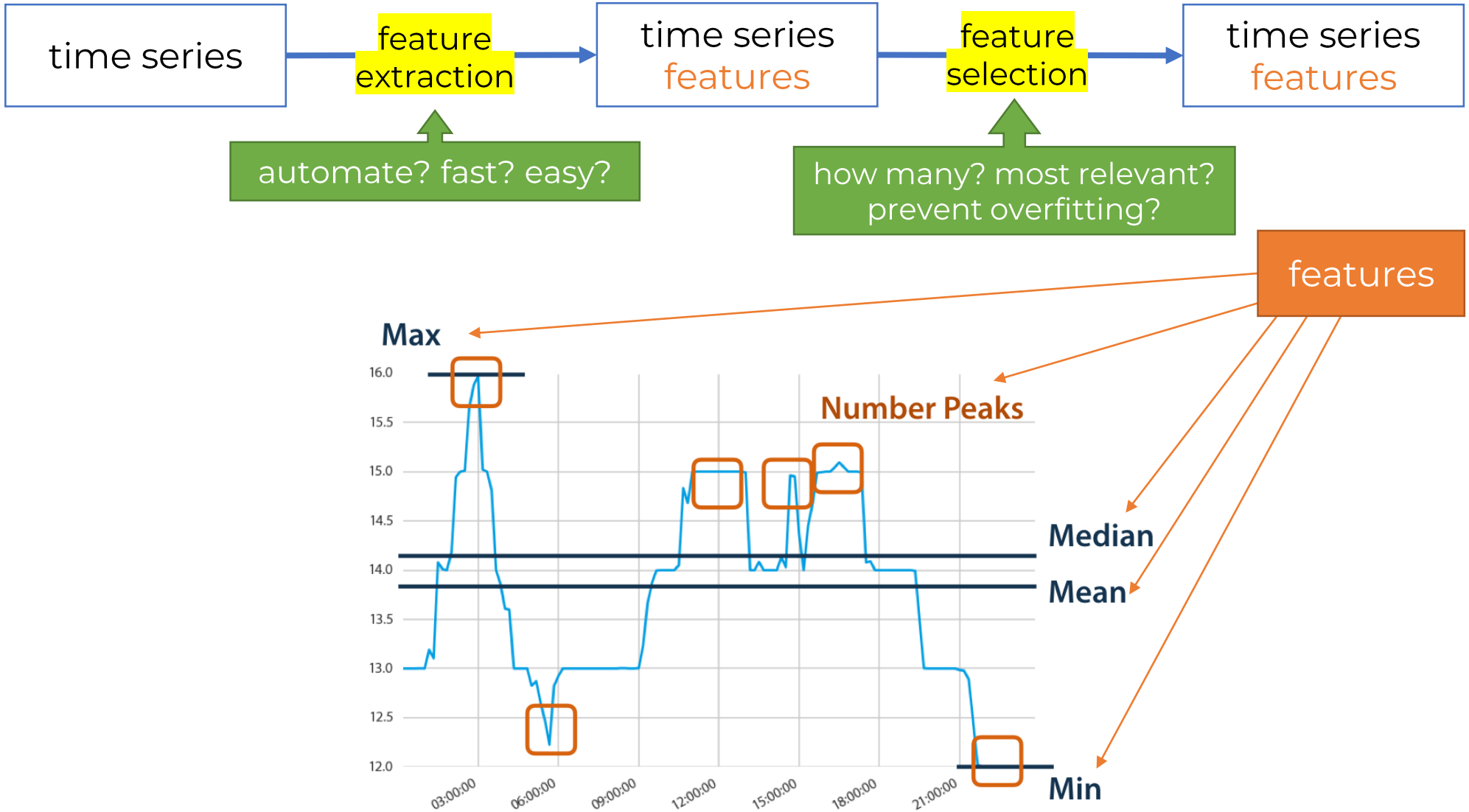
Layout of talk

1. **tsfresh**
   - General idea
   - Example
   - Example on Soufflet data
2. **Handle different-length time series**
3. **AutoEncoder / Soufflet**
   - General idea
   - Applied to find anomaly
   - Approaches
   - Basic AE on Soufflet data
   - CNN & LSTM AE idea
   - Some comments
4. **Clustering / Soufflet**
   - General idea & examples
   - Some basic results on Soufflet data
   - Applied to recipe recommendation
5. **Anomaly detection / Aquassay**

Layout of talk

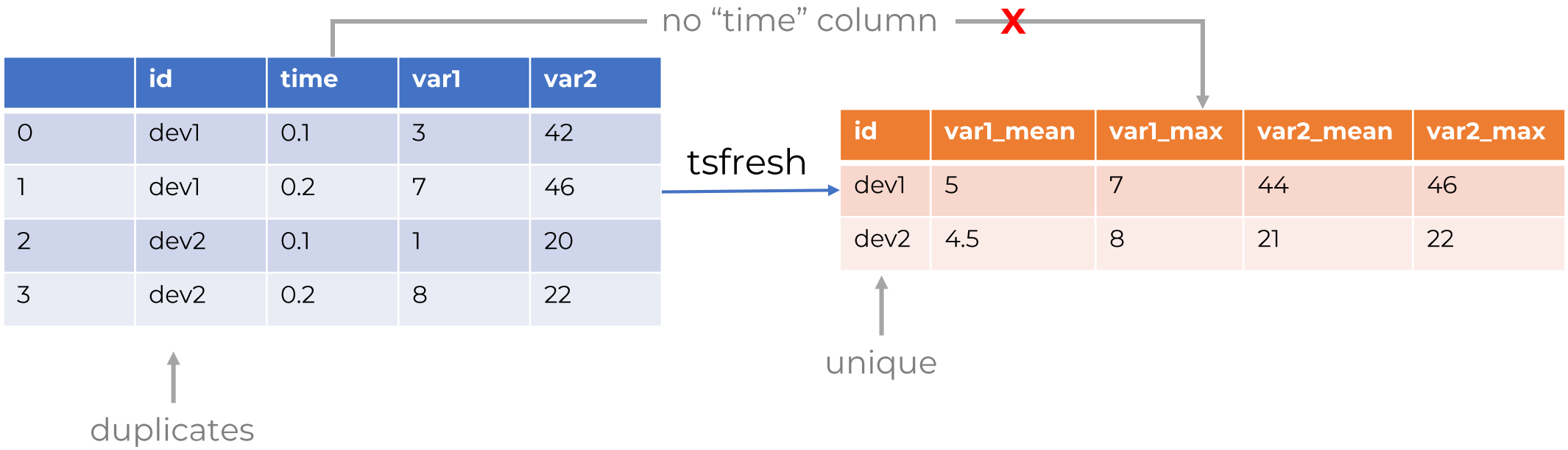
1. **tsfresh**
   - General idea
   - Example
   - Example on Soufflet data
2. **Handle different-length time series**
3. **AutoEncoder / Soufflet**
   - General idea
   - Applied to find anomaly
   - Approaches
   - Basic AE on Soufflet data
   - CNN & LSTM AE idea
   - Some comments
4. **Clustering / Soufflet**
   - General idea & examples
   - Some basic results on Soufflet data
   - Applied to recipe recommendation
5. **Anomaly detection / Aquassay**

tsfresh → general idea



tsfresh → example

tsfresh = feature extraction + feature selection + python + open source
(p-value calculation) (skit-learn)



	id	time	var1	var2
0	dev1	0.1	3	42
1	dev1	0.2	7	46
2	dev2	0.1	1	20
3	dev2	0.2	8	22

id	var1_mean	var1_max	var2_mean	var2_max
dev1	5	7	44	46
dev2	4.5	8	21	22

tsfresh = **T**ime **S**eries **F**eatu**R**e **E**xtraction based on **S**calable **H**ypothesis test

tsfresh → Soufflet

date	id_couche	consigne_temperature_sur_plateau	mesure_temperature_sur_plateau	consigne_temperature_sous_plateau	mesure_temperature_sous_plateau	ouverture_volet_air_neuf	vitesse_v
2013-10-15 05:19:00	P13C1021	0.0	164.0	0.0	168.0	100.0	0.0
2013-10-15 05:20:00	P13C1021	0.0	166.0	0.0	168.0	100.0	0.0
2013-10-15 05:21:00	P13C1021	0.0	166.0	0.0	168.0	100.0	0.0
2013-10-15 05:22:00	P13C1021	0.0	166.0	0.0	168.0	100.0	0.0
2013-10-15 05:23:00	P13C1021						

625029 x 11

tsfresh →

id_couche	amperage_moteur_ventilateur__abs_energy	amperage_moteur_ventilateur__absolute_sum_of_changes	amperage_moteur_ventilateur__c3__lag_1	amperage_moteur_ventilateur__c3__lag_2	amperage
P13C1021	71866667.0	8418.0	1.215348e+06	1.215078e+06	1.214890
P13C1041	68446441.0	8706.0	1.143089e+06	1.142621e+06	1.142162
P13C1051	69310664.0	9218.0	1.154612e+06	1.154382e+06	1.154113
P13C1061	71067848.0	9276.0	1.178115e+06	1.177775e+06	1.177353
P14C1016	66632263.0	9648.0	1.105785e+06	1.104900e+06	4042

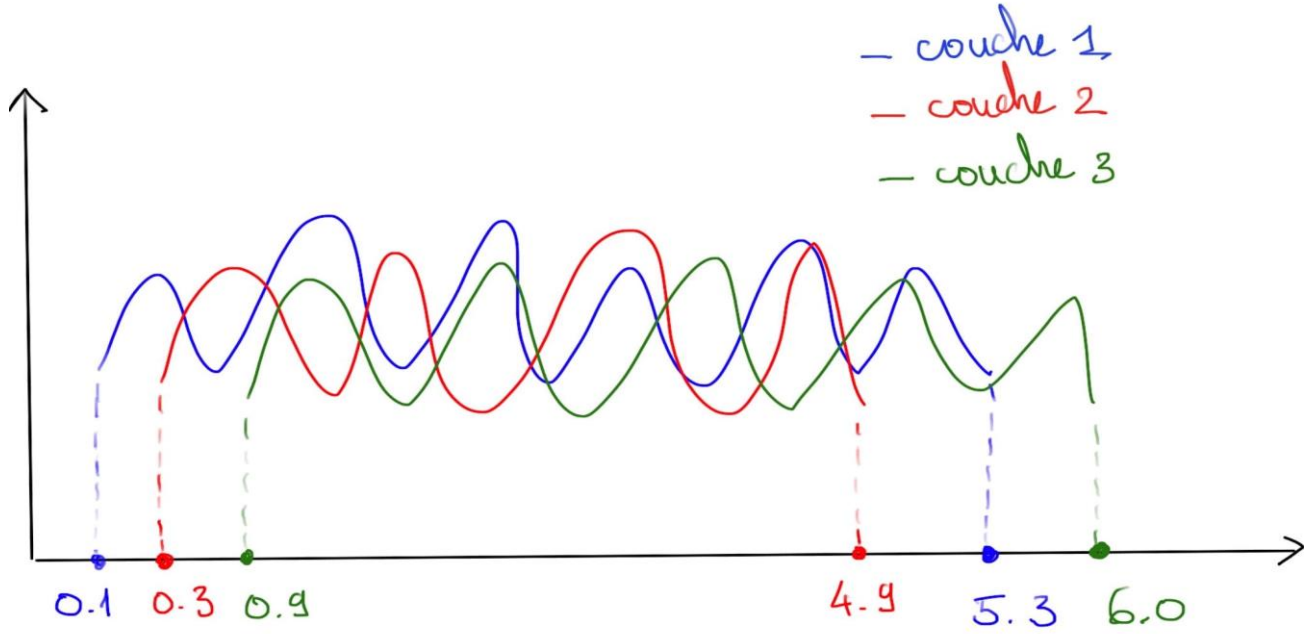
97 x 2390

Layout of talk

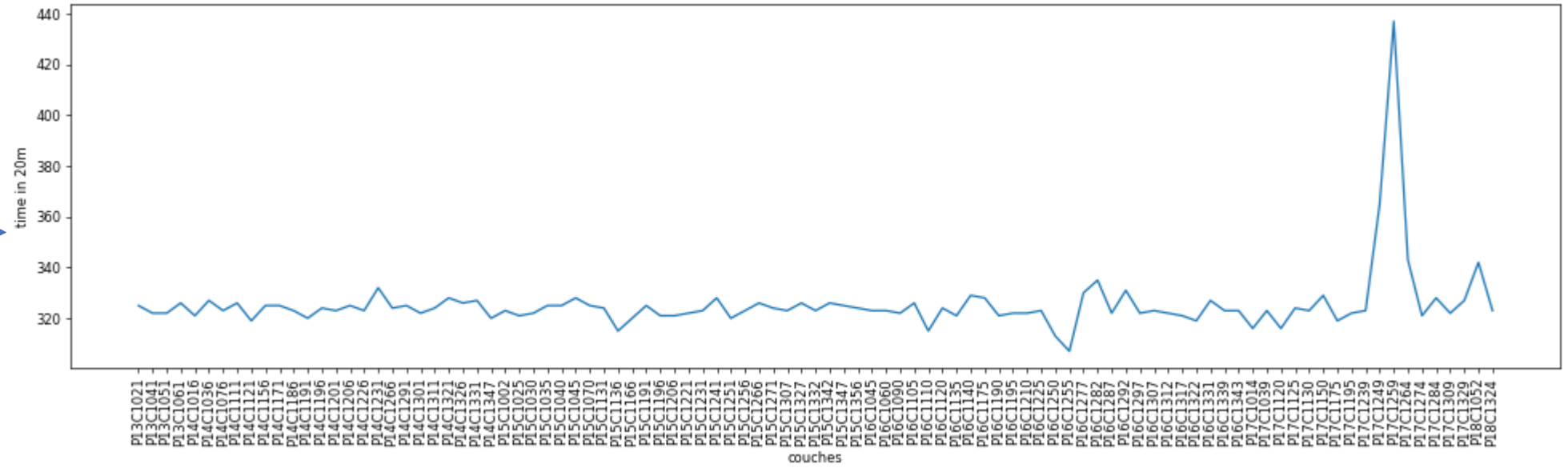
1. **tsfresh**
   - General idea
   - Example
   - Example on Soufflet data
2. **Handle different-length time series**
3. **AutoEncoder / Soufflet**
   - General idea
   - Applied to find anomaly
   - Approaches
   - Basic AE on Soufflet data
   - CNN & LSTM AE idea
   - Some comments
4. **Clustering / Soufflet**
   - General idea & examples
   - Some basic results on Soufflet data
   - Applied to recipe recommendation
5. **Anomaly detection / Aquassay**

Handle different-length TS

- **Problem**:
  - They start / end at different timestamps
  - Their lengths are different
- **Solution**:
  - Resampling to “20-minute” bins + align TS
  - Approaches:
    - Take the shortest
    - Filled by 0 / last / mean



Soufflet data →

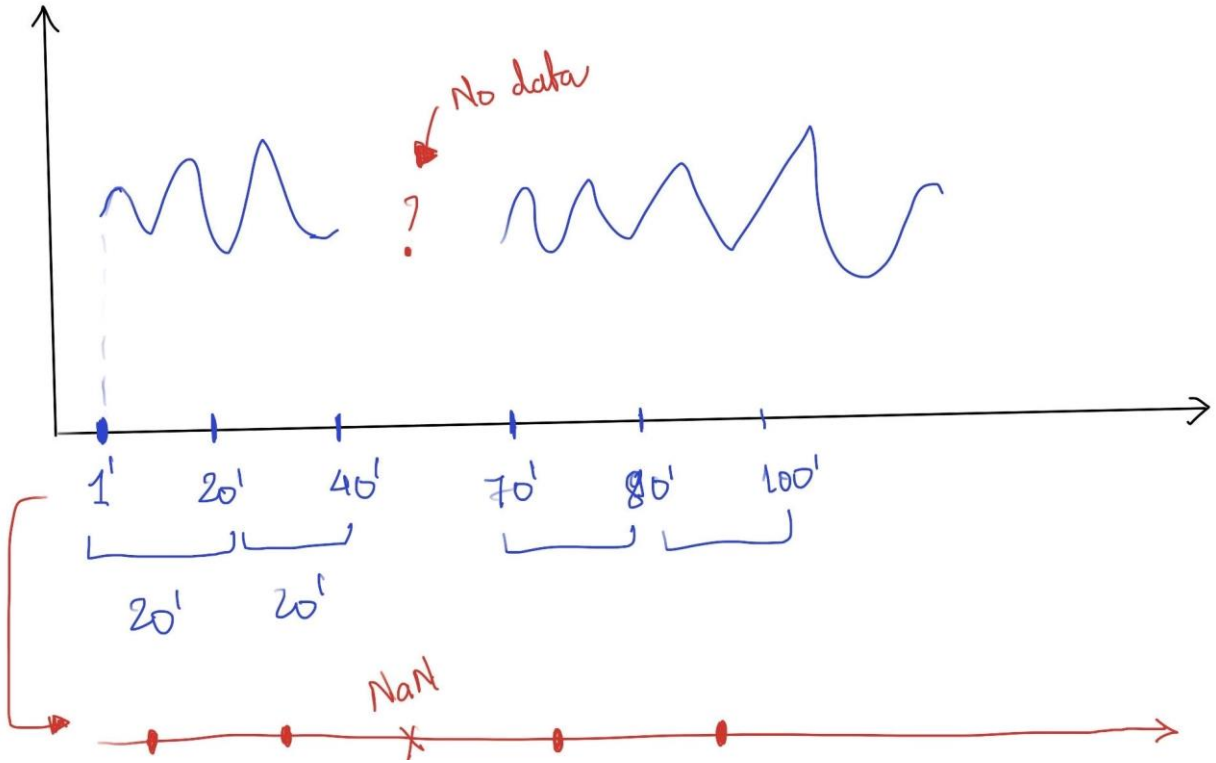


Dataswati

Handle different-length TS → resampling to 20m

popai

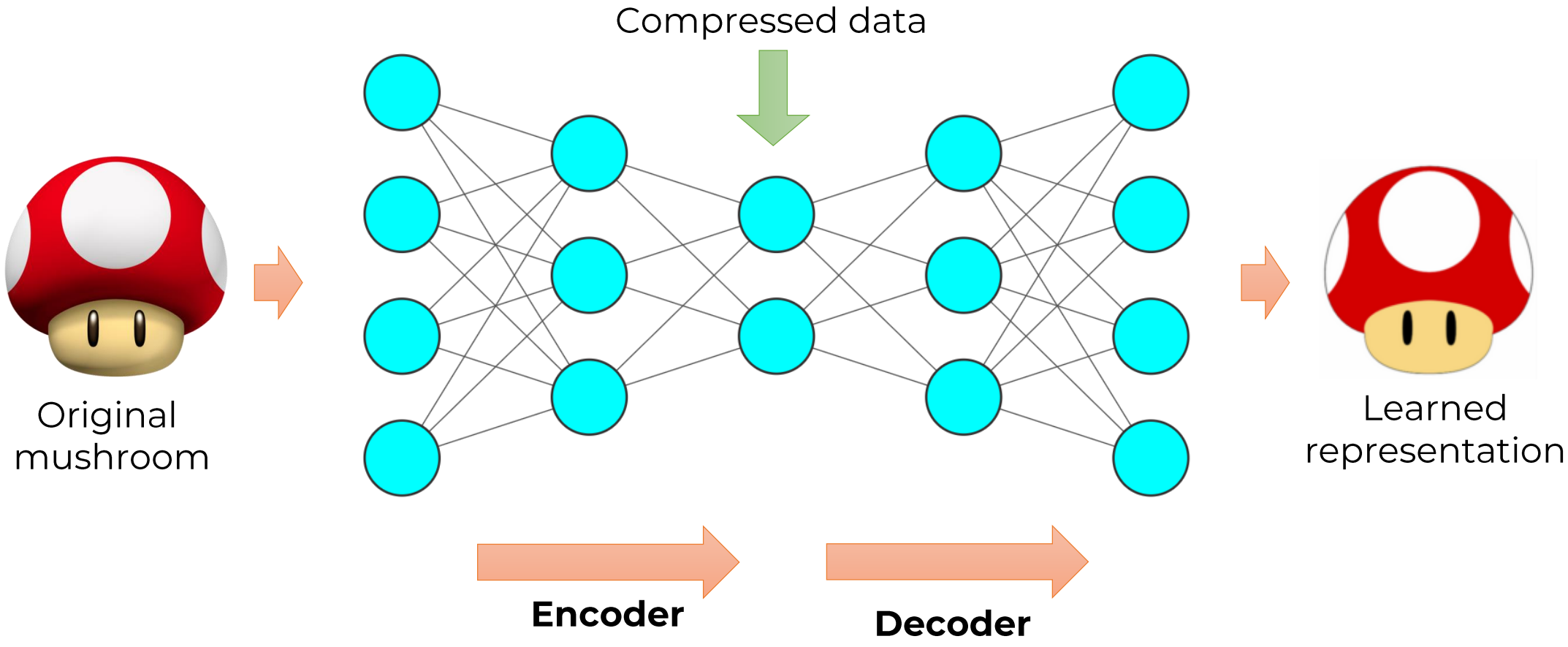
```
params_resampling = {'resample_rule': '20T', # 20 minutes
                     'groupby': ['id_couche'], # group by couches
                     'align_time': True, # align time
                     'resample_fillna': {'type': 'interpolate'}
                    }
prcs = ControllerProcessing('resampling', params_resampling)
```



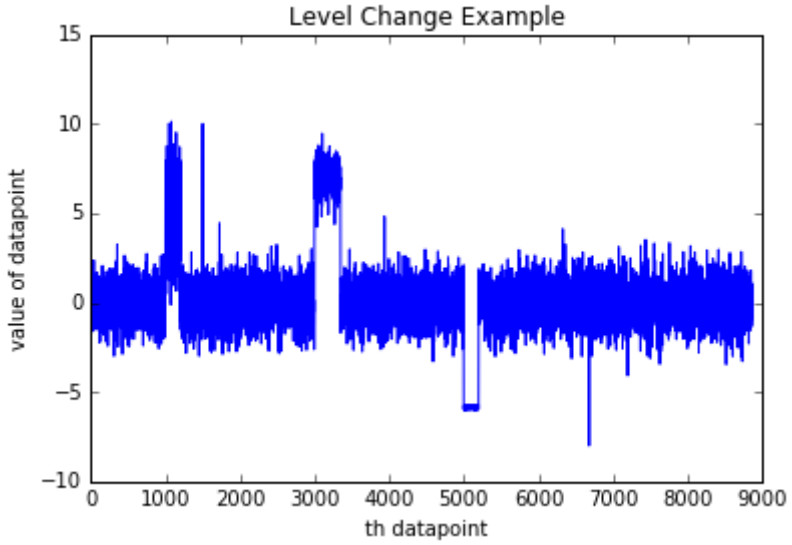
Layout of talk

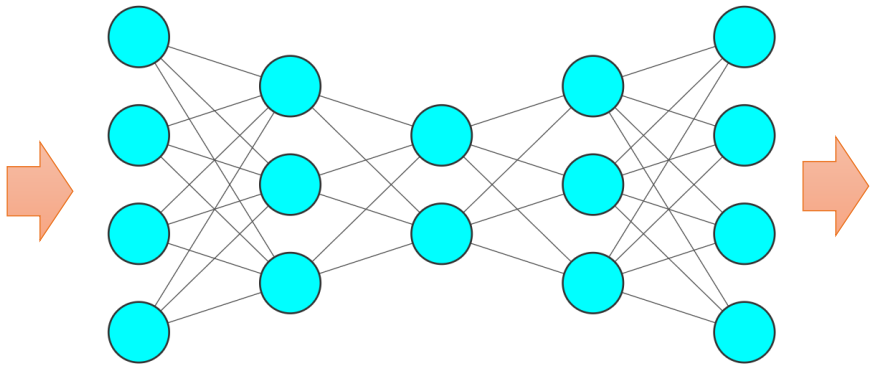
1. **tsfresh**
   - General idea
   - Example
   - Example on Soufflet data
2. **Handle different-length time series**
3. **AutoEncoder / Soufflet**
   - General idea
   - Applied to find anomaly
   - Approaches
   - Basic AE on Soufflet data
   - CNN & LSTM AE idea
   - Some comments
4. **Clustering / Soufflet**
   - General idea & examples
   - Some basic results on Soufflet data
   - Applied to recipe recommendation
5. **Anomaly detection / Aquassay**

AutoEncoder → general idea

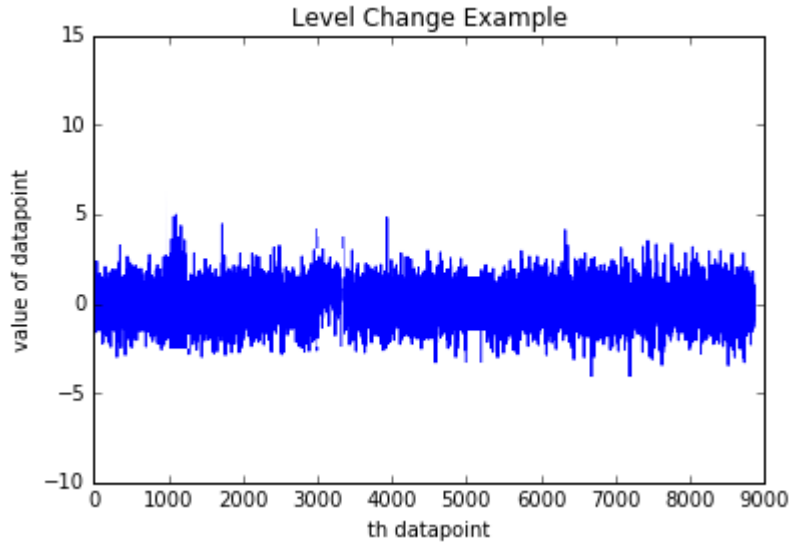


AutoEncoder → find anomaly?





Encoder **Decoder**



difference?

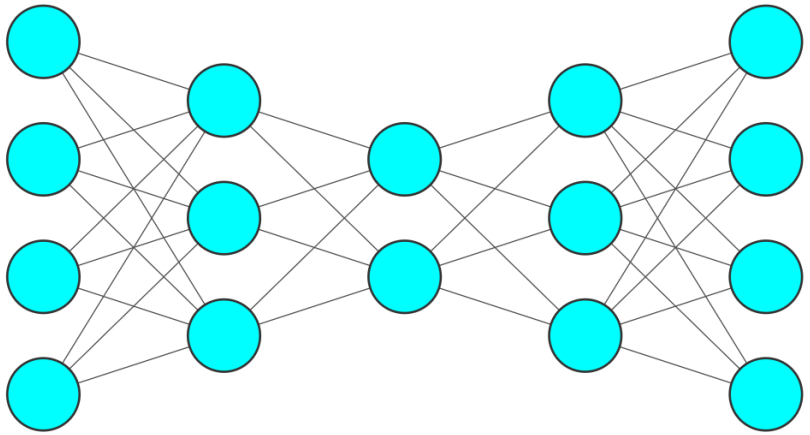
AutoEncoder → approaches?

- **TS → tsfresh → autoencoder** : make the AE after feature extraction/selection
- **TS → autoencoder** : make the AE directly on the time series data

- **Basic AE** : Linear/ReLU layers ⇒ Thi makes some tests/understanding
- **CNN AE** : usually used in image processing
- **LSTM AE** : usually used in forcasting TS

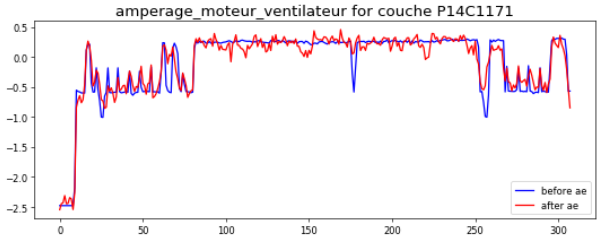
} ⇒ Alice takes in charge

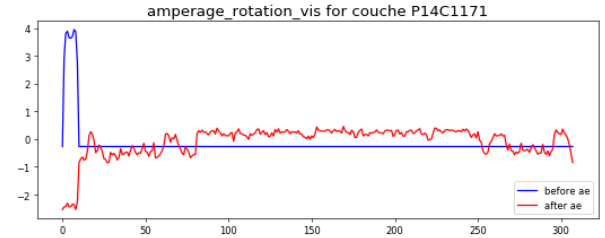
- **VAE** ⇒ not considered yet!

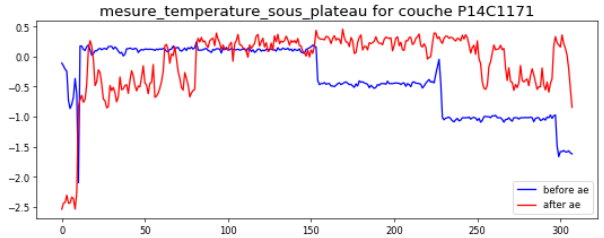


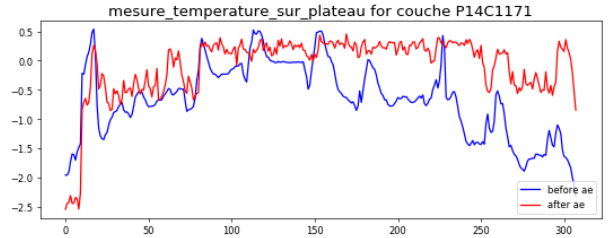
AutoEncoder → Basic AE on Soufflet

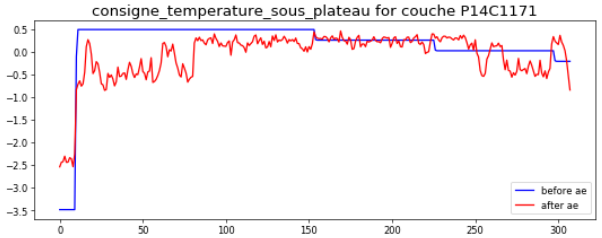
Try : Basic AE on Soufflet (directly on TS data)

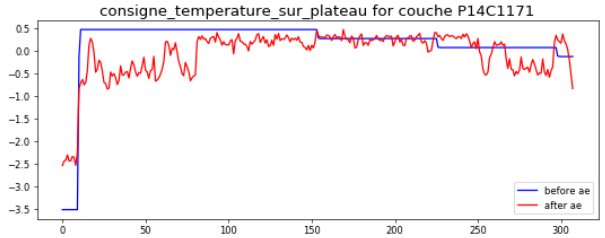


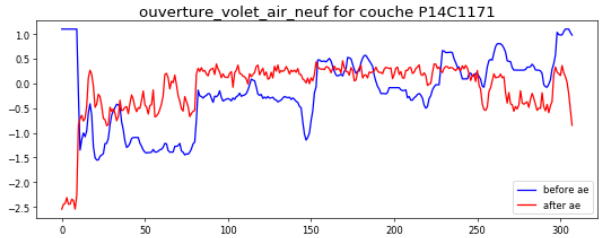


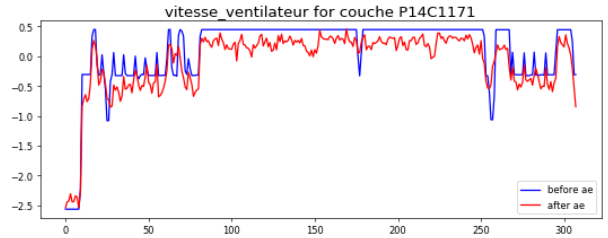


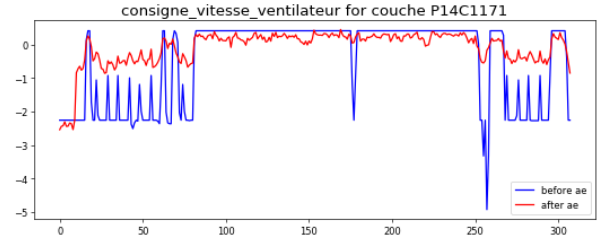


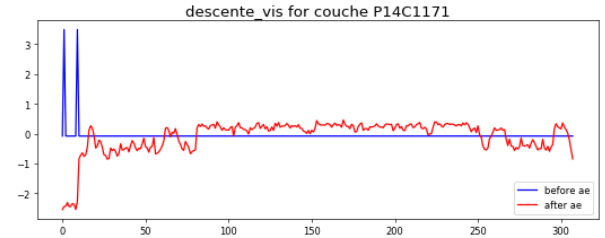


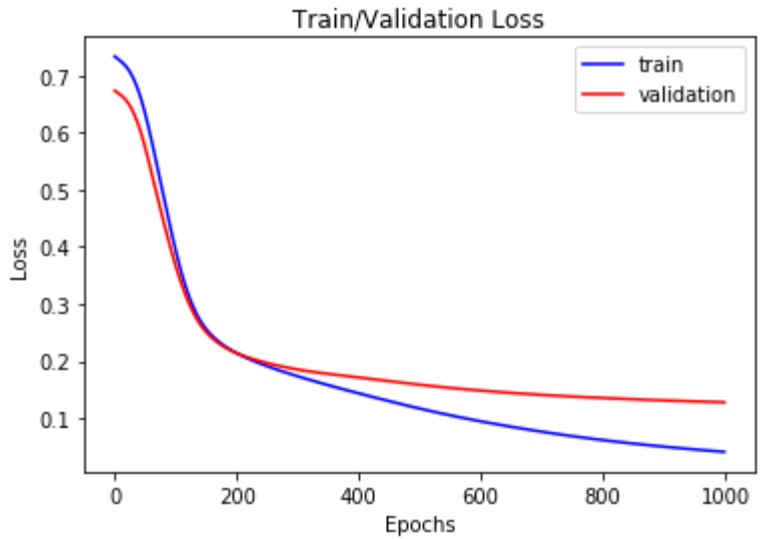












Dataswati

AutoEncoder → CNN / DeepConvLSTM idea

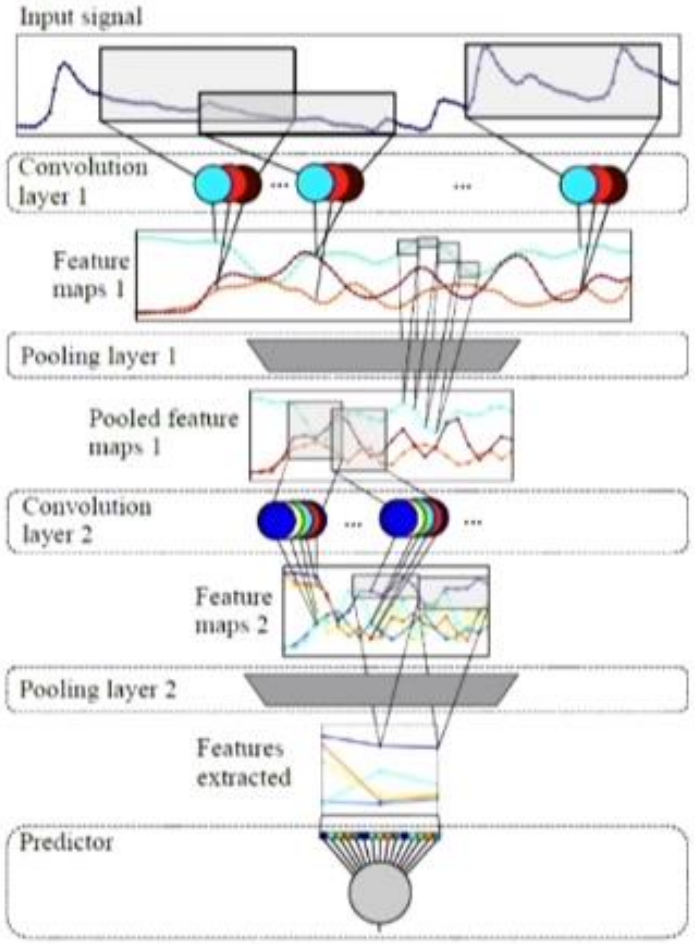


Image from: Martinez 2013

CNN

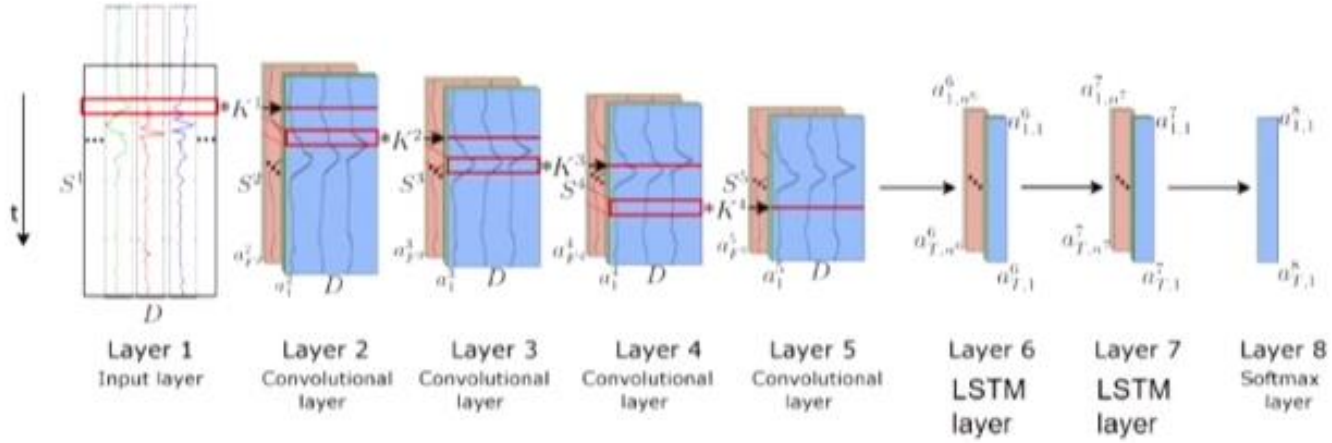


Image from: Ordonez 2016

DeepConvLSTM

Source: https://www.youtube.com/watch?v=9X_4i7zdSY8

AutoEncoder → some comments

- Choice of activations (type of layers)?
- Number of layers?
- Normalize or not? Type of normalization?
- Hyperparameters? (learning rate, regularization)
- Current working with ***Germoir_1/GLOBE-1385-1*** → only **97** couches to train NN → so few

  → Take more data (from other clients)
- Dimensional input (my problem of understanding):
  - *batch_size x channel x height x width* → **CNN**
  - *samples x time_step x features* → **RNN/LSTM**
- Problem with **CUDA / PyTorch**:

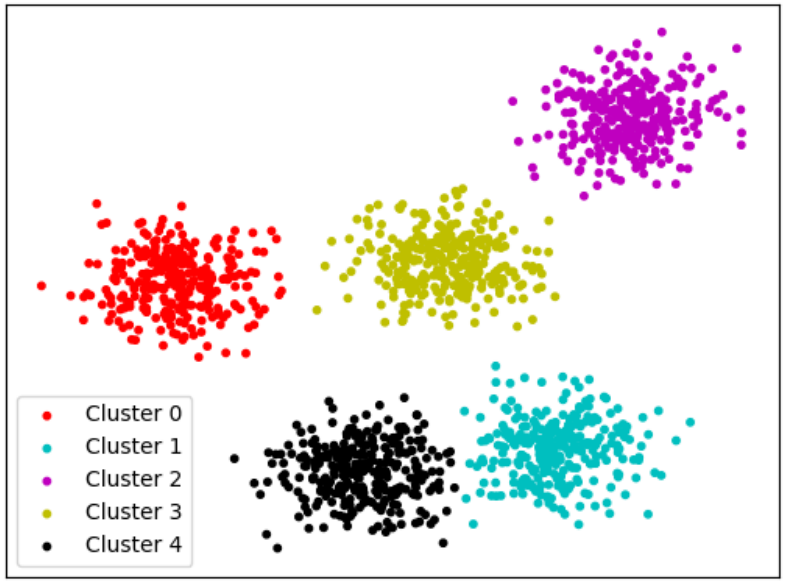
  torch==1.2.0 ← torchvision==0.4.0 ← Pillow<7.0.0

Layout of talk

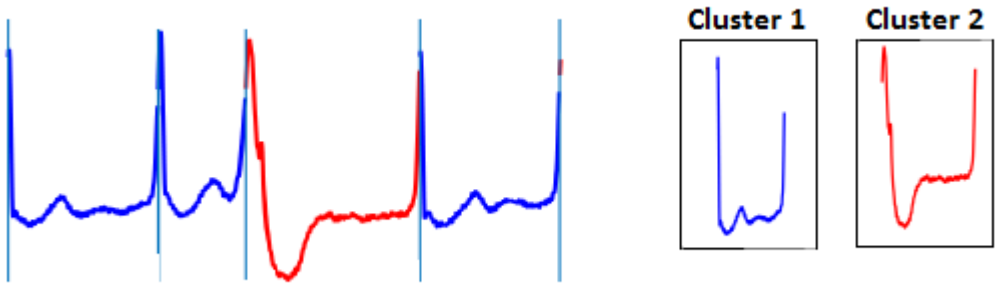
1. **tsfresh**
   - General idea
   - Example
   - Example on Soufflet data
2. **Handle different-length time series**
3. **AutoEncoder / Soufflet**
   - General idea
   - Applied to find anomaly
   - Approaches
   - Basic AE on Soufflet data
   - CNN & LSTM AE idea
   - Some comments
4. **Clustering / Soufflet**
   - General idea & examples
   - Some basic results on Soufflet data
   - Applied to recipe recommendation
5. **Anomaly detection / Aquassay**

Clustering → general idea & examples

- Grouping similar objects into the same clusters
- Same clusters → similar properties

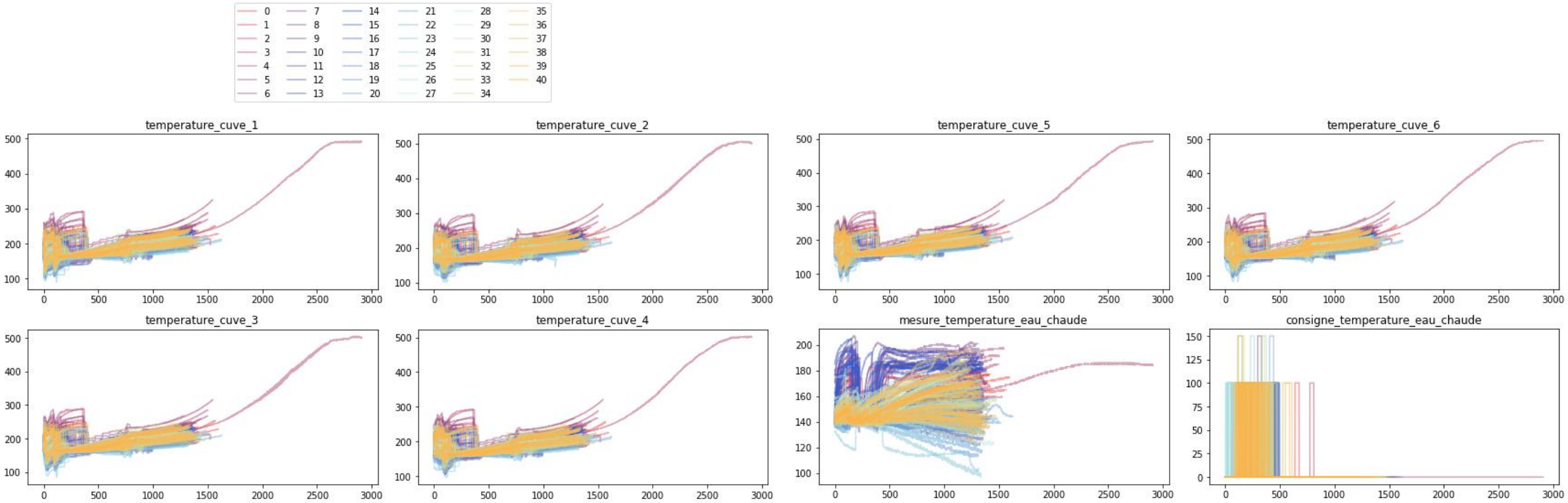


An example of clustering time series



Clustering → general idea & examples

Soufflet data: df_tsfresh + without normalize + DBSCAN + Silhouette + min_cluster_size=3



Clustering → Soufflet

df_tsfresh : Germoir_1 / GLOBE / cdc 1385-1

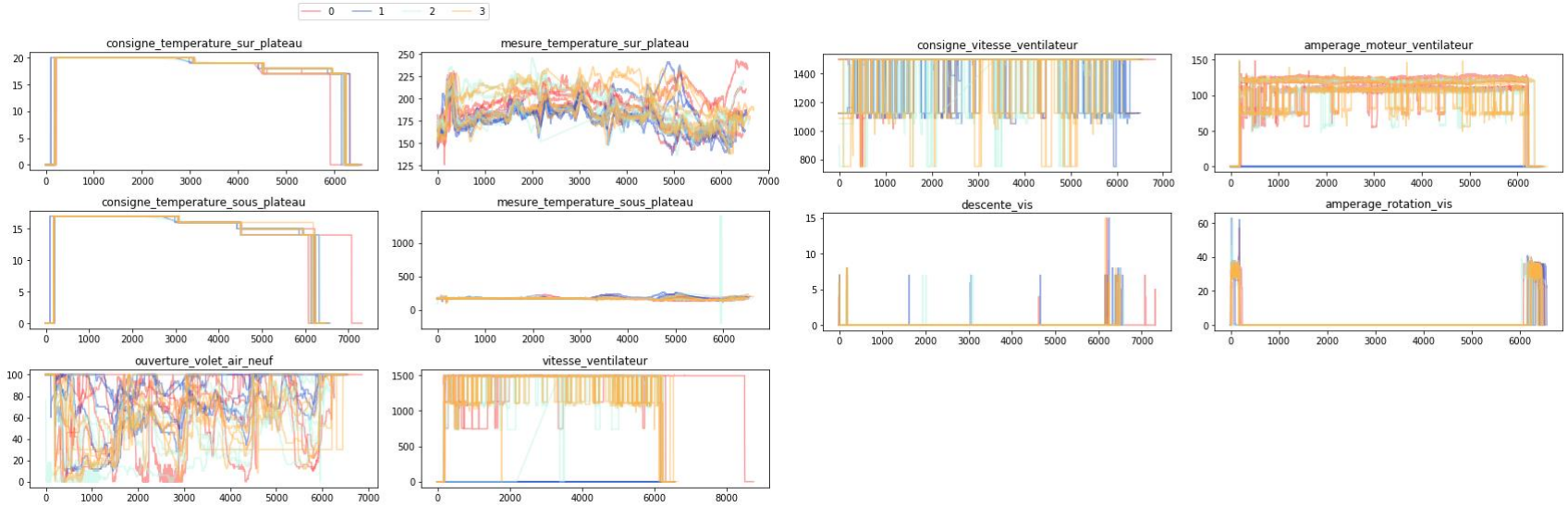
The results are not so exact and good (in visualization)

Result: The Silhouette Score with several clustering methods and dimensionality reduction methods.

	PCA	UMAP	Note
K-Means	0.2202	0.5604	PCA (35 clusters), UMAP (9 clusters)
HDBSCAN	0.2767	0.744	`min_cluster_size=3`

Calinski-Harabasz Index with HDBSCAN:

- PCA data: 12.7616
- UMAP data: 258.08



*Notebook: soufflet/blob/master/notebooks/models/**2020-01-09-Thi-clustering-tsfresh.ipynb***

Clustering

If someone needs?

dataframe

Dimensional reduction?

Normalization?

Clustering

Show clusters?

```
def clustering_couches(df, dim_reduction={}, normalize=None,
                       clustering={}, show_clusters={}):
    """Clustering the couches with several options.
    Input: df
    Output: - id of couches with their clusters
            - plot the clusters w.r.t. df's variables

    Parameters
    ----------
    df: DataFrame
        DataFrame to be used in the clustering.
    dim_reduction: dict
        The parameters to apply to dimensionality reduction model.
        They have keys:
            - 'type': type of dim_reduc model. It can be 'UMAP' or 'PCA'
            - other key parameters of the chosen model (eg. 'n_components')
    normalize: str
        The name of normalisation model. E.g. 'normalize', 'minmax_scale', 'quantile_transform', 'standard_scaler'
    clustering: str
        The parameters to apply to clustering model.
        They have keys:
            - 'key_cluster': type of clustering. It can be 'kmeans', 'hdbscan'.
            - 'range_nb_clusters': (used for 'kmeans') range of number of clusters to be tested,
            - 'random_state': default None.
            - if 'key_cluster' is 'hdbscan', you can use its custom parameters.
    show_clusters: dict
        Option to make/show the clusters w.r.t. variables.
        They have keys:
            - 'show': bool, show the plot or not?
            - 'df': The original dataframe of couches.
    """
```

*Notebook: /soufflet/blob/master/notebooks/eda/**2020-02-03-Thi-trempe-recette_logic-voisins-recette_de_germination-IM**.ipynb*

Clustering → Soufflet : recipe recommendation

df_clean : Germoir_1 / GLOBE / cdc 1385-1

```
id_couche:  P18C1325
```

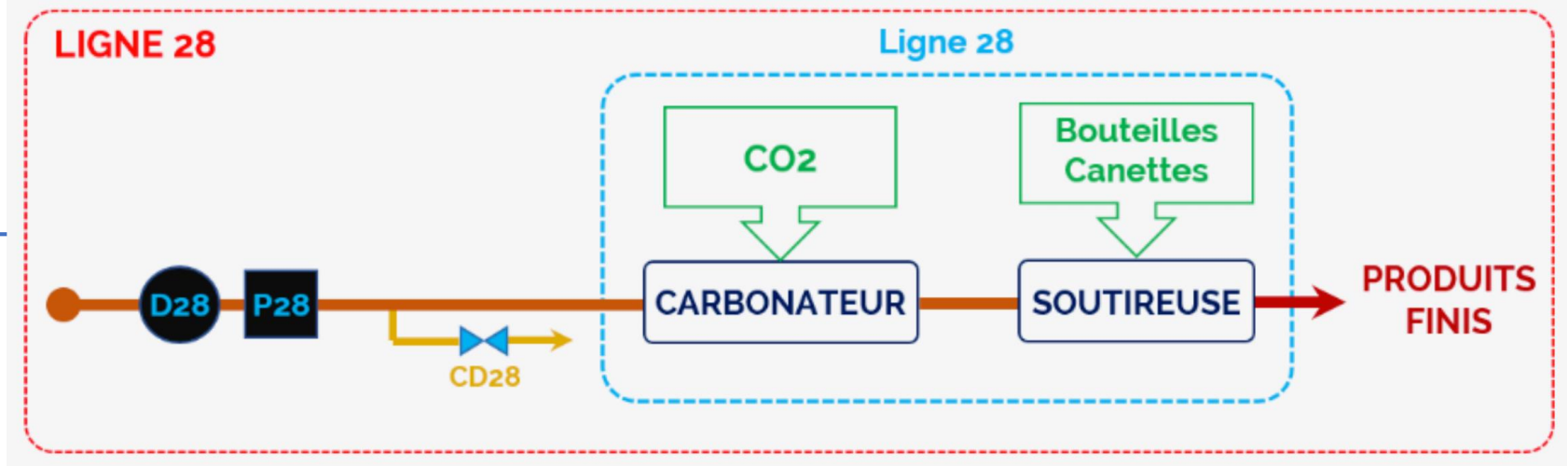
```
recette_germination
26-6rh_2017x2-1714_40-2    96.155
26_-_40-2                  84.620
31_-_40-2                  76.920
40-2                       73.080
27_-_30-2                  69.230
27-6rh_2017x2-1714_30-2    69.230
26_-_40-1                  69.230
Name: im, dtype: float64
```

*Notebook: /soufflet/blob/master/notebooks/eda/**2020-02-03-Thi-trempe-recette_logic-voisins-recette_de_germination-IM**.ipynb*

Layout of talk

1. **tsfresh**
   - General idea
   - Example
   - Example on Soufflet data
2. **Handle different-length time series**
3. **AutoEncoder / Soufflet**
   - General idea
   - Applied to find anomaly
   - Approaches
   - Basic AE on Soufflet data
   - CNN & LSTM AE idea
   - Some comments
4. **Clustering / Soufflet**
   - General idea & examples
   - Some basic results on Soufflet data
   - Applied to recipe recommendation
5. **Anomaly detection / Aquassay**

Anomaly Detection → Aquassay

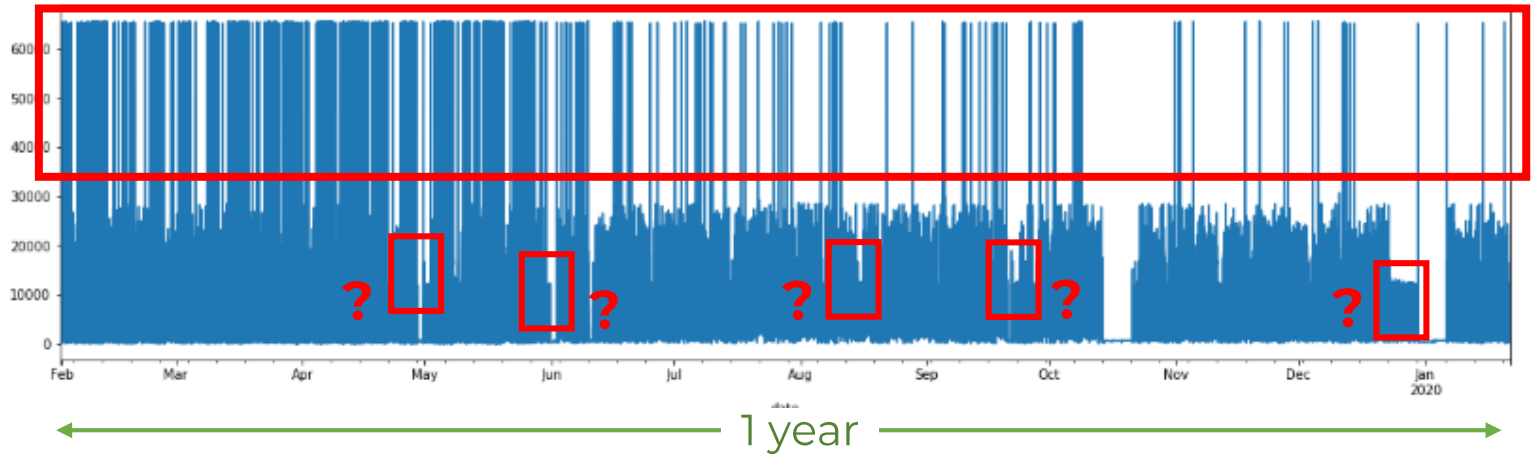


Une ligne d'embouteillage

focus on

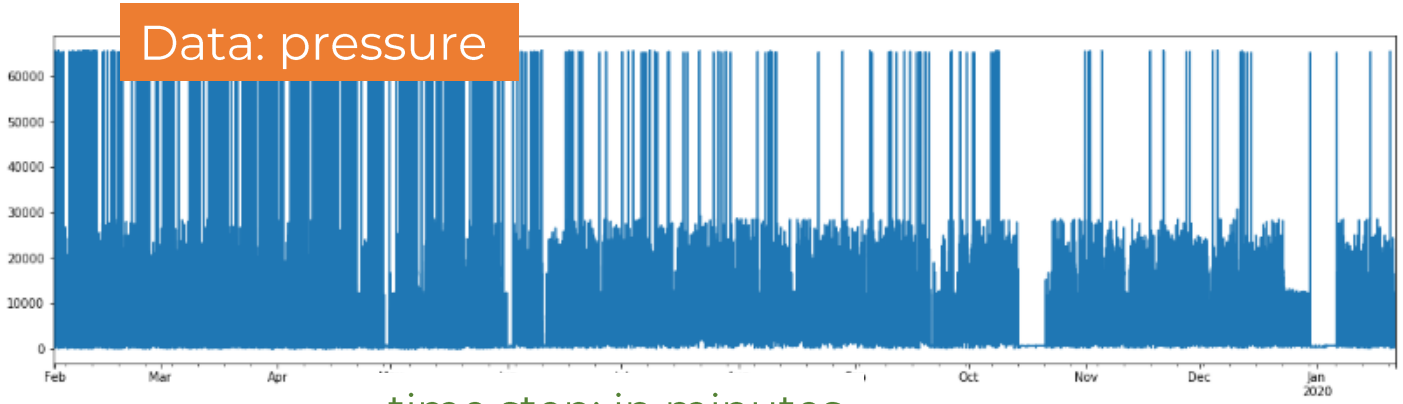
Pression entrée de ligne

Find anomalies



Anomaly Detection → Aquassay

Data: pressure



time step: in minutes

clustering

Anomaly detection

windows of 5 minutes

	id_anomalie	id_customer	ts_start	ts_end	ssii_type	id_ssii	author	type
0	0	1	1548960960000	1548961260000	sensor	1	PowerOP	3.0/8/0.282
1	1	1	1548996960000	1548997260000	sensor	1	PowerOP	3.0/8/0.282
2	2	1	1549051560000	1549051860000	sensor	1	PowerOP	3.0/8/0.282
3	3	1	1549066260000	1549066560000	sensor	1	PowerOP	3.0/8/0.282
4	4	1	1549075260000	1549075560000	sensor	1	PowerOP	3.0/8/0.282

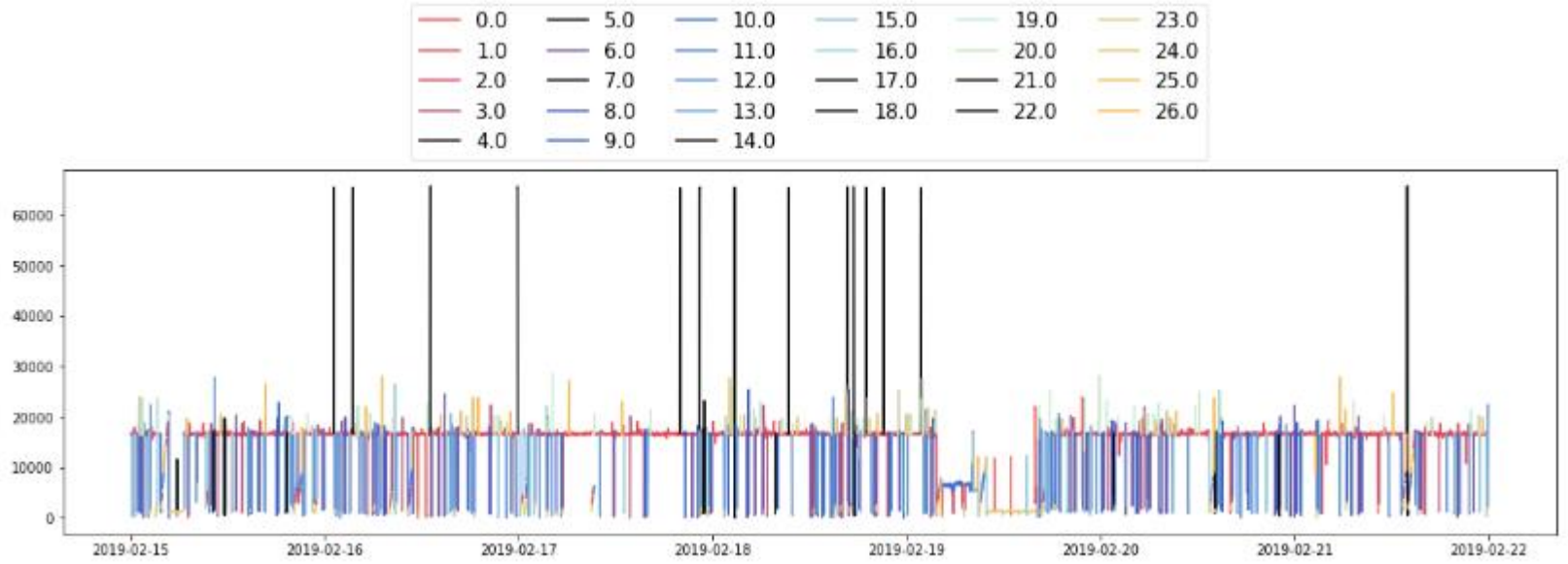
What we want

Anomaly Detection → Aquassay

clusters	f_count	f_percent
1.0	1246	61.836228
23.0	149	7.394541
10.0	67	3.325062
9.0	53	2.630273
8.0	52	2.580645
3.0	46	2.282878
20.0	46	2.282878
11.0	43	2.133995
2.0	39	1.935484
6.0	38	1.885856
13.0	32	1.588089
19.0	29	1.439206
12.0	29	1.439206
0.0	28	1.389578
15.0	27	1.339950
16.0	19	0.942928

Threshold = 0.5

24.0	19	0.942928
26.0	16	0.794045
25.0	12	0.595533
21.0	6	0.297767
4.0	5	0.248139
22.0	4	0.198511
7.0	3	0.148883
5.0	3	0.148883
14.0	2	0.099256
17.0	1	0.049628
18.0	1	0.049628



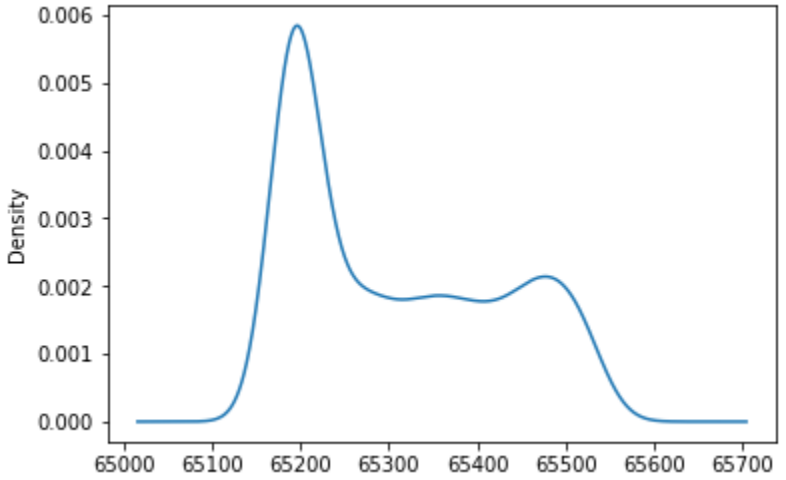
1 week

Anomal clusters

the clusters with least number of elements

(Kmeans + Silhouette + popai)

✔ Almost their elements fall into very high pressure



Dataswati

Anomaly Detection → Aquassay

Define a type? → need more discussion

which cluster / number of clusters / percentange of elements in that cluster

clusters	f_count	f_percent
1	1356	67.295285
6	187	9.280397
2	122	6.054591
8	78	3.870968
3	67	3.325062
9	54	2.679901
7	53	2.630273
0	52	2.580645
10	33	1.637717
4	7	0.347395
5	6	0.297767

	id_anomalie	id_customer	ts_start	ts_end	ssii_type	id_ssii	author	type
0	0	1	1550279100000	1550279400000	sensor	1	PowerOP	5.0/11/0.298
1	1	1	1550287500000	1550287800000	sensor	1	PowerOP	4.0/11/0.347
2	2	1	1550322000000	1550322300000	sensor	1	PowerOP	4.0/11/0.347
3	3	1	1550361000000	1550361300000	sensor	1	PowerOP	5.0/11/0.298
4	4	1	1550433300000	1550433600000	sensor	1	PowerOP	5.0/11/0.298
5	5	1	1550442000000	1550442300000	sensor	1	PowerOP	4.0/11/0.347
6	6	1	1550457600000	1550457900000	sensor	1	PowerOP	5.0/11/0.298
7	7	1	1550481600000	1550481900000	sensor	1	PowerOP	4.0/11/0.347
8	8	1	1550507760000	1550508060000	sensor	1	PowerOP	5.0/11/0.298
9	9	1	1550516160000	1550516460000	sensor	1	PowerOP	5.0/11/0.298
10	10	1	1550523960000	1550524260000	sensor	1	PowerOP	4.0/11/0.347
11	11	1	1550540460000	1550540760000	sensor	1	PowerOP	4.0/11/0.347
12	12	1	1550757060000	1550757360000	sensor	1	PowerOP	4.0/11/0.347

The end!

Thank you for your attention!