Machine Learning applied to CERN's Industrial Control Systems

Spanish High-School Students Internship Programme 2019
Miguel Martin & Celia Lopez

Introduction to Industrial Control Systems

A control system:

- Monitors
- Commands
- Maintains Processes

3 types at CERN depending of what they control:

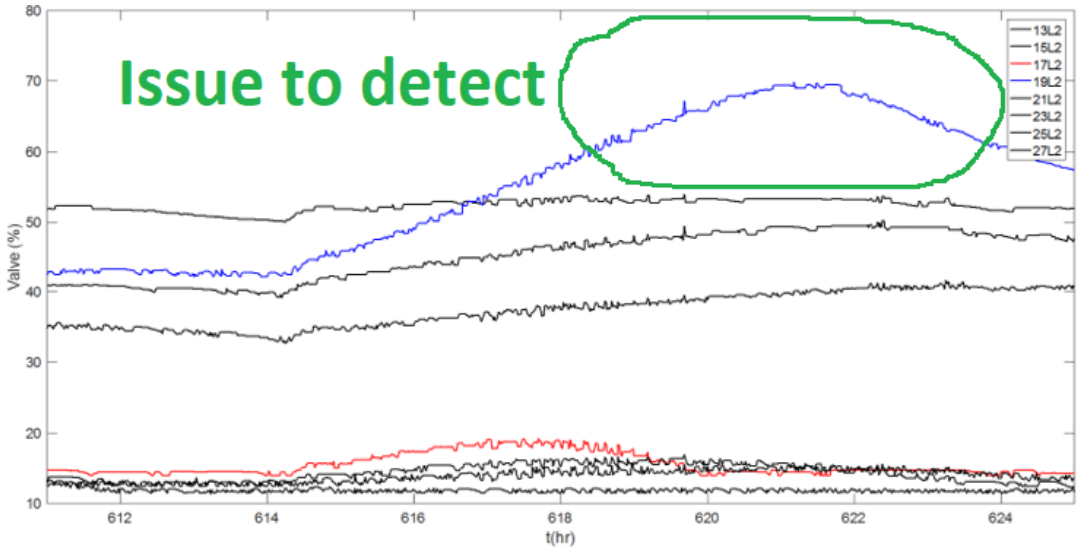
- Experiments
- Accelerator
- Infrastructure

600 control apps, 600 PLCs, ~10 million signals.

Many commercial solutions that can be used easily.

Introduction to Machine Learning

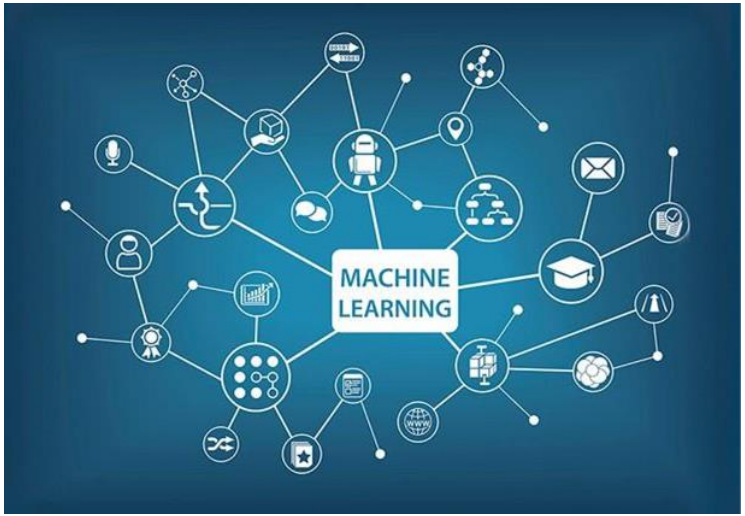
- Analyse large amounts of data automatically to identify errors, component deterioration, poor process optimization, etc…
- Using algorithms, neural networks, one can:
  - group/exclude data.
  - detect oscillations in commands to devices.
  - predict signal behaviors.
  - train neural networks to make them more precise.



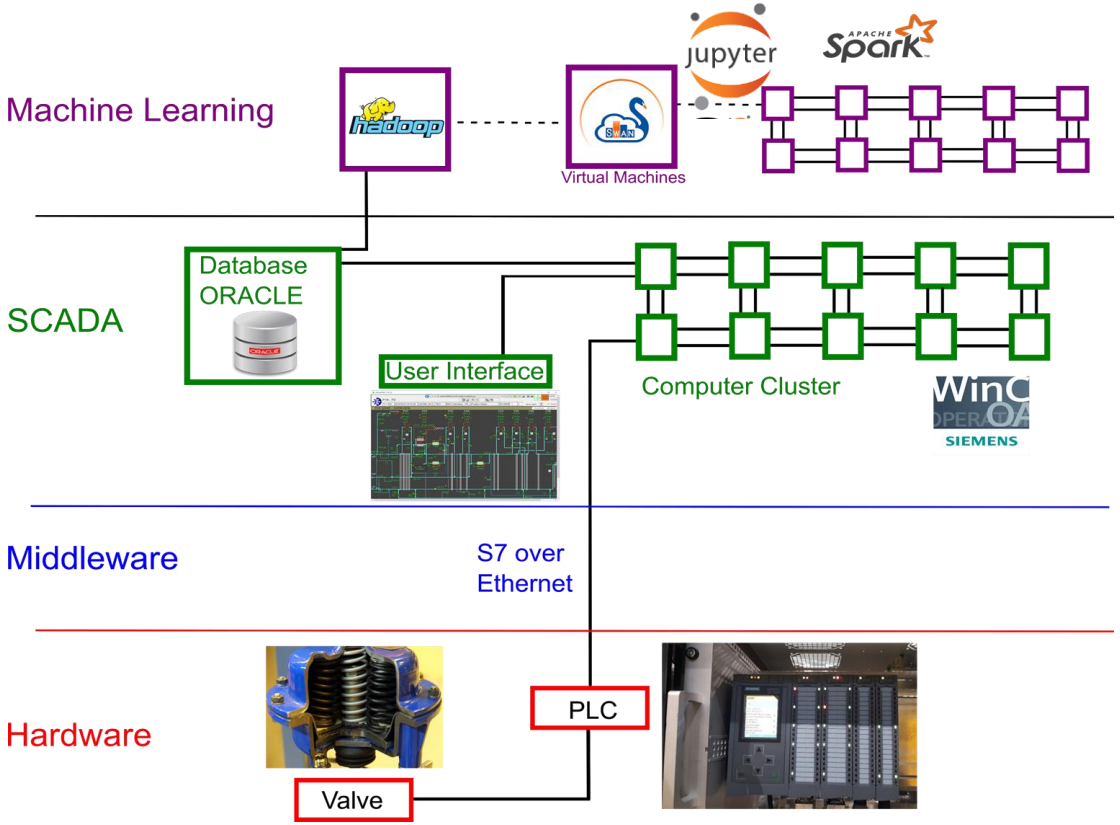
PROJECT OBJECTIVES

Why is this project interesting?

- It uses the latest technologies available in this technology sector.
- It allows us to understand how data can be captured from signals.
- How signals are processed to store them.
- How those signals are used to:
  - Predict when sensors and actuators are going to stop working.
  - Detect deviated signals from millions of other values.

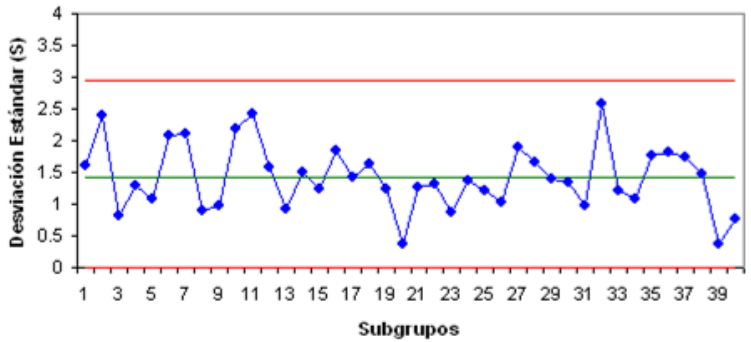


Acquisition of data in a Control System



Machine Learning concepts: Mean and Standard Deviation

- The **Mean** (applied to signals) : the average value of the sampled signal.
- The **Standard Deviation**: the difference between a value and the mean of the signal.



In [17]:
```
mean = 0

for element in x:
   mean = mean + element

mean = mean /  N
print (mean)
```

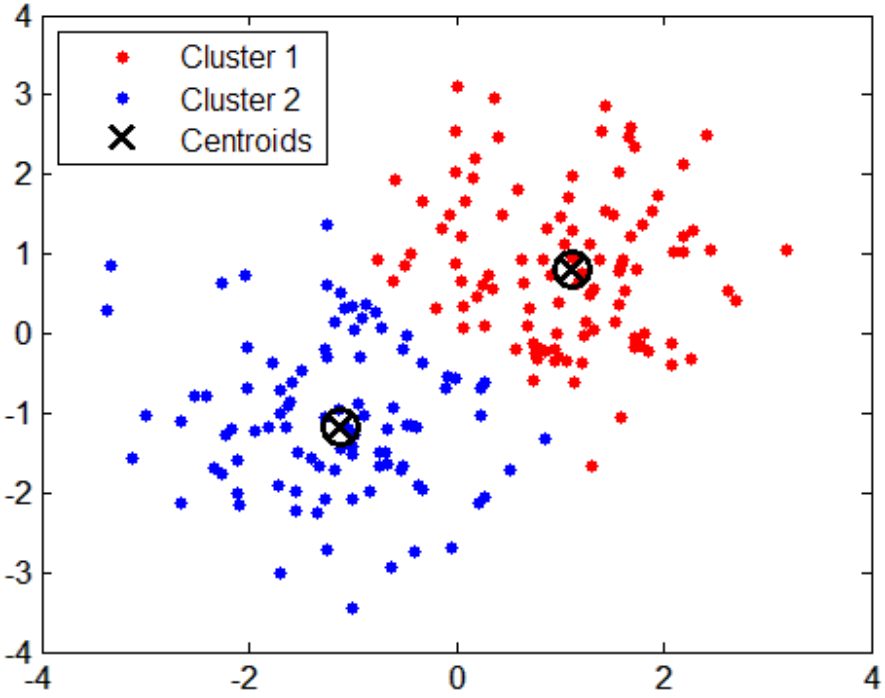
```
5.575
```

In [18]:
```
variance = 0

for element in x:
   variance = variance + (element - mean)**2

variance = variance /  (N - 1)
sd = math.sqrt (variance)
print (variance)
print (sd)
```

```
6.09679487179487
2.4691688625517028
```

Machine Learning Algorithm: kmeans

The kmeans Machine Learning algorithm groups data samples (represented by points) depending on their values so that the area of each cluster (group) is minimal and the number of samples contained in it is the largest possible.



Our ML example: Cryogenics data analysis

1. **The signal comes from different valves**
2. Using jupyter notebook, clean up the data
3. Visually pre-analyse the data
4. Set number of groups and let the machine Learning algorithm do its magic

```
df = pandas.read_csv('TIMBER_DATA_CV910_20170626_processed.csv')
```

```
10000,12.493999360246853,13.058107503996196,12.54702984779533,12.93657241289762
737060547,55.96876796541541,50.42833793487733,54.98828125,56.02733612060547,52.
11000,12.496742365162318,13.054945956507224,12.545313772077344,12.9364068748456
51737060547,55.96786323226516,50.43881185899584,54.98828125,56.02733612060547,5
12000,12.499485370077783,13.051784409018252,12.543597696359361,12.9362413367936
51737060547,55.96695849911491,50.449285783114355,54.98828125,56.02733612060547,

13000,12.502228374993248,13.048622861529278,12.541881620641377,12.9360757987416
37060547,55.96605376596466,50.45975970723287,54.98828125,56.02733612060547,52.7
14000,12.504971379908714,13.045461314040306,12.540165544923394,12.9359102606896
75864101902,55.965149032814416,50.47023363135139,54.98828125,56.02733612060547,
15000,12.507714384824178,13.042299766551334,12.538449469205409,12.9357447226376
323413510716,55.964244299664166,50.480707555469905,54.98828125,56.0273361206054

16000,12.510457389739642,13.03913821906236,12.536733393487426,12.93557918458563
3818600241,55.963339566513916,50.49118147958842,54.98828125,56.02733612060547,5
17000,12.513200394655108,13.035976671573389,12.535017317769443,12.9354136465336
529584941,55.962434833363666,50.501655403706934,54.98828125,56.02733612060547,5
18000,12.515943399570572,13.032815124084415,12.533301242051458,12.9352481084816
```

We have 15600 samples, 1200 for each valve, 13 valves in total, this is just an example, normally there are many, many, many more

Our ML example: Cryogenics data analysis

1. The signal comes from different valves
2. **Using jupyter notebook, clean up the data**
3. Visually pre-analyse the data
4. Set number of groups and let the machine Learning algorithm do its magic

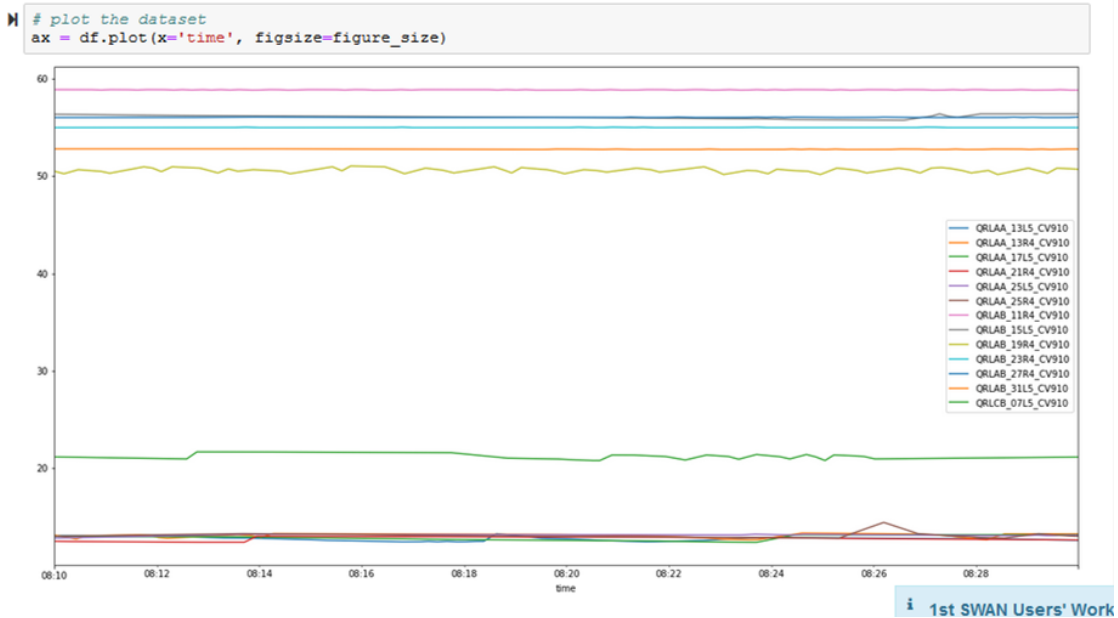
```
In [3]: df.describe()
```

Out[3]:

	QRLAA_13L5_CV910	QRLAA_13R4_CV910	QRLAA_17L5_CV910	QRLAA_21R4_CV910
count	1200.000000	1200.000000	1200.000000	1200.000000
mean	12.780490	13.147989	12.889128	12.810920
std	0.197514	0.179407	0.250931	0.198418
min	12.468750	12.671873	12.414062	12.425781
25%	12.615010	13.051345	12.661909	12.701414
50%	12.799195	13.211892	12.938322	12.886248
75%	12.883938	13.285154	13.195312	12.965741
max	13.308394	13.366751	13.195312	13.042718

Our ML example: Cryogenics data analysis

1. The signal comes from different valves
2. Using jupyter notebook, clean up the data
3. **Visually pre-analyse the data**
4. Set number of groups and let the machine Learning algorithm do its magic

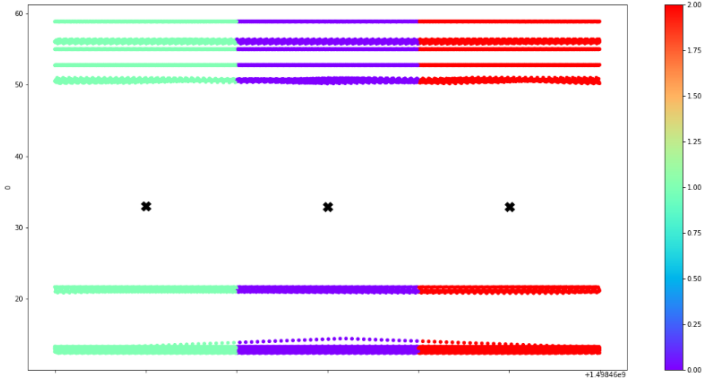


Our ML example: Cryogenics data analysis

1. The signal comes from different valves
2. Using jupyter notebook, clean up the data
3. Visually pre-analyse the data
4. **Set number of groups and let the machine Learning algorithm do its magic**

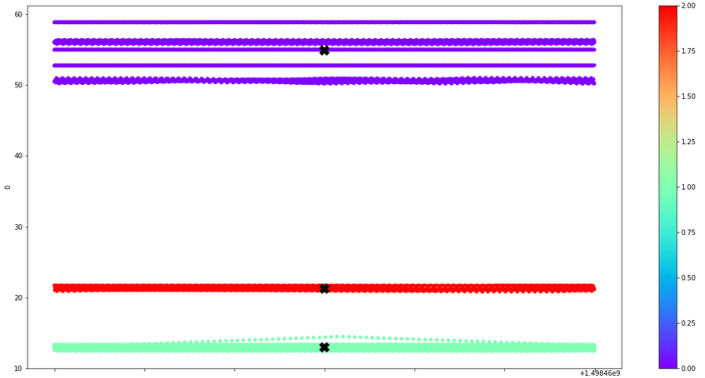
Although the algorithm is mathematically correct the solution is not what we wanted

```
In [6]: whole_set = df.iloc[:,1:].stack().to_frame()
        time_numeric = pandas.Series(pandas.to_timedelta(df.iloc[:,0]).dt.total_seconds())


        tm = pandas.concat([time_numeric]*df.iloc[:,1:].shape[1], ignore_index=True)

        df2 = tm.to_frame().reset_index().join(whole_set.reset_index())

        df2

        df2 = df2.drop(['index','level_0','level_1'],axis=1)
        df2.head()

        mid_time = time_numeric[len(time_numeric)/2]

        startpts=np.array([[mid_time, df2[0].max()], [mid_time, df2[0].min()], [mid_time, df2[0].mean()/2]], np.float64)
        kmeans = KMeans(n_clusters=3)
        kmeans.fit(df2)

        df2.plot.scatter(x='time',y=0,c=kmeans.labels_, cmap='rainbow', figsize=figure_size)
        plt.scatter(kmeans.cluster_centers_[:,0] ,kmeans.cluster_centers_[:,1], color='black', marker="X", s=200)

        print(whole_set[kmeans.labels_==0].describe())
        print
        print(whole_set[kmeans.labels_==1].describe())
        print
        print(whole_set[kmeans.labels_==2].describe())
```



Our ML example: Cryogenics data analysis with initial points

- The signal comes from different valves.
- Using jupyter notebook, clean up the data.
- Visually preanalyse the data.
- Set number of groups needed, **specify starting conditions** and let the Machine Learning algorithm do its magic.



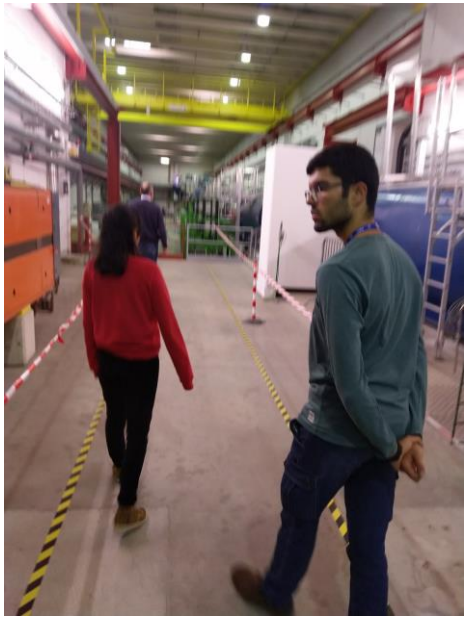
```
In [6]: whole_set = df.iloc[:,1:].stack().to_frame()
        time_numeric = pandas.Series(pandas.to_timedelta(df.iloc[:,0]).dt.total_seconds())


        tm = pandas.concat([time_numeric]*df.iloc[:,1:].shape[1], ignore_index=True)

        df2 = tm.to_frame().reset_index().join(whole_set.reset_index())

        df2

        df2 = df2.drop(['index','level_0','level_1'],axis=1)
        df2.head()

        mid_time = time_numeric[len(time_numeric)/2]

        startpts=np.array([[mid_time, df2[0].max()], [mid_time, df2[0].min()], [mid_time, df2[0].mean()/2]], np.float64)
        kmeans = KMeans(n_clusters=3, init=startpts, n_init=1)
        kmeans = KMeans(n_clusters=3)
        kmeans.fit(df2)

        df2.plot.scatter(x='time',y=0,c=kmeans.labels_, cmap='rainbow', figsize=figure_size)
        plt.scatter(kmeans.cluster_centers_[:,0] ,kmeans.cluster_centers_[:,1], color='black', marker="X", s=200)

        print(whole_set[kmeans.labels_==0].describe())
        print
        print(whole_set[kmeans.labels_==1].describe())
        print
        print(whole_set[kmeans.labels_==2].describe())
```

Conclusions: when we specify the initial points we can obtain exactly what we wanted

Conclusions

- CERN Industrial Control Systems are really complex.
- Standardisation simplifies maintenance and makes component integration easier.

- Signals have to be processed many times before they can be used for Machine Learning.
- Machine Learning makes maintenance more efficient and reduces costs.
- Machine Learning is an emerging topic and there is much work left to do around it.



Do you have any questions?