POLITECNICO
MILANO 1863

Failure Management in Microwave Networks

Francesco Musumeci

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB)

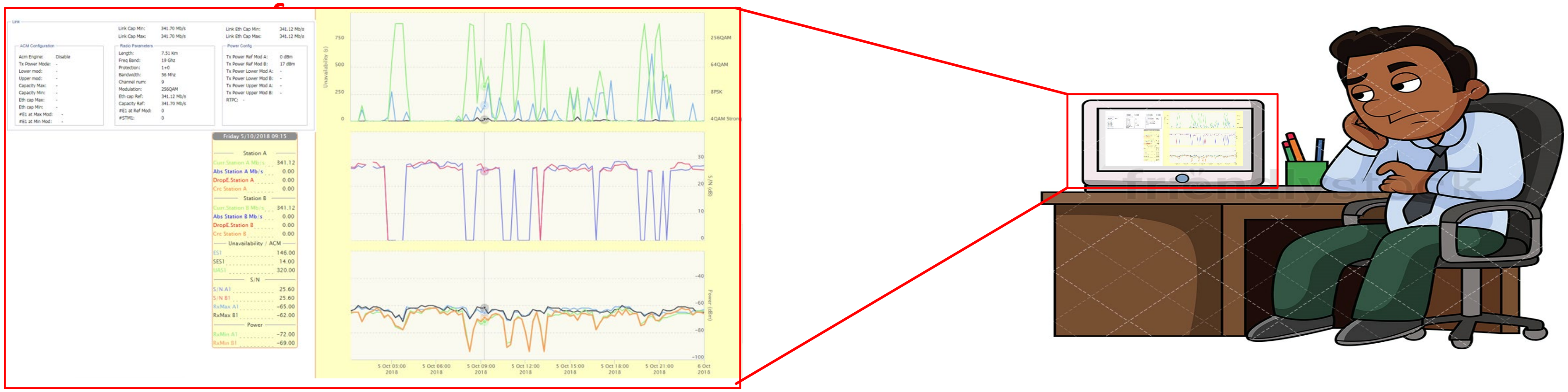
Politecnico di Milano, Milano, Italy

Failure management in microwave networks
Background and motivation

- Current mode of operation: “human-assisted” management

Step 1: Network Management System (NMS) provides measurements related to links’

Step 2: Network experts are required to identify causes by observing and correlating several graphs and statistics



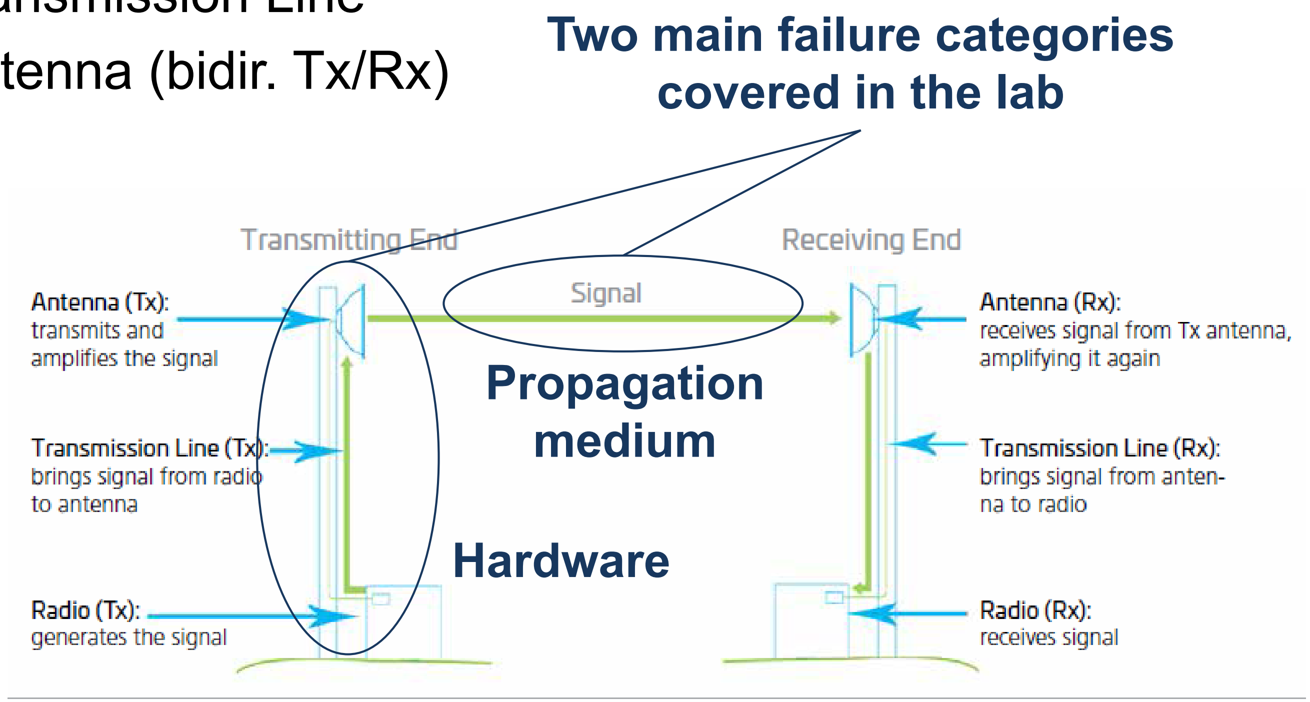
- Highly inefficient (costly and time-consuming)

→ Need for ***automated*** ***failure-cause identification***

Quick failure recovery → reduced service degradation

Failure management in microwave networks
Microwave Links

- Line of Sight (LoS) Microwave Links
  - 3 Basic building blocks:
    - Radio
    - Transmission Line
    - Antenna (bidir. Tx/Rx)

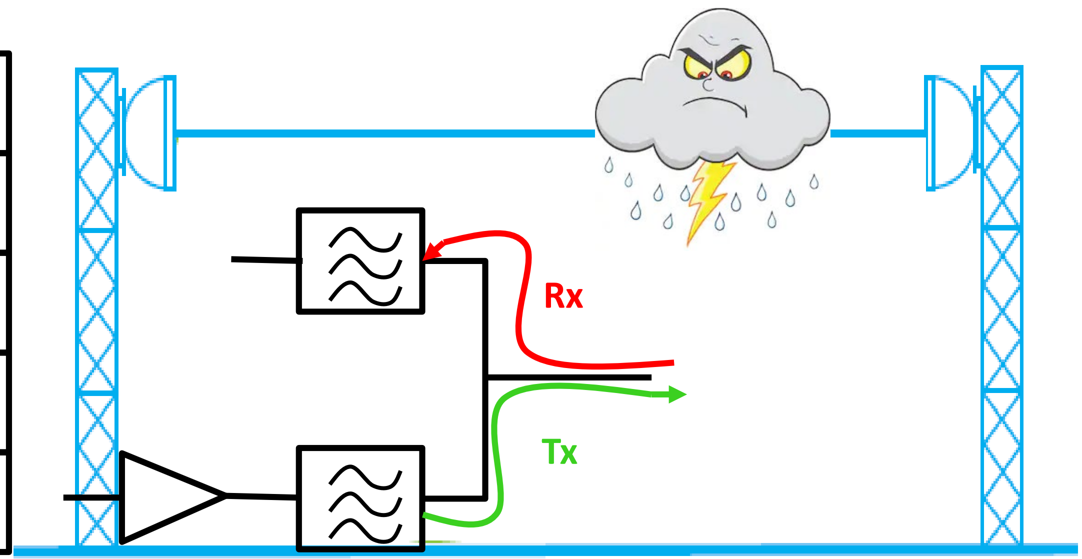


The basic components that allow LOS microwave communications

Failure management in microwave networks
Failure causes

- Category A: 5 Propagation failure causes (5 classes)

Class #0	**Deep Fading**
Class #1	**Extra Attenuation**
Class #2	**Interference**
Class #3	**Low Margin**
Class #4	**Self Interference**



- Category B: 1 Hardware failures (1 class)

Class #5	**Hardware failures**

Raw data

- NMS extracts raw data from several bidirectional links **every 15-minutes**

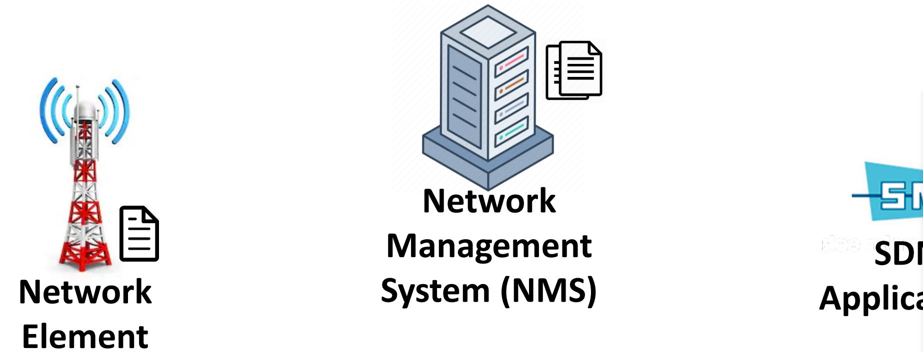
- The elements in the DB are described by 39 attributes

General Information
ITU-T G.828* performance
Tx Power
Other measures
Link design Characteristics
Rx Power
Quality Flags

* ITU-T Rec. G.828 : Error performance parameters and objectives for international, constant bit-rate synchronous digital paths. Available at https://www.itu.int/rec/T-REC-G.828-200003-I/en

Raw data

- NMS extracts raw data from several bidirectional links **<u>every 15-minutes</u>**



- The elements in the DB are descr

General Information

ITU-T G.828* performance

Link design Characteristics

Rx Powe

- Errored Seconds (ES)
- Severely Errored Seconds (SES)
- UnAvailability Seconds (UAS)

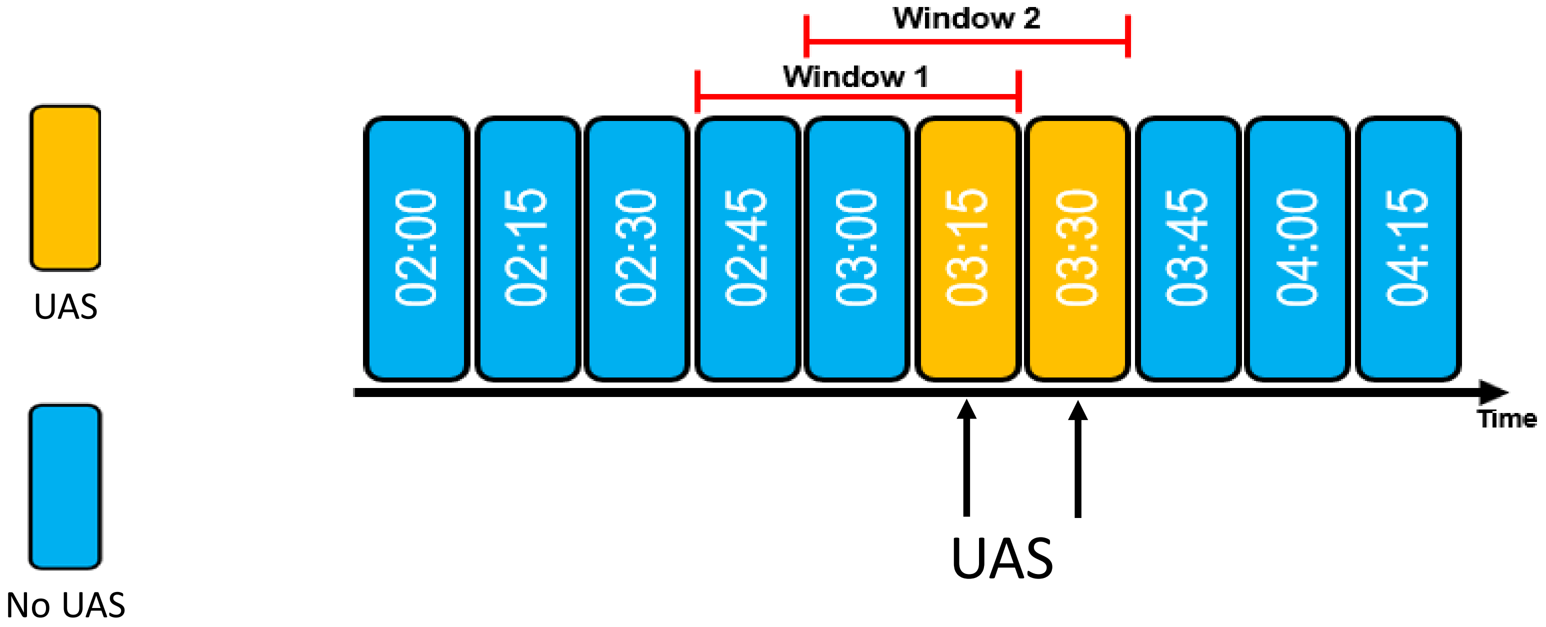
Main difference is in the amount of consecutive seconds with errors above a certain threshold

- **We consider presence of UAS as the failure event**

* ITU-T Rec. G.828 : Error performance parameters and objectives for international, constant bit-rate synchronous digital paths. Available at https://www.itu.int/rec/T-REC-G.828-200003-I/en

Available dataset (partly pre-processed)

- Several bidirectional links, data collected for around 18 months in 2018-2019 in a real microwave network
- One data point: 45-minutes windows (3 consecutive slots of 15 minutes), where the last slot has UAS



Available dataset (partly pre-processed)

	A	B	C	D	E	F	G	H	I	J	K
1	idlink	dataN-2	dataN-1	dataN	eqtype	acmLo	freqba	bandw	acmEr	esN-2	sesN-2
2	287	29/10/2018 15:30	29/10/2018 15:45	29/10/2018 16:00	19	2	18	28	1	0.0	0.0
3	287	29/10/2018 15:45	29/10/2018 16:00	29/10/2018 16:15	19	2	18	28	1	0.0	0.0
4	1882	29/10/2018 16:00	29/10/2018 16:15	29/10/2018 16:30	19	2	18	56	1	0.0	0.0
5	1882	29/10/2018 16:15	29/10/2018 16:30	29/10/2018 16:45	19	2	18	56	1	0.0	0.0
6	1882	29/10/2018 18:45	29/10/2018 19:00	29/10/2018 19:15	19	2	18	56	1	0.0	0.0
7	1882	29/10/2018 19:00	29/10/2018 19:15	29/10/2018 19:30	19	2	18	56	1	0.0	0.0
8	2356	29/10/2018 16:00	29/10/2018 16:15	29/10/2018 16:30	19	2	38	28	1	0.0	0.0
9	7774	02/01/2019 12:45	02/01/2019 13:00	02/01/2019 13:15	19	2	38	28	1	15.0	0.0
10	7774	02/01/2019 13:00	02/01/2019 13:15	02/01/2019 13:30	19	2	38	28	1	1.0	0.0
11	7774	02/01/2019 13:15	02/01/2019 13:30	02/01/2019 13:45	19	2	38	28	1	70.0	9.0
12	7774	02/01/2019 13:30	02/01/2019 13:45	02/01/2019 14:00	19	2	38	28	1	69.0	27.0
13	7774	02/01/2019 13:45	02/01/2019 14:00	02/01/2019 14:15	19	2	38	28	1	106.0	46.0
14	7774	02/01/2019 19:30	02/01/2019 19:45	02/01/2019 20:00	19	2	38	28	1	0.0	0.0

...

AN	AO	AP	AQ	AR
lowthr	ptx	RxNominal	Thr_min	label
-85.0	23.0	-44	-8	0.0
-85.0	23.0	-44	-8	0.0
-82.5	23.0	-52	-8	0.0
-82.5	23.0	-52	-8	0.0
-82.5	23.0	-52	-8	0.0
-82.5	23.0	-52	-8	0.0
-83.0	19.0	-45	-8	0.0
-83.0	19.0	-41	-8	4.0
-83.0	19.0	-41	-8	4.0
-83.0	19.0	-41	-8	4.0
-83.0	19.0	-41	-8	4.0
-83.0	19.0	-41	-8	4.0
-83.0	19.0	-41	-8	1.0

Non-relevant features (link ID, date/time, equipment type, bandwidth …)

LowThr
Ptx
RxNominal
Thr_min
acmEngine

Link (design) characteristics

es
ses
txMaxA
txminA
rxmaxA
rxminA
txMaxB
txminB
rxmaxB
rxminB

3x Link Measures (one for each of the 15-minutes slots: N-2 N-1 N)

Labels:

Class no.	Failure cause	# of instances
0	Deep Fading	284
1	Extra Attenuation	581
2	Interference	49
3	Low Margin	190
4	Self-Interference	187
5	HW Failure	1222

Failure management in microwave networks
Source paper

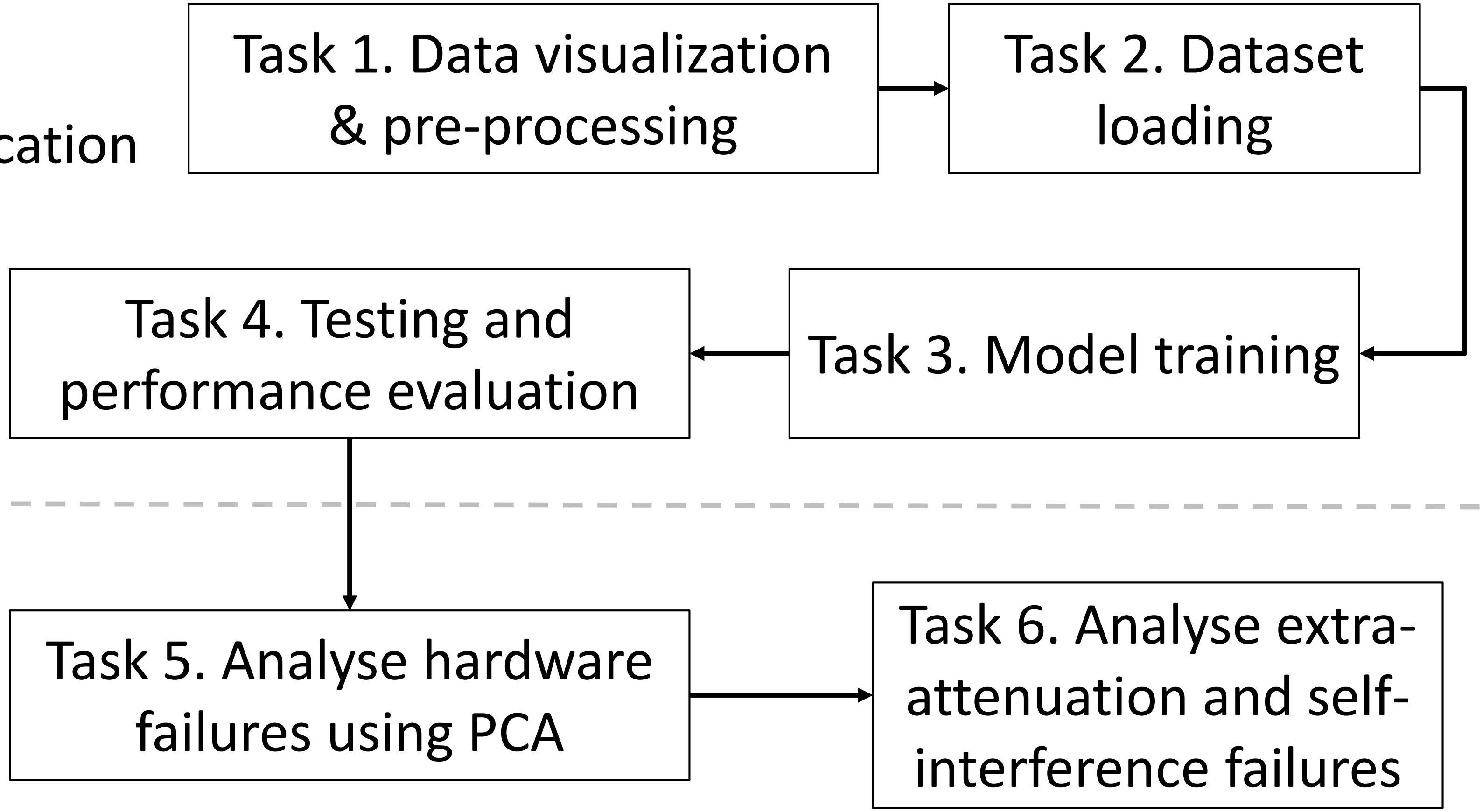
TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, VOL. X, NO. Y, NOVEMBER 19, 2020 1

Supervised and Semi-Supervised Learning for Failure Identification in Microwave Networks

Francesco Musumeci, *Member, IEEE,* Luca Magni, Omran Ayoub, Roberto Rubino, Massimiliano Capacchione, Gabriele Rigamonti, Michele Milano, Claudio Passera, and Massimo Tornatore, *Senior Member, IEEE*

Failure management in microwave networks
Lab overview

Part I: classification



Part II: clustering

- Part I: classification

Failure management in microwave networks
Task 1

1. Dataset pre-processing (use file “Labelled_Multiclass.csv”):
   a) Load data in pandas dataframe, delete unnecessary info (*idlink, eqtype, acmLowerMode, freqband, bandwidth*), and visualize data points in tabular form

Expected output

	acmEngine	esN-2	sesN-2	txMaxAN-2	txminAN-2	rxmaxAN-2	rxminAN-2	txMaxBN-2	txminBN-2
0	1	0.0	0.0	18.0	18.0	-49.0	-49.0	18.0	18.0
1	1	0.0	0.0	18.0	18.0	-49.0	-53.0	18.0	18.0
2	1	0.0	0.0	18.0	18.0	-45.0	-47.0	18.0	18.0
3	1	0.0	0.0	18.0	18.0	-45.0	-52.0	18.0	18.0
4	1	0.0	0.0	18.0	18.0	-45.0	-46.0	18.0	18.0
...	...	...	...	...	...	...	...	...	...
2508	1	0.0	0.0	18.0	18.0	-58.0	-60.0	18.0	18.0
2509	0	0.0	0.0	18.0	18.0	-40.0	-41.0	18.0	18.0
2510	0	0.0	0.0	18.0	18.0	-39.0	-40.0	18.0	18.0
2511	1	0.0	0.0	17.0	17.0	-55.0	-68.0	17.0	17.0
2512	1	0.0	0.0	14.0	14.0	-39.0	-50.0	18.0	18.0

2513 rows × 36 columns

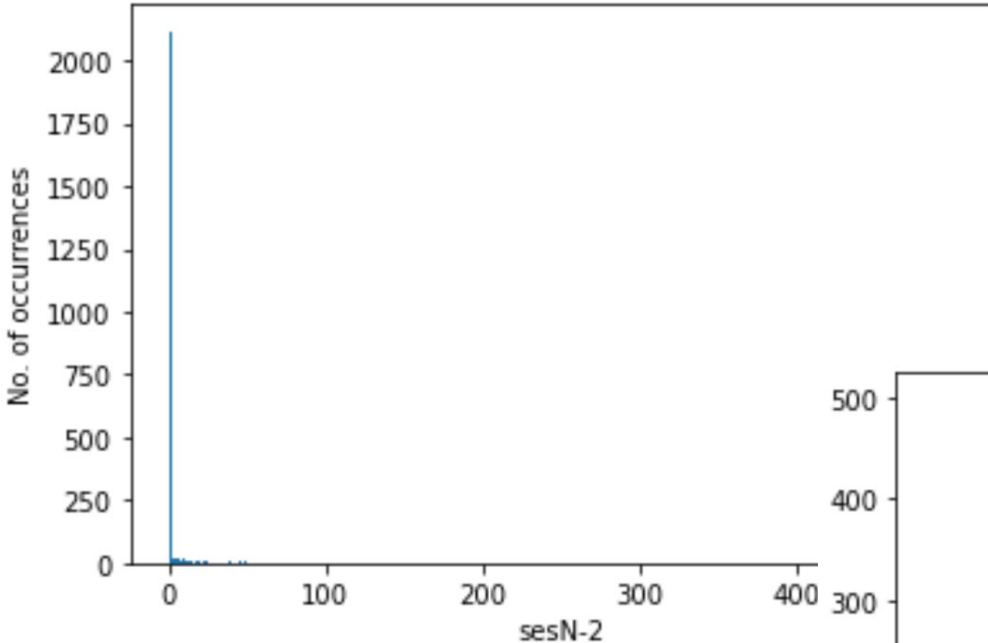
Failure management in microwave networks
Task 1

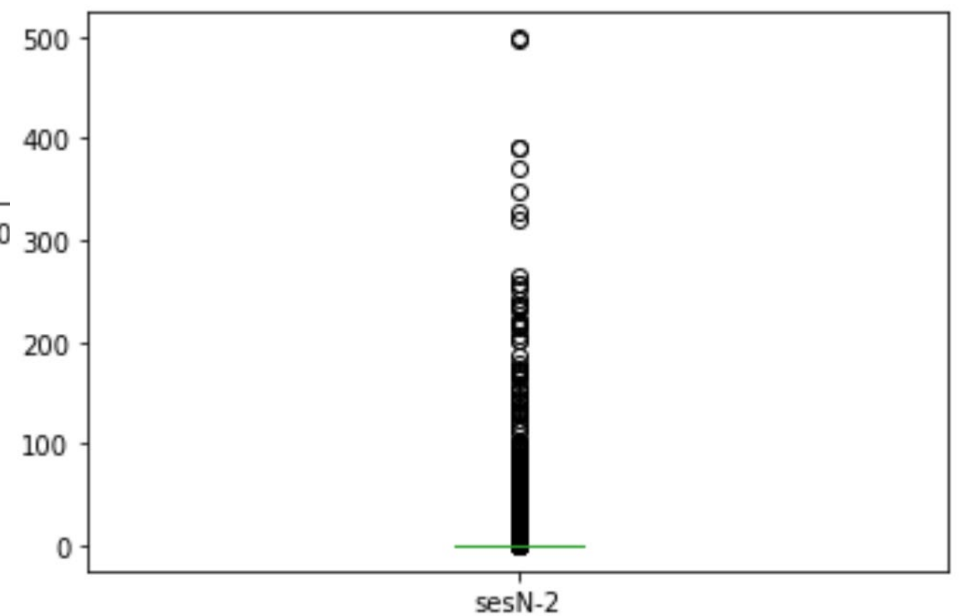
1. Dataset pre-processing (use file “Labelled_Multiclass.csv”):
   b) Define function *plot_feature()* that takes in input dataframe and feature number (passed as integer representing the column index), and plots the distribution of the feature. Distribution should be plotted as histograms and boxplot
      - **Already given in skeleton code**
   c) Call function *plot_feature()* to plot features no. 2 and 4
      - **Already given in skeleton code**

What can we observe?

Failure management in microwave networks
Task 1c): expected outputs (1/2)

```
You are going to see feature 2: sesN-2
```





Failure management in microwave networks

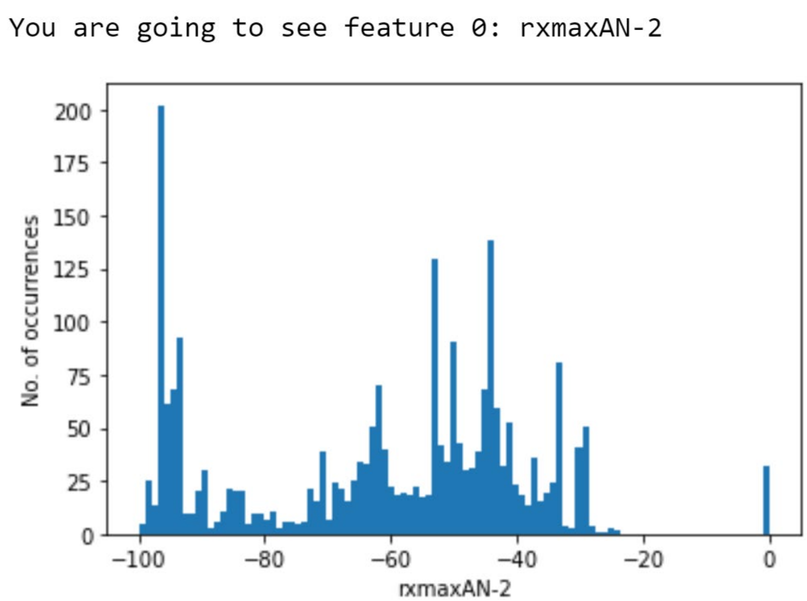
Task 1c): expected outputs (2/2)

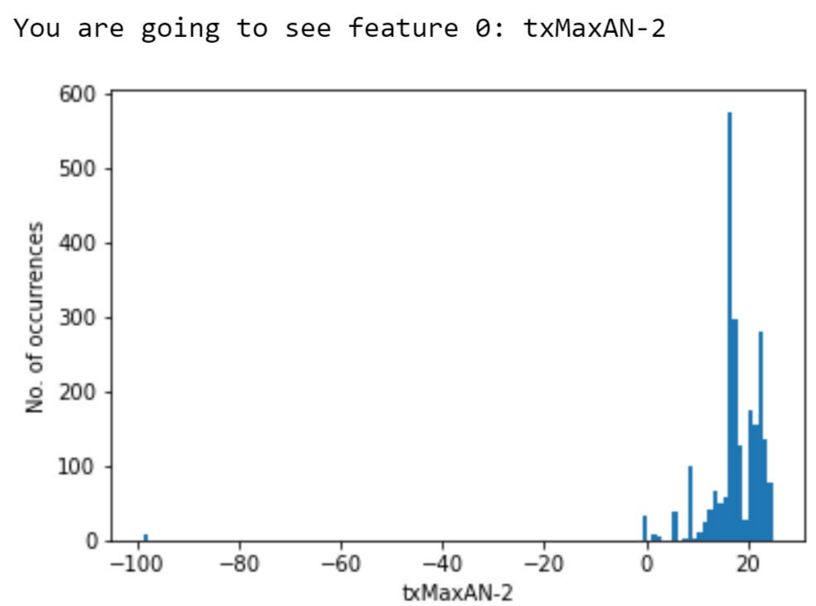
```
You are going to see feature 4: txminAN-2

---------------------------------------------------------------------------
ValueError                                Traceback (mo
C:\Users\FRANCE~1\AppData\Local\Temp/ipykernel_11020/17
      6 plot_feature(data,2)
      7
----> 8 plot_feature(data,4)

C:\Users\FRANCE~1\AppData\Local\Temp/ipykernel_11020/40
urenumber)
     12     maxvalue = feature.max()
     13
---> 14     plt.hist(feature, bins = 1 + int(maxvalue-m
     15     plt.xlabel(dataframe.columns.values[feature
     16     plt.ylabel('No. of occurrences')

ValueError: cannot convert float NaN to integer
```

Failure management in microwave networks
Task 1

1. Dataset pre-processing (use file “Labelled_Multiclass.csv”):
   d) Check which features have missing values. Then, ONLY FOR THOSE FEATURES, use function *plot_feature()* from task 1b) to plot the distribution neglecting missing values.
      - If everything is ok in your code, you should find 24 features with NaN values. Anything in common between these 24 features?

Expected outputs (example):





Failure management in microwave networks
Task 1

1. Dataset pre-processing (use file “Labelled_Multiclass.csv”):
   e) Take features with missing values and substitute NaN with the **median** of the feature without NaN. Print the features and corresponding median values used to replace NaN values
      - Using the median is just a design choice, you can proceed differently

Failure management in microwave networks
Task 1e): expected outputs

Feature, Median value

txMaxAN-2, 18.0
txminAN-2, 18.0
rxmaxAN-2, -55.0
rxminAN-2, -65.0
txMaxBN-2, 18.0
txminBN-2, 18.0
rxmaxBN-2, -53.0
rxminBN-2, -60.0
txMaxAN-1, 18.0
txminAN-1, 18.0
rxmaxAN-1, -58.0
rxminAN-1, -72.0

Feature, Median value

txMaxBN-1, 18.0
txminBN-1, 18.0
rxmaxBN-1, -53.0
rxminBN-1, -66.0
txMaxAN, 18.0
txminAN, 18.0
rxmaxAN, -59.0
rxminAN, -82.0
txMaxBN, 18.0
txminBN, 18.0
rxmaxBN, -55.0
rxminBN, -77.0

Failure management in microwave networks
Task 2

2. Load dataset
   a) Define function *load_dataset()* to load a dataset from a given dataframe in input into given arrays X and y passed in input
      - See skeleton code for the details
   b) Test function *load_dataset()* with datasets created in task 1e)
      - **Already given in skeleton code**

Expected output (task 1f):

```
print(X)
print(y)
print(X.shape)
print(y.shape)
```

```
[[  1.    0.    0. ...   23. -44. -86.]
 [  1.    0.    0. ...   23. -44. -86.]
 [  1.    0.    0. ...   23. -52. -84.]
 ...
 [  0.    0.    0. ...   18. -37. -84.]
 [  1.    0.    0. ...   23. -56. -84.]
 [  1.    0.    0. ...   19. -44. -82.]]
[0. 0. 0. ... 0. 0. 0.]
(2513, 35)
(2513,)
```

Failure management in microwave networks
Task 3

3. ML models selection and training
   a) Split dataset into train/test (80% / 20%) with balanced classes, standardize features and print shapes of train and test sets
      - **Already given in skeleton code**

Expected output:

```
print('Training set shape (features): {}'.format(X_train.shape))
print('Training set shape (labels): {}'.format(y_train.shape))
print('Test set shape (features): {}'.format(X_test.shape))
print('Test set shape (labels): {}'.format(y_test.shape))
```

```
Training set shape (features): (2010, 35)
Training set shape (labels): (2010,)
Test set shape (features): (503, 35)
Test set shape (labels): (503,)
```

Failure management in microwave networks
Task 3

3. ML models optimization and training
   b) Define function *optimize_RF()* that performs hyperparameter optimization with 5-fold crossvalidation, retrains the model on the entire training set with the optimized hyperparameters and returns the trained model
      - **Already given in skeleton code**
   c) Call function *optimize_RF()* and save the trained RF model in a .json file
      - **Already given in skeleton code**

Expected output:

```
Training RF...
100%|████████████████████████████████████████████████| 20/20 [01:05<00:00,  3.26s/trial, best loss: -0.9348258706467663]
{'crit': 1, 'estimators': 1, 'maxd': 8}
{'crit': 'entropy', 'estimators': 50, 'maxd': 17}
best_cv_acc: 0.93
Best number of estimators: 50

Best splitting criterion: entropy

Best max_depth: 17

Crossvalidation accuracy: 0.93

Training duration for RF is 0 s
```

Failure management in microwave networks
Task 3

3. ML models optimization and training
   d) Define function *optimize_KNN()* that performs hyperparameter optimization with 5-fold crossvalidation, retrains the model on the entire training set with the optimized hyperparameters and returns the trained model
      - See skeleton code for the details
   e) Call function *optimize_KNN()* and save the trained KNN model in a .json file
      - **Already given in skeleton code**

Expected output:

```
Training KNN...
100%|████████████████████████████████████████████████| 10/10 [00:03<00:00,  2.98trial/s, best loss: -0.8975124378109453]
{'neighb': 0, 'wgts': 1}
{'neighb': 1, 'wgts': 'distance'}
best_cv_acc: 0.9
Crossvalidation duration for KNN is 3 s

Best number of neighbors: 1

Best weight function: distance

Crossvalidation accuracy: 0.9

Training duration for KNN is 0 s
```

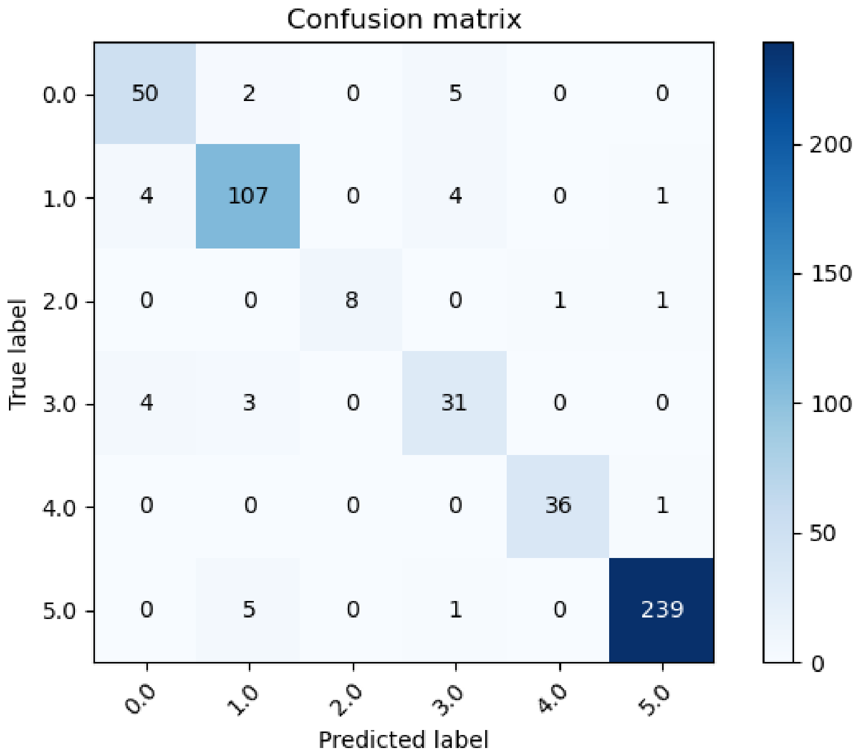
Failure management in microwave networks
Task 4

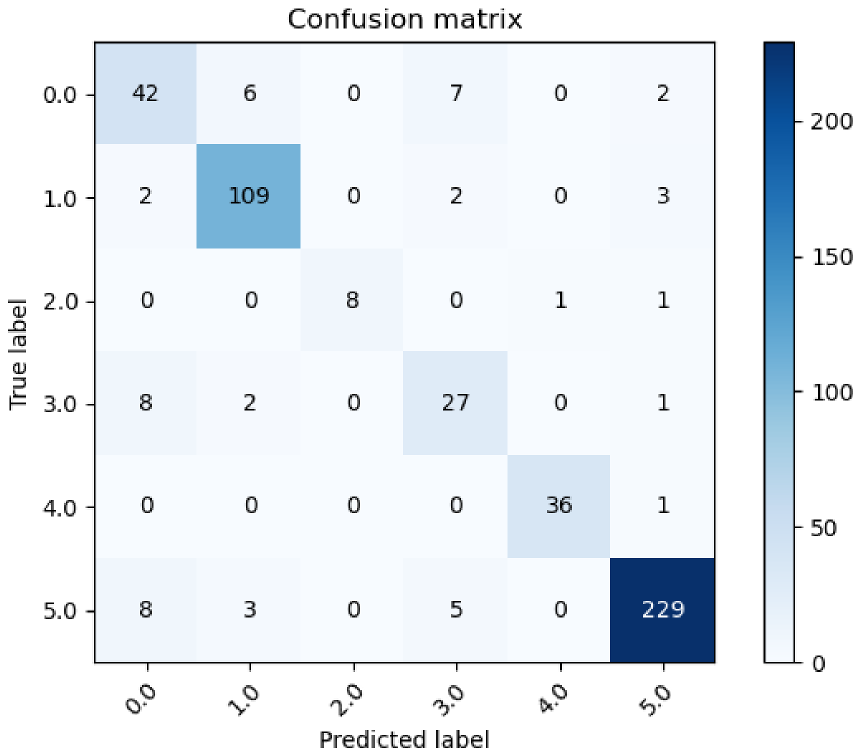
4. Test and performance evaluation
   a) Define function *performance_eval()* that takes in input ground truth and predicted labels, prints results in output, and returns PER-CLASS metrics
      - See details in the skeleton code
      - **N.B. Metrics should be calculated "manually" (applying the specific formulae) AND with sklearn APIs**
   b) Load RF and KNN models saved in .json files in tasks 3c) and 3e) and store into new model objects. Then, perform predictions for the test set and call function *performance_eval()* to evaluate performance of the RF and KNN models
      - **Already given in skeleton code**

Failure management in microwave networks
Task 4a-b): expected outputs

Evaluating RF performance.....
Accuracy: 0.9363817097415507
Manual accuracy: 0.9363817097415507
...

Evaluating KNN performance.....
Accuracy: 0.8966202783300199
Manual accuracy: 0.8966202783300199
...

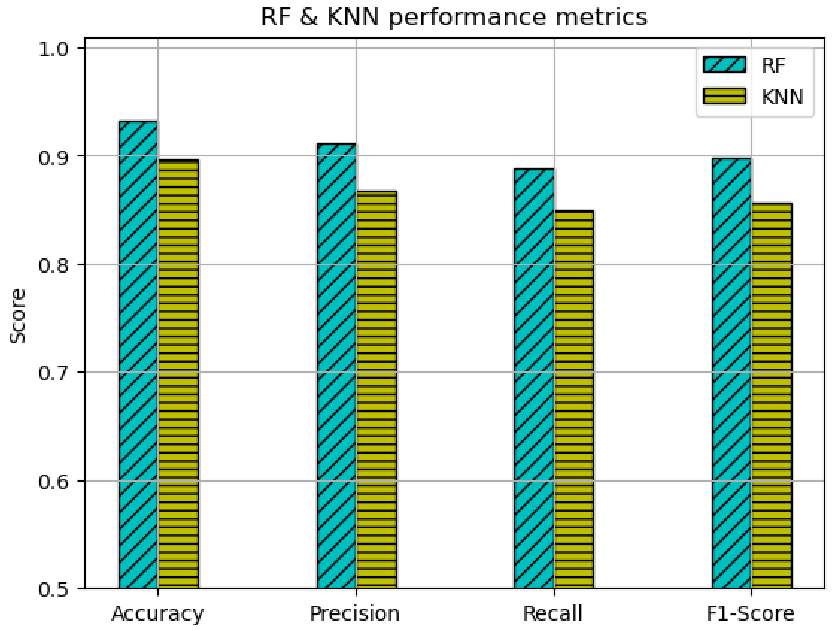




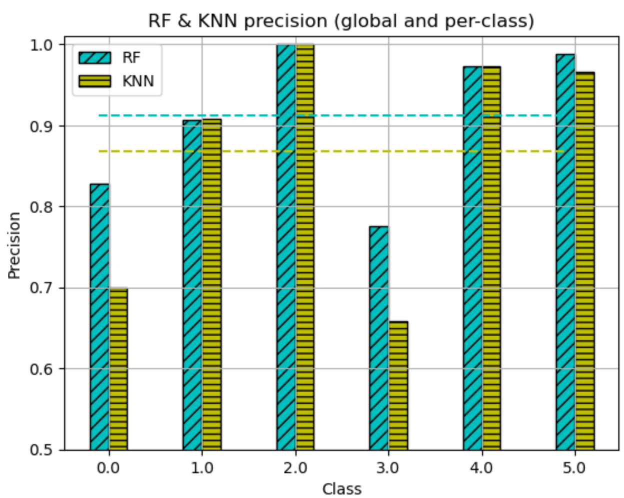
Failure management in microwave networks

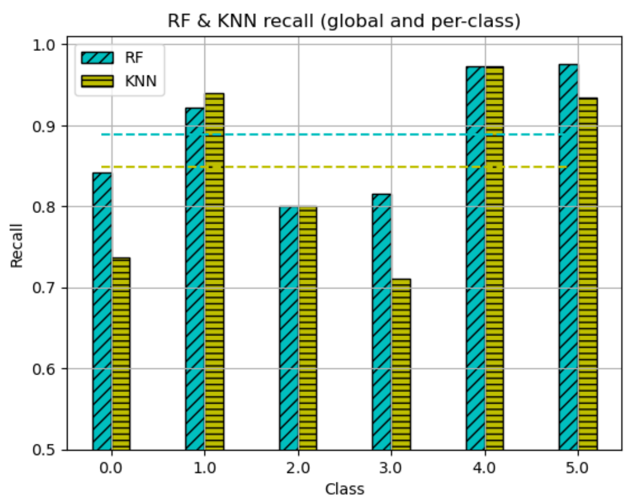
Task 4

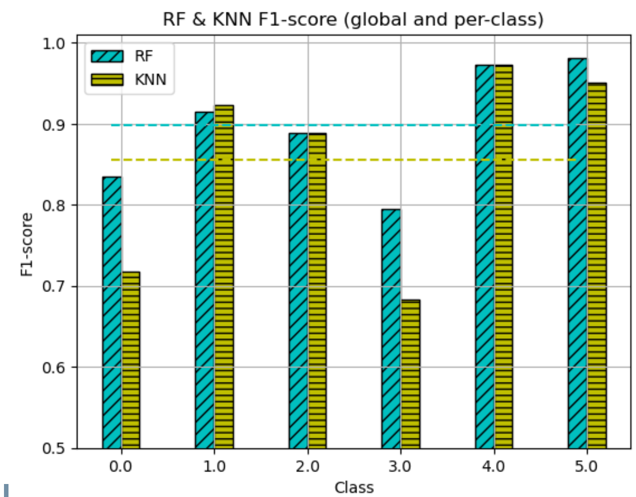
4. Test and performance evaluation
    c) For RF and KNN, compute global precision, recall and F1-score as mean of per-class metrics and plot in 4 separate graphs
        - **Already given in skeleton code**



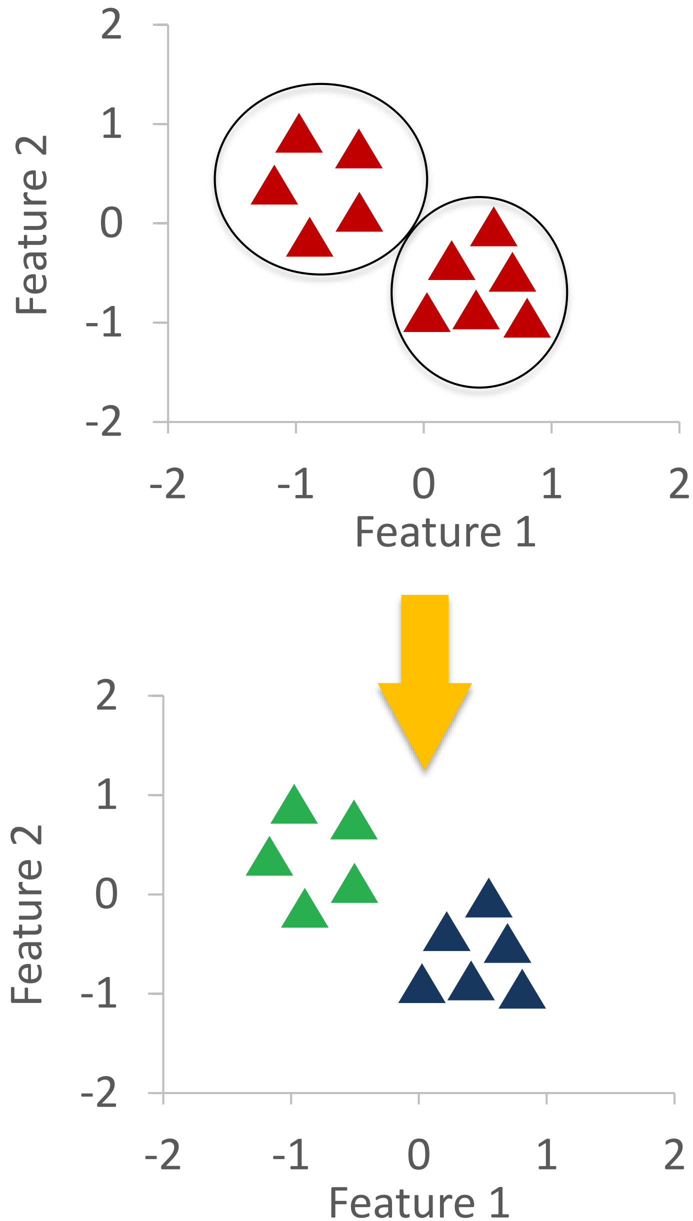
Expected outputs:







- Part II: clustering



Check out more on clustering algorithms and scikit learn APIs:
https://scikit-learn.org/stable/modules/clustering.html

Failure management in microwave networks
Task 5

- Let’s consider only hardware failure class (label = 5)
- We suspect that there can be multiple sub-classes within this group (e.g., HW failures hitting different equipment)
- How can we address this issue?

5. Analyse **hardware failure data** …
   a) Starting from *(X,y)* above, generate new "truncated" dataset by removing all labels different from HW failures (*label* = 5)

Expected output:

```
print(X_truncated.shape)
print(y_truncated.shape)

```

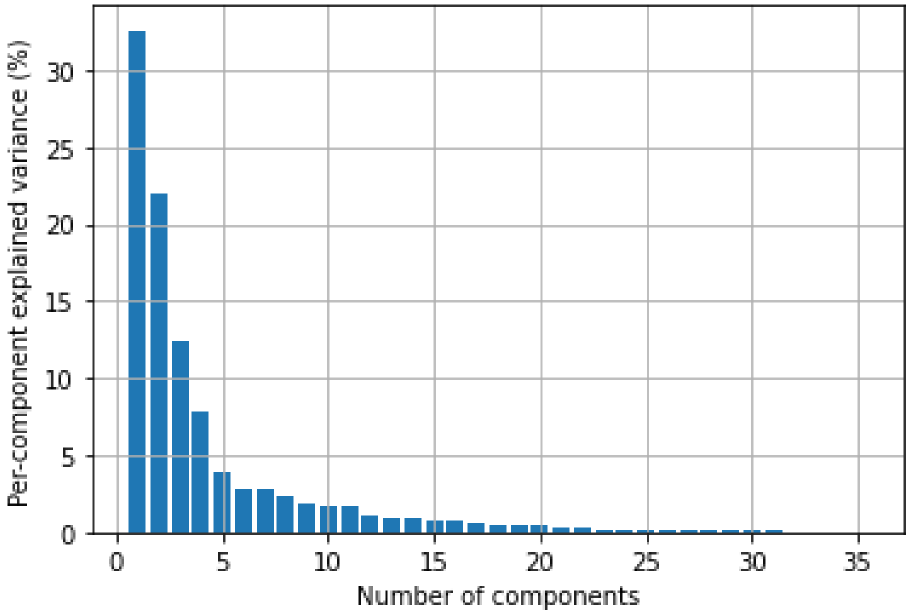
```
(1222, 35)
(1222,)
```

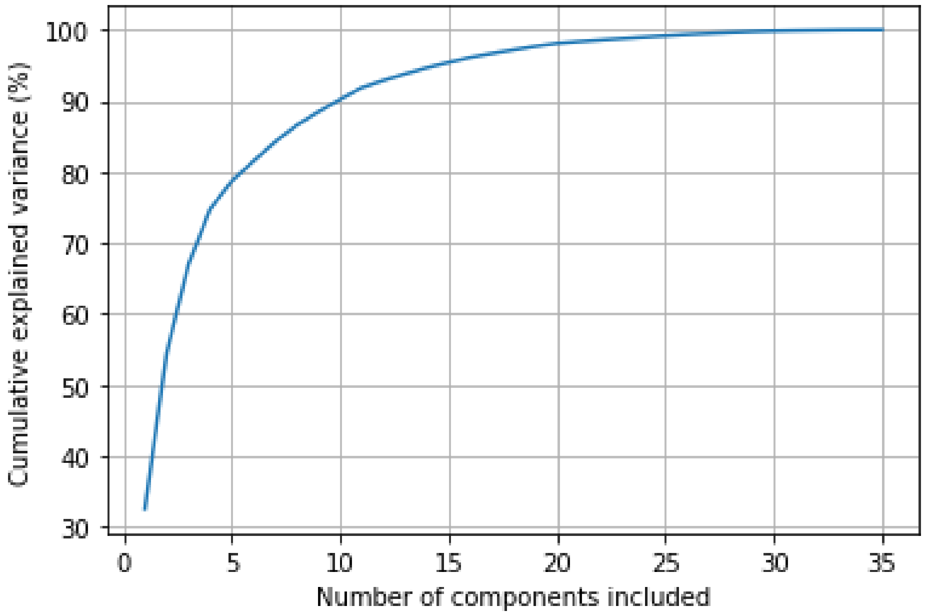
Failure management in microwave networks
Task 5

5. Analyse **hardware failure data** **using Principal Component Analysis (PCA)**
   b) Instantiate and fit a PCA object on the truncated dataset *X_truncated*, then plot the explained variance for all components separately and cumulatively (i.e., adding one component at a time in the dataset)
      - **Already given in skeleton code**

Failure management in microwave networks
Task 5b): expected outputs

Variance represented by each component, %:
[32.6 21.9 12.4 7.9 3.9 2.8 2.7 2.3 1.9 1.7 1.7 1.   0.9 0.9
 0.7 0.7 0.6 0.5 0.5 0.4 0.3 0.2 0.2 0.2 0.2 0.2 0.1 0.1
 0.1 0.1 0.1 0.   0.   0.   0.]

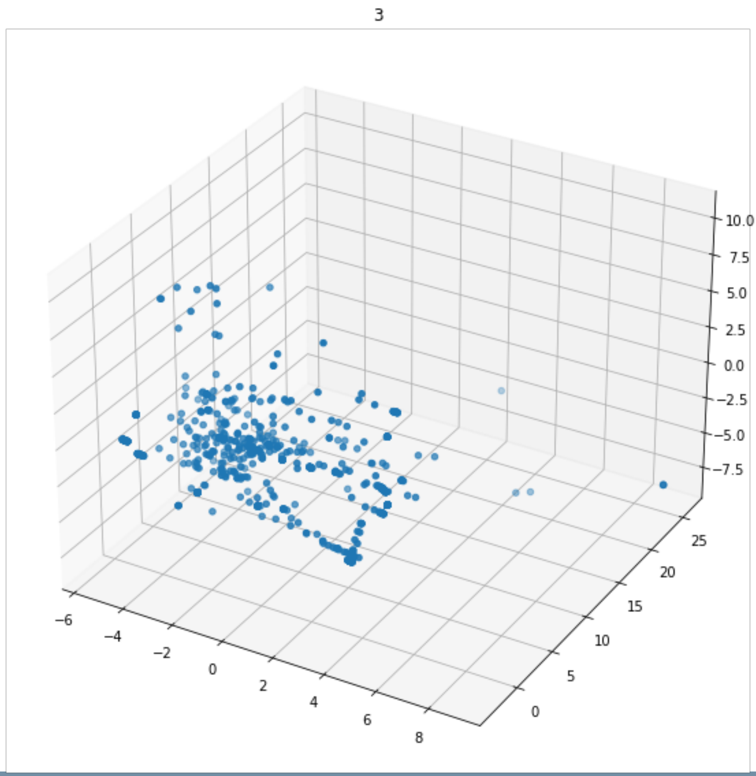




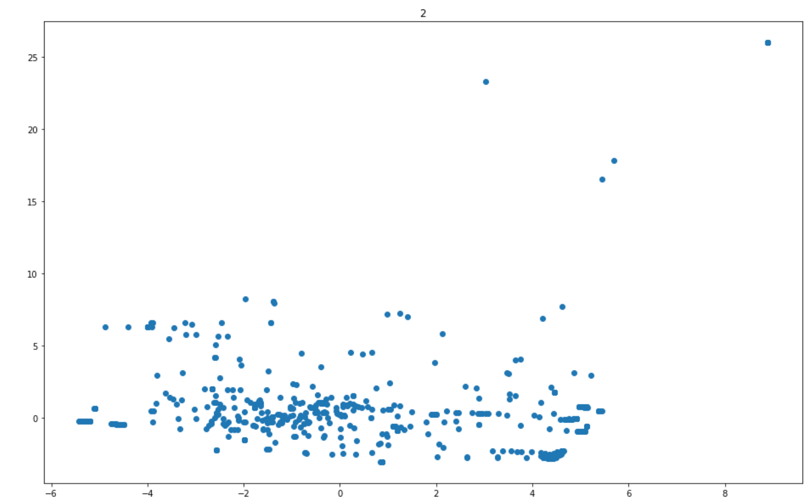
We can try to guess the number of sub-clusters (sub-classes of HW failures representing different causes, all in the HW failure category) by observing these graphs

Failure management in microwave networks
Task 5

5. Analyse **hardware failure data** using Principal Component Analysis (PCA)
    c) Consider the cases of 2 and 3 PCA components to visualize the (transformed) dataset via a scatterplot of the (transformed) features in 2D and 3D graphs
        - **Already given in skeleton code**



Expected outputs:



Failure management in microwave networks
Task 5

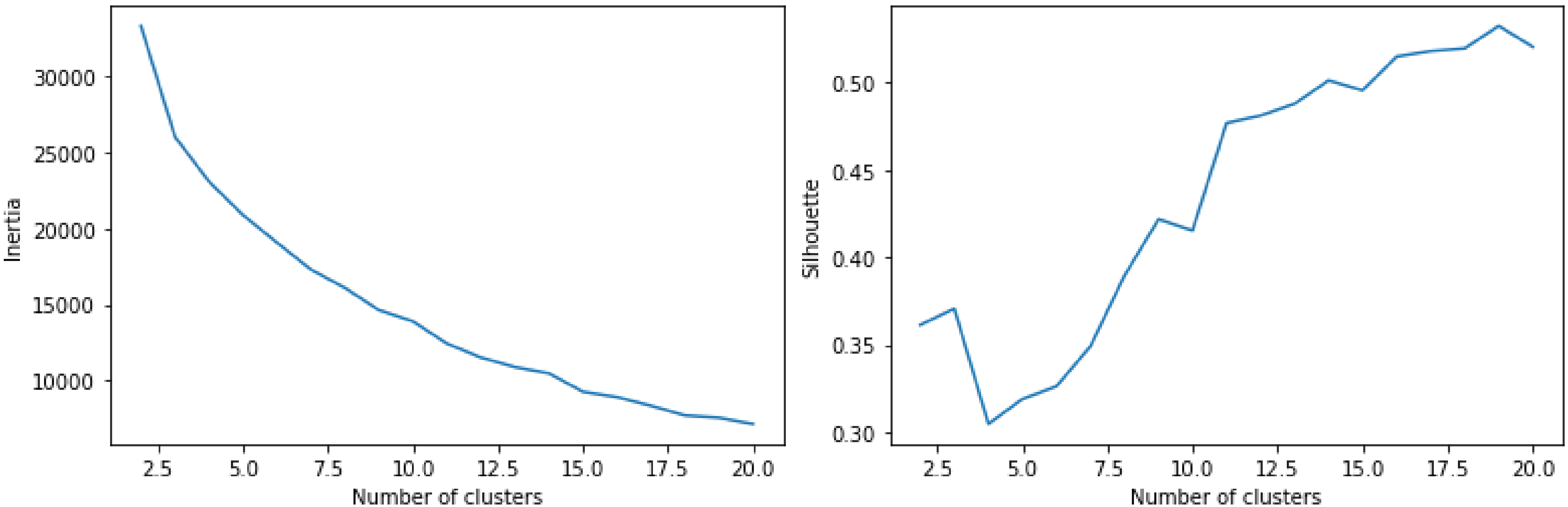
5. Analyse **hardware failure data** using Principal Component Analysis (PCA)
   d) Perform clustering on dataset *X_truncated* with number of clusters *k* in range(2, 21) and considering ***kmeans* algorithm**. Plot ***inertia*** and ***silhouette**** for each value of k
      - **Already given in skeleton code**

**** N.B:***

- ***Inertia*** ≥ 0: measures how much clusters are sparse (the lower the better, BUT monotonically decreases with number of clusters!)
- ***Silhouette*** $\in [0,1]$: measures how much each point is close to points in the same cluster and far from points in the nearest cluster (the higher the better)

Failure management in microwave networks

Task 5d): expected outputs



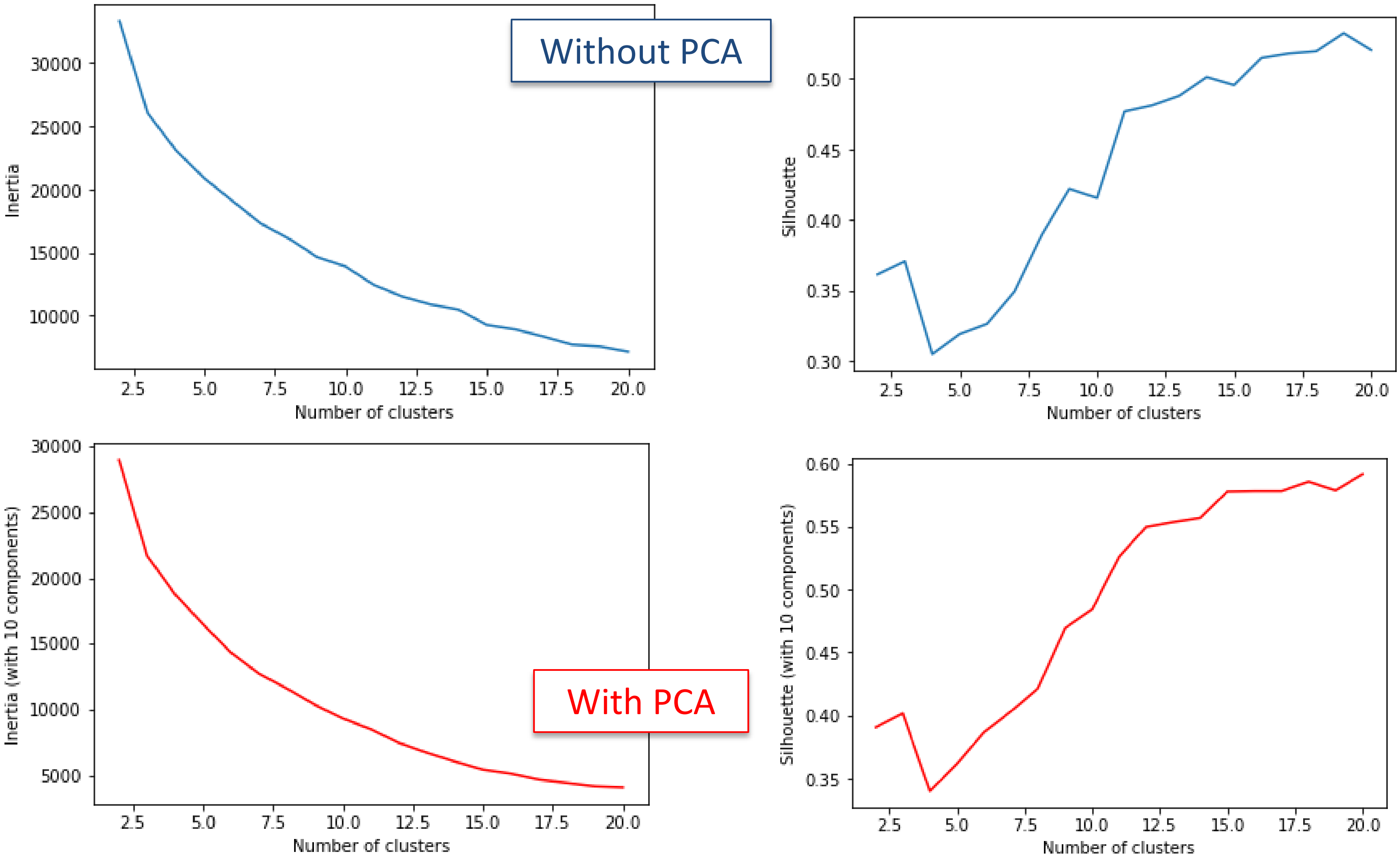
Failure management in microwave networks
Task 5

5. Analyse **hardware failure data** using Principal Component Analysis (PCA)
   e) Repeat task 5d) considering the PCA-modified dataset with a number of components that retains at least 90% variance of the original dataset *X_truncated*

   What can we observe?

Failure management in microwave networks

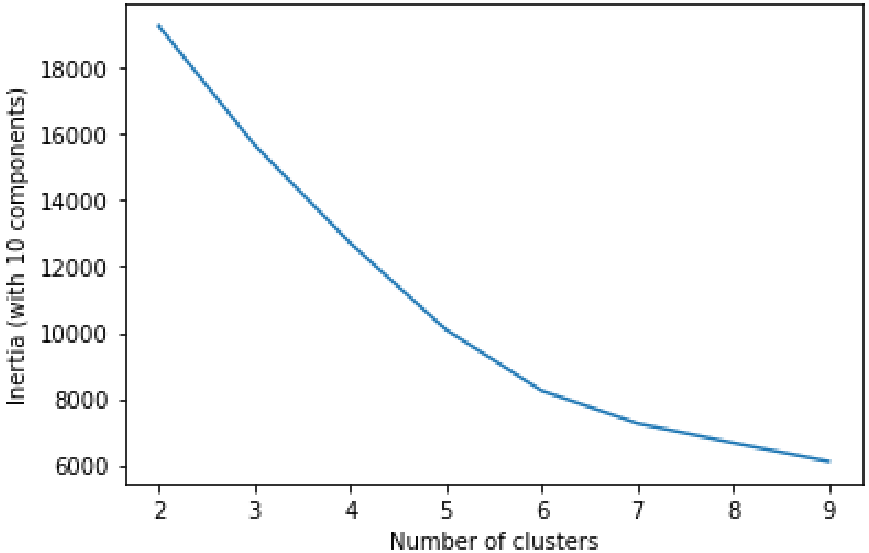
Task 5e): expected outputs

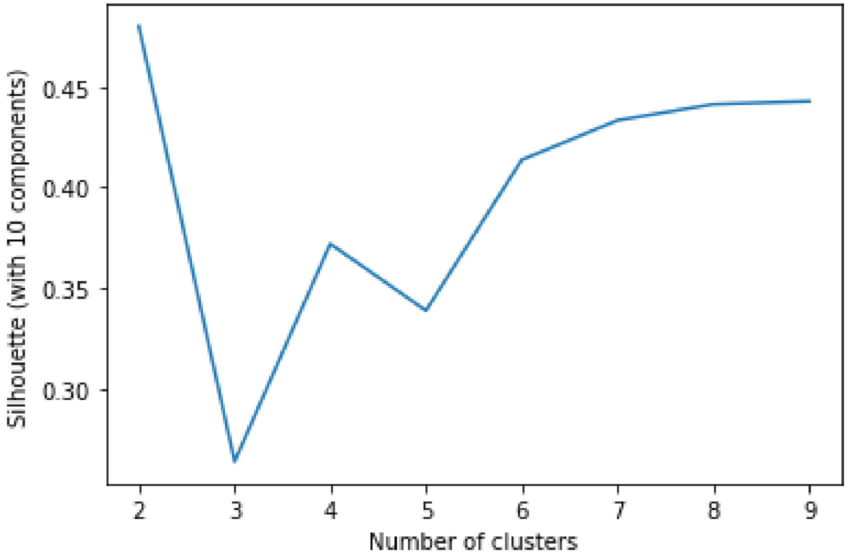


Failure management in microwave networks
Task 6

6. Analyse **extra-attenuation** and **self-interference** failures
   a) Generate a new "truncated" dataset taking only labels 1=Extra-attenuation and 4=Self-interference from the original dataset. Then, perform *kmeans* clustering with the new dataset and considering PCA with 10 components and k in range(2, 10)
      - **Already given in skeleton code**

Expected output:





Failure management in microwave networks
Task 6

6. Analyse **extra-attenuation** and **self-interference** failures
   b) Perform *kmeans* clustering of the truncated dataset of task 6a, considering k=2 and 10 PCA components. Then assign labels to data points and compare this partition with the ground truth (labels 1,4 in *y_truncated*) in terms of *rand_score, homogeneity_score* and *completeness_score*
      - **Already given in skeleton code**

N.B.

- **Rand index** $\in [\mathbf{0}, \mathbf{1}]$: measures the similarity of two partitions
- **Homogeneity** $\in [\mathbf{0}, \mathbf{1}]$: measures how much each cluster contains only members of a single class (the higher, the better)
- **Completeness** $\in [\mathbf{0}, \mathbf{1}]$: measures how much all members of a given class are assigned to the same cluster (the higher, the better)

Expected output:

```
Rand_score (0; 1): 0.69
Homogeneity (0; 1): 0.13
Completeness (0; 1): 0.2
```