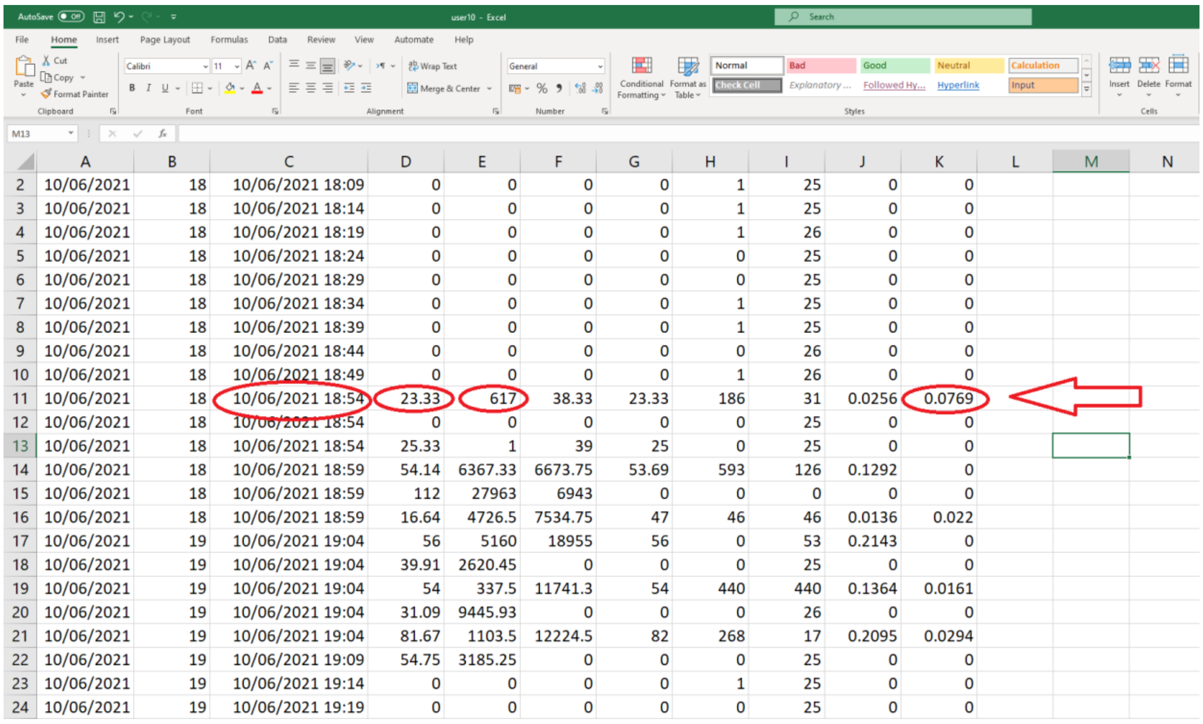
Automated QoE Prediction with AI/ML– Solutions

Downloading the Github Repository and Dataset

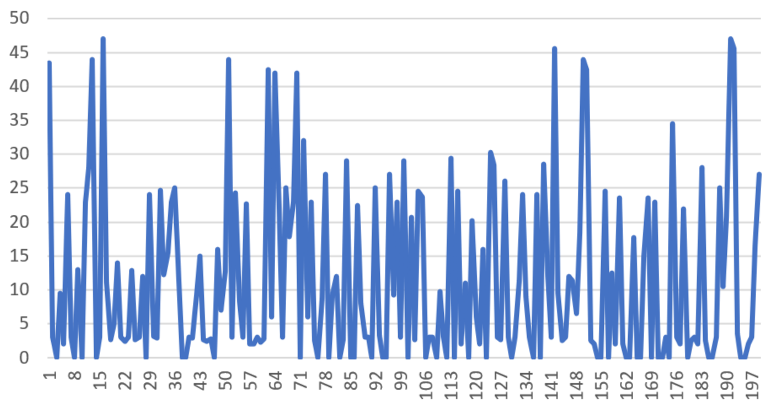
4- 23.33 - Milliseconds (ms) - 617 ms - 0.0769 ~ 8%



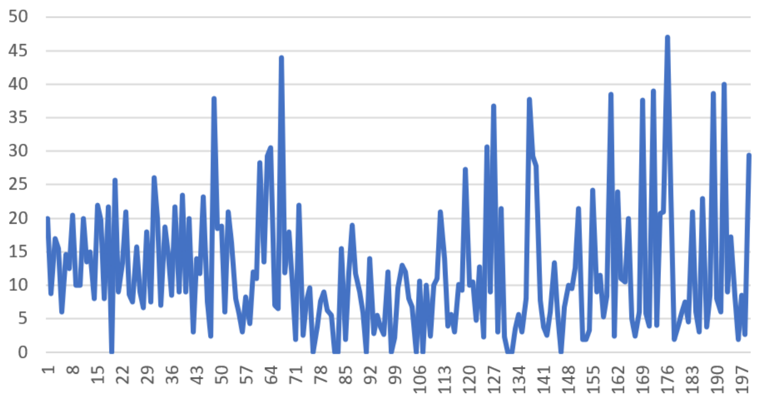
	A	B	C	D	E	F	G	H	I	J	K	L	M	N
2	10/06/2021	18	10/06/2021 18:09	0	0	0	0	1	25	0	0			
3	10/06/2021	18	10/06/2021 18:14	0	0	0	0	1	25	0	0			
4	10/06/2021	18	10/06/2021 18:19	0	0	0	0	1	26	0	0			
5	10/06/2021	18	10/06/2021 18:24	0	0	0	0	0	25	0	0			
6	10/06/2021	18	10/06/2021 18:29	0	0	0	0	0	25	0	0			
7	10/06/2021	18	10/06/2021 18:34	0	0	0	0	1	25	0	0			
8	10/06/2021	18	10/06/2021 18:39	0	0	0	0	1	25	0	0			
9	10/06/2021	18	10/06/2021 18:44	0	0	0	0	0	26	0	0			
10	10/06/2021	18	10/06/2021 18:49	0	0	0	0	1	26	0	0			
11	10/06/2021	18	10/06/2021 18:54	23.33	617	38.33	23.33	186	31	0.0256	0.0769			
12	10/06/2021	18	10/06/2021 18:54	0	0	0	0	0	25	0	0			
13	10/06/2021	18	10/06/2021 18:54	25.33	1	39	25	0	25	0	0			
14	10/06/2021	18	10/06/2021 18:59	54.14	6367.33	6673.75	53.69	593	126	0.1292	0			
15	10/06/2021	18	10/06/2021 18:59	112	27963	6943	0	0	0	0	0			
16	10/06/2021	18	10/06/2021 18:59	16.64	4726.5	7534.75	47	46	46	0.0136	0.022			
17	10/06/2021	19	10/06/2021 19:04	56	5160	18955	56	0	53	0.2143	0			
18	10/06/2021	19	10/06/2021 19:04	39.91	2620.45	0	0	0	25	0	0			
19	10/06/2021	19	10/06/2021 19:04	54	337.5	11741.3	54	440	440	0.1364	0.0161			
20	10/06/2021	19	10/06/2021 19:04	31.09	9445.93	0	0	0	26	0	0			
21	10/06/2021	19	10/06/2021 19:04	81.67	1103.5	12224.5	82	268	17	0.2095	0.0294			
22	10/06/2021	19	10/06/2021 19:09	54.75	3185.25	0	0	0	25	0	0			
23	10/06/2021	19	10/06/2021 19:14	0	0	0	0	1	25	0	0			
24	10/06/2021	19	10/06/2021 19:19	0	0	0	0	0	25	0	0			

5- It may not be easy to predict the QoE of users by a simple look at the indicators. Below are sample plots of Indicators for UBE vs. UGE in the training dataset. Some periodicity can be seen, e.g. similar indicator values at similar times of the day.

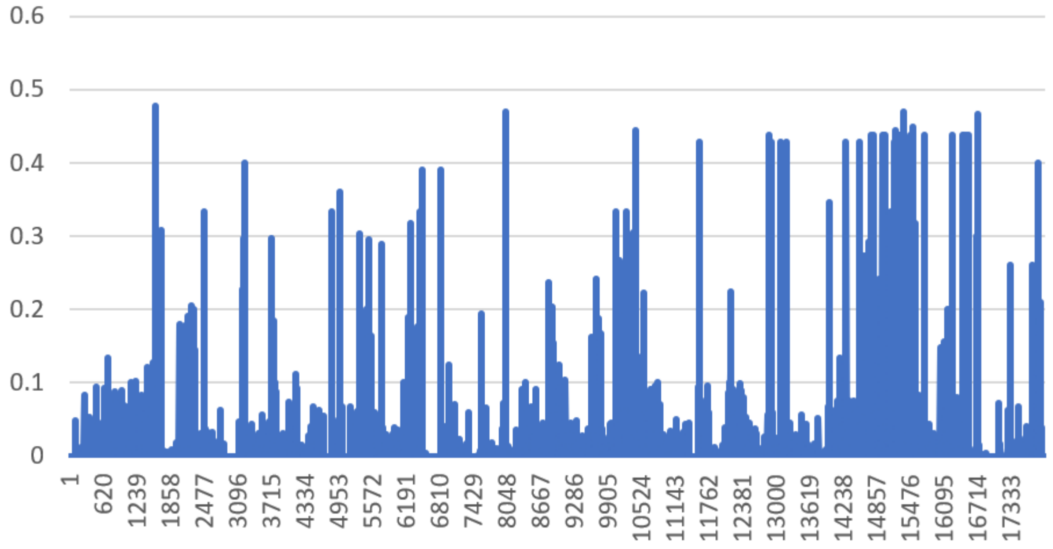
Indicator 1 UBE:



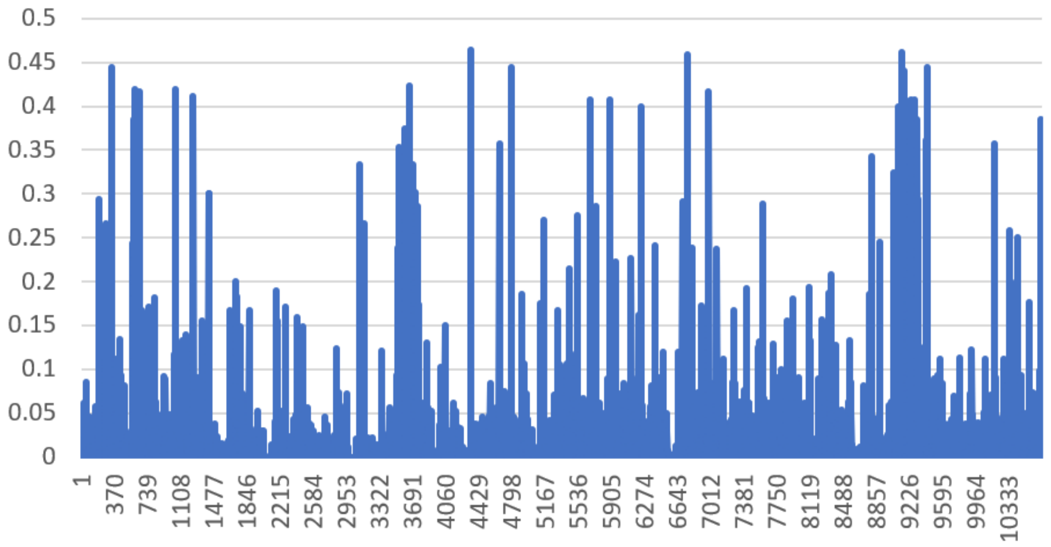
Indicator 1 UGE:



Indicator 8 UBE:



Indicator 8 UGE:



The Classifier Training Pipeline

1-The numerical label value is 0 for UBE, and 1 for UGE. This conversion is carried out via:

```
y.append(1 if("UGE" in _f_path) else 0)
```

2a-This processing is carried out via the following piece of code. It is only applied on indicators 1, 2, 4, 5, and 6. We do expect the packet retransmission probability to be absolute "0" if the network condition is good.

```
# data cleaning (number of dimension at output: 8)
    data_clean  = data_raw[(data_raw["indicator1"] > 0) &
(data_raw["indicator2"] > 0) &
                           (data_raw["indicator4"] > 0) &
(data_raw["indicator5"] > 0) &
                           (data_raw["indicator6"] > 0)] # remove data
if either indicator 1/2/4/5/6 is zero
```

2b-The outlier threshold values are:

```
# definition of outlier ([outlier of indicator1, 2 , ... , 8])
OUTLIER = [48.75, 101.5, 19.1, 50.5, 128, 47.5, 0.184629, 0.212225]
```

And the code to carry out this processing is:

```
for _j in range(8):
        # replace data exceding outlier for the outlier value
        data_clean.loc[data_clean["indicator" + str(_j+1)] >
OUTLIER[_j], "indicator" + str(_j+1)] = OUTLIER[_j]
```

2c-The day is split into 3 time ranges: 0am–7am, 7am–7pm, and 7pm–0am; and there are 2 date ranges: Weekend and midweek. Below is the code used to carry out the splitting.

```
# definition of temporal sample splitting
    list_time_range = [] # definition of time range (T1 ~ T3)
    list_time_range.append((data_group["hour"] >= 0)     &
(data_group["hour"] < 7))
    list_time_range.append((data_group["hour"] >= 7)     &
(data_group["hour"] < 19))
    list_time_range.append((data_group["hour"] >= 19)    &
(data_group["hour"] < 24))
    list_date_range = [] # definition of date range (D1 ~ D2)
    list_date_range.append(data_clean["day"].apply(lambda x:
check_holiday(x)) == True)
    list_date_range.append(data_clean["day"].apply(lambda x:
check_holiday(x)) == False)
```

2d-Dimension of the feature space is 36. In other words, the 8 time series are reduced to 36 features for each user.

3-The classifier used here is “Random Forest” and the code to carry out the training, i.e. model fitting, is as follows:

```
# Random Forest model training
clf = RandomForestClassifier(max_depth=10, n_estimators=5,
max_features='log2', random_state=71666)
clf.fit(X_train, y_train)
```

QoE Prediction and Performance Evaluation

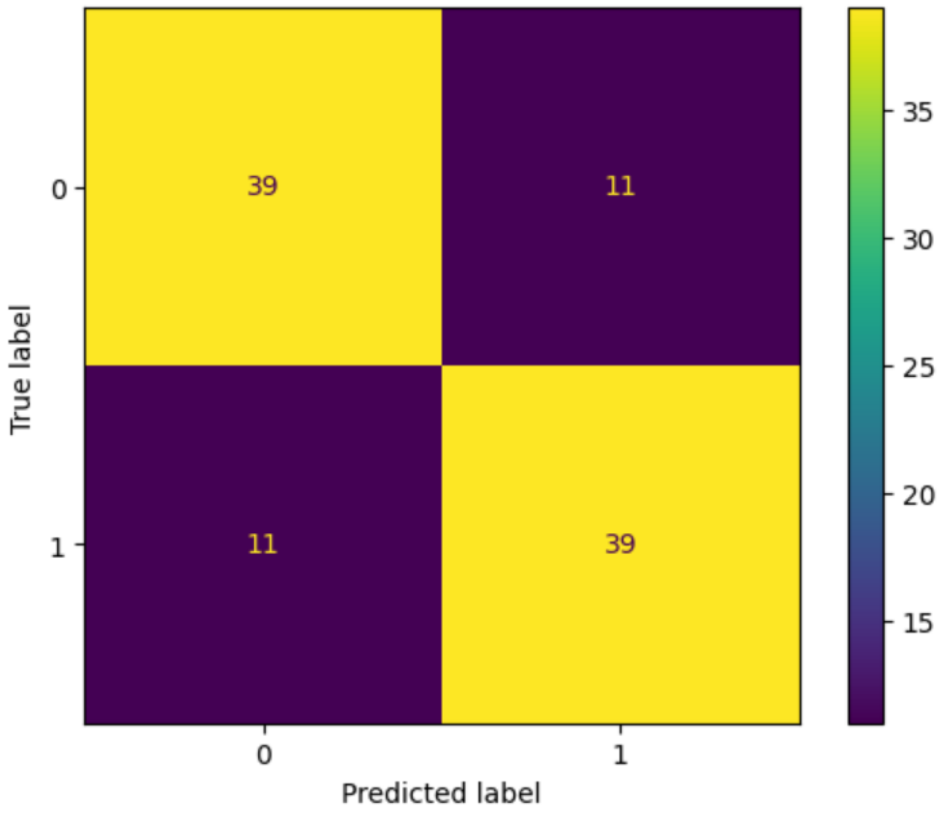
1-Based on the following output generated by the code, the probability of correct prediction of the QoE is 78% which is in agreement with the results in [3]:

```
precision    recall  f1-score   support

           0       0.78      0.78      0.78        50
           1       0.78      0.78      0.78        50

    accuracy                           0.78       100
   macro avg       0.78      0.78      0.78       100
weighted avg       0.78      0.78      0.78       100
```

2-Based on the following confusion matrix generated by the code, the probability that the trained classifier wrongly predicts a UBE to be a UGE is 22%. Similarly, the probability that the trained classifier wrongly predicts a UGE to be a UBE is 22%.



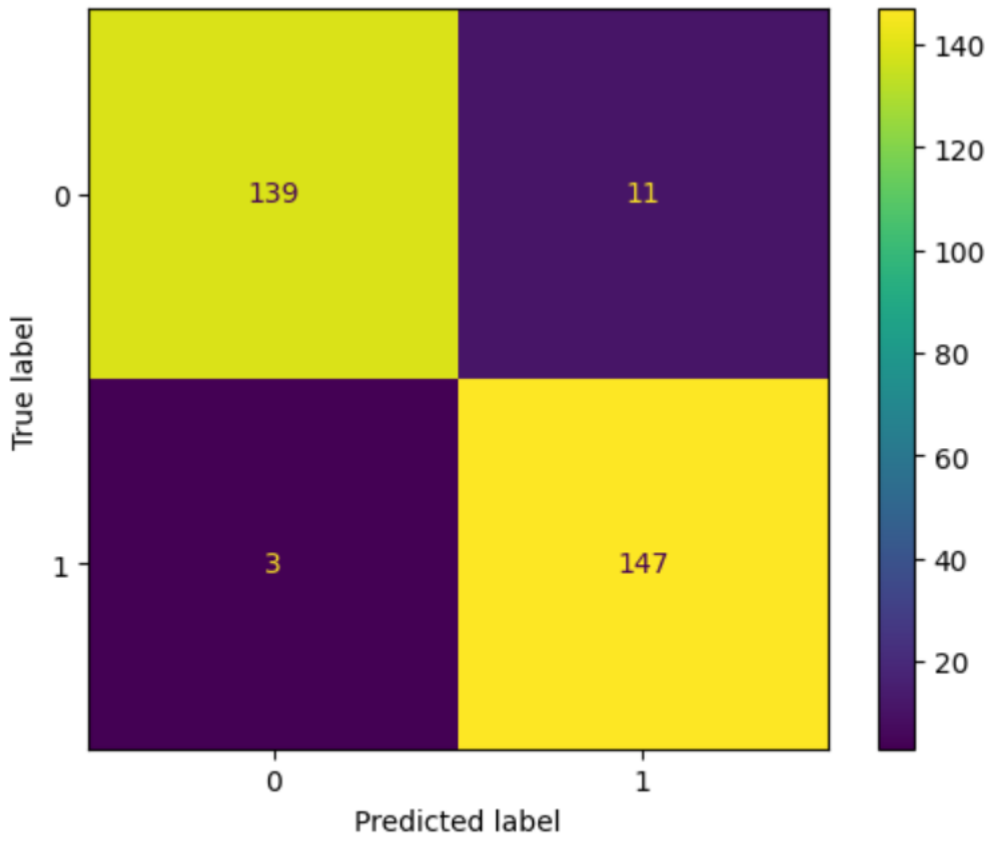
3a- Based on the following output generated by the code, the probability of correct prediction of the QoE on the training data is 95%, which is larger than 78% achieved on the test data. This is expected because evaluating the classifier on the same data it has been trained on is considered “unfair”. In other words, the classifier should generalize well beyond the training dataset and the accuracy we can expect on the unseen data is 78% not 95%.

```
precision     recall   f1-score    support

           0       0.98      0.93      0.95       150
           1       0.93      0.98      0.95       150

    accuracy                           0.95       300
   macro avg       0.95      0.95      0.95       300
weighted avg       0.95      0.95      0.95       300
```

3b-Based on the following confusion matrix generated by the code, the probability that the trained classifier wrongly predicts a UBE to be a UGE is ~7%. The probability that the trained classifier wrongly predicts a UGE to be a UBE is 2%. Based on this, it seems that it is harder to predict a UBE!


True label
Predicted label