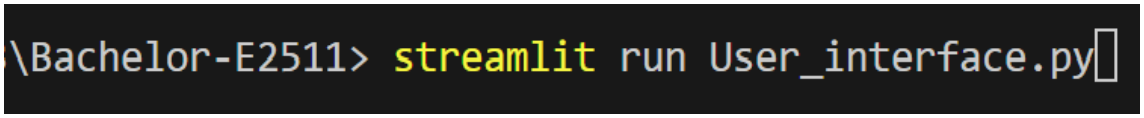
E2511 Bachelor thesis - User interface manual

20th May 2025

Table of Contents

1 How to use and how it works

In this guide it is provided you that you have cloned the repository of the project and no changes are made to the names of the folders or whatnot. To start the user interface run the User_interface.py file with this command in the terminal:
`streamlit run User_interface.py`



This should open a page on your Web browser running on `localhost:8051`.

1.1 Settings

The ML model tab should look like this:

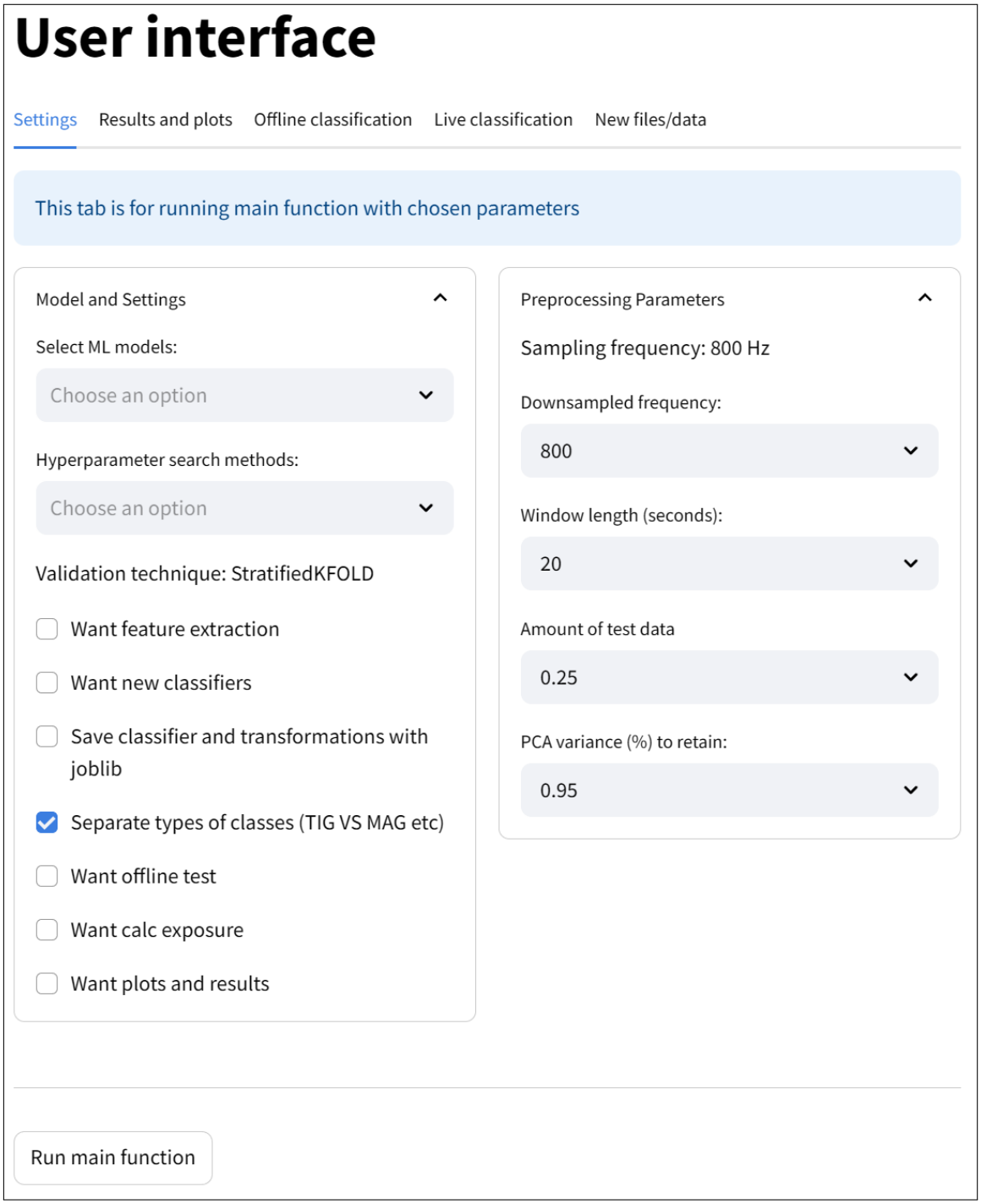


Figure 1: Settings tab

This is the tab with most parameters that can be changed, and perhaps where most confusion of how to use the program occurs. Before continuing, know that only clicking on the settings won't do anything before you click run main function at the bottom of the tab.

Want feature extraction:
Enable this option and click "Run main function" when you want to extract features from collected

data. Where the data are located can be seen in section 1.5. This is needed when you upload new files in the "New files/data" tab, which is explained in its own section. When main function is run with this option extracted features are saved in a csv file which are found here:

```
\Bachelor-E2511\OutputFiles\Separated\800_feature_df.csv
\Bachelor-E2511\OutputFiles\Combined\800_feature_df.csv
```

The first directory is where the file ends up if "Separate types of classes TIG VS MAG etc" is selected. The second directory is where it ends up if it is not. If you have done feature extraction once, you dont need to do it more unless you collect more data. Note that the data we have are sampled with 800 Hz, for other sampling rates, one might have to alter the program slightly.

Want new classifier:
If you want to train a new classifier, perhaps after more data collection, enable this option. You will need to tell the program what models you want to train and what hyperparameter search method to use. You can do this under "Select ML models:" and "Hyperparameter search methods:" (1). You can choose multiple models and search methods. The program will compare the classifiers and choose the best one in regards to F1 score when plotting. If you want info of lower performing classifiers you should use the main program and look in the terminal. The highest performing classifier will be temporarily saved to:

```
\Bachelor-E2511\CLF results\clf_results.joblib
```

I say temporarily because this classifier will always be rewritten when "Want new classifiers" is enabled (meaning it trains a new classifier). If "Want new classifiers" is diabled it will use the "main" classifier used for live and offline predictions, it is located here:

```
\Bachelor-E2511\OutputFiles\Separated\classifier.joblib
\Bachelor-E2511\OutputFiles\Separated\PCA.joblib
\Bachelor-E2511\OutputFiles\Separated\scaler.joblib
```

These are the same classifier and transformations used in the real time program. Therefore from here on out in the document this classifier, will be referred to as the `current classifier`. This way, you can see results from the `current classifier` or the newly trained classifier depending on if "Want new classifiers" is enabled.

Save classifier and transformations with joblib:
If you draw the conclusion that the new classifier is better than the `current classifier`, enable this option. This will save/overwrite the classifier and transformations to:

```
\Bachelor-E2511\OutputFiles\Separated\classifier.joblib
\Bachelor-E2511\OutputFiles\Separated\PCA.joblib
\Bachelor-E2511\OutputFiles\Separated\scaler.joblib
```

Which will become the `current classifier` and transformations used in real time program 1.4.

Separate types of classes (TIG VS MAG etc):
Enable this option if you want to train a classifier on granular lables we made for our project with our data. Like distinguishing TIG vs Mag welding, small angle grinder, big angle grinder etc. If you are only interested in the activity itself like welding, grinding etc, disable this option.

Want offline test:
If you want to test the `current classifier` on a new data file, for example, to test it on an inexperienced worker to check generalizability or whatever else. Enable this option. You can upload the test file to use using the "Test classifier on a file tab" which is explained Section 1.3. Remember to run main function after uploading the file to get updated results.

Want calc exposure:
This option is thought to be a way to link the activities/labels with exposure to vibrations, neurotoxins and more. However this link is missing in research, for now placeholder values are used to

calculate the exposure. It is therefore not an interesting option as of now. The results, nevertheless, will also be displayed in the "Test classifier on a file tab".

Want plots and results:
Enable this option if you want plots and results of the selected classifier. The results and plots will show in the "Results" tab.

Remember, if "Want new classifiers" is enabled it will use the most recently trained classifier in:

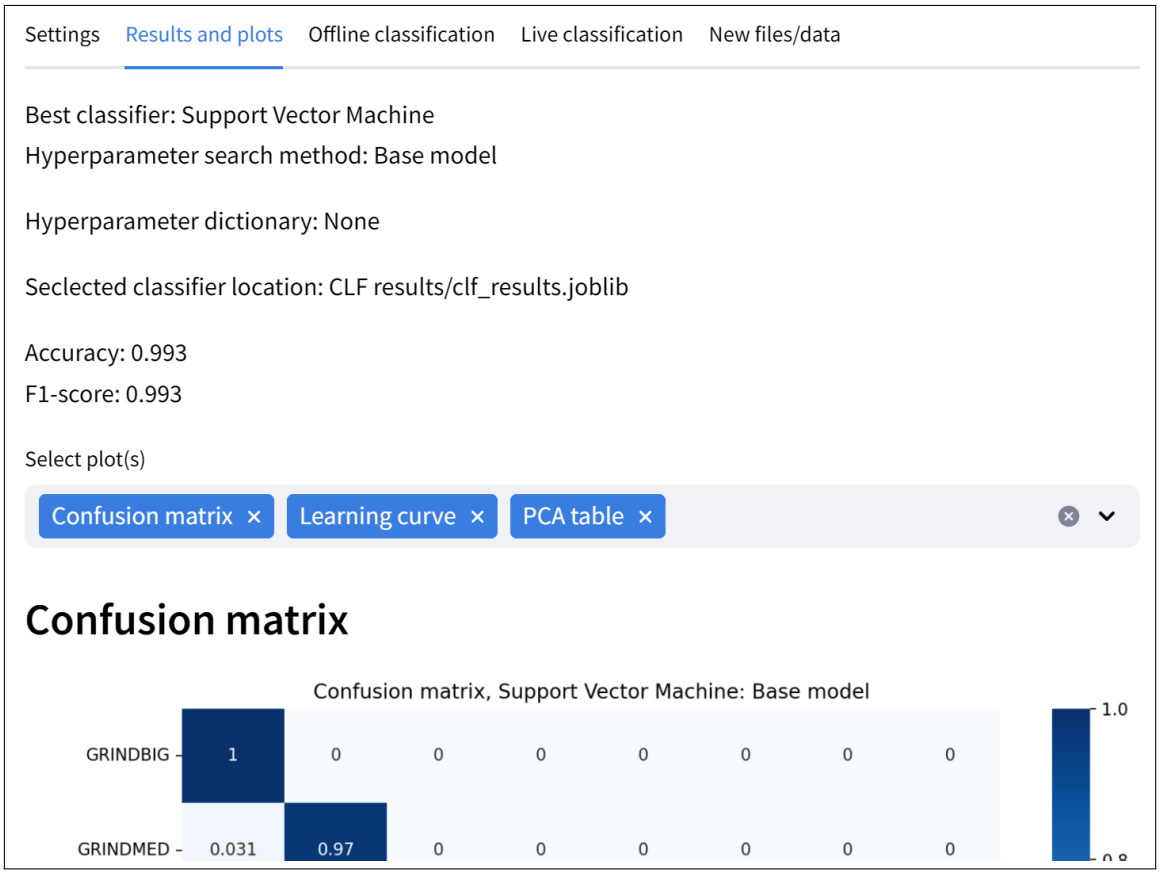
```
\Bachelor-E2511\CLF results\clf_results.joblib
```

If "Want new classifiers" is disabled it will use the `current classifier`:

```
\Bachelor-E2511\OutputFiles\Separated\classifier.joblib
\Bachelor-E2511\OutputFiles\Separated\PCA.joblib
\Bachelor-E2511\OutputFiles\Separated\scaler.joblib
```

1.2 Results and plots

Results tab should look like this if you have ran the main function with plots enabled.



From this tab you can see what the highest performing model was in terms of F1 score, and what hyperparameter search method was used to find it. From here you can also select which plots to examine. Plots included are:

- Learning curve
- Confusion matrix
- PCA table
- Feature importance
- Scree plot
- Biplot
- Biplot 3D
- Decision boundaries
- Distribution of labels

You can choose multiple plots to show, they will appear below each other.

Things to note if some plots arent showing:

- **Confusion matrix**: The preprocessing parameters used for training the classifier must be the same as the ones selected in the ML model tab. Otherwise the confusion matrix will be blank when you run the main function.
- **PCA table**: If the plot for PCA table gives an error, its because there are too many principal components to plot the PCA table in a meaningful way. One could train a new model and reduce the "PCA variance (%) to retain:" to something lower like 0.7, if that is something you want to look at.
- **Decision boundaries**: The plot for decision boundaries requires a classifier trained with 2 PCA components. To do this change "PCA variance (%) to retain:" to 2, and train a new classifier with "Want new classifiers" enabled, with the model and hyperparameter search method you want. Of course this will replace the previous classifier in:

```
    \Bachelor-E2511\CLF results\clf_results.joblib
```

If you want to add more plots than what is currently supported, head over to the main file:

```
\Bachelor-E2511\main.py
```

```python
if want_plots:

    ''' CONFUSION MATRIX '''

    fig_0 = plotLearningCurve(n_results, PCA_train_df, train_labels)

    fig_1 = confusionMatrix(test_labels, PCA_test_df, activity_name, result)

    ''' FEATURE IMPORTANCE '''

    fig_list_0  = plotPCATable(PCA_final, features_per_table=28)

    fig_list_1  = plotFeatureImportance(PCA_final, original_feature_names)

    fig_2       = screePlot(PCA_final)

    ''' PLOTS OF PCA '''

    fig_3 = biplot(feature_df, scaler, window_labels, label_mapping, window_length_seconds)

    fig_4 = biplot3D(feature_df, scaler, window_labels, label_mapping, window_length_seconds)

    fig_5 = plotDecisionBoundaries(PCA_train_df, train_labels, label_mapping, n_results, accuracy_list, cmap_name)

    fig_6 = datasetOverview(window_labels, window_length_seconds, test_size)

    plots = {
        'Learning curve': fig_0,
        'Confusion matrix': fig_1,
        'PCA table': fig_list_0,
        'Feature importance': fig_list_1,
        'Scree plot': fig_2,
        'Biplot': fig_3,
        'Biplot 3D': fig_4,
        'Decision boundaries': fig_5,
        'Distribution of labels': fig_6
    }
```

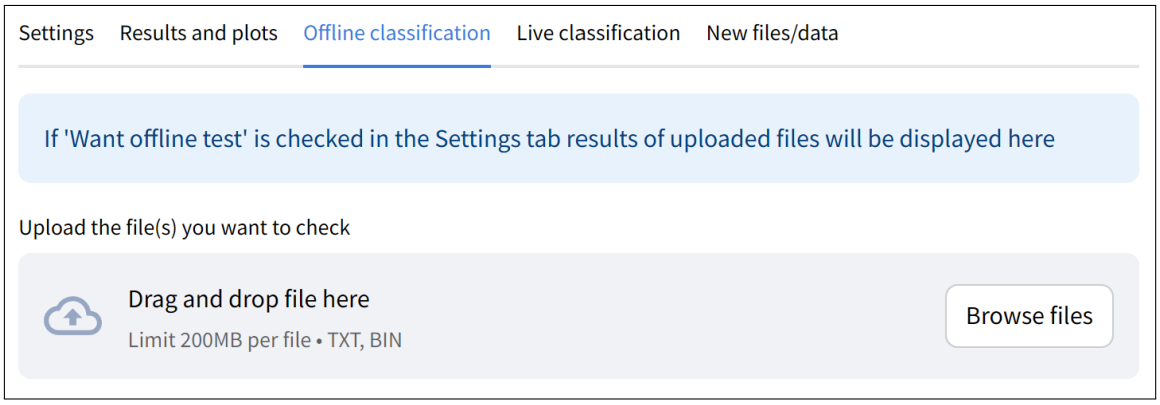
The plot functions exists within

```
\Bachelor-E2511\plotting.py
```

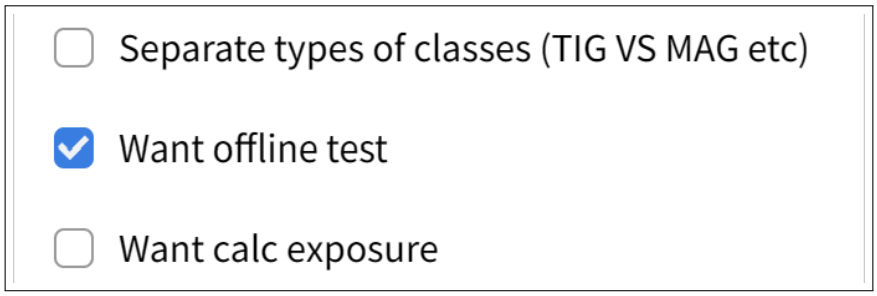
Make your own plot function, return the figure and put it in the "plots" dictionary.

1.3 Offline classification

Initially the tab should look like this:



If you select "Want offline test" from settings tab and run the main function:



The classifier will assess files inside

```
\Bachelor-E2511\testOnFile\testFiles\
```

As displayed in the UI you can upload the file and it will put it in the correct folder. When main function is ran from the settings tab it will save the result here as a csv file:

```
\Bachelor-E2511\testOnFile\predictions.csv
```

The content of the csv file will be displayed automatically in the user interface.

User interface

Real time streaming | ML model | Results | Test classifier on a file | New files/data

Displaying the content of predictions.csv

	Time	Activity	Probability	Top-3	Filtered acti
0	0–20	IDLE	0.3	IDLE (0.30), GRINDSMALL (0.24), GRINDBIG (0.20)	IDLE
1	20–40	GRINDSMALL	0.88	GRINDSMALL (0.88), GRINDMED (0.04), GRINDBIG (0.03)	GRINDSMAL
2	40–60	GRINDSMALL	0.81	GRINDSMALL (0.81), GRINDBIG (0.07), GRINDMED (0.05)	GRINDSMAL
3	60–80	GRINDMED	0.24	GRINDMED (0.24), GRINDSMALL (0.21), GRINDBIG (0.19)	GRINDMED
4	80–100	GRINDMED	0.96	GRINDMED (0.96), GRINDBIG (0.02), GRINDSMALL (0.01)	GRINDMED
5	100–120	GRINDMED	0.98	GRINDMED (0.98), GRINDBIG (0.01), IMPA (0.01)	GRINDMED
6	120–140	IDLE	0.48	IDLE (0.48), GRINDBIG (0.15), IMPA (0.12)	GRINDMED
7	140–160	GRINDBIG	0.82	GRINDBIG (0.82), GRINDSMALL (0.13), GRINDMED (0.03)	IDLE
8	160–180	IMPA	0.6	IMPA (0.60), IDLE (0.16), GRINDSMALL (0.07)	IMPA
9	180–200	IMPA	0.98	IMPA (0.98), WELDALTIG (0.01), GRINDMED (0.00)	IMPA

Upload the file(s) you want to check

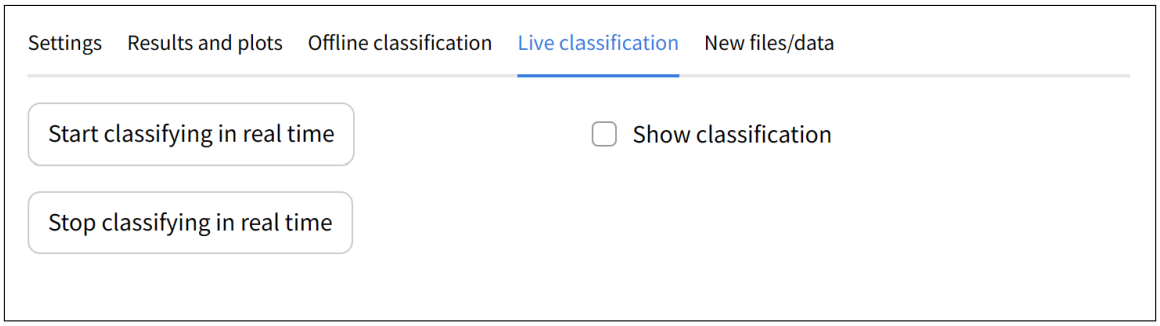
Drag and drop file here

Limit 200MB per file • TXT

Browse files

1.4 Live classification

The live prediction tab looks like this:



Make sure to enable bluetooth on your computer before continuing. The button "Start classifying in real time" initiates the real time program by calling `RT_main()`

```
if st.button("Start classifying in real time"):
    asyncio.run(RT_main())

if st.button("Stop classifying in real time"):
    shutdown_event.set()
```

Figure 2: From User_interface.py

`RT_main()` is an imported function from:

```
\Bachelor-E2511\realtime.py
```

When the "Start classifying in real time" button is clicked, the real time program will look for bluetooth devices selected in

```
\Bachelor-E2511\realtime.py
```

The device it looks for is given by `device_name`.

```
device_list = ["Muse_E2511_GREY", "Muse_E2511_RED", "muse_v3_3", "muse_v3"] # List of bluetooth devices
device_name = device_list[0]                                                # Choose device to connect to from listq
```

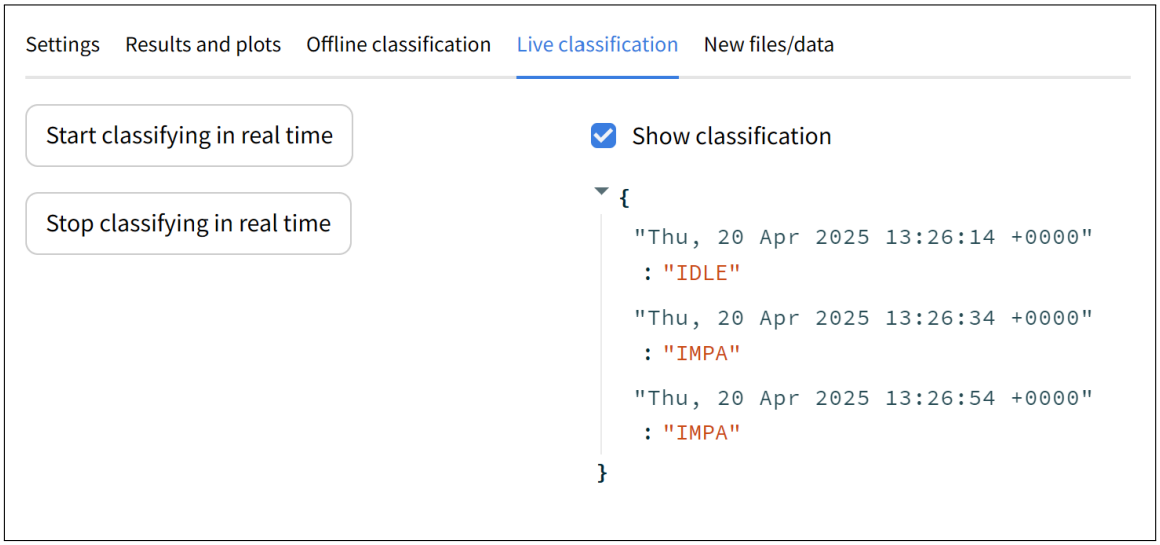
Make sure the name of the muse you have corresponds to the name selected. The name of the muse can be found and changed in Muse viewer.

If the program is unable to connect it is most likely due to a timeout error, click the button again. If its still unable to connect its most likely because the device is already connected from a previous session. In this case disable and enable bluetooth on your device and try again.

When connected, it will immediately start to classify. It will use the transformations and classifier saved in:

```
\Bachelor-E2511\OutputFiles\Separated\classifier.joblib
```

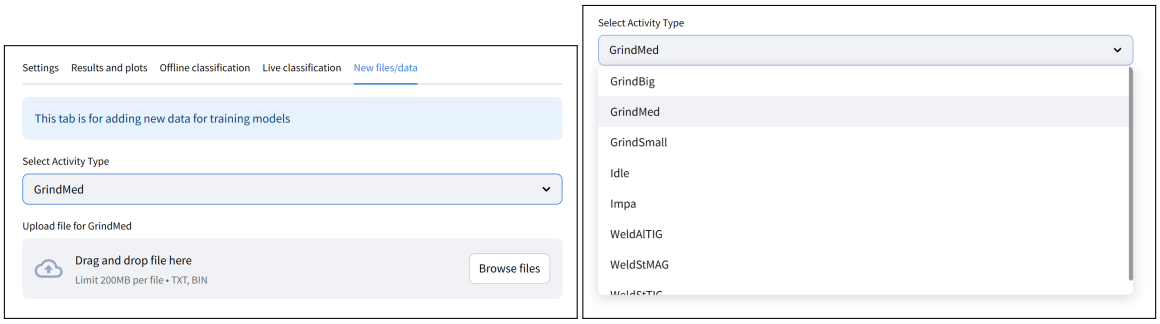
```
\Bachelor-E2511\OutputFiles\Separated\PCA.joblib
\Bachelor-E2511\OutputFiles\Separated\scaler.joblib
```



The predictions will also show in the terminal. "Stop classifying in real time" stops the sensor from streaming/classifying by calling an imported function from the developers of the Muse sensor `shutdown_event.set()`.

1.5 New files/data

From this tab you can add new data in correct folders. It should look like this.

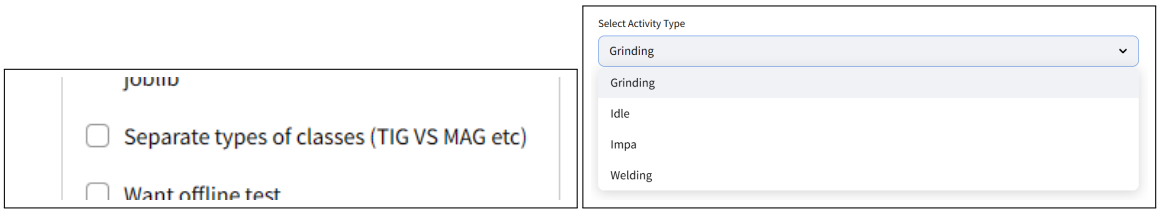


When logging data the muse sensor should be in a configuration where it logs IMU data, temperature, pressure and Light (Lum VIS + Lum IR + Range), this can be done in Muse viewer. The data file should have these columns:

```
1  Timestamp  Gyr.X  Gyr.Y  Gyr.Z  Axl.X  Axl.Y  Axl.Z  Mag.X  Mag.Y  Mag.Z  Temp  Press  Lum VIS Lum IR  Range
```

A newly logged file will also have a header, but these are removed for you when you do feature extraction. The only thing to worry about is logging data in the correct configuration and placing the file in the correct directory such that labels become correct.

If you have collected data, select the label of activity and upload the raw file from the sensor. Afterwards you should do feature extraction from the "Settings" tab, this will process everything else for you. If you want to train a classifier only separating activities (less granular labels) then disable the "Separate types of classes (TIG VS MAG etc)" from the "Settings" tab.



This will change the options and directories in the user interface.

The directories connected to all the options in either case can can be found in:

```
\Bachelor-E2511\User_interface.py
```

```
path_granular = "Datafiles/DatafilesSeparated_Aker"
path_combined = "Datafiles/DatafilesCombined_aker"

if use_granular_labels == True:

    category_dirs = {
        "GrindBig": path_granular + "/GrindBig",
        "GrindMed": path_granular + "GrindMed",
        "GrindSmall": path_granular + "/GrindSmall",
        "Idle": path_granular + "/Idle",
        "Impa": path_granular + "/Impa",
        #"SandSim": path + "/DatafilesSeparated/SandSim",
        "WeldAlTIG": path_granular + "/WeldAlTIG",
        "WeldStMAG": path_granular + "/WeldStMAG",
        "WeldStTIG": path_granular + "/WeldStTIG"
    }

else:
    category_dirs = {
        "Grinding": path_combined + "/Grinding",
        "Idle": path_combined + "/Idle",
        "Impa": path_combined + "/Impa",
        "Welding": path_combined + "/Welding"
    }
```