**Overview of scenario and details about different classes and run script.**

**AISignalMonitor**

Contains ML model, SQLite DB to collect training data and operational mode for classification.

*States:*

* 'collecting\_good' – collecting signal data in normal state
* 'collecting\_bad' - collecting signal data in critical state
* 'collecting\_additional' - collecting signal data of slight variation of the normal state
* 'check\_stored\_data' – check amount of collected data in SQLite DB
* 'train' – train ML model with all available signal data (using test\_train\_split parameter)
* 'save\_model' – save trained ML model
* 'load\_model' – load trained ML model
* 'evaluate' – operational mode classifying incoming signal data with trained ML model
* 'purge\_db' – delete all collected data
* 'purge\_db\_good' - delete collected data of class “good”
* 'purge\_db\_bad' - delete collected data of class “bad”
* 'purge\_db\_add' - delete collected data of class “additional”

*Methods:*

* onSignal – triggered by incoming signal data
* onStateChange – used to change state of AISignalMonitor

**AImonitorLogger**

Simple class with “print” statements as communication interface with AISignalMonitor.

*Methods:*

* print - print normal message
* stateChange - print message concerning state change
* printEval- print message concerning classification result in operational mode

**AISignalProducer**

Class to simulate data source

*States:*

* 'good' – sending signal data of class “good”
* 'bad' – sending signal data of class “bad”
* 'additional' – sending signal data of class “additional”

*Methods:*

* onFrame – returning on signal data set
* onStateChange – used to change state of AISignalProducer

**ProducerLogger**

Simple class with “print” statements as communication interface with AISignalProducer.

*Methods:*

* print - print normal message
* stateChange - print message concerning state change

**helper\_functions.py**

Contains functions to provide parameters and fft transformation

*Functions:*

* get\_db\_parameter- Function to provide parameter for DBs and tables
* get\_AI\_parameter - Function to provide parameter for ML module
* get\_operational\_parameter - Function to provide parameter for operational mode
* get\_parameter - Function to provide parameter for data signals and generation of simulated data
* get\_fft\_values- Function to perform/simulate FFT module

**helper\_db\_functions.py**

Contains all “helper” functions for SQLite DB interaction

*Functions:*

* create\_connection- Create connection to DB
* show\_tables - Show all tables in DB
* show\_data(db\_conn, table) - Show all data of table
* show\_some\_data - Show data of first 10 rows of table
* delete\_data - Delete all rows in table
* vacuum\_db - Cleans up database and releases memory after deleting data
* trans\_ndarray2blob - Transform ndarray to blob
* trans\_blob2ndarray - Transform blob to ndarray
* insert\_blob\_data - Function to insert data signal (blob) into table of SQLite db
* get\_blob\_data- Function to get data signals (blob) from table of SQLite db of specific class
* commit\_data - Commit all changes to DB
* close\_connection - Function to close SQLite DB connection
* state\_is\_UTF8 - Function to check UTF-8

**create\_DBs\_and\_tables.py**

Script to create DBs and tables for both AISignalMonitor and AISignalProducer

**generate\_simulated\_data.py**

Script to generate and save simulated test data.

**visualize\_simulated\_data.py**

Script to visualize some data sets for good, bad and additional data as plots

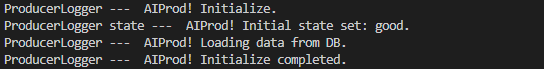
**run\_AISignalMonitor.py**

Script to run simulated scenario using AISignalMonitor and AISignalProducer.

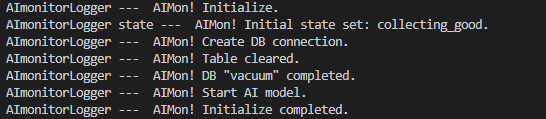
Collecting data, training ML model and simulating operational mode for all signal data classes.

**Details of executing run script with explanations**

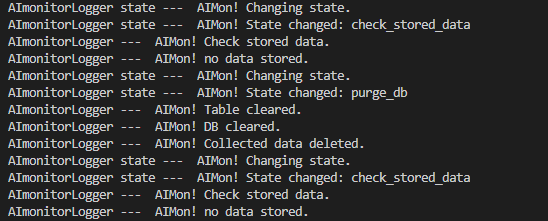
Initialize AISignalProducer



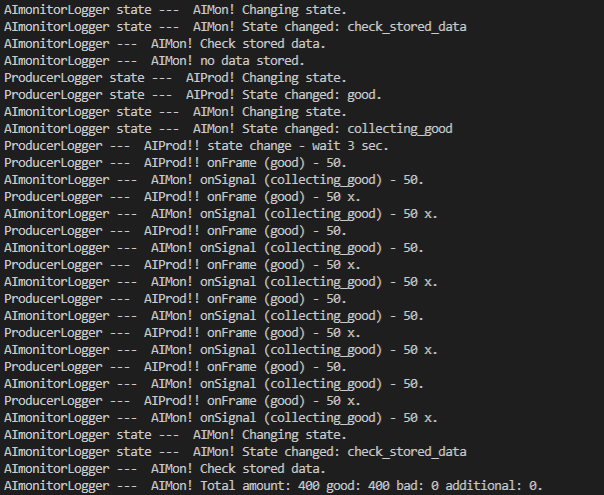
Initialize AISignalMonitor



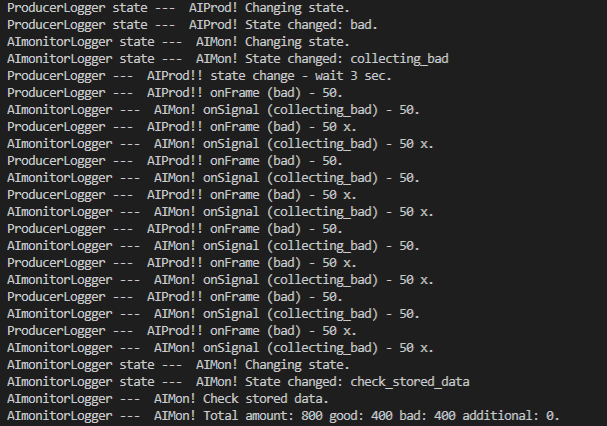
Check stored data and clean table of AISignalMonitor



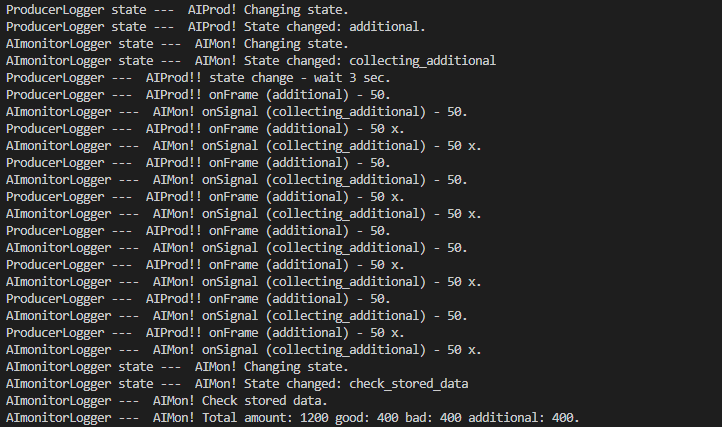
Collecting “good” data and check DB



Collecting “bad” data and check DB

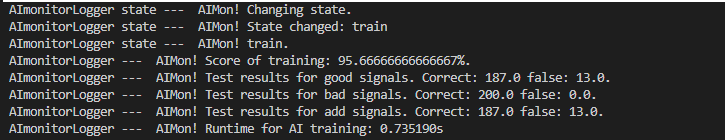


Collecting “additional” data and check DB

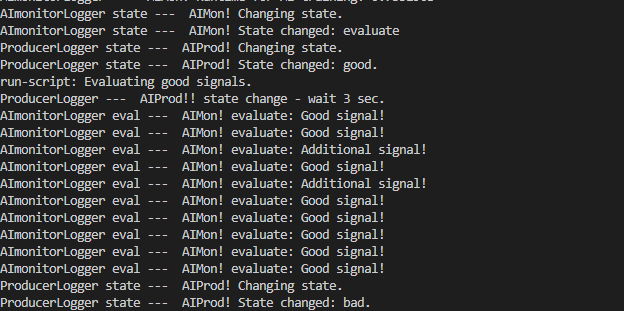


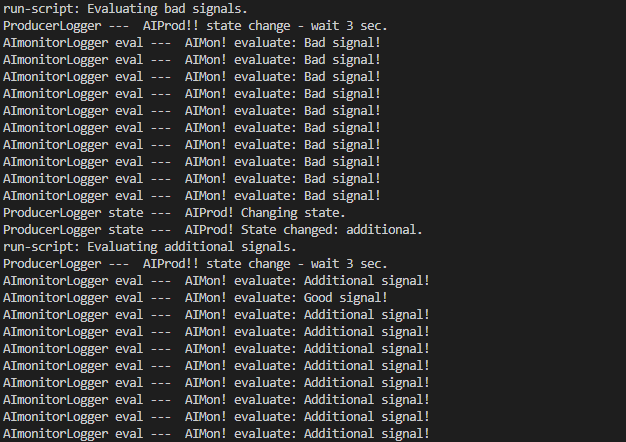
Train ML model and evaluate training result

Comment: ~ 95.7 accuracy with slight problems between “good” and “additional” class which are really close together. “bad” class clearly separated by classification.



Simulating operational mode





End with deconstructor

