Exploring NI AutoML Application for Simulated Waveforms

Eliza Maria Olariu[*] and Horia Hedesiu[*]

[*] Technical University of Cluj-Napoca / Electrical Engineering Department, Cluj-Napoca, Romania,
Eliza.Olariu@campus.utcluj.ro

Abstract – **This paper presents the importance of finding the suitable configurations for Artificial Intelligence and Machine Learning algorithms and correct data preprocessing for a waveform problem. In the Artificial Intelligence and Machine Learning area, this step is one of the most important and it influences the performance result of the model. The experiments of different configurations were done using National Instruments Automated Machine Learning (NI AutoML), a web application created for everyone that allows us to easily change the configurations of the model by just clicking some buttons. This work shows how the model performance is influenced by modifying what columns of data to use, by data splitting or by adding or deleting preprocessing steps in the pipeline. All the results obtained for the different experiments are analyzed in this paper. The proposed flow is generic enough to be applied for all the use cases. To exemplify the whole process, a synthetic data set obtained by generating current and voltage in an RL circuit was chosen and the experiments part was created. The data represent two waveforms: one for current and one for voltage and they represent data recorded during the test time. In the end process each test has a label associated: Pass or Fail. The classification problem was defined for help in improving the fail detection rate.**

Cuvinte cheie: *NI AutoML, forme de undă simulate, problemă de clasificare, preprocesarea datelor, inteligență artificială și învățare automată.*

Keywords: *NI AutoML, simulated waveforms, classification problem, data preprocessing, Artificial Intelligence and Machine Learning.*

I. Introduction

In the spotlight these days is Artificial Intelligence and Machine Learning area. This technique is on the increasing of development, and it is used in many domains to improve the quality of life. Better solutions can be created based on Artificial Intelligence and Machine Learning in different domains: in industry like the batteries industry from research [1] to production [2] and to monitoring [3] or in the healthcare for diagnostics, predictive analytics, personalized medicine and administration application [4].

In all cases, the data obtained can create a model that finds patterns and learns from previous information and helps in identifying the aspects of the new data [5]. There are many types of Artificial Intelligence and Machine Learning algorithms: supervised, unsupervised; for classification, regression; for tabular data, images, waveforms; exists something for each problem [6].

The first aspect in solving problems in the modern approach is to obtain data. It should be possible to use real data, recorded directly from the sensors or measurements from the environment, or it is possible to simulate them in laboratory [7]. Simulated data can be also applied together with real data or used in the beginning for training the models that will be deployed after that in the real systems.

An example of simulated waveforms data used in a predictive classification model is presented in [8]. Data generated is a way to simplify a battery data model and can help in presenting the full flow for a system that helps in monitoring battery performance [9]. This infrastructure can be used in Prognostics and Health Management [10] or in the testing phase of manufacturing [11].

After having the data, the next phase is understanding and transforming it into information. There are more preprocessing steps needed to clean, transform and prepare data for Artificial Intelligence and Machine Learning algorithms [12]. Last but not least, it is also important to find the correct parameters to the predictive model. Do more experiments and identify the suitable configuration for each problem to solve. National Instruments Automated Machine Learning (NI AutoML) is one of the web applications that allows the customers to create all the environment for defining all the experiments for finding the best configuration and the best model and also be able to monitor and change it with time passing and environment changing [13].

II. Classification Problem To Solved

A. Data set

The data set contains simulated waveforms generated using an RL series circuit that represents an object into the testing phase into a factory. Each measurement contains two waveforms that represent the current and the voltage and it is associated with a label that represents if the test was with success or failed (Fig. 1) [8].

The voltage (V) signal represents the input in the RL circuit. It is created as a sum of three sinus wave components with different frequencies and added on top a low-level noise signal. The output of the RL circuit generated the current signal (A). In the middle of the test time, parameters of the RL circuit can vary, and, in this case, it was considered that data provided from a failed test [8].

The data set contains 520 waveforms with 1000 test points and 5 columns:

- Current (A)
- Voltage (V)
- Index
- Unit
- Label.

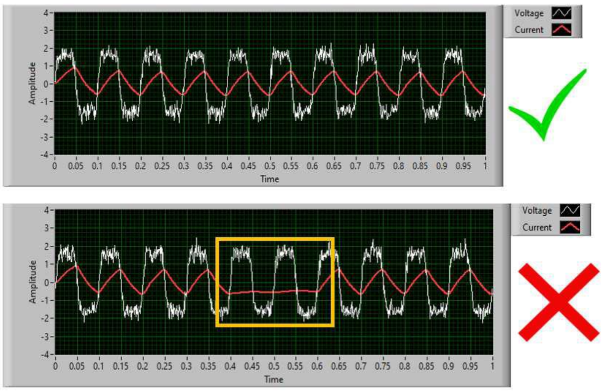


Fig. 1. Simulated waveform for success (top figure) and failed (bottom figure) having voltage (V) and the current (A) waveforms from the RL circuit [8].

B. *NI AutoML*

For defining the Artificial Intelligence and Machine Learning model, it was used NI AutoML application [13]. It has many configuration aspects that can be easily modified with simple clicks and trained in 9 different models and shows the best one [14]:

- AdaBoost Classifier – Adaptive Boosting - a statistical classification meta-estimator
- Baseline Classification – predicting on classes' distribution
- Decision Tree – create one decision tree with conditions rules
- Gradient Boosting – additive model base on decision trees
- LightGBM – based on Gradient Boosting
- Logistic Regression – algorithm for one-vs-rest
- Random Forest – create more decision trees with conditions rules
- Support Vector Classifier – divided the space in regions for each label
- XGBoost – Extreme Gradient Boosting – based on Gradient Boosting

For identifying the best model, one criterion can be selected. The application offers 6 different metrics [14]:

- Accuracy – correct classification divided to all classifications
- AUC – area under the ROC curve
- Recall – true positive divided to true positive and false negative
- Precision – true positive divided to true positive and false positive
- Balanced Accuracy - average recall obtained on each class
- F1 – harmonic mean of the precision and recall

Another criteria to select the best algorithm is Run Time, some of the algorithms take more time to be train comparing with the others. For some cases, time is also very important, like real time problem.

III. Experiments

NI AutoML configuration that was kept the same for all the experiments:

- Target selected value: label
- Prediction type: Binary Classification
- For the first part of the experiments, the metric used to select the best algorithm is area under the curve "AUC" (secund column in the table results). This metric will rank the positive label higher than the negative one [15].

A. *Experiments based on waveform used*

1) Used just Current (A): drop the Voltage column and create the prediction model using just features obtained from Current waveform.

Obtained results can be seen on Table I.

➔ Best model: Random Forest 0.865 AUC

2) Used just Voltage (V): drop the Current column and create the prediction model using just features obtained from Voltage waveform

Obtained results can be seen on Table II.

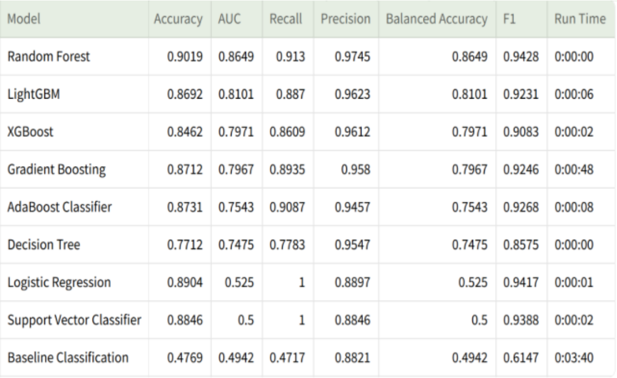
➔ Best model: Decision Tree 0.53 AUC

3) Used both Current (A) and Voltage (V): do not drop any columns and create the prediction model using features obtained from both Current and Voltage waveforms

Obtained results can be seen on **Error! Reference source not found.**II.

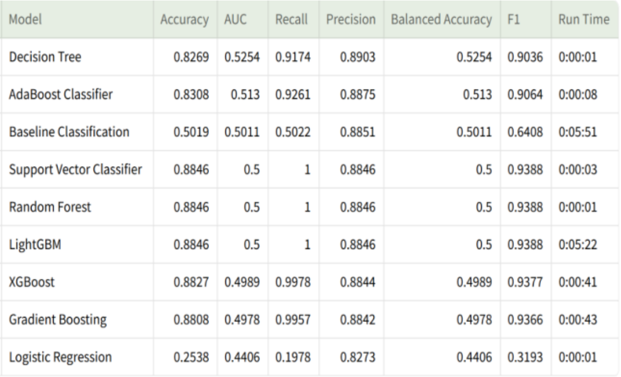
➔ Best model: Random Forest 0.859 AUC

TABLE I.
Results obtained for Experiment A 1) order by AUC column, best model on the first line



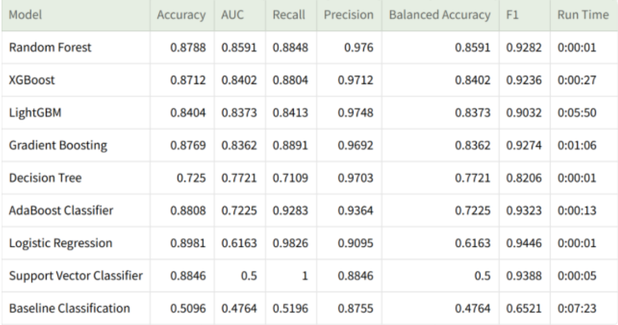
Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.9019	0.8649	0.913	0.9745	0.8649	0.9428	0:00:00
LightGBM	0.8692	0.8101	0.887	0.9623	0.8101	0.9231	0:00:06
XGBoost	0.8462	0.7971	0.8609	0.9612	0.7971	0.9083	0:00:02
Gradient Boosting	0.8712	0.7967	0.8935	0.958	0.7967	0.9246	0:00:48
AdaBoost Classifier	0.8731	0.7543	0.9087	0.9457	0.7543	0.9268	0:00:08
Decision Tree	0.7712	0.7475	0.7783	0.9547	0.7475	0.8575	0:00:00
Logistic Regression	0.8904	0.525	1	0.8897	0.525	0.9417	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:02
Baseline Classification	0.4769	0.4942	0.4717	0.8821	0.4942	0.6147	0:03:40

TABLE II.
Results obtained for Experiment A 2) order by AUC column, best model on the first line



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Decision Tree	0.8269	0.5254	0.9174	0.8903	0.5254	0.9036	0:00:01
AdaBoost Classifier	0.8308	0.513	0.9261	0.8875	0.513	0.9064	0:00:08
Baseline Classification	0.5019	0.5011	0.5022	0.8851	0.5011	0.6408	0:05:51
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:03
Random Forest	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:01
LightGBM	0.8846	0.5	1	0.8846	0.5	0.9388	0:05:22
XGBoost	0.8827	0.4989	0.9978	0.8844	0.4989	0.9377	0:00:41
Gradient Boosting	0.8808	0.4978	0.9957	0.8842	0.4978	0.9366	0:00:43
Logistic Regression	0.2538	0.4406	0.1978	0.8273	0.4406	0.3193	0:00:01

TABLE III.
RESULTS OBTAINED FOR EXPERIMENT A 3) ORDER BY AUC COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.8788	0.8591	0.8848	0.976	0.8591	0.9282	0:00:01
XGBoost	0.8712	0.8402	0.8804	0.9712	0.8402	0.9236	0:00:27
LightGBM	0.8404	0.8373	0.8413	0.9748	0.8373	0.9032	0:05:50
Gradient Boosting	0.8769	0.8362	0.8891	0.9692	0.8362	0.9274	0:01:06
Decision Tree	0.725	0.7721	0.7109	0.9703	0.7721	0.8206	0:00:01
AdaBoost Classifier	0.8808	0.7225	0.9283	0.9364	0.7225	0.9323	0:00:13
Logistic Regression	0.8981	0.6163	0.9826	0.9095	0.6163	0.9446	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:05
Baseline Classification	0.5096	0.4764	0.5196	0.8755	0.4764	0.6521	0:07:23

Conclusion:

The best results were obtained for the case when it was used just Current waveform. This happened because the Current is the output of the simulated RL circuit and just the Current is influenced by R, L value modifications during the test.

➔ Best model: Random Forest 0.86 AUC

For the next experiment, it will be used just Current waveform.

B. Experiments basd on selection positive class

1) Used "Pass" value: in the previous examples it was used "Pass" value.

Obtained results can be seen on Table I.

➔ Best model: Random Forest 0.86 AUC

2) Used "Fail" value: change the value for positive class into "Fail"

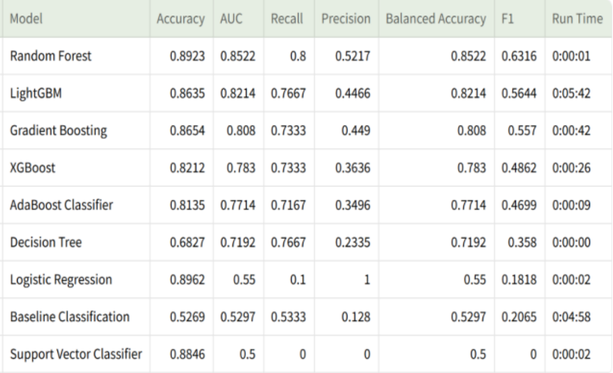
Obtained results can be seen on Table IV.

➔ Best model: Random Forest 0.85 AUC

Conclusion:

The best results were obtained for the case when it was used "Pass" value for positive class. This is expected because the "Pass" value represents the success test simulated.

TABLE IV.
RESULTS OBTAINED FOR EXPERIMENT B 2) ORDER BY AUC COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.8923	0.8522	0.8	0.5217	0.8522	0.6316	0:00:01
LightGBM	0.8635	0.8214	0.7667	0.4466	0.8214	0.5644	0:05:42
Gradient Boosting	0.8654	0.808	0.7333	0.449	0.808	0.557	0:00:42
XGBoost	0.8212	0.783	0.7333	0.3636	0.783	0.4862	0:00:26
AdaBoost Classifier	0.8135	0.7714	0.7167	0.3496	0.7714	0.4699	0:00:09
Decision Tree	0.6827	0.7192	0.7667	0.2335	0.7192	0.358	0:00:00
Logistic Regression	0.8962	0.55	0.1	1	0.55	0.1818	0:00:02
Baseline Classification	0.5269	0.5297	0.5333	0.128	0.5297	0.2065	0:04:58
Support Vector Classifier	0.8846	0.5	0	0	0.5	0	0:00:02

➔ Best model: Random Forest 0.86 AUC

For the next experiment, it will be used "Pass" value for positive class.

C. Experiments based on preprocessing steps in pipeline

1) Used default set up for waveforms: in the previous examples it was selected:

- add_missing_indicator
- infinity_to_nan
- mean_median_imputer
- nan_column_dropper
- datetime_features
- high_cardinality_dropper
- min_max_scaler
- match_variables
- waveforms_feature_extractor
- remove_special_json_characters
- id_label_encoder
- ordinal_encode_target
- smote

Obtained results can be seen on Table I.

➔ Best model: Random Forest 0.86 AUC

2) Delete SOTE step and data time feature extraction: deleted steps:

- datetime_features
- smote

Obtained results can be seen on Table V.

➔ Best model: Random Forest 0.82 AUC

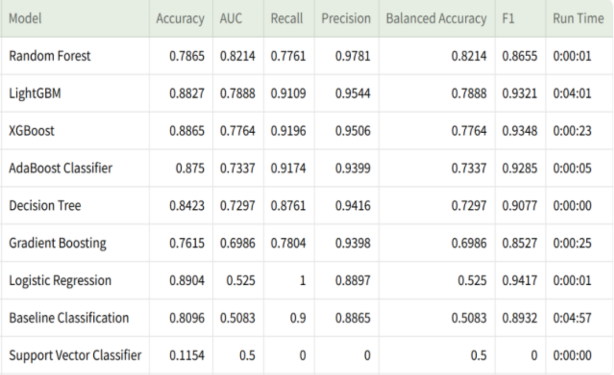
3) Delete also MinMaxScaler step and high_cardinality_dropper step: deleted steps:

- min_max_scaler
- high_cardinality_dropper
- datetime_features
- smote

Obtained results can be seen on Table VI.

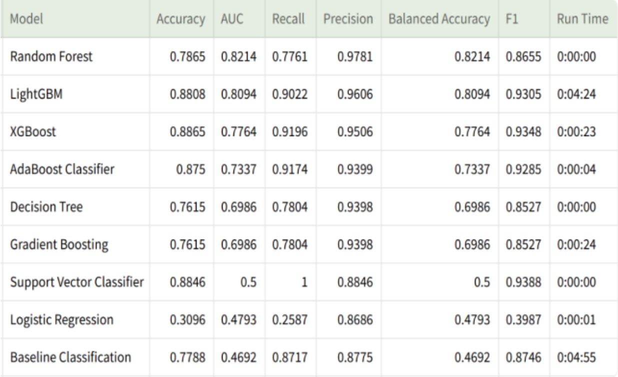
➔ Best model: Random Forest 0.82 AUC

TABLE V.
RESULTS OBTAINED FOR EXPERIMENT C 2) ORDER BY AUC COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.7865	0.8214	0.7761	0.9781	0.8214	0.8655	0:00:01
LightGBM	0.8827	0.7888	0.9109	0.9544	0.7888	0.9321	0:04:01
XGBoost	0.8865	0.7764	0.9196	0.9506	0.7764	0.9348	0:00:23
AdaBoost Classifier	0.875	0.7337	0.9174	0.9399	0.7337	0.9285	0:00:05
Decision Tree	0.8423	0.7297	0.8761	0.9416	0.7297	0.9077	0:00:00
Gradient Boosting	0.7615	0.6986	0.7804	0.9398	0.6986	0.8527	0:00:25
Logistic Regression	0.8904	0.525	1	0.8897	0.525	0.9417	0:00:01
Baseline Classification	0.8096	0.5083	0.9	0.8865	0.5083	0.8932	0:04:57
Support Vector Classifier	0.1154	0.5	0	0	0.5	0	0:00:00

TABLE VI.
RESULTS OBTAINED FOR EXPERIMENT C 3) ORDER BY AUC COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.7865	0.8214	0.7761	0.9781	0.8214	0.8655	0:00:00
LightGBM	0.8808	0.8094	0.9022	0.9606	0.8094	0.9305	0:04:24
XGBoost	0.8865	0.7764	0.9196	0.9506	0.7764	0.9348	0:00:23
AdaBoost Classifier	0.875	0.7337	0.9174	0.9399	0.7337	0.9285	0:00:04
Decision Tree	0.7615	0.6986	0.7804	0.9398	0.6986	0.8527	0:00:00
Gradient Boosting	0.7615	0.6986	0.7804	0.9398	0.6986	0.8527	0:00:24
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:00
Logistic Regression	0.3096	0.4793	0.2587	0.8686	0.4793	0.3987	0:00:01
Baseline Classification	0.7788	0.4692	0.8717	0.8775	0.4692	0.8746	0:04:55

Conclusion:

The best results were obtained for the case when it was used all the default steps for waveform. If steps are deleted the performance goes down. Normalization, balance, adding more features or deleting high cardinality improves the results.

➔ Best model: Random Forest 0.86 AUC

For the next experiment, default preprocessing steps will be used for waveform.

D. Experiments based on split configuration

1) Used "Random" type with 60% training set, 20% validation set and 20% test set: in the previous examples it was used this configuration.

Obtained results can be seen on Table I.

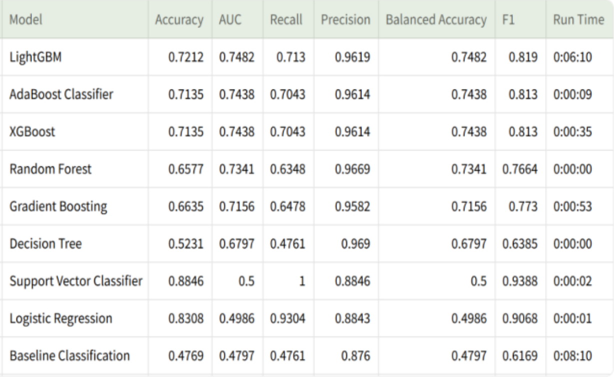
➔ Best model: Random Forest 0.86 AUC

2) Used "Random" type with 80% training set, 10% validation set and 10% test set: change values to use more data from training

Obtained results can be seen on Table VII.

➔ Best model: LightGBM 0.75 AUC

TABLE VII.
RESULTS OBTAINED FOR EXPERIMENT D 2) ORDER BY AUC COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
LightGBM	0.7212	0.7482	0.713	0.9619	0.7482	0.819	0:06:10
AdaBoost Classifier	0.7135	0.7438	0.7043	0.9614	0.7438	0.813	0:00:09
XGBoost	0.7135	0.7438	0.7043	0.9614	0.7438	0.813	0:00:35
Random Forest	0.6577	0.7341	0.6348	0.9669	0.7341	0.7664	0:00:00
Gradient Boosting	0.6635	0.7156	0.6478	0.9582	0.7156	0.773	0:00:53
Decision Tree	0.5231	0.6797	0.4761	0.969	0.6797	0.6385	0:00:00
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:02
Logistic Regression	0.8308	0.4986	0.9304	0.8843	0.4986	0.9068	0:00:01
Baseline Classification	0.4769	0.4797	0.4761	0.876	0.4797	0.6169	0:08:10

3) Used "Cross-Validation" option: check this option in the UI of NI AutoML.

Obtained results can be seen in Table VIII.

➔ Best model: Random Forest 0.78 AUC

Conclusion:

The best results were obtained for the case when it was used "Random" type with 60% training set, 20% validation set and 20% test set.

➔ Best model: Random Forest 0.86 AUC

For the next experiment, it will be used "Random" split type with 60% training set, 20% validation set and 20% test set.

E. Experiments based on metric to optimize

1) Used "AUC" option: in the previous examples it was used this configuration.

Obtained results can be seen on Table I.

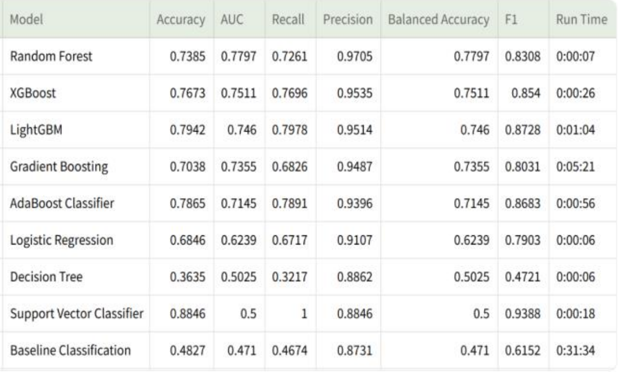
➔ Best model: Random Forest 0.86 AUC

2) Used "Accuracy" option:

Obtained results can be seen on Table IX.

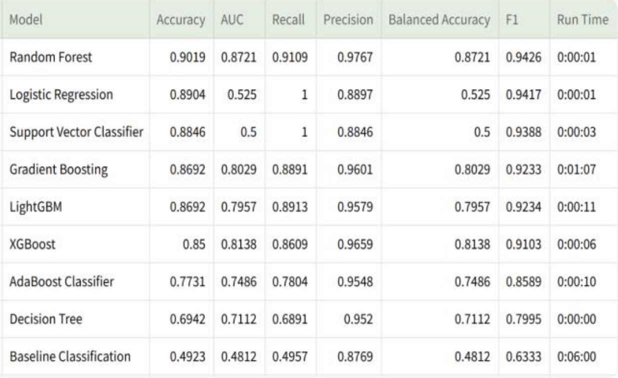
➔ Best model: Random Forest 0.9 Accuracy

TABLE VIII.
RESULTS OBTAINED FOR EXPERIMENT D 3) ORDER BY AUC COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.7385	0.7797	0.7261	0.9705	0.7797	0.8308	0:00:07
XGBoost	0.7673	0.7511	0.7696	0.9535	0.7511	0.854	0:00:26
LightGBM	0.7942	0.746	0.7978	0.9514	0.746	0.8728	0:01:04
Gradient Boosting	0.7038	0.7355	0.6826	0.9487	0.7355	0.8031	0:05:21
AdaBoost Classifier	0.7865	0.7145	0.7891	0.9396	0.7145	0.8683	0:00:56
Logistic Regression	0.6846	0.6239	0.6717	0.9107	0.6239	0.7903	0:00:06
Decision Tree	0.3635	0.5025	0.3217	0.8862	0.5025	0.4721	0:00:06
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:18
Baseline Classification	0.4827	0.471	0.4674	0.8731	0.471	0.6152	0:31:34

TABLE IX.
RESULTS OBTAINED FOR EXPERIMENT E 2) ORDER BY ACCURACY COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.9019	0.8721	0.9109	0.9767	0.8721	0.9426	0:00:01
Logistic Regression	0.8904	0.525	1	0.8897	0.525	0.9417	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:03
Gradient Boosting	0.8692	0.8029	0.8891	0.9601	0.8029	0.9233	0:01:07
LightGBM	0.8692	0.7957	0.8913	0.9579	0.7957	0.9234	0:00:11
XGBoost	0.85	0.8138	0.8609	0.9659	0.8138	0.9103	0:00:06
AdaBoost Classifier	0.7731	0.7486	0.7804	0.9548	0.7486	0.8589	0:00:10
Decision Tree	0.6942	0.7112	0.6891	0.952	0.7112	0.7995	0:00:00
Baseline Classification	0.4923	0.4812	0.4957	0.8769	0.4812	0.6333	0:06:00

3) Used “Recall” option:

Obtained results can be seen on Table X.

➔ Best model: Logistic Regression and Support Vector Classifier 1 Recall

4) Used “Precision” option:

Obtained results can be seen on Table XI.

➔ Best model: Random Forest 0.97 Precision

5) Used “Balanced Accuracy” option:

Obtained results can be seen on Table XII.

➔ Best model: Random Forest 0.86 Balanced Accuracy

6) Used “F1” option:

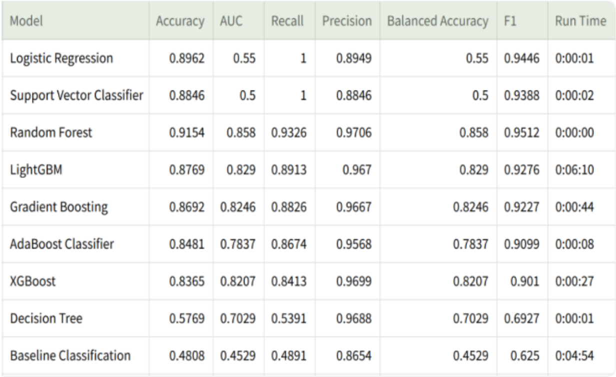
Obtained results can be seen on Table XIII.

➔ Best model: Random Forest 0.95 F1

Conclusion:

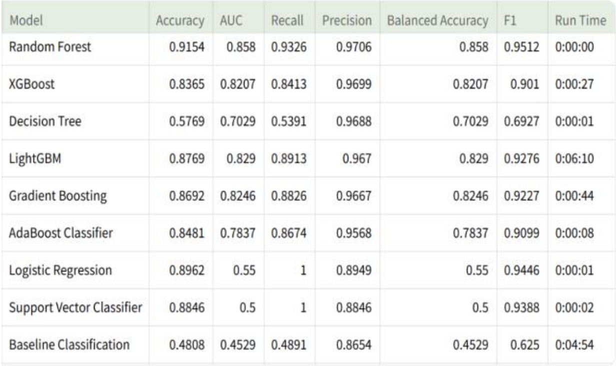
The best results were obtained from the case when it was used Recall metrics for Logistic Regression and Support Vector Classifier. On this aspect, the metric can be chosen based on the problem that we want to solve, not based on the value obtain for it. In our case the best meaning is using AUC.

TABLE X.
RESULTS OBTAINED FOR EXPERIMENT E 3) ORDER BY RECALL COLUMN, BEST MODEL ON THE FIRST LINE



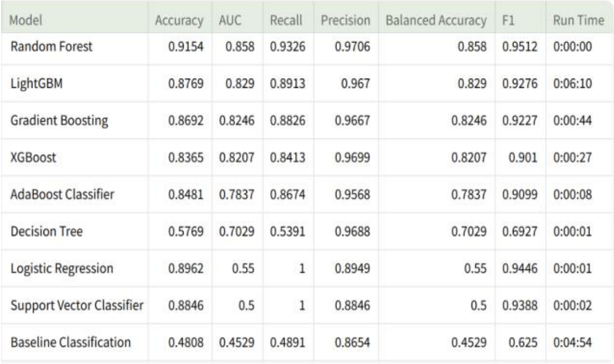
Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Logistic Regression	0.8962	0.55	1	0.8949	0.55	0.9446	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:02
Random Forest	0.9154	0.858	0.9326	0.9706	0.858	0.9512	0:00:00
LightGBM	0.8769	0.829	0.8913	0.967	0.829	0.9276	0:06:10
Gradient Boosting	0.8692	0.8246	0.8826	0.9667	0.8246	0.9227	0:00:44
AdaBoost Classifier	0.8481	0.7837	0.8674	0.9568	0.7837	0.9099	0:00:08
XGBoost	0.8365	0.8207	0.8413	0.9699	0.8207	0.901	0:00:27
Decision Tree	0.5769	0.7029	0.5391	0.9688	0.7029	0.6927	0:00:01
Baseline Classification	0.4808	0.4529	0.4891	0.8654	0.4529	0.625	0:04:54

TABLE XI.
RESULTS OBTAINED FOR EXPERIMENT E 4) ORDER BY PRECISION COLUMN, BEST MODEL ON THE FIRST LINE



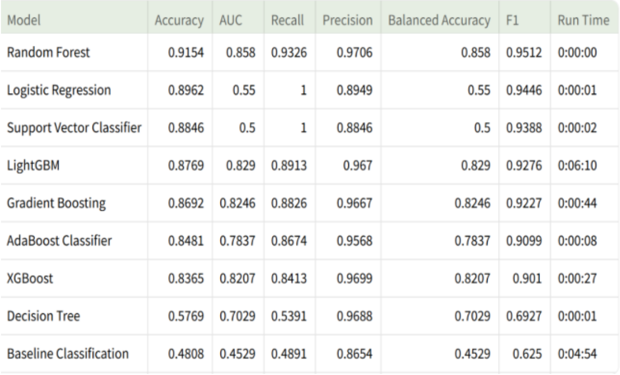
Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.9154	0.858	0.9326	0.9706	0.858	0.9512	0:00:00
XGBoost	0.8365	0.8207	0.8413	0.9699	0.8207	0.901	0:00:27
Decision Tree	0.5769	0.7029	0.5391	0.9688	0.7029	0.6927	0:00:01
LightGBM	0.8769	0.829	0.8913	0.967	0.829	0.9276	0:06:10
Gradient Boosting	0.8692	0.8246	0.8826	0.9667	0.8246	0.9227	0:00:44
AdaBoost Classifier	0.8481	0.7837	0.8674	0.9568	0.7837	0.9099	0:00:08
Logistic Regression	0.8962	0.55	1	0.8949	0.55	0.9446	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:02
Baseline Classification	0.4808	0.4529	0.4891	0.8654	0.4529	0.625	0:04:54

TABLE XII.
RESULTS OBTAINED FOR EXPERIMENT E 5) ORDER BY BALANCED ACCURACY COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.9154	0.858	0.9326	0.9706	0.858	0.9512	0:00:00
LightGBM	0.8769	0.829	0.8913	0.967	0.829	0.9276	0:06:10
Gradient Boosting	0.8692	0.8246	0.8826	0.9667	0.8246	0.9227	0:00:44
XGBoost	0.8365	0.8207	0.8413	0.9699	0.8207	0.901	0:00:27
AdaBoost Classifier	0.8481	0.7837	0.8674	0.9568	0.7837	0.9099	0:00:08
Decision Tree	0.5769	0.7029	0.5391	0.9688	0.7029	0.6927	0:00:01
Logistic Regression	0.8962	0.55	1	0.8949	0.55	0.9446	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:02
Baseline Classification	0.4808	0.4529	0.4891	0.8654	0.4529	0.625	0:04:54

TABLE XIII.
RESULTS OBTAINED FOR EXPERIMENT E 6) ORDER BY F1 COLUMN, BEST MODEL ON THE FIRST LINE



Model	Accuracy	AUC	Recall	Precision	Balanced Accuracy	F1	Run Time
Random Forest	0.9154	0.858	0.9326	0.9706	0.858	0.9512	0:00:00
Logistic Regression	0.8962	0.55	1	0.8949	0.55	0.9446	0:00:01
Support Vector Classifier	0.8846	0.5	1	0.8846	0.5	0.9388	0:00:02
LightGBM	0.8769	0.829	0.8913	0.967	0.829	0.9276	0:06:10
Gradient Boosting	0.8692	0.8246	0.8826	0.9667	0.8246	0.9227	0:00:44
AdaBoost Classifier	0.8481	0.7837	0.8674	0.9568	0.7837	0.9099	0:00:08
XGBoost	0.8365	0.8207	0.8413	0.9699	0.8207	0.901	0:00:27
Decision Tree	0.5769	0.7029	0.5391	0.9688	0.7029	0.6927	0:00:01
Baseline Classification	0.4808	0.4529	0.4891	0.8654	0.4529	0.625	0:04:54

IV. CONCLUSIONS

This paper presented the importance of finding the best parameters and the best preprocessing steps for Artificial Intelligence and Machine Learning model creation. The experiments presented used the simulated waveform data set.

NI AutoML was used as an Artificial Intelligence and Machine Learning application.

It was defined and presented results for 17 experiments that play with different configurations:

A. Experiments based on waveform used

1) Used just Current (A)

2) Used just Voltage (V)

3) Used both Current (A) and Voltage (V)

B. Experiments based on selection positive class

1) Used “Pass” value

2) Used “Fail” value

C. Experiments based on preprocessing steps in pipeline

1) Used default set up for waveforms

2) Delete SOTE step and data time feature extraction

3) Delete also MinMaxScaler step and high_cardinality _dropper step

D. Experiments based on split configuration

1) Used "Random" type with 60% training set, 20% validation set and 20% test set

2) Used "Random" type with 80% training set, 10% validation set and 10% test set

3) Used "Cross-Validation" option

E. Experiments based on metric to optimize

1) Used "AUC" option

2) Used "Accuracy" option

3) Used "Recall" option

4) Used "Precision" option

5) Used "Balanced Accuracy" option

6) Used "F1" option

The best results were obtained from experiment A 1) having:

- Used just Current (A) waveform
- Used "Pass" value for positive class
- Used default set up for preprocessing steps for waveforms
- Used "Random" type with 60% training set, 20% vali-dation set and 20% test set
- Using AUC metric

➔ Best model: Random Forest 0.86 AUC

In the future, using data taken directly from a real system can be used in this infrastructure to confirm all the aspects and analysis the differences and the similarity with simulated data.

ACKNOWLEDGMENT

This work used NI AutoML, one application from NI's AI infrastructure that the customers can obtain.

Source of research funding in this article: Research program financed by the Technical University of Cluj-Napoca.

Contribution of authors:

First author – 50%

First coauthor – 50%

Received on September 17,2024

Editorial Approval on December 2, 2024

REFERENCES

[1] T. Lombardo, et al. "Artificial intelligence applied to battery research: Hype or reality?," *Chem Rev*, June 2022, vol. 122, iss. 12, pp. 10899-10969, doi: 10.1021/acs.chemrev.1c00108.

[2] M. Faraji Niri, K. Aslansefat, S. Haghi, M. Hashemian, R. Daub, J. A. Marco, "A review of the applications of explainable machine learning for lithium–ion batteries: From production to state and performance estimation", *Energies,* 16, 6360, doi: 10.3390/en16176360, 2023.

[3] A. Prisacaru, P. J. Gromala, M. B. Jeronimo, Bongtae Han and Guo Qi Zhang, "Prognostics and health monitoring of electronic system: A review," *2017 18th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE*), Dresden, Germany, 2017, pp. 1-11, doi: 10.1109/EuroSimE.2017.7926248.

[4] G. S. Nadella, S. Satish, K. Meduri, and S. S. Meduri, "A systematic literature review of advancements, challenges and future directions of AI and ML in healthcare", *International Journal of Machine Learning for Sustainable Development*, vol.5, iss. 3, pp. 115-130, 2023.

[5] C. M. Bishop, *Pattern Recognition and Machine Learning,* New York: Springer, 2016.

[6] R. Agrawal, "Fundamentals of machine learning," *Machine learning for healthcare: Handling and managing data*, p. 1, 2020.

[7] T. Chamunorwa, D. Ursutiu, C. Samoila, H. Hedesiu, and H. A. Modran, "Electronic educational laboratory platform for students," in *Online Engineering and Society 4.0: Proceedings of the 18th International Conference on Remote Engineering and Virtual Instrumentation*. Springer, 2022, pp. 311–322.

[8] O. E. Maria, and H. Horia. "Improving manufacturing testing steps using artificial intelligence and machine learning algorithms trained on simulated waveforms" *2024 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR).* Cluj-Napoca, Romania, 2024, pp. 1-5.

[9] J. Peng, X. Zhao, J. Ma, et. al, "Enhancing lithium-ion battery monitoring: A critical review of diverse sensing approaches", *eTransportation*, vol. 22, 2024.

[10] M. Ahsan, S. Stoyanov and C. Bailey, "Prognostics of automotive electronics with data driven approach: A review," *2016 39th International Spring Seminar on Electronics Technology (ISSE)*, Pilsen, Czech Republic, 2016, pp. 279-284, doi: 10.1109/ISSE.2016.7563205.

[11] E. M. Olariu, C. Vlasin, O. Balaj, and H. Hedesiu, "Cloud based realtime data acquisition for industrial applications," in *2023 IEEE 29th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, Craiova, Romania, 2023, pp. 324-328.

[12] L. Pierson, *Data Science For Dummies*, 3rd Edition, New Jersey: Wiley, 2021.

[13] E. M. Olariu, and H. Hedesiu, "Artificial intelligence and machine mearning using NI AutoML in industry-case study: Simulated waveforms", *2024 International Conference on Applied and Theoretical Electricity (ICATE),* Craiova, Romania, 2024, pp. 1-4.

[14] scikit-learn, Machine Learning in Python, [cited 2024 December 8], Available from: https://scikit-learn.org/stable/

[15] Machine Learning, "Classification: ROC and AUC", [cited 2024 December 8], Available from: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc