CLASSify User Guide

Contents

Overview

CLASSify is a self-service tool developed by the Center for Applied Artificial Intelligence at the University of Kentucky. It is designed to provide accessible and easy-to-use machine learning algorithms that can be applied to a variety of tabular data. CLASSify provides the ability to train and evaluate multiple models on your own data. These models can analyze a variety of data features and produce class label predictions on new data. It also includes additional features such as synthetic data generation, visualizations, and explainability scores. CLASSify is open-source and free to use for your own research purposes. It is available on a per-user basis. If you don't have access, submit a request here.

When to Use CLASSify

The machine learning models provided by CLASSify are suitable for both binary and multiclass classification problems with tabular data.

'Tabular' data means that the data takes the form of a table, with rows representing individual data points/observations and columns representing features/variables. If your dataset can be represented as a table in a .csv or .xlsx file, then it is the right format for CLASSify.

Additionally, the dataset must be suitable for classification. Classification problems are those in which each data observation belongs to a classification which you are interested in predicting. For instance, you may have patient observations containing demographic and medical data, and you hope to train a model that can diagnose whether a patient has a certain condition. The data you provide must contain a column that indicates each patient's diagnosis. This column would be the 'class' because it is the variable you are interested in predicting for new patients. To train a model to do this prediction, it must be provided with already-labeled data so that it can learn from the examples. If you have collected data but do not already know the class labels for them, then classification models cannot be trained.

The above example represents a **binary** classification problem, where there are only two distinct classes (positive diagnosis or negative diagnosis). CLASSify also supports **multiclass** classification problems. This is a situation in which you have more than two distinct classes. For example, if there is a disease with four distinct stages and you are interested in classifying patients by what stage of the disease they are in, this would be multiclass, and each stage of the disease would be its own class label.

Note that binary and multiclass classification is different from other machine learning methods, such as regression or time series forecasting. Classification implies **distinct** classes that you hope to analyze and predict with. Predicting the heart rate of a patient, for example, would not be a classification problem, because the heart rate is a continuous numerical measurement rather than a discrete classification. However, continuous measurements can be transformed into discrete classifications using thresholds. For example, the heart rate could be divided into two classes, 'Low Heart Rate' and 'High Heart Rate', and these classes could be used instead of the original heart rate measurement.

CLASSify is meant to be a self-service tool, providing users with a great deal of customizability when running jobs. Therefore, when uploading a dataset, you will choose which models to train and what types of analysis to perform. CLASSify can provide an open playground for experimentation with different types of models and analysis, and it allows for easy comparison between runs and understandable visualizations to explain model performance. However, if you need quick and effective results, it is helpful to go in with ideas about your goals and what you expect to see in results. When you upload a dataset and submit a training job, the process of training models and generating results is automated, so our team at CAAI is not directly involved with the jobs' completion and do not perform analysis on the data ourselves. However, always feel free to reach out with any questions about CLASSify.

How to Use CLASSify

Logging In

CLASSify can be found by going to https://data.ai.uky.edu/. On this page, click the CLASSify button to access the CLASSify interface. This will take you to the CILogon page. For the identity provider, choose your institution (such as University of Kentucky), and then you can login. If your account has been created already, this will bring you to the CLASSify interface. If you reach an unauthorized page, please reach out.

What to Upload

Any uploaded dataset must be in the .csv file format. If your dataset is in the .xlsx format, this can be easily converted to a .csv through Excel. There is a 500 MB limit on the size of the dataset you upload. If you have a separate dataset you wish to use as a testset, you will be able to upload that at a later stage. The first dataset you upload should be the training dataset.

The only requirement for an uploaded dataset is that one of the columns must have the name 'class'. This column represents the classification you want to focus on, and it must be named this way so that it can be identified as the column of interest. If you only have two distinct class labels (binary classification), these labels should be represented with 1/0 or True/False. Do not use custom class labels, such as 'Positive Diagnosis'/'Negative Diagnosis', as these will not be recognized by the system. If you have more than two distinct labels (multiclass classification), these should be represented as integers (0, 1, 2…). The order of the integers and how they are assigned to each class does not matter. These classes should still be represented with a single column labeled 'class'. Currently, CLASSify does not support multilabel classification, where a single observation may belong to multiple classes. Each row of your data should only belong to a single class label.

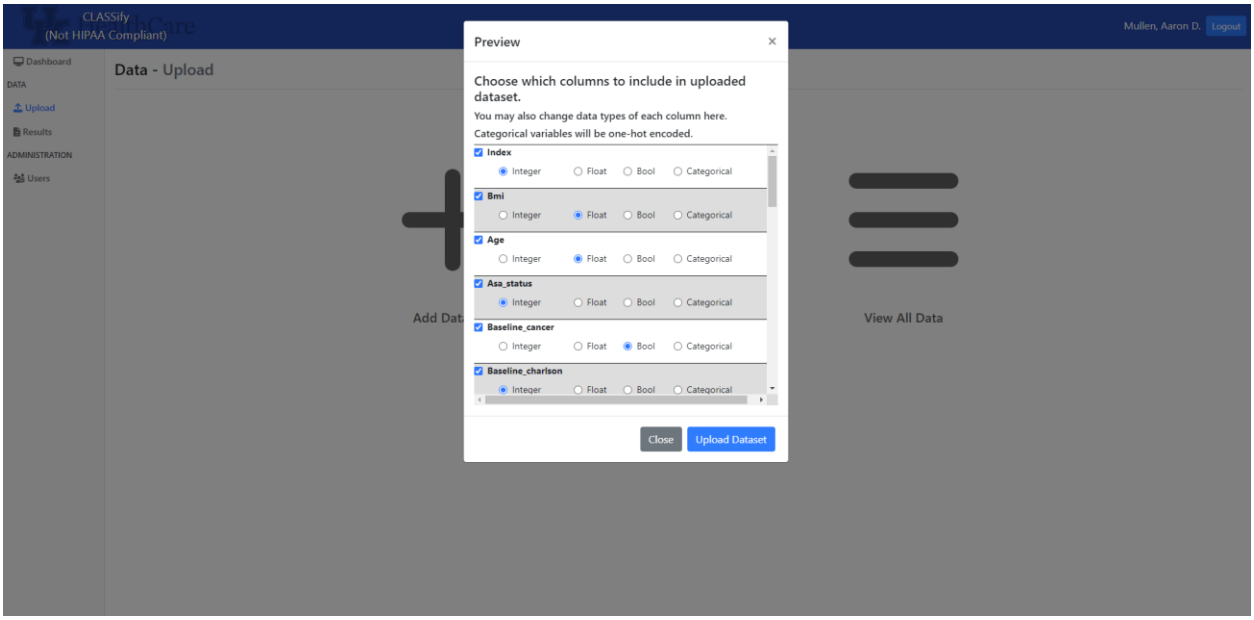
You can name your dataset anything you wish, but it cannot have the same name as another dataset that you have already uploaded. If this is the case, you will need to either rename the new file or delete the old dataset from the CLASSify system.

There are no other requirements for dataset upload, but depending on your data, there may be other pre-processing steps you wish to take before uploading. For example, if your dataset has missing values, you should consider how you want these to be treated. CLASSify provides options to synthetically generate and fill missing values, or simply drop rows containing missing values. However, if you wish to use a different encoding method, such as a constant fill value for missing data, you will want to perform that processing before uploading your dataset.

How to Upload

From the home page, click the ‘Add Data File’ button to upload a new dataset.

This will bring up a modal that allows you to choose a .csv file from your file system. Once chosen, hit the Preview button.



After Previewing, you will have a list of every feature of your dataset. In this modal, you can do two things: 1) choose which columns to include in your dataset, and 2) change the data types of any columns.

You can uncheck the box next to any column (except for the class column) to remove it from your dataset upload. This is helpful if you have columns you know are irrelevant to the class label. For example, if you have a column 'Patient ID' that contains a unique identifier for each row, this column should likely be dropped, as the value of a Patient ID is likely irrelevant for predicting the class label. For this reason, any column labeled 'Index' will automatically be dropped, as the inclusion of unique identifiers like this can confuse the model and lead to worse or misleading performance. In general, **models will perform better if the dataset has only a few relevant columns** rather than many columns of varying relevancy, so it is best to drop any columns that won't provide much information about the class label to the model. If you're not sure which of your columns are relevant, you can run an initial training job with all columns and analyze CLASSify's explainability score results, which provide information about each feature's impact on the model. This can be used to determine the most relevant columns, and you can then re-run the data with only those features.

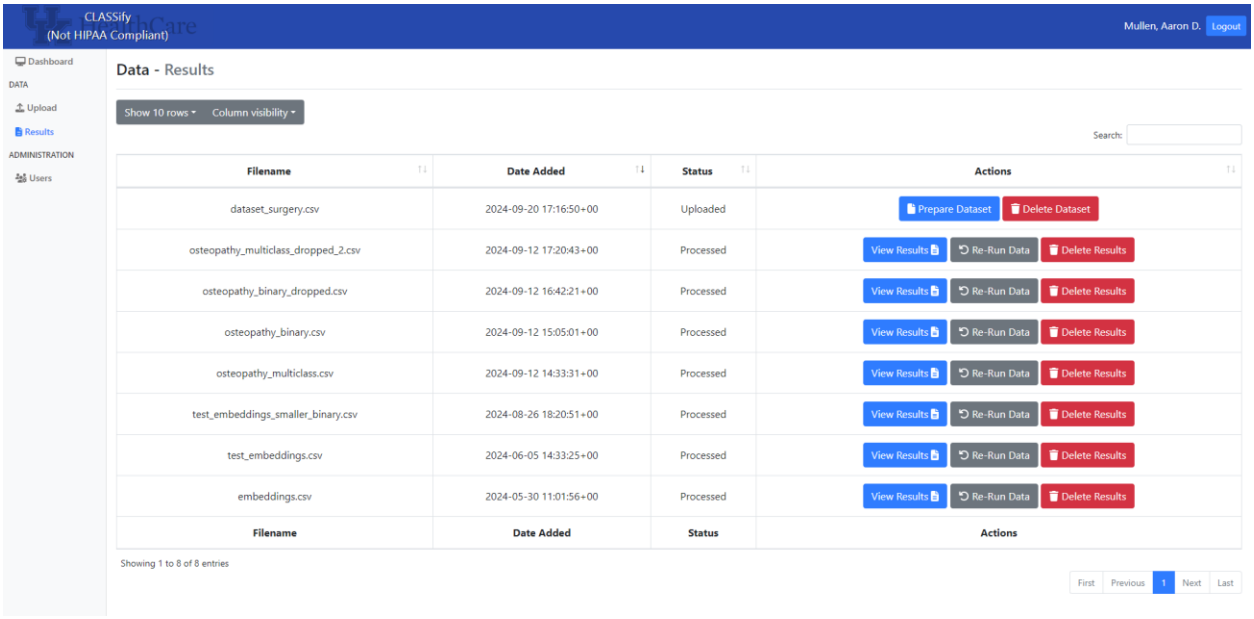
You can also change the data type of any column, although you likely will not have to, as the data type can be automatically detected by the system. However, it can be useful to double check and ensure all columns' types look correct. As a brief guide, **Integer** is a whole number, **Float** is a decimal number, **Bool** is a True/False value, and **Categorical** contains distinct string categories.

Categorical variables will be automatically converted using a process called one-hot encoding. This is necessary because most machine learning models can only work with numerical or binary data. If you have a categorical variable such as Race, one-hot encoding will convert each unique category into its own binary column. For instance, this would create a new binary column each for White, Black, Asian, etc., where each column contains only values of 1 or 0, depending on whether a given row belongs to that race. With this method, all of the information from the categorical column is retained, it is just split across multiple new columns.

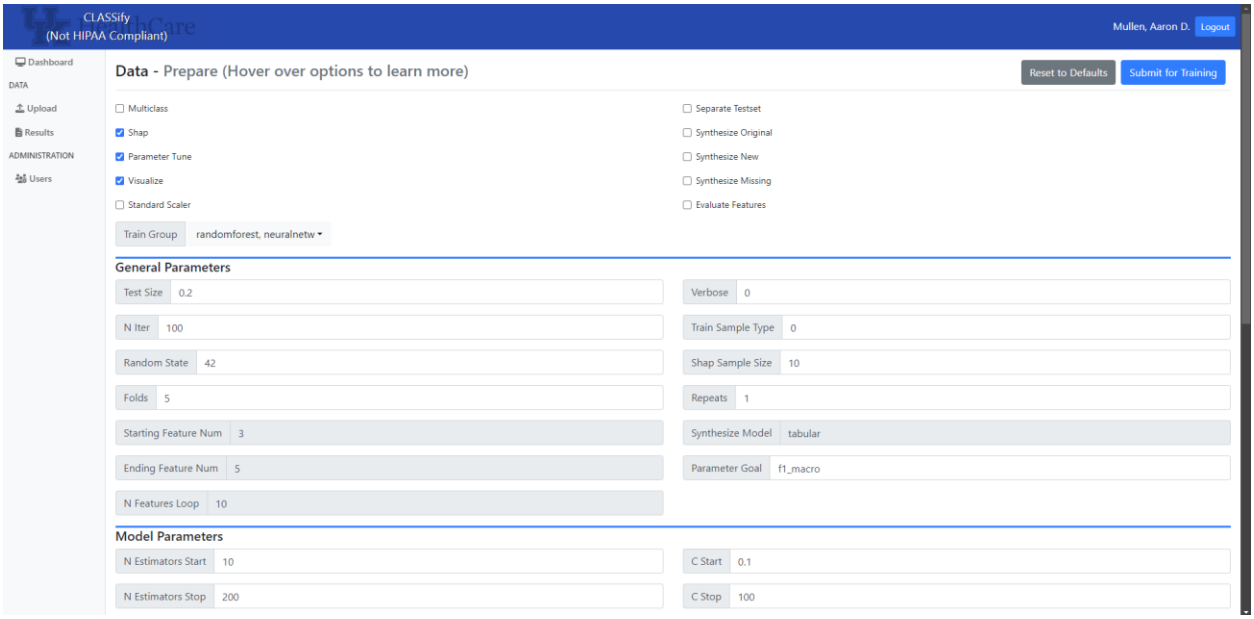
Because of this method, if you have any categorical columns with many categories, it may be best to drop them rather than encoding. For example, if you have a column indicating a patient's county of residence, each distinct county will become its own column, creating a large number of additional columns that may not be useful.

Once you are satisfied with the included columns, click 'Upload Dataset', and then 'View Uploaded Data'.

Starting a Training Job



On the results page, you can view any previous jobs you've run. You can delete old jobs, and you can re-run them to experiment with different parameter settings and models. Doing this will duplicate the original, so you can easily compare between runs. Any datasets you've uploaded but haven't run training yet will show up here too, and you can click 'Prepare Dataset' to choose the training job options.



On the preparation page, there are a variety of options for customizing the training job before you submit it. It may look overwhelming at first, but many of these options are best

left at their default values. If you want more information about any of these options, you can hover over them with the cursor on the site to see more information.

Many of the most important options are given in the checkboxes at the top.

- **Multiclass**- whether the dataset is multiclass (unchecked for binary).
- **Shap**- SHAP scores are the explainability measurements for each feature of your data. Keep this on if you want to analyze which features of your dataset are most relevant to the class label. Switch it off if you want to prioritize minimizing training time, or if feature explainability is unimportant.
- **Parameter Tune**- parameter tuning will retrain each model for many iterations, altering the models' parameters each time to determine the most optimal model settings. This will increase the training time but will likely improve the final results.
- **Visualize**- whether to create explanatory visualizations for the results.
- **Standard Scaler**- alternate data scaling method. Leave unchecked unless you have a reason not to.
- **Separate Testset**- checking this will allow you to upload a separate .csv file to use as a testset for model evaluation. Ensure that this testset has the same columns and format as your initial training set.
- **Synthesize Original**- generate synthetic data to balance the class labels. Imbalanced class labels can worsen performance as the models tend to focus on the majority class. Synthetic balancing can allow the model to treat all classes equally, which can improve performance. Leave unchecked if your class labels are already relatively balanced or you want to focus on real data only.
- **Synthesize New**- train models on an entirely new, synthetically generated dataset. This synthetic dataset can be downloaded later for future use.
- **Synthesize Missing**- fill in missing values with synthetic imputing algorithms.
    - All synthetic data options allow you to upload a metadata file in JSON format containing column types. This is not required, as column types can be automatically inferred from the data.
- **Evaluate Features**- train and evaluate multiple versions of each model using different feature combinations. This can help to determine which features are most relevant for your class label. Due to exhaustive combinations of features, it is not recommended with many features due to long training times.

The dropdown allows you to choose which models you want to train. If experimenting or unfamiliar, leave all models checked, and then compare results to determine which models are best for your dataset. If training time is a concern, the Neural Network and TabPFN models are the most complex and take the longest time to train.

The section titled General Parameters includes further training options that apply to all models, such as the train/test split ratio and the number of iterations of parameter tuning.

The last section contains parameters for individual models. These allow you to customize the parameter ranges that are used in parameter tuning. Irrelevant options will be grayed out depending on which models you've chosen. It is recommended to only change these parameters if you understand what they do.

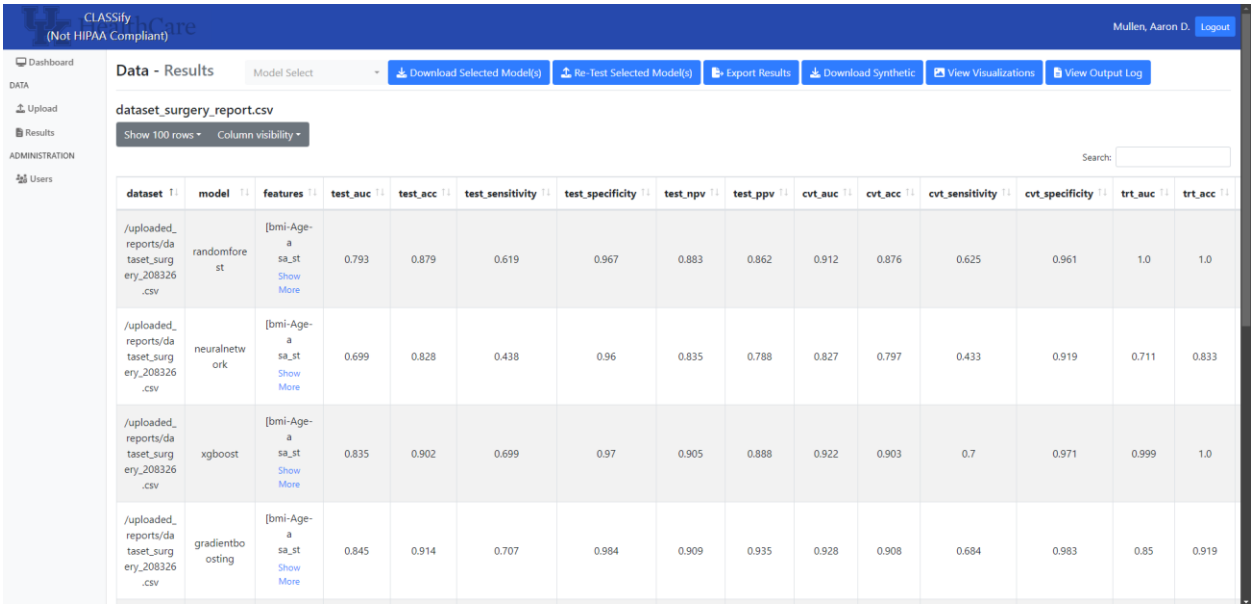
When all options have been chosen, click Submit for Training at the top of the page to submit the training job. Your job will now begin, using the dataset you uploaded and options you selected. The training and evaluation of each model is run on our DGX Cluster, a high-performance computing center housed on campus. No processing is performed on the machine you use to interact with the website.

You can monitor training progress on the results page. The Status column will update after each chosen model is successfully trained. Therefore, if you chose to train all 10 models, after the third model has successfully completed training, the Status will say '3/10 processed'.

The time for a job to complete will depend on your dataset size, number of features, and the training options you chose. With a small dataset and no additional processes, such as parameter tuning or SHAP score calculation, all 10 models can be trained in a matter of a couple minutes. If additional processes are included or the dataset is large, this will increase, and each model may take several minutes on its own, with the more complex models (Neural Network, TabPFN) possibly taking several hours to train. Choose your parameter options based on your time frame. You do not need to keep the site open for the models to train, so you can always check back in later or the next day.

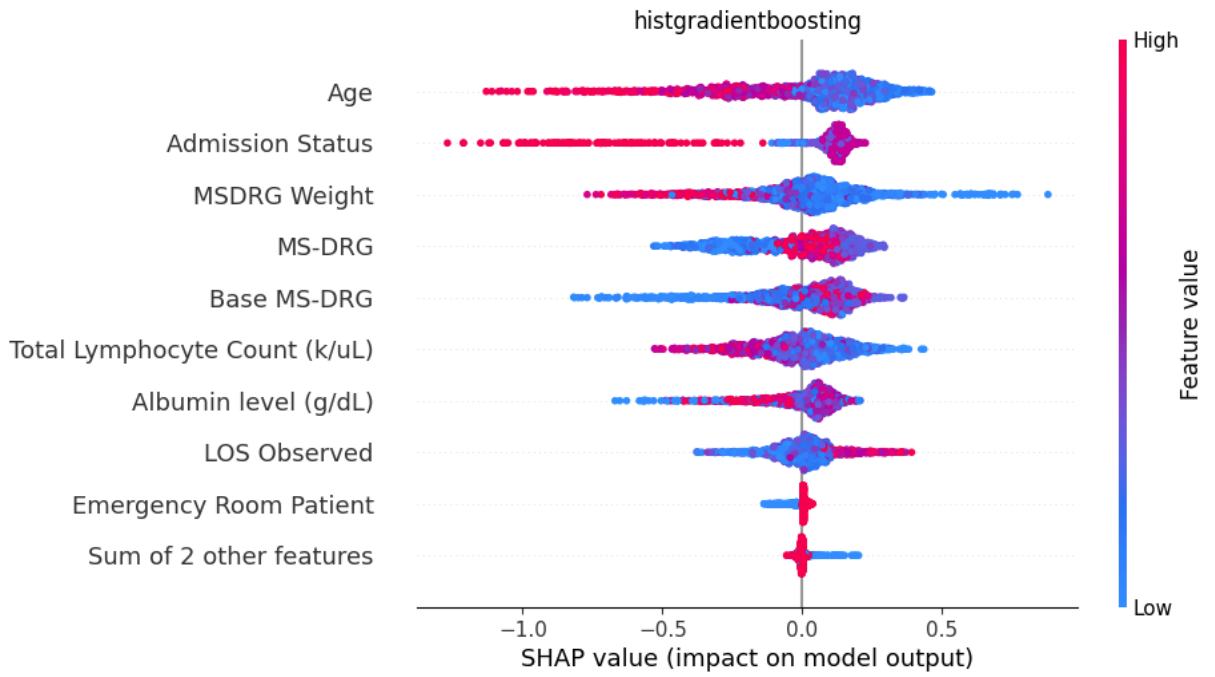
Viewing Results

When your job has completed, click on View Results.



You will see a page like this, with a large table showing a variety of performance metrics. This includes accuracy, Area Under the ROC Curve (AUC), sensitivity, specificity, negative predictive value (NPV), and positive predictive value (PPV). Many of these metrics are given for both the testset, a cross-validation set, and the original training set. They can be compared between models or feature groups with this table, but the View Visualizations button will show more explainable charts and graphs to compare performance.

These charts include heatmaps to compare performance metrics, bar graphs to showcase True/False Positive and True/False Negative rates for each model, and comparisons between performance on the test, cross-validation, and training sets. ROC curves will be generated for each model as well. If the SHAP option was chosen on the parameter page, you can view SHAP diagrams that can explain feature importances.



This is an example of a SHAP diagram. It lists the most important features ranked on the left side. To interpret the graph, you can look at each feature's points. For the Age feature at the top, there are more blue points on the right side, and more red points on the left side. The scale on the right shows that red values indicate high feature values, while blue indicates low feature values. The right side of the y-axis indicates positive impact on the model's classifications (influenced to predict 1 instead of 0), while the left side indicates a negative impact (influenced to predict 0 instead of 1). So, the top row can be interpreted as showing that low values of age are associated with a positive classification with this model and dataset, while high values of age are associated with a negative classification.

These diagrams can be helpful for understanding how the features impact the model's output. Different diagrams will be generated for each model, so they can be compared to see what they have in common. Note that by default, some model's graphs will appear less filled-in with points. This is because the SHAP algorithm is optimized to work well with some model architectures but not others. For the models it is not optimized for, a general SHAP explainer can be used, but this algorithm takes a long time and must run on each individual data point. Therefore, it is run only on a small subset of points. The size of this subset can be customized on the parameter page if runtime is less important.

Outside of the visualizations page, there are other actions you can perform when viewing results. You can download the table of performance metrics, as well as any synthetic data generated. You can also download any of the models you have trained as .joblib files, which can be easily read into your own Python code for further analysis if necessary.

If you want to re-test any of the models you've trained with new data, you can do that from this page as well. Select the models you want from the dropdown and click Re-Test Selected Models. This allows you to upload a new dataset. If this dataset already has a class column, it will produce new performance metrics for each model and download those. If the dataset you upload does not have a class column, each model you choose will generate its classification predictions for each data observation. This allows you to utilize the models you've trained through the web interface, analyzing new data and generating class label predictions.

Finally, you can view an output log on this page. This is helpful if your job ran into any errors. Sometimes, errors will be because of incorrect parameter choices, such as failing to check the Multiclass option when your dataset has more than two classes. CLASSify is not a completed system, so other bugs may arise that hinder the training of your models. If you encounter any errors that you do not understand or seem out of your control, please reach out to us so we can help.

Other Notes

CLASSify is a work-in-progress, and updates are continually made to the site, so available features are subject to change.

Security

We hope to soon achieve HIPAA compliance with CLASSify, but for now, do not upload any HIPAA protected data to CLASSify.

Contact Us

If you have any questions when using CLASSify, reach out to ai@uky.edu .

Citation

If writing a publication containing research performed with CLASSify, please cite the following paper: https://arxiv.org/abs/2310.03618