




A European Cancer Image Platform Linked to Biological and Health Data for Next-Generation Artificial Intelligence and Precision Medicine in Oncology

## Deliverable D5.3:

# Integrative machine learning toolbox

Reference	D5.3_ EuCanImage_UB_v1
Lead Beneficiary	UB
Author(s)	Kaisar Kushibar, Socayna Jouide, Martijn Starmans
Dissemination level	Public
Type	Report
Official Delivery Date	31 October 2022
Date of validation of the WP leader	31 October 2022
Date of validation by the Project Coordinator	31 October 2022
Project Coordinator Signature	

EuCanImage is funded by the European Union's H2020 Framework Under Grant Agreement No 952103

## Version log

Issue Date	Version	Involved	Comments
October 10, 2022	V0.1	Socayna Jouide	Initial draft
October 24, 2022	V0.2	Kaisar Kushibar	Structure and more content
October 28, 2022	V0.3	Kaisar Kushibar	Liver cancer classification results using the VRE-MLToolbox
October 31, 2022	V1	Anais Emelie & Karim Lekadir	Revised and corrected final version.

## Executive Summary

Deliverable D5.3 presents the Machine learning toolbox for integrated predictive modelling, which is available at the EuCanImage Virtual Research Environment (VRE) platform. In this deliverable, detailed functionality and specifics of the toolbox are provided. The design of the toolbox has focused on user friendliness and user experience to ensure the usability of the tool by users with no prior experience in machine learning, aiming for the democratisation of machine learning tools especially for clinicians.

The document is divided into the following sections: (1) the design of the toolbox, (3) its implementation, (3) examples of machine learning workflows.

## Table of Contents

<b>Version log</b>	2
Executive Summary	2
<b>1 Introduction</b>	4
<b>2 Machine learning toolbox</b>	5
2.1 Tool overview	5
2.1.1 Input files	8
2.2 ML mixer deck	8
2.3 MLToolbox output	12
<b>3 Application to liver cancer classification</b>	13
3.1 Dataset	13
3.2 ML mixer deck settings	14
3.3 Results	14
3.3.1 Classification performance	14
3.3.2 Statistical tests	15
3.3.3 Feature importance	15
3.3.4 Feature decomposition	16
<b>4 Conclusions</b>	17
<b>5 References</b>	17

## Acronyms

Name	Abbreviation
ML	Machine Learning
VRE	Virtual Research Environment
SVM	Support Vector Machine
RF	Random Forest
LR	Linear Regression
LDA	Linear Discriminant Analysis
QDA	Quadratic Discriminant Analysis
Gaussian NB	Gaussian Naive Bayes
XGB (XGBoost)	Extreme Gradient Boosting
WORC	Workflow for Optimal Radiomics Classification

# 1 Introduction

Artificial Intelligence (AI) has become one of the leading research fields in oncology with the aim to assist radiologists and medical doctors with decision support systems. Due to the large number of modalities and organs it is crucial to have a universal computational framework where a commonly used and standardised basis for any AI tool is already developed. Radiomics concerns the use of quantitative medical image features to predict clinically relevant outcomes. The MLToolbox will enable researchers and doctors to easily build pipelines without prior expert knowledge. The MLToolbox is built on top of the existing WORC framework [1] and integrated in the VRE.

In this document, we demonstrate the general pipeline that includes five steps:

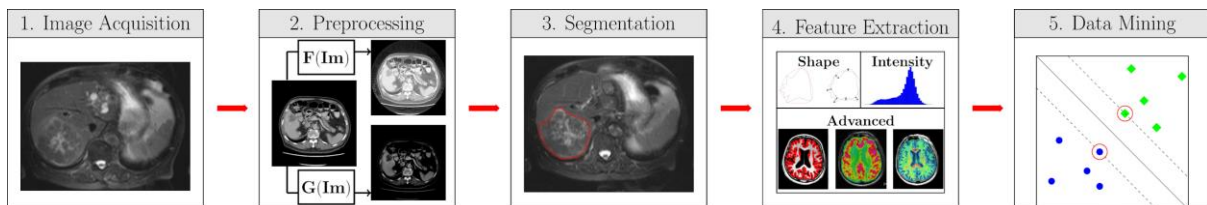


Figure 1: The toolbox includes, after (1) image acquisition, (2) preprocessing, (3) segmentation, (4) feature extraction and (5) data mining. (2) Preprocessing is anything that is done to the image before the feature extraction. This may include normalisation, cropping, masking, and registration. (5) Data mining or machine learning is defined as anything that is used to go from features to the actual label or outcome. This may include feature selection, feature imputation, feature scaling, dimensionality reduction, and oversampling.

The existing WORC framework was built using FASTR<sup>1</sup> as a node in order to standardise radiomics in components to create a modular workflow. To illustrate the principles of FASTR, one of the simpler workflows created is shown in Figure 2.

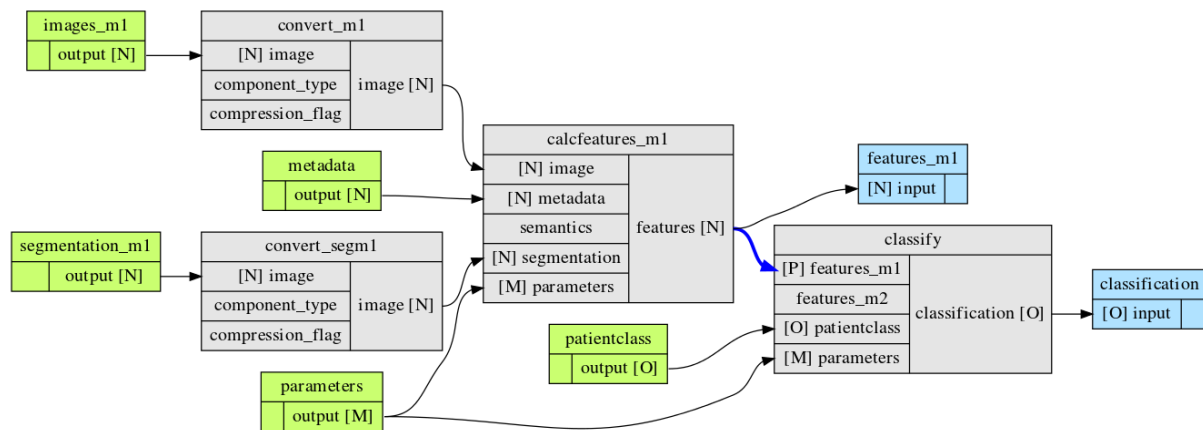


Figure 2: The radiomics workflow is split in nodes. In each node, the inputs, outputs and thereby the task is fixed. However, the actual algorithm that is used within a node is not fixed. For example, in a feature extraction node, the inputs are an image, possibly a segmentation and parameters, from which features are extracted, which are thereby the output.

When starting a new radiomics study, e.g. trying to find an imaging biomarker on a new dataset, you would normally have to design a workflow yourself. Not only would you have

<sup>1</sup> <https://fastr.readthedocs.io/en/stable/> - FASTR is a framework that helps create workflows of different tools.

to determine which methods you are going to use, but also which parameters for all methods. Many studies have shown that these are often not independent: the performance of many methods highly depends on the parameters used. Thus, we need to simultaneously look for the best combination of these.

As these parameters are set before the actual learning step in the machine learning part, we refer to these as hyperparameters. As the performance of the machine learning algorithm may depend on the previous steps (e.g. preprocessing, feature extraction, and all other steps in the data mining node), we decided to include all parameters from all nodes as hyperparameters.

Optimization is done through a variant of combined algorithm selection and hyperparameter (CASH) optimization problems. Currently, the MLToolbox uses WORCs optimization pipeline to solve this problem as displayed in Figure 3.

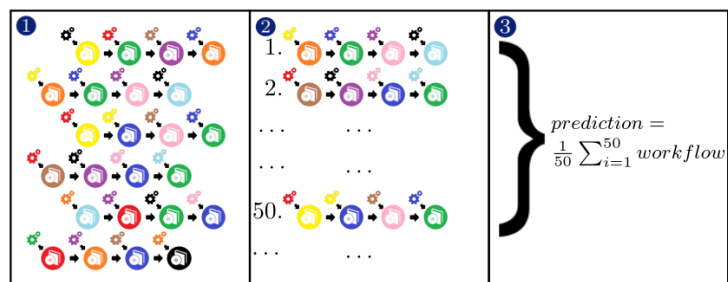


Figure 3: The optimization process: (1) Generate a large number of different pipelines (2) Execute them and rank them (3) Create an ensemble of the 50 best performing pipelines.

## 2 Machine learning toolbox

### 2.1 Tool overview

Overall, the workflow of the MLToolbox will consist of a search space depicted in Figure 4. The search space consists of various sequential sets of algorithms, where each algorithm may include various hyperparameters, as indicated by the leaves in the trees. An example of a workflow, i.e., a specific combination of algorithms and parameters, is indicated by the grey nodes. The search workflow is integrated in the VRE MLToolbox.

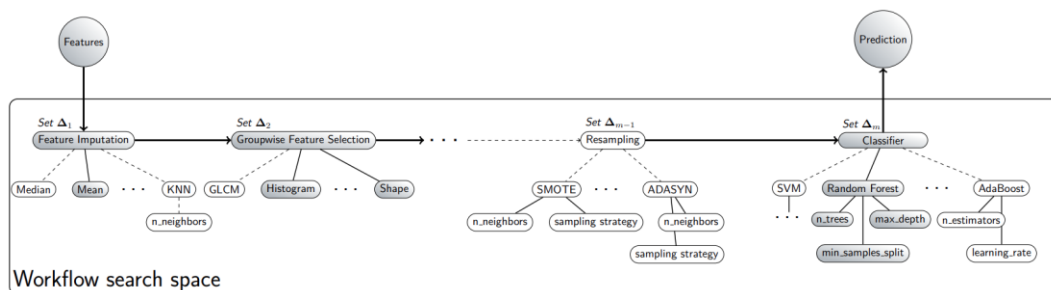


Figure 4: Schematic overview of the workflow search space in the MLToolbox. Abbreviations: AdaBoost: adaptive boosting; ADASYN: adaptive synthetic sampling; KNN: k-nearest neighbour; GLCM: grey level co-occurrence matrix; SMOTE: synthetic minority oversampling technique; SVM: support vector machine. Figure courtesy of Starmans et al [9], EuCanImage partner.

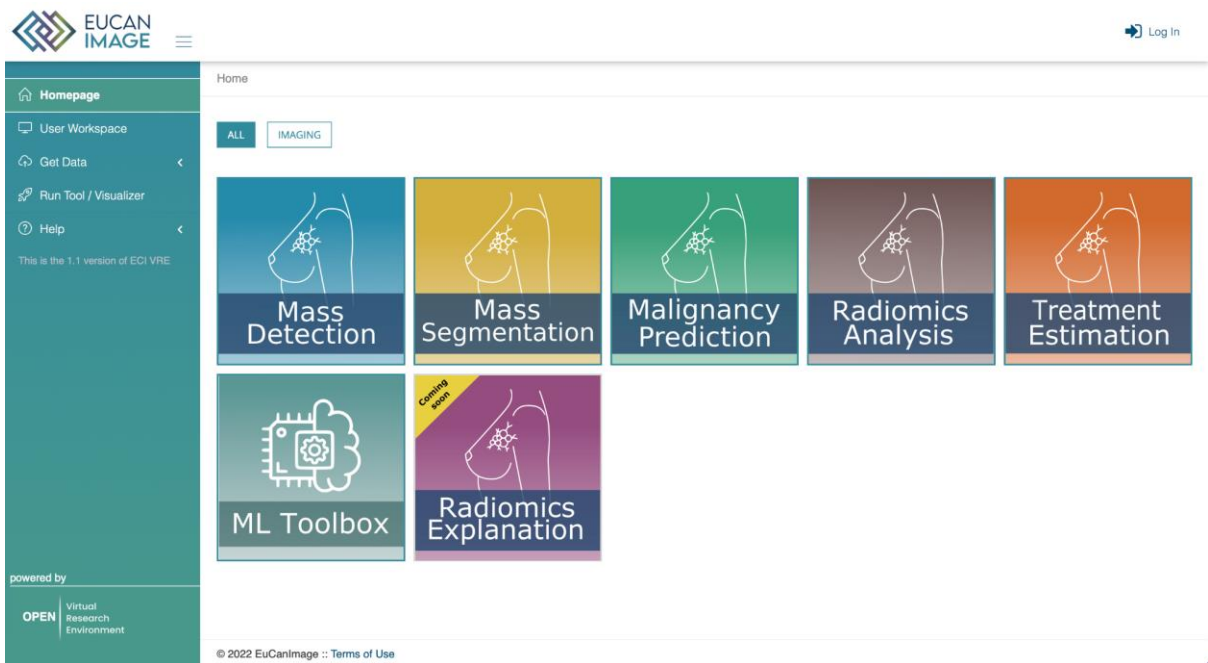


Figure 5: Homepage of the VRE. In this page, the available tools are listed and can be assessed. Other available or under development tools can also be viewed with the coming soon tag.

Figure 5 provides an overview of the current state of the VRE homepage and the tools included. Each tool can be accessed by (i) the homepage (Figure 5), (ii) the “User workspace” tab (Figure 6) if the appropriate file types are chosen, i.e. the file types that each tool accepts as input, and (iii) the “Run Tool/Visualizer” tab (<https://vre.eucanimage.eu/vre/launch/>). Each tool has its own front page (Figure 7), where it can be configured and when the user provides the adequate input and presses the Compute button, the page will be redirected to the “User Workspace”. The status of the job changes from pending to running to finishing. When the job finishes, the result(s) appears under the given experiment name and can be downloaded from the Workspace (Figure 6). The next section dives into the internal structures of each tool and the model training procedures that are used to achieve robust automated annotation tools.

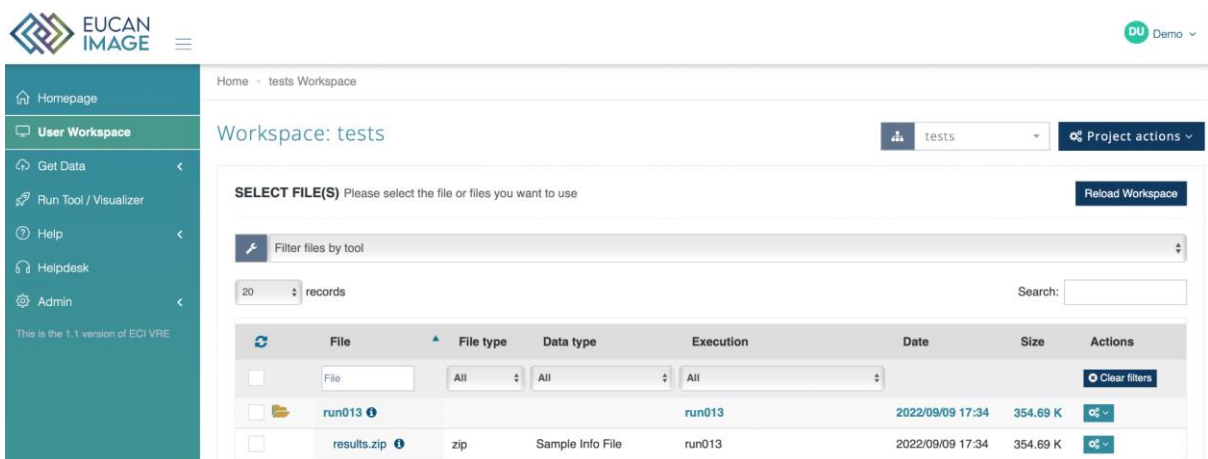


Figure 6: "User workspace" tab. In this page the user can find the previously uploaded data and the results of the experiments. In addition, these files can be selected to see available tools to process them. Lastly, the running jobs can be monitored while they change status from pending to running to finishing.

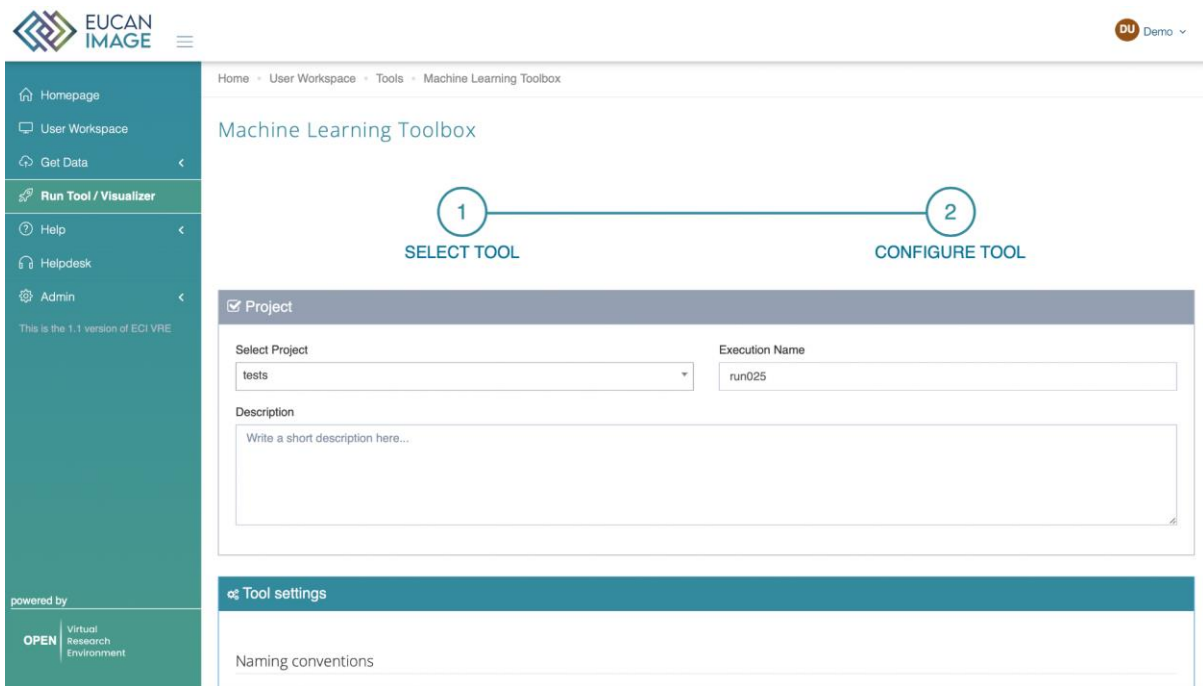


Figure 7: Main page and configuration settings for the MLToolbox.

Figure 10 shows the front page of the Machine Learning Toolbox, including the inputs and arguments that can be selected in the ML mixer deck.

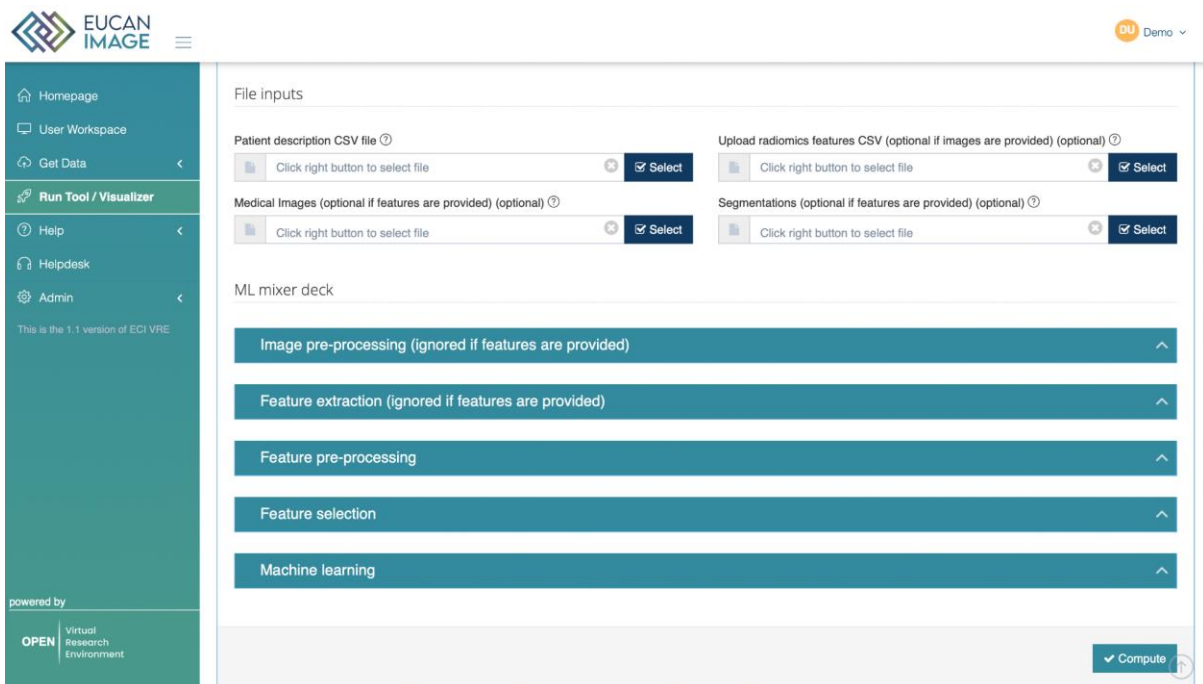


Figure 10: Main page and configuration settings for the MLToolbox.

### 2.1.1 Input files

To perform the analyses the user must provide some parameters. The following list describes each of the inputs, as well as some requirements they must comply in order to work:

1. **Patient description CSV file:** tabular data, in .csv format. Tabular data, where each row must correspond to a patient and each column to a specific feature. Any clinical tabular data can be included for analysis, such as genomics, demographics, environmental factors, etc. In the case of uploading the tabular data, the user must ensure the data complies with the structure of rows and columns presented below. **The first column MUST be named 'Patient'** so that the MLToolbox recognises the patient IDs.
2. **Radiomics features:** If you already computed your features, e.g. from a previous run, you can directly supply the features instead of the images and segmentations and skip the feature computation step. These should be stored in .hdf5 files matching the MLToolbox output format (see Section 2.3). Conversion from an arbitrary CSV format to the MLToolbox format is a work in progress.
3. **Images and Segmentations:** The file names for these two inputs should follow the naming convention shown in Figure 11.

Naming conventions

Patient description file must have the first column title as 'Patient' and each row must have the same ID as the corresponding images and segmentation masks.

Images and segmentation mask file names must have the following structure: "PatientID\_image\_\*.nii.gz" and "PatientID\_mask\_\*.nii.gz". "\*" is a wildcard that can be added in case the same patient has two or more images and masks.

	A	B	C	D	E	F
1	Patient	imaginary_label_1	imaginary_label_2	Hospital	Age	complement_label_1
2	HN1004	0	0	1	99	1
3	HN1077	0	0	0	29	1
4	HN1088	0	0	1	25	1
5	HN1146	1	1	0	77	0
6	HN1159	1	1	1	5	0
7	HN1192	1	1	1	47	0
8	HN1259	0	0	1	44	1
9	HN1260	1	1	0	22	0
10	HN1323	1	1	1	52	0
11	HN1331	0	0	0	71	1

Figure 11: Naming conventions for the input files. Images and their corresponding segmentation masks must have the same prefix as in the first column of the CSV file.

In conclusion, the minimal input for a Radiomics pipeline consists of either images plus segmentations, or features, plus a label file. If you supply images and segmentations, features will be computed within the segmentations on the images. They are read out using SimpleITK<sup>2</sup>, which supports various image formats such as DICOM, NIFTI, TIFF, NRRD and MHD.

## 2.2 ML mixer deck

The most important part of the MLToolbox is to be able to modify the search space by choosing what steps and what algorithms should be used in the ML workflow. The ML mixer deck is designed for this purpose. It contains sections as shown in Figure 12.

<sup>2</sup> <https://simpleitk.org>





Figure 12: Machine Learning Mixer Deck sections that allows users to specify custom ML Workflows. Once all the options are set, MLToolbox runs all the possible combinations and hyperparameter tuning, then returns a compiled output with the results.

Each section in the ML mixer deck contains specific configurations that users can select and tune. Below are the descriptions of each section.

**[General]** It contains the structure as shown in Figure 13, which is responsible for the broad settings for the ML workflow. *Segmentix* - whether segmentation preprocessing should be performed or not. *Save temp folder* - Determines whether after every cross-validation iteration the result will be saved in addition to the result after all iterations. Especially useful for debugging. *Assume same image and mask metadata* - Make the assumption that the image and mask have the same metadata. If True and there is a mismatch, metadata from the image will be copied to the mask. *ComBat* - Whether to use ComBat feature harmonisation on the entire dataset, i.e. not in a train-test setting<sup>3</sup>. *Image type* - used in image normalisation process that can be different depending on the modality. Currently contains: CT, MRI, PET, ADC, MG, DWI, US, Thermography.

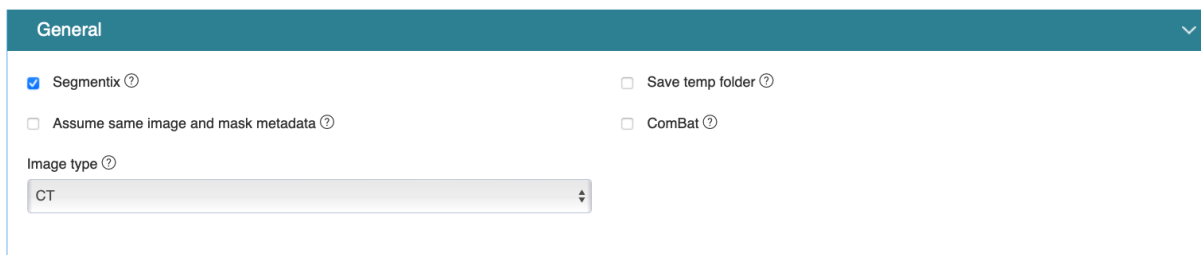


Figure 13. Options for the General section.

**[Image pre-processing]** Contains settings for the commonly used image pre-processing steps such as *Normalisation* (*z\_score* or *minmed*), *Clipping* with the defined intensity *clipping range*, *Resampling* of the pixel/voxel spacings to a specified *resampling spacing*, and *bias field correction*. The UI in the VRE looks as shown in Figure 14. This step is ignored if the features are provided instead of the images.

<sup>3</sup> <https://github.com/Jfortin1/ComBatHarmonization>

Figure 14. Options for the Image pre-processing section. This section is ignored if the features are provided.

**[Feature extraction]** This step is also ignored if the features are provided instead of images. Users can select from a variety of features that correspond to intensity, texture, and shape. The UI is illustrated in Figure 15,

Figure 15. Options for the types of features to be extracted from the images. This section is ignored if the features are provided.

**[Feature harmonisation]** Contains a single field (Figure 16) where the variable name should be specified, which will be corrected for by ComBat. The variable name must be one of the columns in the CSV file provided by the user. For example, "Hospital" - which is a common variable to correct for due to the inter-centre imaging protocol variabilities.

Figure 16. Field to specify the variable that requires feature harmonisation.

**[Feature selection]** Currently, ML mixer deck contains five feature selection algorithms and all of them are applied during the optimisation process. Users can specify in the UI (Figure 17) the probability of each feature selection method to be used throughout the ML workflow. The number should be set to 0.0 if a particular feature selection method should be omitted. And it should be set to 1.0 to include it in every iteration of hyperparameter optimization.

Figure 17. Field to specify the variable that requires feature harmonisation.

**[Machine learning]** This section contains four different subsections related to the ML setup (Figure 18):

[Task and target specification] *Mode* - binary or multi-class classification. *Label to predict* - which variable in the CSV file we are trying to predict. *Probability of class balancing* - a number between 0 and 1 that specifies the chance of using oversampling, undersampling, SMOTE, etc. at each iteration of hyperparameter search.

[Classifiers] Defines which classifiers should be selected for the training. Currently, the options are SVM, RF, LR, LDA, QDA, Gaussian NB, AdaBoost, and XGB.

[Cross validation settings] Allows to specify the test size, type of split (random or leave one out), the number of cross-validation iterations, and fixing a seed to make the experiment reproducible.

[Hyperparameter optimisation settings] Allows to set the validation test size that will be used for model optimisation, number of cross validation splits, number of iterations, and finally, the number of jobs per core. The last option has limits in terms of the VRE computational resources. We recommend users to keep this setting as it is, but we let tuning this parameter as the number of jobs per core should be adjustable depending on the number of iterations.

Figure 18. UI for Machine Learning settings. Contains task and target specification, classifier selection, cross-validation settings, and hyperparameter optimisation settings.

### 2.3 MLToolbox output

Once the process of ML workflow is finished, VRE outputs a single compressed file - results.zip. The content of the zip file is shown in Figure 19. Each file is accompanied with a JSON file (filename.prov.json) that shows the history of the execution, which is handy to debug.

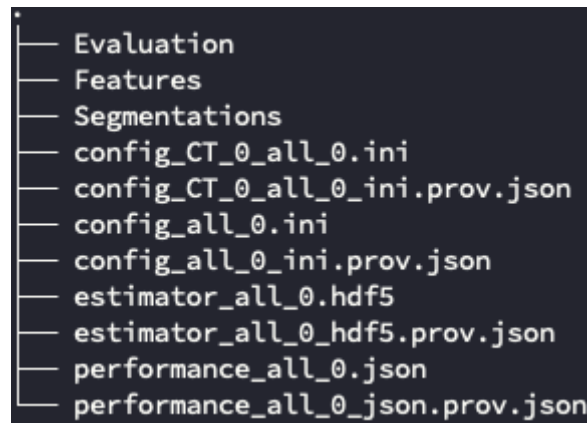


Figure 19. Contents of the results.zip produced by the MLToolbox.

The root directory contains three main files:

1) performance\_all\_{num}.json - where {num} corresponds to the class label. The performance file contains the mean and 95% confidence intervals of several performance metrics: (i) Area under the curve (AUC) of the receiver operating characteristic (ROC) curve; (ii) in a multiclass setting, the multiclass AUC from the [TADPOLE Challenge](#); (iii) Accuracy; (iv) Balanced Classification Accuracy (BCA), based on [2]; (v) F1-score; (vi) Sensitivity or recall or true positive rate; (vii) Specificity or true negative rate; (viii) Negative predictive value (NPV); (ix) Precision or Positive predictive value (PPV).

Moreover, rankings of the samples are included as the “Percentages” that shows how often a sample was classified correctly when that sample was in the test set. The number of times the sample was in the test set is also listed. Those samples that were always classified correctly or always classified incorrectly are also named, including their ground truth label. Additionally, the metric values for each train-test cross-validation iteration are presented, which are used to calculate the confidence intervals.

2) config\_{type}\_{num}.ini and config\_all\_{num}.ini - where {type} shows the modality and {num} is the class label. These are the configuration files that were generated from the user inputs specified in the ML mixer deck.

3) estimator\_all\_{num}.hdf5 - where {num} is the class label. This file contains a pandas dataframe with the best classifier hyperparameters.

Then, the three folders contain the following:

1) **Features** folder contains hdf files per image and has the following naming convention: features\_{featuretoolboxname}\_{image\_type}\_{num}\_{sample\_id}.hdf5. Each file contains a pandas series with feature labels, feature values, parameters, and image type.

2) **Segmentations** folder contains the corrected segmentation masks (by Segmentix) for each sample in NIFTI format.

3) **Evaluation** folder contains a number of files with different evaluation metrics and plots:

- a) Receiver Operating Characteristic (ROC) and Precision-Recall (PR) curves in csv, png, and tex formats.
- b) Univariate statistical testing of the features in csv and tex formats. It includes student t-test, Welch test, Wilcoxon test, Mann-Whitney U test.
- c) Overview of hyperparameters used in the top ranked models in csv format.
- d) Boxplot of the features in png format.
- e) Ranking patients from typical to atypical as determined by the model. Along with the csv file, for each patient a png is provided with the middle slice snapshot of image and segmentation. This allows one to scroll through the patients from typical to atypical to check for patterns.
- f) A barchart of how often certain features groups or feature selection groups were selected in the optimal methods. The plots are provided in png and tex.
- g) Decomposition of your feature space by Principal Component Analysis (PCA), Sparse PCA, Linear Kernel PCA, Polynomial Kernel PCA, Radial Basis Function Kernel PCA, and t-SNE. These decompositions can help to get insight into the dataset on how it can be separated.

### 3 Application to liver cancer classification

We tested the MLToolbox for liver cancer classification and here we present this use-case in more detail on how MLToolbox can help to perform radiomics based machine learning without any programming knowledge.

#### 3.1 Dataset

For the liver cancer dataset we used the WORC database, which is part of the EuCanImage project [3]. It consists of 186 patients with either a malignant (N = 94) or benign (N = 93) primary solid liver tumour. There are segmentation masks of the tumours for each patient annotated on T2-w MRI (see Figure 20). Patients have label 1 if the lesion was malignant, and label 0 if the lesion was benign. All the images with the corresponding segmentation masks are uploaded to the user workspace in the VRE along with the CSV file with patient ids and diagnosis.

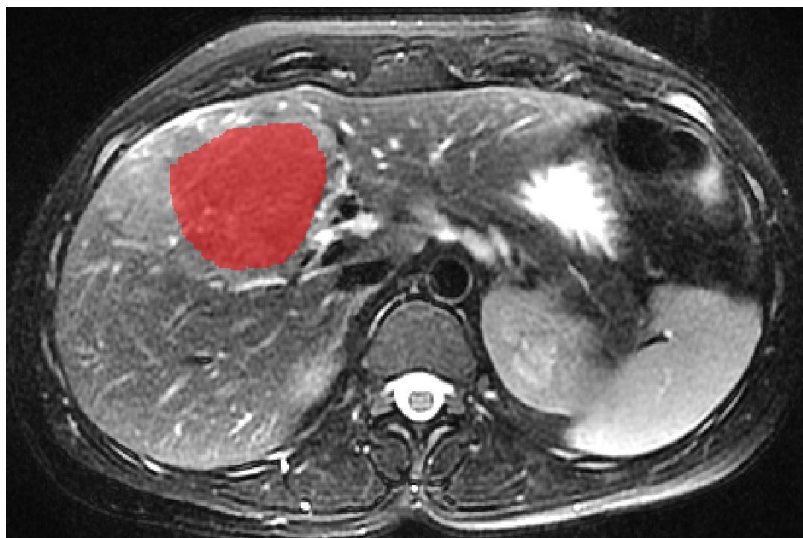


Figure 20. Example of a liver MRI with corresponding lesion segmentation mask overlaid on top of the tumour.

### 3.2 ML mixer deck settings

For the General configurations, we selected Segmentix correction and set the image type to MRI; the rest of the options are set to unchecked. In image processing we only selected Normalise with the z\_score method. Then in feature extraction, all types of features were selected except for Gabor filter, Vessel, Phase, LoG. Feature harmonisation was not selected because the patients were not grouped by hospital or imaging protocol differences. In feature selection, the probability of using the Variance based method was set to 1.0, and all the rest were set to 0.275. Then, we trained SVM, AdaBoost, and XGBoost for binary classification tasks. Cross validation and hyperparameter optimization settings were set to default. The computation time for finding the optimal parameters for three classifiers in 1000 iterations took approximately 7.5 hours in a 6 core CPU and 12 GB RAM virtual machine. The MLToolbox performs hyperparameter optimization in parallel, hence, having more CPU cores will improve the computational time.

### 3.3 Results

Here we present some of the interesting results of the MLToolbox for the sake of brevity. However, more detailed results can be assessed as described in Section 2.3.

#### 3.3.1 Classification performance

Table 1 and Figure 21 show the performance of classifying benign and malignant tumours from Liver MRI images. As can be seen, the user input in the mixer deck for the number of iterations allows defining the 95% confidence intervals (CI) for a multitude of metrics at a single operating point as well as the ROC and PR curves.

Table 1. Mean and 95% confidence intervals for various classification metrics over 10 iterations of cross-validation.

Evaluation metric	Mean (CI <sup>95</sup> <sub>low</sub> , CI <sup>95</sup> <sub>hi</sub> )
AUC	0.752 (0.696, 0.806)
Accuracy	0.689 (0.629, 0.749)
Balanced Classification Accuracy	0.689 (0.629, 0.749)
F1-score	0.687 (0.625, 0.749)
Negative Predictive Value	0.693 (0.605, 0.781)
Precision	0.693 (0.646, 0.739)
Sensitivity	0.678 (0.527, 0.830)
Specificity	0.700 (0.624, 0.775)

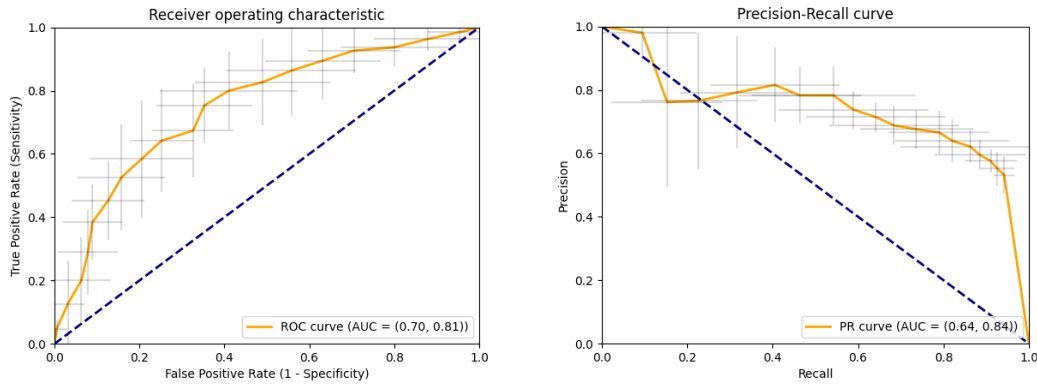


Figure 21. Receiver Operating Characteristic and Precision-Recall Curves with the 95% confidence intervals over 10 cross-validation runs.

### 3.3.2 Statistical tests

In Figure 22, we can see the statistical significance test on the selected features grouped by their type. By default, MLToolbox provides the plot for Mann-Whitney U test, however, a csv file with other supported tests is provided in the Evaluation folder.

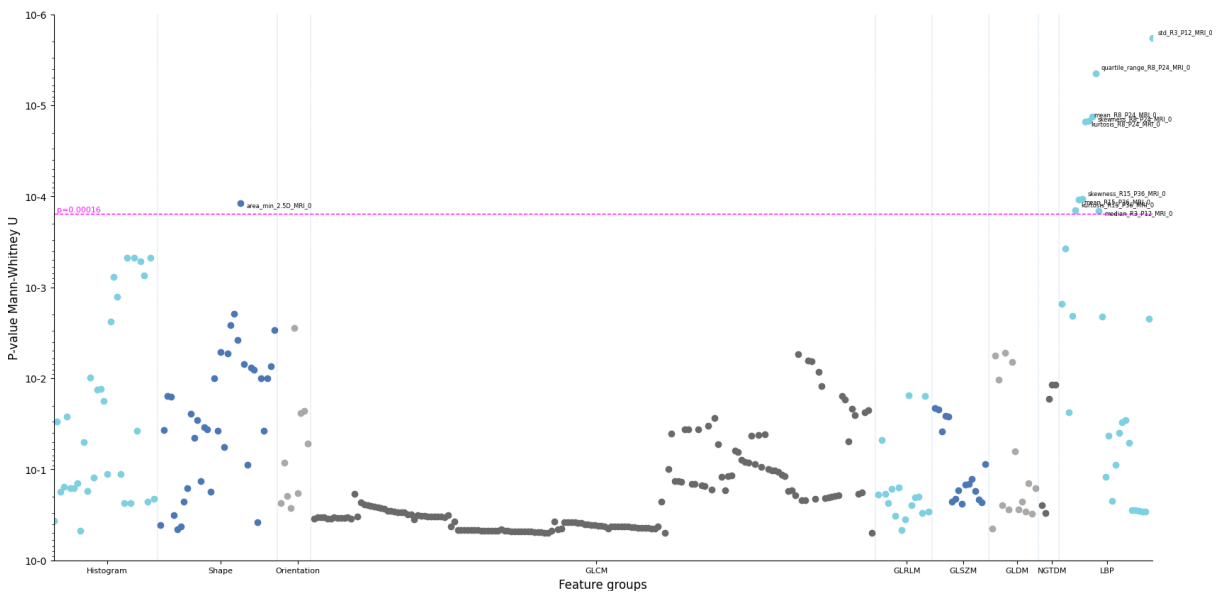


Figure 22. Mann-Whitney U test for all the selected feature groups.

### 3.3.3 Feature importance

Figure 23 illustrates the percentage of times each tunable component in the ML mixer deck has been used over all hyperparameter optimization iterations. This plot in particular useful for overall analysis of the optimization process and to draw conclusions on what features are more important than others.

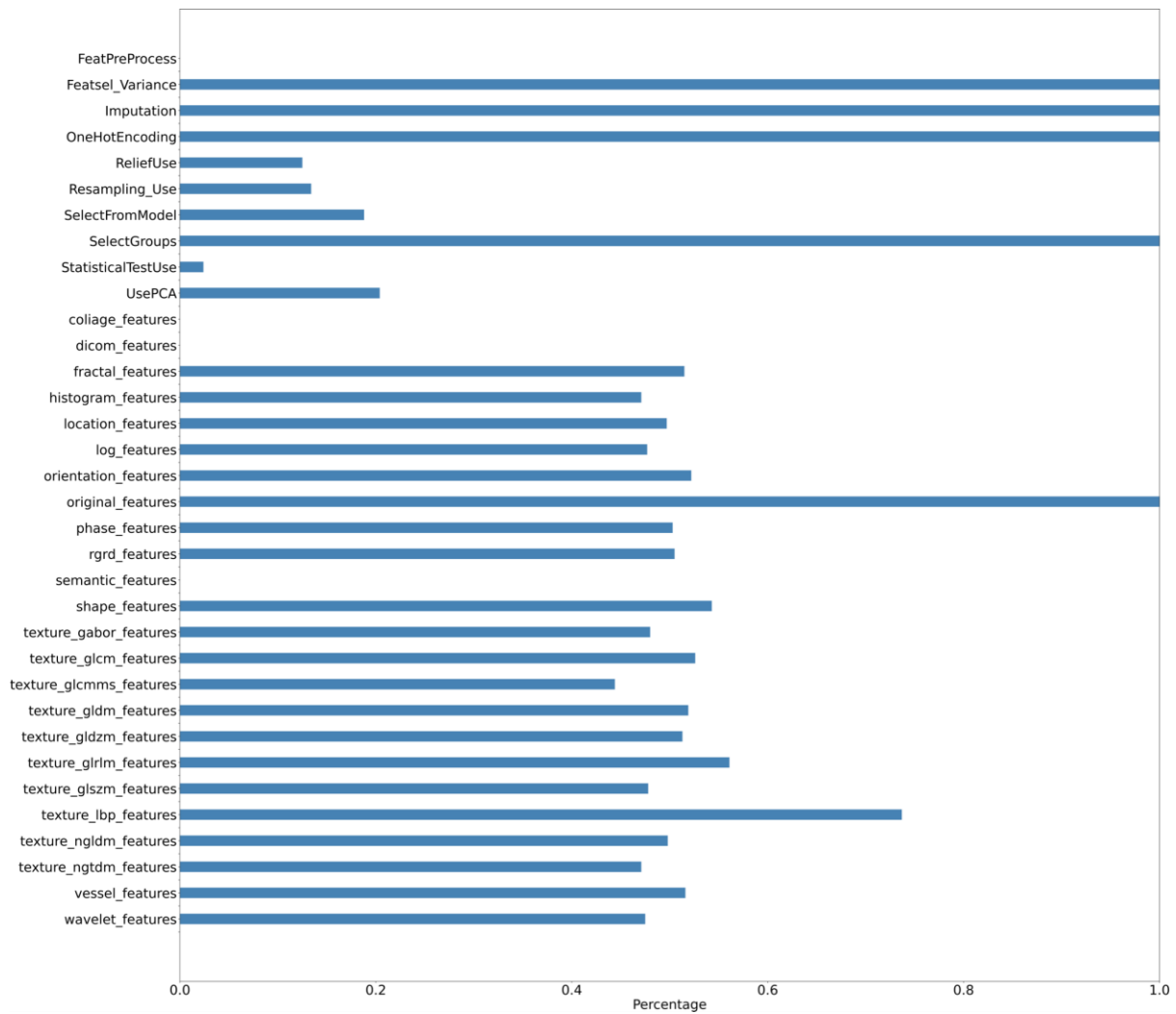


Figure 23. The percentage of the times that a particular feature and feature selection method has been selected through all the hyperparameter optimization iterations.

### 3.3.4 Feature decomposition

Figure 24 shows the decomposition of the features using PCA, Sparse PCA, and Linear Kernel PCA. In general, if the regular PCA shows a good separation of the classes, then the classes can be split using linear combinations of the features. In the Liver cancer classification use case, we can see that the principal components of the features for two classes are somewhat clustered but no clear linear separation is possible. Overall, as can be seen from the average ROC-AUC of 0.752 in Figure 21, cancer classification in Liver MRI is a challenging problem.



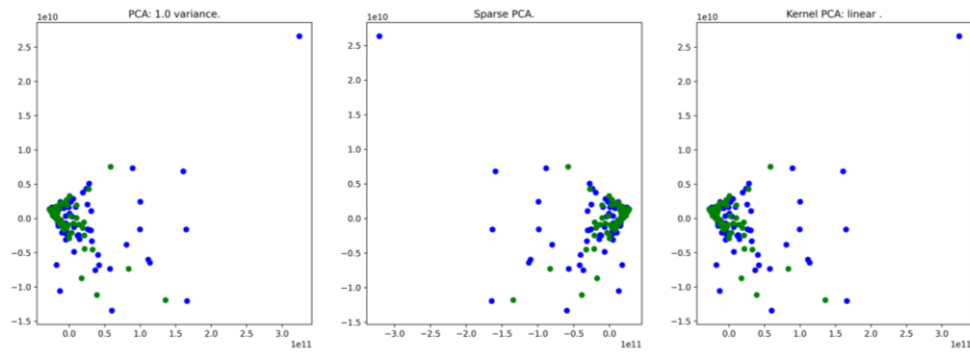


Figure 24. Feature decomposition using PCA and its variants.

## 4 Conclusions

Overall, this cloud-based open-access VRE tool is an excellent option for clinicians or any other users to get hands-on experience in ML by designing their own workflows and experiments without knowing how to code. However, we also encourage ML practitioners to use this tool to set a baseline for their own research to save an immense amount of time spent for manual hyperparameter optimization for different clinical tasks. The additional evaluation features make it even more user friendly by providing tabular and visual assessment of the results along with the feature space exploration.

## 5 References

- [1] Starmans, M, et al., 2021, Reproducible radiomics through automated machine learning validated on twelve clinical applications. arXiv preprint arXiv:2108.08618.
- [2] Tharwat, A., 2021. Classification assessment methods. Applied Computing and Informatics 17, 168–192.
- [3] Starmans, M, et al., 2021, The WORC\* database: MRI and CT scans, segmentations, and clinical labels for 932 patients from six radiomics studies. Submitted, preprint available from <https://doi.org/10.1101/2021.08.19.21262238>