\section{Model validation Pipeline}

\subsection{Importance}

Validation of the ML algorithms used in the platform is a key step in the direction of building trust for the application of ML in the medical field.

In this context, a part of the project focused on showing that the algorithms used performed over a certain baseline established by other, fine-tuned algorithms, on the same dataset. Much criticism has been drawn over ML models in medical field being evaluated on a specific dataset which is unavailable and without having been bench marked against any other model or dataset. In an effort to go against this trend and provide end-users with a certain notion of reliability of the model, the task of validating the platform algorithms was undertaken.

in order to determine adequate and robust ways of comparing ML models on a similar task, different methods were implemented.

An investigation on the importance of this task was done. It is important to note that this validation study is fully linked to the first part of the project which focuses on interpretability and explainability of the platform’s models. It is vital to be able to know if the models used perform well on “standard” data sets before using them on any other dataset.

The creation of new models using the same data pre-processing pipeline as the platform was done. These models were tuned and the results obtained were compared to those of the platforms in order to establish a baseline of what should be expected: do the algorithms provided perform well enough compared to fully specialized and fine-tuned algorithms on the same dataset for the same task. The goal of this validation is to show that the algorithms can be used on this type of dataset with a guarantee of being not too different at a certain threshold from bench-marking algorithms.

\subsection{Pre-Processing}

Before pre-processing data, some features were dropped for two main reasons, either because they lacked too much data or because they were detrimental to the comparison goal. As an example of the latter case, if trying to predict a disease, the features related to a test diagnosing said disease should be discarded in order to let the model learn from features that are not extremely highly correlated to the target label. These dropped features will also be discarded when running the platform models. The selected features were kept as a basis for the rest of the investigation.

Pre-processing was done in two manners. First, the same methods applied to the platform models were chosen, in order to get a comparison of the model performance with the same exact input dataset.

These methods include one-hot-encoding of categorical variables, label encoding of binary variables, min-max scaling of numerical variables, as well as missing value imputation with median values and finally random over-sampling of the minority label class.

Second, further pre-processing and feature engineering was performed in order to figure out if the platform pre-processing could lead to model performance differences. These methods include adding PCA component to the list of features, trying out different sampling techniques such as SMOTE, or a combination of these. The Hyper-Parameter Tuning and Selection was repeated for these models.

\subsection{Hyper-parameter Tuning}

The models' hyper-parameters were tuned in order to find a well-performing set of hyper-parameters. The method implemented for this objective was a grid search on a set of values in the neighborhood of typical values used for classification tasks.

The models will be tuned using a grid-search method for optimal hyper-parameters.

The hyper-parameters we will vary for the tuning can be summarized in the following tables.

First, the models used will be designed based on the models available on the platform, as the objective of the comparison is to evaluate how the platform models fare on a sample dataset against tuned-models that are handcrafted.

The models used will be Naive Bayes Bernoulli, Logistic Regression, Random Forest as well as a MLP (ANN).

Second, some other models will be used as they are extensively used in the literature in the medical field and could be useful to integrate to the platform later on. The models used will be a SVM, kNN and a different type of ANN which includes convolutional layers. Ensemble models could also be constructed to see if they perform better. These will both set a baseline for possible future additions to the platform model selection, while also serving as a reference as a gold-standard (if they perform much better) or as a literature-reference (as they are used a lot).

It is to be noted that some of these models are not expected to perform as well on this dataset as others. Moreover, the nature of the dataset used makes it so that the use of certain architectures (e.g. ANN and CNN) are not typically recommended: they do not give such an edge over other models which are less computationally expensive to train and optimize. Moreover, since the platform models are not tuned, then using ANNs should probably not be the priority on this type of data.

The models will be tuned using a grid-search method for optimal parameters.

The parameters we will vary for the tuning can be summarized in the following tables.

\paragraph{Bernoulli Naive Bayes}

The parameters are: fit\\_prior "Whether to learn class prior probabilities or not. (uniform by default)" and alpha "Additive (Laplace/Lidstone) smoothing parameter (0 for no smoothing)."

\paragraph{Logistic Regression}

The parameters are C ("Inverse of regularization strength. Like in support vector machines, smaller values specify stronger regularization."), fit\\_intercept ("Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.") and penalty ("Used to specify the norm used in the penalization. ").

\paragraph{Random Forest}

The parameters used are n\\_estimators("The number of trees in the forest."), criterion ("The function to measure the quality of a split."), max\\_depth ("The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min\\_samples\\_split samples.") and max\\_features (" The number of features to consider when looking for the best split:").

\paragraph{Multi Layer Perceptron Classifier}

The parameters used are alpha ("The ith element represents the number of neurons in the ith hidden layer."), activation ("Activation function for the hidden layer."), solver ("The solver for weight optimization.") and alpha("L2 penalty (regularization term)") parameter.

\paragraph{kNN}

\paragraph{SVM}

\subsection{Model Selection}

The best set of hyperparameters was chosen for each model architecture, and for each pre-processing technique.

The models were then run on a hold-out test set, both the models from the platform and the models constructed here.

The performance of the various models is asserted \textbf{through AUC scoring, accuracy, precision, recall, sensitivity, specificity, F1 score, Cohen's Kappa, and time of run- actually this might be changed} on the validation data set.

It is important to note that while these parameters are the best, they may also be more prone to over-fitting than worse-performing models. Therefore maybe a second criteria could be added based on the generalization property of the model; the difference between training and testing scores. In this case, to deal with this the model's performance was checked and if the difference was not too large then the parameters were kept.

\subsection{Model Comparison}

Compare results from models that were optimized to results obtained through the platform - on the same dataset.

Run times should also be compared, and the pre-processing of the data should be part of the comparison at one point.

First models should be compared with the same pre-processing. Second, the platform models should be compared to hand-tailored models (of the same type).\textbf{ or just do all at once?}

In a first part, all results will be displayed and analyzed individually. The best-performing models will be tested and metrics will be computed for each one.

In a second part, the results will be compared to one-another.

The best-performing model of each aforementioned architecture / type will be selected and run. Results that will be computed should be the same as available in the platform - namely AUC, accuracy, precision, recall, and some other such as ROC curve. Aside from the model metrics, the time for run will be computed as well.

Once these models have been run and the results computed, the same will be done on the platform models.

With results from both the platform and this project computed, a comparison will be made. AUC curves can be plotted for each model and their counterpart, and a table comparing metrics highlighted above will be constructed.

The models with similar architecture were compared side-by-side, based on several criteria: accuracy, AUC score, sensitivity, specificity, \textbf{maybe shouldn't include both AUC and spec/sens}, F1 Score, Cohen's Kappa and time of run.

A table comparing the models was constructed, as well as a radar plot for easier visualization.

In order to understand if a model truly performs better than its counterpart however, a statistical test should be performed. To this end, a paired T-test will be computed on the models, side-by-side.

\textbf{well this remains to be seen lol}

A conclusion can then be inferred on the best performing model.