Automatic machine learning of multidimensional biological data using tree-based models with application to immune-evasion events prediction in PanCancer

Andreia Rogério[†], andreia.rogerio@tecnico.ulisboa.pt

Supervisor: Cláudia Antunes[†], claudia.antunes@tecnico.ulisboa.pt

[†] Department of Computer Science and Engineering, Instituto Superior Técnico - University of Lisbon, Lisbon, Portugal

ABSTRACT

The incidence of cancer in society has become a growing concern and has encouraged the development of new treatments, like immunotherapy, which boosts the body's natural defences to fight cancer. Unfortunately, cancer cells have developed strategies to evade patients' immune system, called immune-evasion mechanisms. To tackle this problem, a machine learning pipeline was developed, AutoTCGA, which applies classification techniques to The Cancer Genome Atlas data and considers 2 immune-evasion mechanisms for prediction: depletion of neoantigens ("CA") and enrichment of Tregs ("IS").

The Decision Tree algorithm achieved cross-validation accuracy scores of 81% and AUC-ROC of 88%, for IS binary classification, and accuracy scores of 73% and AUC-ROC of 78% for CA binary classification, across the PanCancer population. A feature importance study is available using the tree-based model, revealing the characteristics of the patients with detected immune-evasion events.

These conclusions promote personalized medicine and the application of immunotherapy when suitable. This tool brings the possibility of testing well-defined hypothesis and answer open questions in the cancer biology field, for any user with or without machine learning expertise.

1. Background

In the past twenty years we have seen a revolution in genome sequencing with large increases in speed and efficiency coupled with massive reductions in cost, creating the "omics" field in biology [1]. More than 10,000 tumours have now been analysed and stored at The Cancer Genome Atlas (TCGA) with the purpose of increasing the understanding of the genetic basis of this disease, through data analysis and visualization.

Even though TCGA provides the resources needed to advance cancer research at a multidisciplinary level, extracting the biological knowledge has become a challenge, not just due to the massive size of the data but also due to the diversity of data types. Many computational tools have emerged to face this problem, usually through data visualization techniques and building correlations, distributions across cancer types and Kaplan-Meyer plots [2]. However, there is a shortage of powerful automated data mining tools designed to analyse these data with the ability to find connections between all dimensions of attributes, going beyond a hypothesis testing tool and providing insights on novel sub-types of cancer.

Machine learning algorithms have proven to be strong allies in effective analytical workflows that guarantee reproducibility, statistical significance and result provenance, which are essential characteristics in software tools handling biological data. However, the most powerful algorithms are usually "black-boxes", hard to visualize and therefore interpret, and may even require more advanced programming skills which some investigators may be lacking. Decision Trees, on the other hand, are algorithms proven to be very efficient handling diverse data types, including distinct numerical ranges, and leverage on a tree-based structure which can be visualized and interpreted by a non-expert in machine learning. Beyond classification, these algorithms can be used to model a population, testing a single feature in each node until the last nodes (leaf nodes) have the most homogeneous group of samples possible, considering the values of a chosen feature (class) as the criteria of homogeneity. Random Forest and XGBoost are ensemble algorithms that use Decision Trees as their basic unit, therefore gaining in performance but losing in interpretability [3].

The aim of this study is to build a machine learning pipeline able to assemble data from TCGA, train tree-based classifiers to predict cancer features in an automated fashion and export the results through a web app that allows the visualization of the models and respective interpretation by cancer biology experts who may not be familiar with machine learning techniques.

As motivation for the development of the tool, a biologically relevant feature was chosen to be predicted by the tree-based classifiers: the predominance of an immune-evasion event. These are non-incompatible mechanisms that have been observed in cancer cells able to evade the patients' immune system. The purpose of immunotherapy is to boost the body's natural defences to fight cancer, therefore disabling its evasion strategies [4]. However, the interactions between the tumour and the patient's immune system are not yet well understood and depend on many biological characteristics, which is a problem that a machine learning algorithm can help solving.

At the moment, there are bioinformatic tools that analyse immune features of samples from cancer patients. None is available as a web application service nor as a combined tool able to analyse the predominance of each mechanism. The tools available with web-based interface are only data repertoires [5,6]. Most of the analyses published are comparisons between biological features and immune-evasion mechanisms, supported by p-value. Some more broad analyses use hierarchical clustering to group several samples [5], Cluster-Of-Cluster-Assignments (COCA)[6] to combines the results of several clustering approaches, or even model-based clustering [7] to understand the patterns between several features, and a smaller number uses classification techniques [5] to predict the cytolytic activity of tumours. This tool will allow the analysis of all available features and understanding of their relationship with the mechanisms, instead of studying each feature individually (as the statistic tests do), or independently of the events (as the clustering techniques do).

In conclusion, in this case study the aim of the tool is to answer the question: **Can cancer TCGA genomic features predict immune-evasion mechanisms?**

2. Implementation

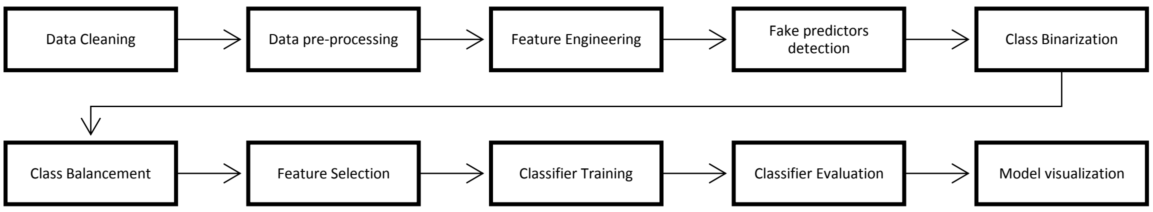

Figure 2: The 10 modules of the automatic system to build models of prediction of a chosen biological class.

The AutoTCGA tool was developed using Flask framework, which connects a visual interface displayed in the browser, with the python scripts that implement the machine learning pipeline – the system itself.
The biological requirements for this system include the choice of binary attribute under study (the class), the possibility of filtering the dataset using biological features values (such as demographic features, genomic features or cancer type), and the visualization of the biological features distribution across the chosen class values, as well as their importance when build the models.
The system is highly modular, composed by six main steps: Data Integration, Data Exploration, Data Preparation, Training, Evaluation and Visualization. Apart from the Data Integration module, which was developed using Pentaho Workbench [8], the remaining modules of the system were built in Python, using the following libraries: NumPy, pandas, scikit-learn. These steps can be found in the architecture of the system (Figure 1), which describes how they are connected to each another. The three main modules are data preparation, training and evaluation, and are the core of the code of the system, distributed over 10 sub-modules, represented in Figure 2. The first 6 sub-modules are what composes Data preparation, while the remaining are the training and evaluation modules.

2.1. Dataset

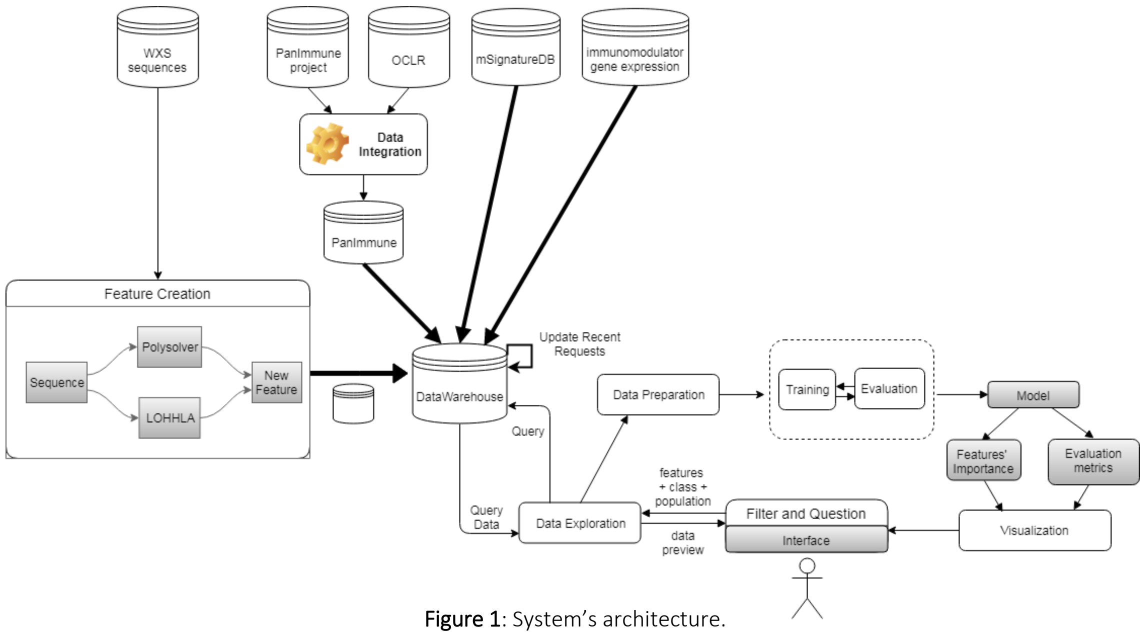

Figure 1: System's architecture.

The user is the one who queries the system and filters the data that is cleaned and processed by the data exploration tools. The first module allows a preview of the dataset that will be used for training and evaluation of the classifier.
The user has 3 datasets available in the system, which can be used individually or merged, and the variable to be predicted can be any of the features of the chosen dataset(s): PanImmune (clinical features and cancer genomic features available at the publication page [9], together with mDNAsi data obtained from publication page [10], Immunomodulator's signatures (same source as PanImmune) and Mutation's Signatures [11]. All the datasets concern features from patients identified at TCGA.
Additionally, two other features were included in the PanImmune dataset: "Treg_cluster", and "Neoantigens_total". The first is a binary variable, obtained from clustering the dataset using the Kmeans algorithm (k=2) of sklearn library, and a set of 32 highly variable genes differentially expressed by Tregs, as implemented by Givenchian et. al [12]. This column splits the patients into two groups: enriched (1) and depleted (0) in regulatory T-cells. The second column is given by the sum of two other columns, two neoantigen types: 'Neoantigens_total' = 'numberOfBindingExpressedPMHC' + 'Neoantigen_num'.
For all the datasets, the Data Cleaning module includes handling missing values (replacing by median in continuous columns or "Unknown" in categorical columns) and word standardization (detecting synonyms and spelling errors). This step is of particular importance in Python, as the sklearn library doesn't handle missing values, or non-numerical values.
The dataset's columns can be filtered by the user, choosing only a subset of the dataset's features to give to the system. This can be done either by using the parameters "Sub-set of features to include" or the parameter "Sub-set of features to exclude". On the other hand, by choosing the population of interest as a cancer type, instead of using the whole sample (PanCancer dataset), the user is filtering the rows given to the system.
Additionally, the user can input new features, as long as they have a column named "sample_code" with the TCGA identification code to enable the merging with the chosen dataset. When merging, the dataset's rows are filtered by the interception between the original dataset and the given input (the join is done in a 'inner' fashion).
After filtering the dataset and before training the classifier, the dataset is pre-processed according to the user's instructions. There are 6 possible pre-processing techniques that handle numerical and categorical data in distinct ways (available only if "PanImmune" is one of the chosen datasets, otherwise only two modes are available, since the other datasets have only numerical variables). The combinations are detailed in the tool and include normalization of numerical variables, categorization of numerical variables, removal of categorical variables and creation of dummy features from categorical variables.
After pre-processing, several optional feature engineering and feature selection techniques are available, which can be activated or deactivated by the user (although "Feature Engineering" option is only available if "PanImmune" is selected as the dataset or one of the datasets).

Once approved by the user, the dataset will be given as input to the Data preparation module that further prepares it for modelling according to the user previous instructions.

2.1.1. Feature Selection

There are three approaches implemented: Logistic Regression Coefficients, Recursive Feature Elimination and PCA. The first is applied to the entire dataset and the features are considered important if they have coefficients higher than 0.2. The second approach works by recursively eliminating each feature from the dataset and measuring the performance of a chosen classifier, which should decrease for important features. The algorithm used was Logistic Regression with cross-validation (cv=10). The PCA approach works by considering a feature to be more important as it contributes most to the principal components' equations (threshold=0.5).
The final set of selected features is given by the features that belong to at least one of the three sets previously obtained.

2.1.2. Feature Engineering

There are two approaches implemented: Log-transformation and Sum of features.The numerical features are log-transformed using the formula y=log(x+1) where y is the new feature and x the old feature. Not all are transformed, only those who show significant difference in distribution: 'LOH_n_seg', 'LOH_frac_altered', 'MutationsSilent', 'MutationsNonSilent', 'CNV_segs', 'CNV_frac', 'HRD', 'Virus_HCV', 'Virus_HPV', 'numberOfNonSynonymousSNP', 'numberOfImmunogenicMutation', 'numberOfBindingExpressedPMHC', 'Indel_num', 'Immunogenic_indel_num', 'Neoantigen_num', 'T.cells.CD4.memory.resting','totTCRa_reads', 'totTCRb_reads', 'mDNAsi'.
The sum features created were "Viral_Total", "Immune_cells_total" and "Mutations_total" using several variables:
Immune_cells_total = B-cells + T-cells + Dendritic cells + NK cells + Neutrophils + Plasma cells + Eosinophils + Macrophages;
Mutations_total = numberOfNonSynonymousSNP + MutationsSilent + MutationsNonSilent + Indel_num
The "**Virus_total**" variable uses all variables with the name "Virus_X" where X stands for a virus name or acronym.
This module is only available when the PanImmune dataset is used, since the techniques rely on features present in that dataset.

2.2. Class

The column chosen to be predicted can be one from the original dataset or one given as input by the user.
Due to implementation purposes, the classes have to be numerical and binary, which requires that all categorical attributes are converted in dummy variables and that the numerical variables are binarized (fifth sub-module in Figure 2).
The class binarization is done automatically by the system by splitting the samples in half after being ordered by the given class value. Exceptions to this are the numerical column "mDNAsi" and all columns regarding viral content. For the "mDNAsi" the value 0.2 was used as the threshold below which is 0 and above is 1, because this variable showed a stepwise behaviour and not continuous. The viral content columns were considered 1 every time the value was not 0. When binarized automatically, the class is also balanced. In case the binarization is not automatic (dummy features and original binary features), the balancement is done by an up-sampling procedure which increases the number of samples in the minority class until reaching the number of samples of the majority class (by repeating existing samples).
The system includes a module of removal of fake-predictors which is class dependent. The system will recognise as a fake predictor all features with a correlation higher than 0.75 with the chosen class, and will remove them from the dataset before training.

2.3. Learning

The learning step (training and evaluation modules) has 3 tree-based algorithms available for classification: Random Forest, Decision Trees and XGBoost classifier.
This optimization is made with RandomSearchCV and GridSearchCV functions of sklearn library which implement cross-validation (cv=10) to train several instances of the classifier and return the one with best evaluation metrics (accuracy being the chosen criteria). RandomSearchCV tests a set of values significantly away from each other, for each hyperparameter of the classifier, and then GridSearchCV will test a grid of values close to the value that obtained the best scores in previous step. In contrast to

GridSearchCV, not all parameter values are tried out by RandomSearchCV, but rather a fixed number of parameter settings is sampled from a given range.
Some parameters were limited to guarantee a good visibility of the tree model, and therefore easy interpretability of it, namely the tree-depth in all algorithms is bound to 6 levels and the number of trees in the Random Forest algorithm and XGBoost is bound to 10 trees:

RandomSearch
```
param_grid = {
        'bootstrap': [True, False],
        'max_depth': [3,5],
        'max_features': ['auto', 'sqrt', None],
        'min_samples_leaf': [2, 6, 10],
        'min_samples_split': [2, 10, 20],
        'n_estimators': [3,5]
    }
```
GridSearch
```
param_grid = {
        'bootstrap': [bootstrap],
        'max_depth': [max_depth-2, max_depth-1,max_depth, max_depth+1],
        'max_features': [max_features],
        'min_samples_leaf': [min_samples_leaf-1, min_samples_leaf, min_samples_leaf+1],
        'min_samples_split': [min_samples_split-2, min_samples_split, min_samples_split+2],
        'n_estimators': [n_estimators-1, n_estimators, n_estimators+1, n_estimators+2]
    }
```
As previously mentioned, the modules work automatically, regardless of the user input parameters. The output of the classification will be in the form of weights given to each feature by the algorithm (feature importance), the distribution of each of the 5 most important features by the chosen class values, in the chosen population, and a tree structure that represents the model (Model visualization module). To sustain the models, the evaluation metrics (accuracy and AUC-ROC) will be displayed as well as the p-value of those metrics obtained with a permutation test. The tree model has nodes coloured according to the predominant class (two different colours in binary classification, one for each class) and each node opacity is given by its purity in the predominant class. By looking at the tree model the user is able to traceback the patterns in the feature's values that describe each class and complement those observations with their rank in the feature importance list.

3. Results

3.1. Default pre-processing parameters

Using the 6 possible datasets created from the original PanImmune dataset with PanCancer population, a study was conducted to conclude which combination of pre-processing techniques provided best results in the system, and therefore should be used as default parameter.
Five algorithms were trained to predict the class "Neoantigens depletion" (CA) with cross-validation (cv=5): Naïve Bayes (using the Gaussian Distribution for numerical variables), Logistic Regression, Decision Tree Classifier (CART implementation) and Random Forest ensemble. The criteria used to choose the algorithms was the ability for interpretability of the models created, which can be done looking at the weights given to each feature by the model (possible in all of them), or looking directly at the model structure (which can be done using a tree-based algorithm). The algorithms were evaluated by their accuracy, precision, recall and AUC-ROC.
After analysing the results, the algorithm with best performance in accuracy and AUC-ROC metrics was Random Forest, using pre-processing 5, achieving 76% and 84% respectively. The scores for precision and recall for Random Forest algorithm were 73,1±2,4% and 71,6±2,5% correspondently, across the 6 pre-processing techniques (results not shown). For this reason, the default pre-processing in the system was set as mode number 5, which includes all categorical features transformed in dummy features and discretization of "Age" into labels but doesn't include normalization of numerical variables. It is possible that due

to the skewed distribution of many variables, the normalization squeezes the range and overlaps the values of many samples, increasing the difficulty in using the variables for prediction.

3.2. Default feature techniques

Taking the algorithms with the best results (Random Forest), and the corresponding pre-processing approach (number 5), several combinations of Feature Selection and Engineering techniques were applied to the dataset and optimized the classifiers for each one. Three combinations were applied: Feature Selection, Feature Engineering and Feature Engineering followed by Feature Selection.
All the three algorithms available in the system were tested and evaluated using the accuracy and AUC-ROC metrics. Opposed to the pre-processing study, the algorithms were not used with the default parameters, but instead they were optimized for each of the three datasets produced, using a restricted range of hyperparameters (maximum of 6 levels in each tree and maximum of 10 estimators in ensembles).
For this study only the biologically relevant classes were tested: “CA” and “IS”; only one dataset was given to the algorithms: the PanImmune dataset; and three populations were chosen: the PanCancer population with 9977 patients, the Breast Cancer (BRCA) population with 1026 patients and the Lung Adenocarcinoma (LUAD) population with 498 cancer patients.
Using both feature selection and feature engineering was only beneficial in one binary classification situation: when training a Decision Tree to predict the class “IS” with the PanCancer population (Figure 4). The application of feature engineering alone was beneficial when training a Random Forest to predict the “CA” class with the LUAD population (Figure 3).
Since activating the feature selection and feature engineering dataset processing is both computational and time expensive, and since most of the algorithms had no benefit with their activation (and some even had worst results), both feature selection and feature engineering are deactivated by default in the system. Ideally the parameters would be set depending on the algorithm used and population under study, but after discussing the system’s usability with staff from the Cancer Bioinformatics group, at King’s College London, it was agreed that having constant default values for the parameters was beneficial for the user experience and understanding of the system.

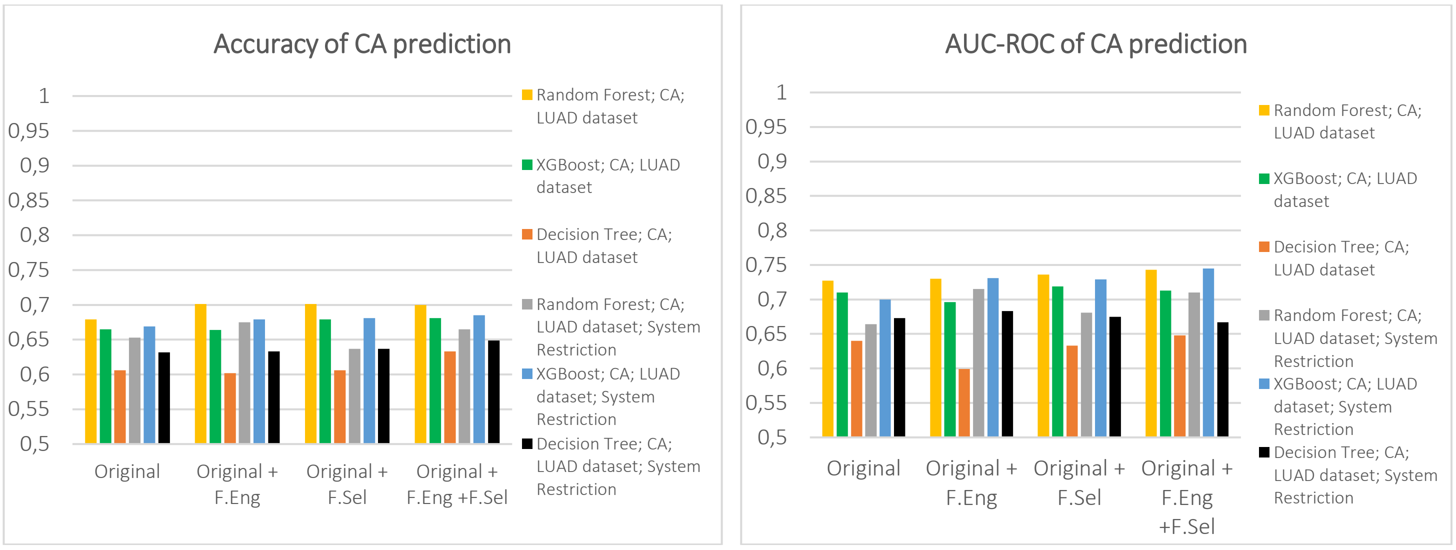

Figure 3: AUC-ROC and accuracy scores obtained when training Random Forest algorithm, XGBoost and Decision Tree with PanCancer dataset, applying Feature Selection and/or Feature Engineering techniques to predict IS.

3.2.1. IS class achieves better results than CA class

Given that the IS class is obtained with a previously tested and approved method [12], it was expected that this was a well-defined classification problem, as opposed to the prediction of the “CA” class, which is automatically binarized in the population (dividing the population in half according to the values in the “Neoantigens_total”). It is possible that the automatic division of the

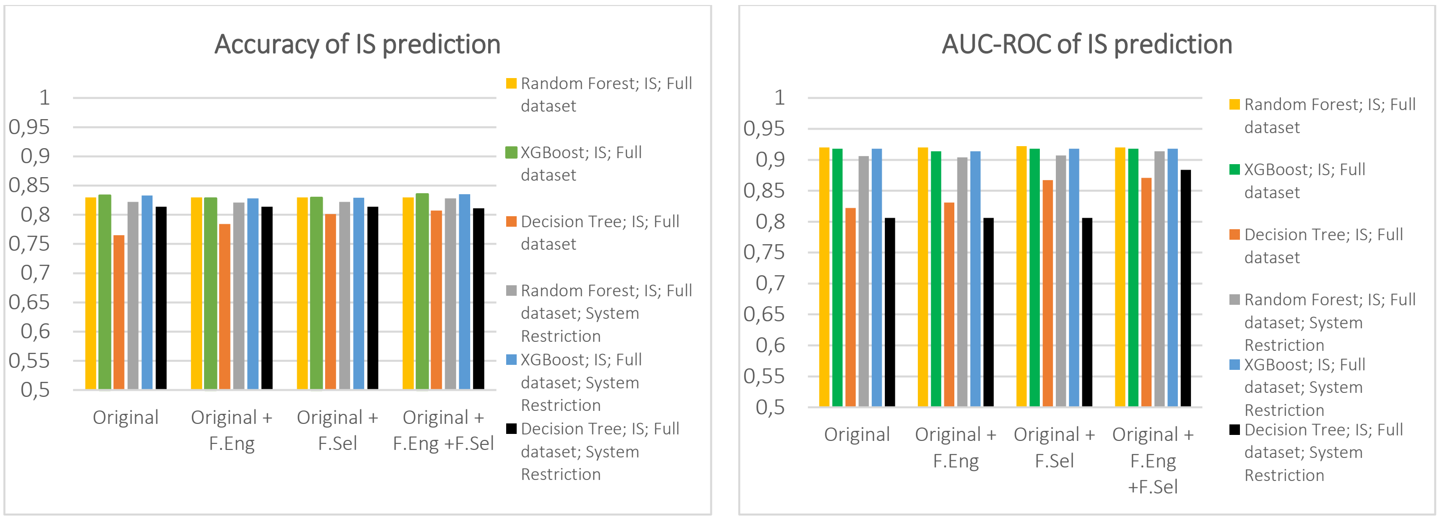

Figure 4: AUC-ROC and accuracy scores obtained when training Random Forest algorithm, XGBoost and Decision Tree with LUAD dataset, applying Feature Selection and/or Feature Engineering techniques to predict CA.

population is not the best approach, and should be revised. Nonetheless, it is possible that the prediction of "IS" was only higher due to the presence of the highly important feature "T.cells.regulatory..Tregs.", as previously mentioned.

3.2.2. Full dataset achieves better results than BRCA and LUAD dataset

There was a decrease in both accuracy and AUC-ROC, for the prediction of both classes "CA" and "IS", when using the dataset BRCA, compared to the PanCancer population results. The dataset filtered for BRCA patients has 1023 rows, which is approximately 10% of the full dataset, while the LUAD dataset is roughly 5% of the PanCancer population. The classifiers chosen are ensemble approaches that use as estimators Decision Tree classifiers, which are algorithms that rely on the size of the dataset to achieve good performance metrics. When faced with a smaller dataset these algorithms may overfit and lose the ability to generalize their prediction power, therefore having worst scoring results. To avoid overfitting problems the models created with the BRCA and LUAD datasets have to be simpler (lower number of nodes in each tree), which explains the overall lower performance obtained, compared to the classifiers trained with the PanCancer population.
Additionally, when filtering the PanImmune dataset by cancer type, the system adds a set of new features specific of that cancer type, which may interfere with the performance of the models. By having too many features it is possible that the algorithms don't choose the most important ones in each node of the tree, since the feature's choice is done from a random sub-set of features from the dataset.
Nonetheless the values obtained are all close to 0,7 or higher (both accuracy and AUC), and the feature's importance together with the tree model obtained with the Decision Tree algorithm are of high relevance to understand the immune-evasion mechanisms in the specific case of BRCA and LUAD patients.

3.2.3. XGBoost and Random Forest outperform Decision Tree

As expected, the ensemble classifiers outperformed the Decision Tree classifier in all combinations of class chosen, feature processing techniques applied and population of interest. Nonetheless, the results achieved by Decision tree classifier were very satisfactory (most of the results are higher than 0,7 for accuracy and AUC).
Given that the Decision Tree algorithm outperforms the remaining algorithms in interpretability by the user, and speed of training and validation, this algorithm was chosen as the default classifier for the system. By doing this the scores are lowered in 0,05 for accuracy and 0,03 for AUC on average, compared with the algorithm that achieved best scores in each population/class combination, under unrestricted conditions.

4. Discussion

4.1. Tool’s performance

Predicting each immune-evasion individually resulted in models with very high accuracy and AUC-ROC scores. Besides being a good binary classification pipeline for immune-evasion mechanisms, the tool has shown to be highly robust and able to predict any other attribute equally well, which is a consequence of two important characteristics of the tool: code abstraction and modular structure. Therefore, this tool has
accomplished a better than expected in software engineering performance, as it can be applied to many biological objects of study, and not only to the primary object of study of this project, which were the immune-evasion mechanisms.
To access the performance of the tool amongst other tools, a comparative study was conducted (Table 1), including tools that analyse TCGA cancer patients, focused on several biological features, besides the immunological features (which were the main initial focus of AutoTCGA tool). Many of the tools are data visualization tools with very good user experience, that allow building variables distributions, correlations and Kaplan-Meyer curves. These tools have the main data types available at TCGA: gene expression, genomic instability and heterogeneity, and demographic information (clinical follow-up).
AutoTCGA is to our knowledge the first tool to combine the interpretability power of Decision Trees algorithms, trained in an automatic machine learning pipeline, with the TCGA genomic data, generating statistically and biologically significant conclusions. With the tree-based models the user can traceback and understand the patterns of patients with a feature of interest. Those results provide useful insight into associations between the variables, which may be a cause or a consequence of the feature under study.
AutoTCGA leverages the easy interpretation and visualization that tree-base models provide. However, the biggest disadvantage is the poor web interface development. The user experience is of high importance in a field where most professionals are not from IT and may not be experienced with programming. Secondly, AutoTCGA doesn’t allow downloading as this is not a data visualization tool, so it does not fit the purpose of data repertoire, as many of the remaining tools do.

Table 1: Comparison of AutoTCGA with other tools.

	Features Available						Dataset personalization					Data Analysis				Data Visualization		
Tool/Characteristic	mic Instability (CNV, muta	leterogeneity (ploidy, pur	Gemic Expression	Viral Content	Immune-cells burden	Immune events	Download	Feature selection	tients selection by Cancer	s selection by demographi	Feature upload	Distributions	Correlations	Kaplan-Meyer	Machine-Learning	Web available	User-friendly App	Visual ML model
AUTOTCGA	X	X	X	X	X	X		X	X		X	X			X		X	X
TCIA (THE CANCER IMMUME ATLAS)	X	X	X		X		X	X	X	X	X	X	X	X	X	X	X	X
TCPA	X	X	X				X	X	X	X		X	X	X		X	X	
XENA	X	X	X					X	X	X	X	X	X	X		X	X	
TUMORMAP	X		X				X	X	X	X	X	X			X	X	X	X
CRC AGGRESSIVENESS EXPLORER	X	X	X													X	X	
ICLUSTER	X		X					X	X	X								
CBIOPORTAL FOR CANCER GEMICS	X	X	X				X	X	X	X		X	X	X		X	X	

4.2. Future work

One of the most important conclusions taken with the system’s parameters analyses concerns the low relevance of both feature selection and feature engineering techniques. In the future, these data analysis techniques should be revised and improved, for instance by testing other approaches such as statistical methods of feature selection (ex: f_classification [13]).

Another important observation concerned the prediction of class “CA” which is created based on an automatically binarization of the values, therefore assuring a balanced population. All other numerical variables will be processed in the same way if the user chooses them as classes. This approach was not proved to be biologically correct, and in fact the model’s scores were lower for the prediction of “CA” comparing to the prediction of “IS”. Furthermore, the class generation model comes after the data preparation modules (which includes cancer type filtering), and consequently one patient can be considered to have an immune-evasion mechanism active in its cancer type population, and not in the PanCancer population, or vice-versa. In the future, the biologically correct values for each class should be collected (if known) and added to the system’s module “Class binarization”.
From the performance analysis of AutoTCGA, when compared with other tools, it was possible to conclude that there were important characteristics present in other tools but absent in AutoTCGA, namely the possibility of filtering the dataset using demographic features and building Kaplan-Meyer curves with the data.
Regarding the visualization of the results, there are still a lot to improve as well, regarding the user experience of the web-app, as well as more particular aspects such as the visualization of the XGBoost models which is not available, and the visualization of the features’ importance in both Random Forest and XGBoost models, which should be calculated based on all the estimators features’ importance.

5. Conclusions

We have developed a python-based tool, AutoTCGA, to lower the barriers between the application of interpretable machine learning models and the analysis of large levels of biological data from TCGA, providing a user-friendly platform that integrates classification and model visualization in an automated manner.
This web tool can be described as a platform for multiple hypotheses testing, where a user with technical expertise in the biological field wishes to study the immune-evasion events (one of the classes) occurring for a particular cancer type (a specific population or the full PanCancer population), training a tree-based classifier with a set of features (personalized dataset). To accomplish such a system, the architecture was built in a modular fashion and each module counts on a high level of code abstraction. Consequently, the system is highly robust and reproducible, as it is possible to recreate every result using a same combination of input parameters. Furthermore, the system is highly flexible and extensible, as the modules can be activated or deactivated, and new modules can be added to the pipeline without changing the remaining modules in the system. This automatic process leverages on domain information to remove fake predictors and create potential engineered features.
The system built has the completeness of a general system, but it is also characterized by its high level of personalization, as the user is able to choose all the classification parameters and the dataset itself. In AutoTCGA, any feature from the dataset can be chosen as a class to be predicted, not just an immune-evasion event, while the remaining features will be the training attributes. There is a wide selection of features available for prediction, which can be filtered or removed, and it is possible to extend the dataset by uploading user features. The strategy used to deal with different data types included a class binarization approach, when continuous variables are chosen as classes, together with the creation of dummy features of all categorical attributes. Given the good results of the algorithm’s performance metrics, obtained in a binary classification scenario, this strategy was considered successful.
The system’s parameters tuning analyses showed choosing the Decision Tree as the default algorithm and restricting the algorithm’s hyperparameters (maximum of 6 levels in each tree), decreased the models’ evaluation scores (AUC-ROC and accuracy) in only 5% on average, while increasing significantly their interpretability and visualization.
AutoTCGA outperforms the existing tools in its interpretability power applied to the TCGA data, and is the first to apply machine learning techniques to further understand immune-evasion mechanisms. Even though some tools have applied clustering techniques with good visualization platforms and others have applied powerful classification algorithms, none is able to provide what AutoTCGA does. With the tree-based models the user is able to traceback and understand the patterns of patients with a feature of interest. Those results provide useful insight into associations between the variables, which may be a cause or a consequence of the feature under study.
Overall, the tool was designed to handle any biological feature as the chosen class to be predicted but was tailored to predict immune-evasion mechanisms and has shown great potential for application in targeted immunotherapy. It allows not only testing hypothesis on the importance of certain features, but also studying open questions on what features are important to predict a chosen class. In this study, it has been demonstrated that AutoTCGA is an efficient proof-of-concept of the great

application of machine learning pipeline to TCGA data and integrating the computational power of this tool with the high quality of the data visualization tools already developed has great future potential for success.

6. Availability and requirements

The detailed instructions on how to run AutoTCGA are in the README file located at the tool's page at https://github.com/andreiarog/AutoTCGA . It is important that all files in the repository folder are left unmodified and in their original places, to guarantee the tool's normal functioning.

7. References

1. Gaye Lightbody, et. al (2018) *Review of applications of high-throughput sequencing in personalized medicine: barriers and facilitators of future progress in research and clinical application*, Briefings in Bioinformatics
2. TCGA Computational Tools, https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga/using-tcga/tools (accessed October 2018)
3. Pivotal Engineering Journal, *Interpreting Decision Trees and Random Forests*, http://engineering.pivotal.io/post/interpreting-decision-trees-and-random-forests/ (accessed July 2019)
4. American Cancer Society, https://www.cancer.org/treatment/treatments-and-side-effects/treatment-types/immunotherapy.html (accessed October 2018)
5. Charoentong P, et. al (2017) *Pan-cancer Immunogenomic Analyses Reveal Genotype-Immunophenotype Relationships and Predictors of Response to Checkpoint Blockade*
6. Hoadley KA, et. al (2018) *Cell-of-Origin Patterns Dominate the Molecular Classification of 10,000 Tumors from 33 Types of Cancer*
7. Thorsson V, et. al (2018) The Immune Landscape of Cancer
8. Pentaho Hitachi Ventara software, https://www.hitachivantara.com/go/pentaho.html
9. PanImmune data repertoire, https://gdc.cancer.gov/about-data/publications/panimmune
10. Malta t., (2018) *Machine Learning Identifies Stemness Features Associated with Oncogenic Dedifferentiation*
11. Mutations signatures platform, *mSignaturedb*, http://tardis.cgu.edu.tw/msignaturedb/ (accessed June 2019)
12. Givechian, Kevin B. , et. al (2018) *Identification of an immune gene expression signature associated with favorable clinical features in Treg-enriched patient tumor samples*
13. F_classification approach for feature selection implemented in sci-kit learn library https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.f_classif.html#sklearn.feature_selection.f_classif