Knowledge Distillation from Random Forests: A performance comparison

Oscar Aguilar
School of Computer Science and Electronic Engineering
University of Essex
Colchester CO4 3SQ, UK
Email: oa18525@essex.ac.uk

Abstract—**Modern algorithms have succeeded in different tasks that involve large and complicated data sets; however, they have brought some economical and efficiency issues. High costs and slow-running times are some examples. Random Forests (RF) are one of the ensemble methods that allow researchers to tackle these complications with top-performance and high accuracy. Nonetheless, this classifier is a black-box model; hence it is hard to interpret. Also, depending of the chosen hyperparameters it can become very slow. Therefore, this study focuses on distilling its knowledge and injecting it into a simpler model such as Decision Tree (DT). This technique has been used with success in other ensemble methods such as Neural Networks. The main objective of this work is to analyze if a distilled DT (that learned from the RF class probabilities) performs as well as the RF *per se*. Also, these classifiers were contrasted in terms of accuracy with other supervised learning methods. After training, tuning and testing with three different medical data sets, the distilled DTs classifiers achieved to perform similarly to RF; furthermore, they managed to outperform some of the other classifiers. In addition, the plots of the distilled DT are included as well.**

Keywords—Knowledge Distillation, Random Forests, Decision Trees, Machine Learning, Supervised Learning, Graphviz

I. Introduction

Machine learning algorithms have increased its size and complexity due to the evolution of modern data sets. The latter has helped these algorithms to reach high levels of accuracy and high-performance; however, it has risen different problems such as: expensive costs to store information, slow evaluation process, difficulties whilst integration to other systems, and hard interpretation [1]. Traditionally, in machine learning the main focus has been prediction tasks or Supervised Learning (SL). In this area, Random Forests (RF) have proven a more-adequate performance than other models [2].

RF are categorized as an ensemble learn method. By this, it is meant that RF aggregate results of a simpler estimator, which in this case are Decision Trees (DT) [3]. In other words, this group of randomly-collected-grown-trees yields a final prediction based on the aggregation of each individual tree prediction [4]. As abovementioned, RF are well-known for their high accuracy, their ability to work with small samples and high-dimensional features spaces, and their simplicity to use [5]. Generally, the RF algorithm produces good results with its default parameters, for both: classification and regression tasks [4].

As abovementioned, RF consist on a collection of DTs, which is a common classifier for SL tasks [4]. A DT is a tree-based graphical representation of a flowchart; where the internal nodes represent features, the branches decision rules, and the leaf nodes outcomes or classes. This classifier is fast to train, and its complexity depends on the number of features of the data set. Furthermore, DTs do not depend upon probability distributions assumptions. The DT algorithm methodology works in the following order: selection of the best attribute for the first split (based on information gain or gini index), converting that attribute into a decision node and breaking data into smaller subsets. After that, the continuous repetition of the last steps will yield a tree. The algorithm will stop when there are no more remaining attributes or instances, or when the remaining data belong to the same class [6]. In the presence of perturbations of the training data, highly-different models might result, causing variability and poor performance [2].

Although RF are tree-based models, there are some differences between RF and DTs. For instance, High-Complex and Deep DTs tend to suffer from overfitting [4]. RF address this problem by combining multiple overfitted DT, which reduces the effect. The latter is known as bagging, where an ensemble method averages the results to find better predictions [3]. In the other hand, DTs are easy to interpret (descriptive tool) while RF are not (exclusively predictive) [4], [5].

As established, RF achieve high accuracy and their non-parametric model makes them easy to use. Highly-accurate predictions require more DTs, especially in real-word applications. However, in these real-time predictions, a high number of estimators (DTs) can make the algorithm slower, thus inefficient [4]. The applications of RF are distributed in several areas; e.g. banking, medicine, stock market, e-commerce, marketing, etc.

Those differences rely in the fact that DTs are a white-box-based algorithm, while RF are black-box-based. Recently, there has been considerable interest in finding a method to retrieve an “X-ray” of the black-box algorithm.

In essence, attempts have been made to find understandable ways that explain how ensemble methods such as Neural Networks (NN) and RF reach their predictions [2]. A first approach trained different models with the same data, and then average the predictions. However, it resulted inconvenient to generate different predictions with a whole ensemble of models, as well as expensive [7]. A different technique uses an intelligible “student” model to mimic the output of a predictive black box "teacher" model [2]. The latter is rephrased by [1], where the concept of knowledge distillation is defined as the extraction of knowledge contained in an ensemble method and the injection of it into a more convenient model.

The main challenge of knowledge distillation consists on constructing a convenient model that works as good as the ensemble method (from which it learned). Therefore, the scope of this work consists on extracting the knowledge of a RF classifier and use it to train an easier-to-interpret DT. The successful construction of a distilled tree with the knowledge of a RF classifier might be an explanation of how predictions are made within the RF. In order to interpret the performance of the distilled DT (as an explanation of the RF modus operandi), the results and explanations should be reproducible [2]. Different versions of this process have been explored, [7] states that results on the MNIST data were surprisingly good; while [1] mentioned that the distillation of a NN speeded up, considerably, the calculation of needed messages to be passed within a graphical model. Another goal of this study is to show that RF classifier is one of the SL methods with the highest values of accuracy.

Medical data sets have been selected to perform this work because is an area where RF have been used widely. Diagnosing correctly and highlighting lifestyle habits that threat patient's health were part of the motivation behind this decision. In the medical field, in order to diagnose and make decisions, is important to understand and interpret the model correctly. Therefore, understanding the processes within the RF is a high-interest topic for researches.

Throughout this report, an overview of previous works based on RF and knowledge distillation will be included. Then, the utilized sources and implemented methodology to achieve the main scope of this work are going to be enlisted and detailed. Also, a section with the logic and explanation of the experiments performed will be presented, as well as the obtained results. Next, a discussion section will follow. The discussion will consist on the analysis and interpretation of the obtained results for every built classifier. Finally, a section with final remarks, lessons learned, and future work will conclude the report.

II. Background

As previously established, RF classifier have been widely used in medicine field for different purposes, i.e. to construct models that aid clinicians to diagnose patients. In this section, a review of works that utilize RF models for different medical applications is going to be presented. Followed by different studies that have tried to distil the knowledge of ensemble methods into a more convenient model.

Medical images such as CT scans are crucial for physicians to diagnose patients. In [8], a RF classifier was trained after extracting the optimal image features. The selected features were histogram-based and grey-level-based, some examples are: mean, variance, kurtosis, gradient, grey distribution, etc. The RF model reached 95.33% of accuracy when recognizing liver lesions from CT images. Likewise, [9] classify metabolomic data for osteosarcoma diagnosis. In this study, three different machine learning methods were used: linear regression, support vector machine and RF. Although all classifiers classified correctly between healthy images and tumour instances, RF outperformed the other two with 95% overall accuracy; furthermore, RF got an accuracy rate of 97% for cross-validation in the training set.

The analysis and interpretation of an electrocardiogram (ECG) with the aid of RF has become a tendency. In [10] and [11] RF models have been used to diagnose arrhythmia and congestive heart failure respectively. The experiment in [10], proposed a new feature based on amplitude differences in different segments that achieved 98.68% of accuracy. On the other hand, [11] used five different classifiers to classify EG signals from two different data sets (BIDMC Congestive Heart Failure and PTB Diagnostic ECG). Afterwards, the performance of each model was analysed with statistical measures such as: sensitivity, specificity, accuracy and F1-score.

The article concludes that the RF model performed the best, with an accuracy close to 100% when detecting heart failure. Finally, it concludes by expressing the importance and value of this kind of models to aid knowledge and medicine evolve.

These examples confirm the robustness and high-levels of accuracy that RF can achieve. Nonetheless, as mentioned above, several attempts to understand the paths that ensemble methods follow to reach a prediction have been done. Principally, due to training time and improvement costs Therefore some Knowledge distillation previous approaches are going to be discussed.

In [1] different approaches to extract knowledge for complex models into simpler ones are stated. First, an attempt to compress a model into a single neural network was performed in 2006. This framework used flexible density estimators trained on the data and had low storage requirements. The latter allowed the student model to label the input data like the teacher model (in low size batches). Papamakairos in [1], includes an automatic speech recognition problem, where a deep neural network model (with high success) was mimicked in order to have less-memory-limited system. The replacement system was a smaller shallow neural network that minimizes the disparities between unlabeled instances of a data set. However, shallow networks are not commonly used in practice because they have a tendency to overfit results when directly trained on the data.

III. Methodology

This work aims to extract the knowledge within a RF and inject it into a simpler model: a DT classifier. To achieve the latter, the class probabilities of the RF are going to be obtained and used to yield a new data set; which will be used to train the new DT.

Apart from the distilled DT, another one will be built and trained with the original data sets (such as the RF classifier). This means that there will be two different DT that can be compared directly.

When the models behave properly, the performance of the RF and distilled DT will be contrasted in terms of accuracy and its confusion matrices (which were plotted and included in Fig. 1-Fig. 6). Similarly, other models such as: SVM and k-NN will be tested and compared versus the ones previously stated.

A. Data Sets

The description of each utilized data set will be covered in this section. All of them contain medical data and are suitable for binary classification. It is important to mention that these descriptions correspond to the data sets before a cleansing process. On the other hand, Fig. 1-3 are after data cleansing.

1) *Breast Cancer Coimbra Set (BCC)*

This set of data was gathered in 2018 by the research staff of the University of Coimbra in Portugal. It is made up of nine different anthropometric features, which means that they measure proportions of the human body. In this particular case, the attributes correspond to information that could be obtained from a blood analysis. Some examples are: Glucose, Insulin, Homeostatic Model Assessment of Insulin Resistance (HOMA), Leptin, Adiponectin, Resistin, among others. This data set aims to predict (from a blood sample) if a person has a cancerous breast tumor; therefore, target classes are ‘1’ for healthy patients (52 instances) and ‘2’ for breast cancer patients (64 instances) (See Fig. 1) [12].

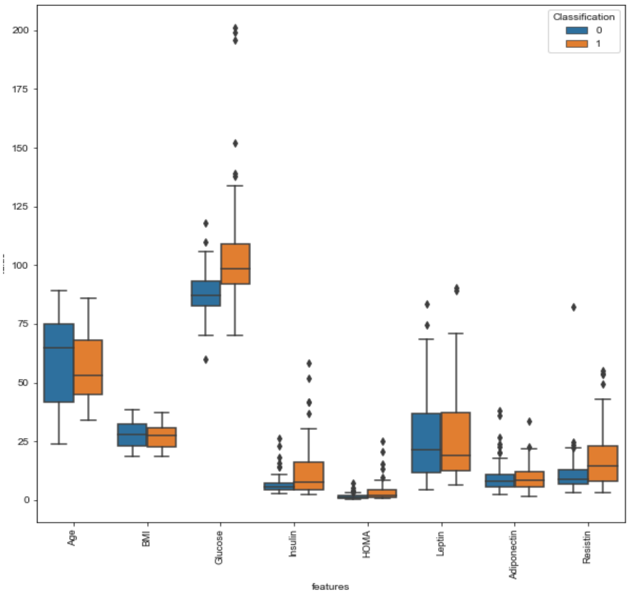


Fig. 1. BBC Data Set Plot

2) *Fertility Set*

This data set was gathered by a research group in 2012 at the University of Alicante, Spain with the help of 100 volunteers who provided a semen sample. Each sample was clinically analyzed and labeled as ´N´ for normal (88 instances) and ´O´ for altered (12 instances). Their goal was to relate the fertility diagnosis with socio-demographic and environmental aspects and the health status and life habits of each volunteer. All the data is quantitative since each attribute was numerically labelled; for example, for surgical intervention ‘1’ represents ‘yes’ and ‘2’ stands for ‘no’. Some attributes are: season in which the analysis was performed, accidents or trauma, surgical intervention, frequency of alcohol consumption, smoking habit, and the number of hours spent sitting per day, among others (See Fig. 2) [13].

3) *Mamammographic Mass Set (MM)*

This data set was obtained between 2003 and 2006 by two German doctors, one from the image Processing Medical Engineering Department (BMT) and the other from the Institute of Radiology. This data is composed of 4 predictive attribute and one non-predictive. It contains also a target or classification column. The predictive values are all numerical labels; where some of them represent a specific quality. For instance, shape has numerical labels for descriptive information, where round=1, oval=2, etc. Other attributes are: age, shape, margin and density. The target labels are ‘0’ for benign (516 instances) and ‘1’ for malignant (445 instances) (See Fig. 3) [14].

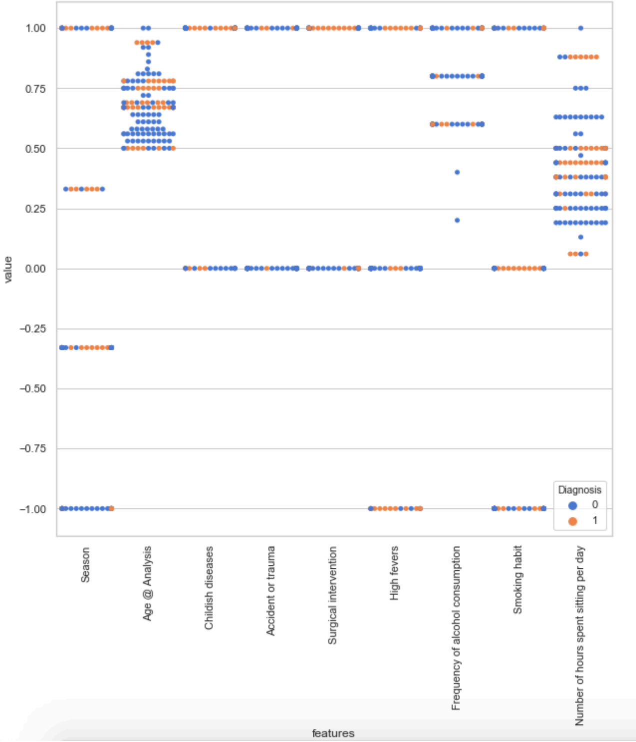


Fig. 2. Fertility Data Set Plot

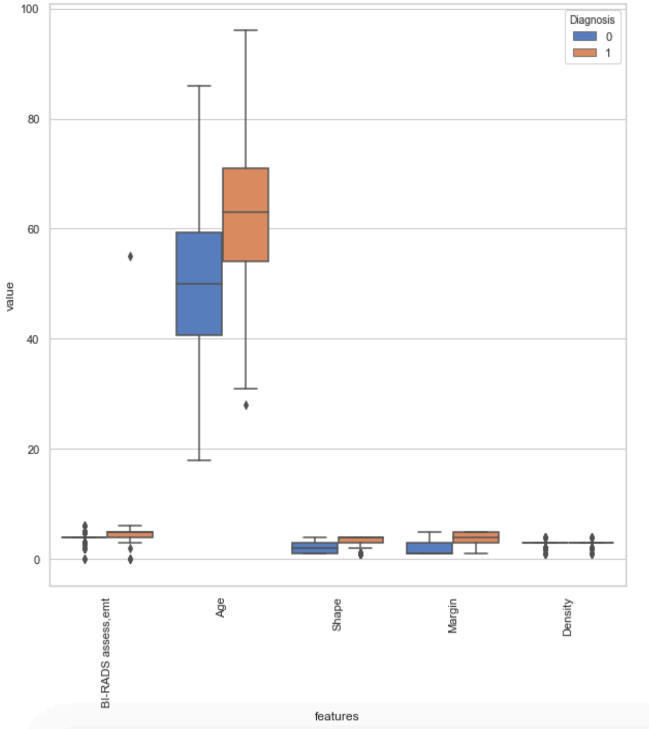


Fig. 3. MM Data Set Plot

B. *Data Cleansing*

As part of a previous stage of this work, the three data sets were cleansed before utilized. During this process, each data set was inspected to look for missing values. The only one that had them was the MM set but was fixed with median imputation. This technique avoids altering considerably the data. Other aspect that was taken into consideration was class imbalance. This was found in the fertility set, where the data had 88 instances for one class and 12 for the other. However, to fix it, an upsampling method was performed [15].

C. Libraries and Classifiers

The chosen working environment was Spyder from Anaconda, which allowed the import of different useful libraries. For instance, to perform data manipulation 'Numpy' and 'Pandas' were crucial. On the other hand, for machine learning purposes (classifiers building and data splitting) 'Sklearn' was imported. Similarly, for data visualization 'Matplot' was chosen; finally, 'Graphviz' was utilized to plot the final decision trees.

As aforementioned, one of the main goals of this project is to analyze if a student model is capable of learning the knowledge of an ensemble classifier. In this case, a 'distilled tree' will learn from the class probabilities of a RF. Simultaneously, a binary DT will learn the same information as the RF. At this moment, three classifier will be built: a RF, a binary DT and a distilled DT.

However, analyzing the performance of this distilled classifier versus the ensemble method and the binary DT is not the only objective. This study, similarly, aims to contrast the distilled classifier versus other supervised learning methods. Hence, an SVM and k-NN classifiers were built as well.

In section 3.D the chosen hyperparameters for each classifier are discussed. These chosen values vary depending on the data set. Mainly, because the final hyperparameters were the ones that allowed a better performance during the tuning phase.

D. Classifier Hyperparameters

As previously established, five different classifiers were fitted for each data set. Nonetheless, their hyperparameters vary depending on the data set.

For the BCC set, the RF has 500 estimators and uses the gini criterion to stop the splitting. Both, the DT and the distilled DT use, as well, the gini criterion and a minimum impurity decrease of 1x10^{-4}. Finally, the SVM has a linear kernel and the k-NN uses 3 nearest neighbors.

For the Fertility data the RF, DT and distilled DT contain the same hyperparameters as the BCC set. However, the SVM has a gamma of 1x10^{-2} and a cost of misclassification equal to 1000. Also, the number of the nearest neighbors for the k-NN is 4.

Finally, for the MM set the hyperparameters changed slightly. First of all, the stop criterion for the RF, DT and distilled DT was entropy; however, the number of estimators and minimum impurity decrease remained the same. Also, the number of nearest neighbors changed to 10 in the k-NN. Finally, the SVM has the same cost as the one used in the fertility set, but gamma changed to 1x10^{-3}.

IV. Experiments and Results

This section of the report aims to provide a detailed explanation of each followed step during experimentation. It covers from splitting the data to final testing. Any named function in this section is included in the previously-mentioned libraries of Section III.C

A. Train, Validation and Test Sets

Each data set was splitted into three different subsets: training, validation and testing. As in [16], the training set includes the data used to fit the models. In the other hand, the validation set provided a method to evaluate the fitted models unbiasedly while tuning their respective hyperparameters. Finally, the test set was used to perform an unbiased evaluation of the final models; which was data that the classifiers have never seen before, and allowed a fair and direct comparison between performances.

The splitting ratio, considering the data set as a 100% of the information, was as follows: training 80%, validation 20%, and testing 20%. Primarily, because it was intended to provide as much information as possible to the models when training. The latter was necessary since the data sets have few instances. In order to do so, the model selection function from Sklearn was utilized twice.

It is important to mention that each subset had features' information and target values. Furthermore, in order to predict the probabilities of the RF classes, all the information (except the target/class column) was utilized.

B. Building, fitting and tunning of classifiers

All the classifiers were built with their respective functions from Sklearn. Similarly, they were trained with the functions fit and tested with predict (one for tuning and other for final testing). During the tuning process, the hyperparameters were modified. The final values were reached when the models reached top-performance (high values of tuning accuracy).

C. Binning, binarization, and plotting

In order to extract the knowledge from the RF, the probabilities of each class were obtained with the function 'predict proba' (all the feature data was used). Since the addition of the classes for each instance is equal to 1, there was no need to keep the information of both classes. For instance, if the probability for class 1 is 0.7, the probability of class 2 will be strictly 0.3. Therefore, the column of the second class was dropped. After that, the function histogram from the numpy library was used as a binning method.

From here, the histogram values and edges were saved on variables, and allowed the binary classification problem to shift to a multiclass problem. The previously-mentioned variables were plugged in the function digitize, which returned the specific bin number where each probability belonged. After that, a new data set was created, splitted and used to fit a previously-tuned DT.

Finally, the multiclass predictions where binarized with the numpy function 'where'. In the last function, whenever a prediction was below the middle bin edge, it was given a '1' which stands for unhealthy. In the opposite case, the given class was a '0'or healthy. Depending on the dropped-class-probabilities the latter can change; since in this work the second class (unhealthy) probabilities were removed, a low bin edge stands for "low probability of being healthy". Please refer to Code I.

D. Final testing

Since the results of the distilled DT can be binarized (as mentioned above), it is possible to test all the five classifiers with the remaining set of data. As established in IV.A, the purpose of the validation set was tuning, but the final testing requires an unseen data set. Therefore, the previously fitted classifiers are used to make predictions with the test set. After that, the results and true values are used to calculate accuracies and confusion matrices.

Tuning and final accuracies were calculated with the function 'accuracy_score' from the metrics section of Sklearn. From this same section, the function 'confusion_matrix' was used to obtain the confusion matrices of the classifiers. Additionally, the library 'matplot' was utilized to visualize the confusion matrices as plots.

In order to obtain the average accuracy, the cross-validation score was obtained. The reported value is the mean of the accuracy obtained in 10 folds. Finally, to plot the binary DT and the distilled DT, the steps in Code II were followed. For this task, the required library was graphviz. The number of plotted trees were 6, where half of them are binary decision trees and the other half distilled decision trees. For pictures please refer to Appendix 1.

CODE I. BINNING AND BINARIZATION

```
#Get probabilities
prob=clf.predict_proba(x)
#Convert to data frame
df=pd.DataFrame(prob)
#Drop probabilities of one of the classes
p1=df.drop(1, axis=1)
#Convert to numpy array
p2=np.array(p1, dtype=float)
#Bining process
hist, bin_edges = np.histogram(p2,
bins=num_bins)
#Retrieve the bin number of each probability
bin_number= np.digitize(p2,bin_edges)
#Create new data set for multiclass
classification
prob_dataset=np.concatenate((x,bin_number),
axis=1)
#Data splitting
x_train, x_test, y_train, y_test =
train_test_split(data,bin_number,test_size)
#Training multiclass classification DT
(distilled)
clf=clf.fit(x_train,y_train)
#Testing the classifier
dt_multi=clf.predict(x_test)
#Binarize process (from multi to a binary
problem)
binarize=np.where(dt_multi >= (num_bins/2),0,1)
```

CODE II. TREE PLOTTING

```
dot_data =
tree.export_graphviz(clf,out_file=None,
feature_names=attributes,class_names=['0','1'],
filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph.render("Title", directory='/path/')
```

E. Results

In this section of the report, the relevant obtained results are presented in tabular form. Table I, II and III include tuning, average and final accuracies, while Fig. 4-9 show the confusion matrices of each RF and distilled DT classifiers. The accuracy values show how well does the student model performed against other supervised learning classifiers; while the confusion matrices contrast the student model behavior against the ensemble method. The interpretation and discussion of each result is included later in Section V.

TABLE I. BCC SET ACCURACIES

	Accuracies (%)		
Classifier	***Tuning***	***Average***	***Final***
Random Forest	69.56	52.88	83.33
Binary Decision Tree	78.26	58.80	66.66
Distilled Decision Tree	25.71	52.80	79.17
SVM	60.87	34.62	62.5
k-NN[a]	21.74	34.62	62.5

[a] Trained with 3 Nearest Neighbors

TABLE II. FERTILITY SET ACCURACIES

	Accuracies (%)		
Classifier	***Tuning***	***Average***	***Final***
Random Forest	92.59	93.24	88.88
Binary Decision Tree	81.48	88.79	85.19
Distilled Decision Tree	72.5	88.79	88.88
SVM	77.77	63.08	70.37
k-NN[b]	77.77	73.52	77.77

[b] Trained with 4 Nearest Neighbors

TABLE III. MM SET ACCURACIES

	Accuracies (%)		
Classifier	***Tuning***	***Average***	***Final***
Random Forest	72.92	78.36	77.72
Binary Decision Tree	68.75	77.63	75.65
Distilled Decision Tree	88.24	77.63	76.17
SVM	78.65	82.42	80.83
k-NN[c]	72.92	79.72	78.76

[c] Trained with 10 Nearest Neighbors

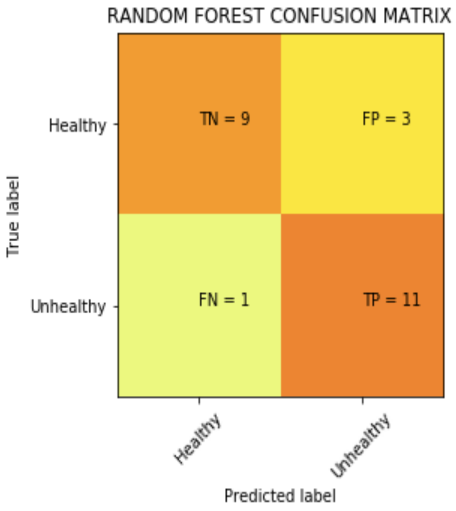


Fig 4. BBC Set RF Confusion Matrix

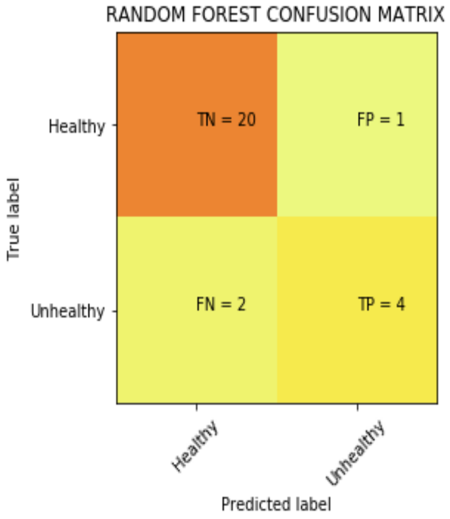


Fig 5. Fertility Set RF Confusion Matrix

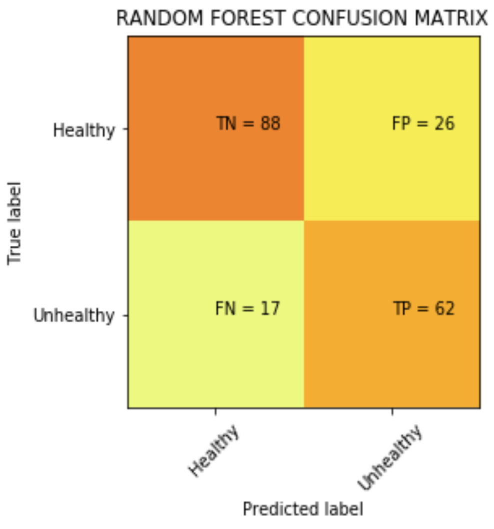


Fig 6. MM Set RF Confusion Matrix

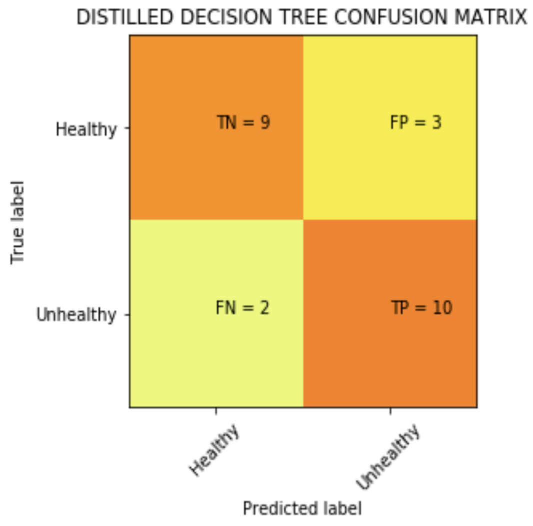


Fig 7. BBC Set Distilled Decision Tree Confusion Matrix

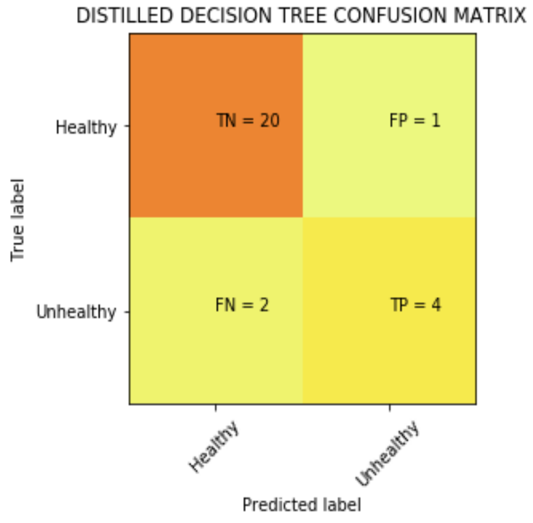


Fig 8. Fertility Set Distilled Decision Tree Confusion Matrix

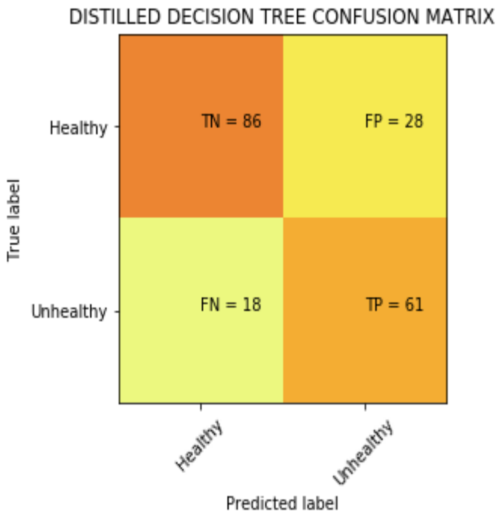


Fig 9. Fertility Set Distilled Decision Tree Confusion Matrix

In addition to these tables and plots, the classification report of each model was obtained with a function from the metrics section of Sklearn. These reports include the precision, recall, support and F1-score of each class per classifier. The latter are located in Appendix 2; however, in Table IV to Table VI are included the relevant values for discussion.

TABLE IV. BCC SET WEIGHTED CLASSIFICATION REPORT

	Precision	Recall	F1-Score
RF	0.84	0.83	0.83
DDT[d]	0.79	0.79	0.79

[d] Distilled Decision Tree

TABLE V. FERTILITY SET WEIGHTED CLASSIFICATION REPORT

	Precision	Recall	F1-Score
RF	0.88	0.89	0.89
DDT	0.88	0.89	0.89

TABLE VI. MM SET WEIGHTED CLASSIFICATION REPORT

	Precision	Recall	F1-Score
RF	0.78	0.78	0.78
DDT	0.77	0.76	0.76

V. DISCUSSION

From the previous results it is obvious that each distilled DT performed equally good as every RF. The average accuracy of both classifiers in each data set was remarkably similar and relatively high. For instance, in the BCC set, the average accuracy of the RF was 83.33% while the distilled DT got 79.17%. It was clearer on the other two data sets, were the accuracies were even closer. In the Fertility set, both classifiers got 88.88%, while in the third one the difference was less than 2% (76.17% vs. 77.72%).

Looking at Tables I-III, it is shown that the distilled DT outperformed the binary DT in all data sets. The latter indicates that training a DT with the class probabilities of the RF yield better predictions than fitting a DT with raw data. Taking the above mentioned into consideration, it can be said that it is possible to extract the knowledge from an ensemble method and use it to fit a student model with success. In this case, the information within a black-box classifier (RF) was injected into a white-box model (DT).

From Tables IV-VI, it can be said that precision and recall values (specificity and sensitivity) are relatively high. The aforementioned is important, because a high precision is useless if the recall value is low, and vice versa. Also, all F1-Scores are close to 1, which proves that the relation between precision and recall, for every classifier and each data set, is adequate.

When comparing the performance of the ensemble and student models against other supervised learning classifiers, results highlighted the qualities of RF. In two of the data sets RF got the highest accuracy values (BBC and Fertility sets).

In the MM set, the best model was the SVM with 80.33%. However, in this particular data set, all the classifiers behaved very similar. The lowest accuracy was 75.65% (binary DT). Therefore, it is proven that RF are excellent classifiers due to their high accuracy values, as mentioned in [5].

Another important aspect to discuss is the visual representation of the DT models. As mentioned above, those graphics are in Appendix 1. The binary decision trees for the BCC and Fertility sets have 10 and 9 splits respectively, they do not seem to be over or underfitted. On the contrary, the MM DT had 19 splits and looks overfitted. The multiclass DTs (distilled) before binarization look good, except the one of the MM set (which looks overfitted as well). A possible explanation could be that the classes' means within attributes were to close, causing the classification task to be more complex.

As [2] mentioned, in order to confirm that the distilled method learned correctly from the ensemble method, the results should be reproducible and consistent. Since this experiment showed consistently that the distilled DT can perform and achieve similar results to the RF classifier, it is possible to affirm that the extracted knowledge was passed to a simpler model successfully. In fact, [2] also achieved to construct a DT that mimicked the behavior of a RF. As well, [7] was able to extract the knowledge of complex neural networks into a single one. Therefore, it is necessary to continue the research on finding simpler models that learn from more complex methods.

VI. Conclusion

Different attempts to reduce the complexity of machine learning models have been done. In this work, knowledge distillation was the studied approach. This technique aims to use the knowledge of complex method to train a simpler one. For this study, the classes' probabilities of a RF were used to train a DT.

In order to do so, a python function was coded in Spyder. The program reads the data set and split them into different sets. It trains, tune and predict classes with different classifiers. It also uses a binning technique to transform a binary classification problem into a multiclass one, and then binarize the result to yield a new data set that trains the student model.

This "distillation technique" was tested in three different data sets from the medical field. After building the distilled DTs, their performance was contrasted with the original RF and the other supervised learning classifiers. The main finding showed that the student model was able to perform classification tasks within the same range of accuracy as the ensemble method (RF). The latter occurred in the three cases, which gives credibility to the study due to reproducibility. Some concerns were risen when the DTs were plotted, since two out of six returned overfitted. Data distribution and the closeness of the values are possible causes of this situation.

On other matters, the qualities of RF such as top-performance and high-accuracy values, were corroborated. The reason is because it was the model with the best performance in 2 out of three data sets. Therefore, it was useful to achieve a white-box model that provided an insight of how RF make their final predictions internally. This should encourage research to find models that reduce the complexity of other ensemble methods. As mentioned before, in medicine this is of high interest, since black-box algorithms (such as RF) are being used to diagnose.

References

[1] G. Papamakarios, *Distilling Model Knowledge,* Edinburgh: Cornell University, 2015.

[2] Y. Zhou, Z. Zhou and G. Hooker, "Approximation Trees: Statistical Stability in Model Distillation," *Arxiv ,* vol. 1808.07573, pp. 1-30, 2018.

[3] J. VanderPlas, "Python Data Science Handbook," 25 March 2016. [Online]. Available: https://jakevdp.github.io/PythonDataScienceHandbook/05.08-random-forests.html. [Accessed 09 April 2019].

[4] N. Donges and N. Donges, "The Random Forest Algorithm," Towards Data Science, 22 February 2018. [Online]. Available: https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd. [Accessed 09 April 2019].

[5] G. Biau and E. Scornet, "A Random Forest Guided Tour," vol. 25, no. 2, pp. 1-42, 2016.

[6] A. Navlani, "Decision Tree Classification in Python," DataCamp, 28 December 2018. [Online]. Available: https://www.datacamp.com/community/tutorials/decision-tree-classification-python. [Accessed 10 April 2019].

[7] G. Hinton, O. Vinyals and J. Dean, "Distilling the knowledge in a neural network," *Arxiv,* pp. 1-9, 2015.

[8] X. Jin, T. Zhang, L. Li, H. Wu and B. Sun, "Lesion Recgonition Method of Liver CT Images Based on Random Forest," in *8th International Conference on Information Technology in Medicine and Education (ITME)*, Fuzhou, China, 2016.

[9] Z. Li, S. Reza, Y. Hua, M. Mao, Y. Qiu and K. Najarian, "Classifying osteosarcoma patients using machine learning approaches," in *39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Seogwipo, South Korea, 2017.

[10] J. Park, S. Lee and K. Kang, "Arryhtmia detection using amplitude difference features based on random forest," in *37th Annual International Conference of the IEEE Enginering in Medicine and Biology Society (EMBC)*, Milan, Italy, 2015.

[11] Z. Masetic and A. Subasi, "Congestive heart failure detection using random forest classifier," *Pub Med,* pp. 54-64, 2016.

[12] M. Patrício, J. Pereira, P. Cirsóstomo, P. Matafome, R. Gomes, R. Seica and F. Caramelo, "Using resistin, glucose, age and BMI to predict the presence of breast cancer," 2017 December 05. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Coimbra. [Accessed 15 February 2019].

[13] D. Gil, J. Girela, J. De Juan, J. Gomez-Torres and M. Johnson, "Predicting seminal quality with artificial intelligence methods," Expert Systems with Applications, 2012. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Fertility.. [Accessed 2019 February 15].

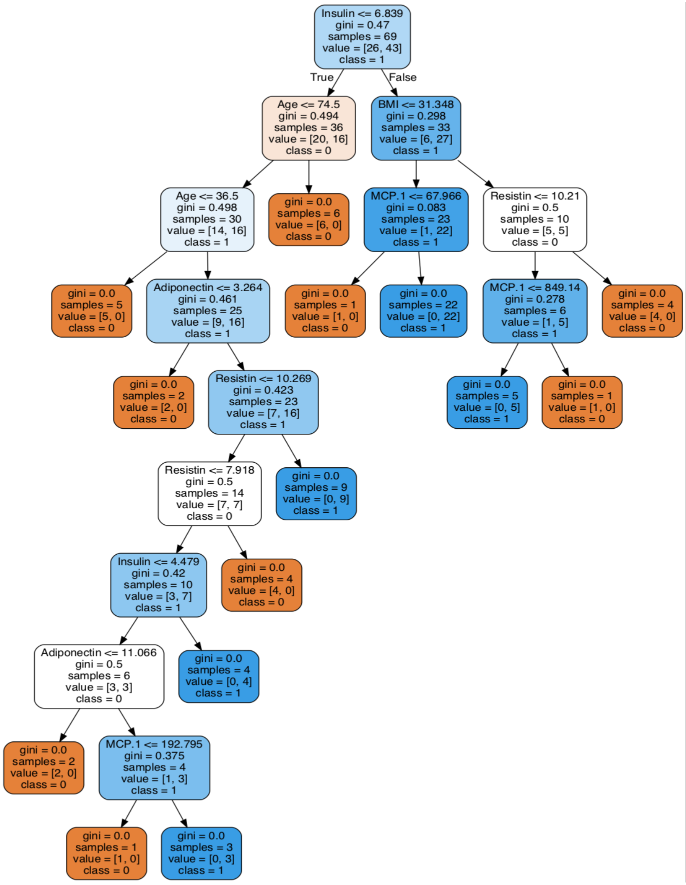
[14] M. Elter, R. Schulz-Wendtland and T. Wittenberg, "The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process," Medican Physics, 2007. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Mammographic+Mass.. [Accessed 2019 February 15].

[15] O. Aguilar, *Project 3: Knowledge distillation from random forests (Supervised Learning),* Colchester: University of Essex, 2019.

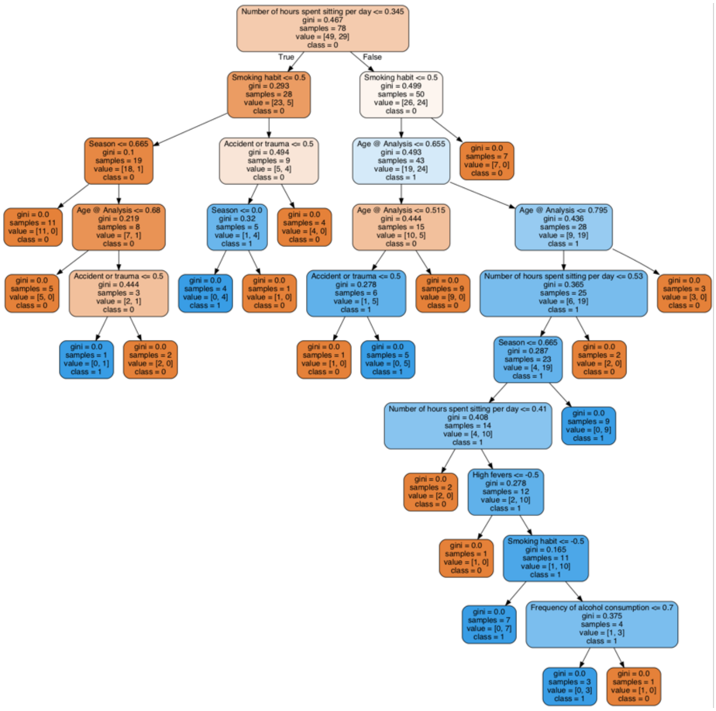
[16] T. Shah, "About Train, Validation and Test Sets in Machine Learning," Towards Data Science, 06 December 2017. [Online]. Available: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7. [Accessed 12 April 2019].

APPENDIX

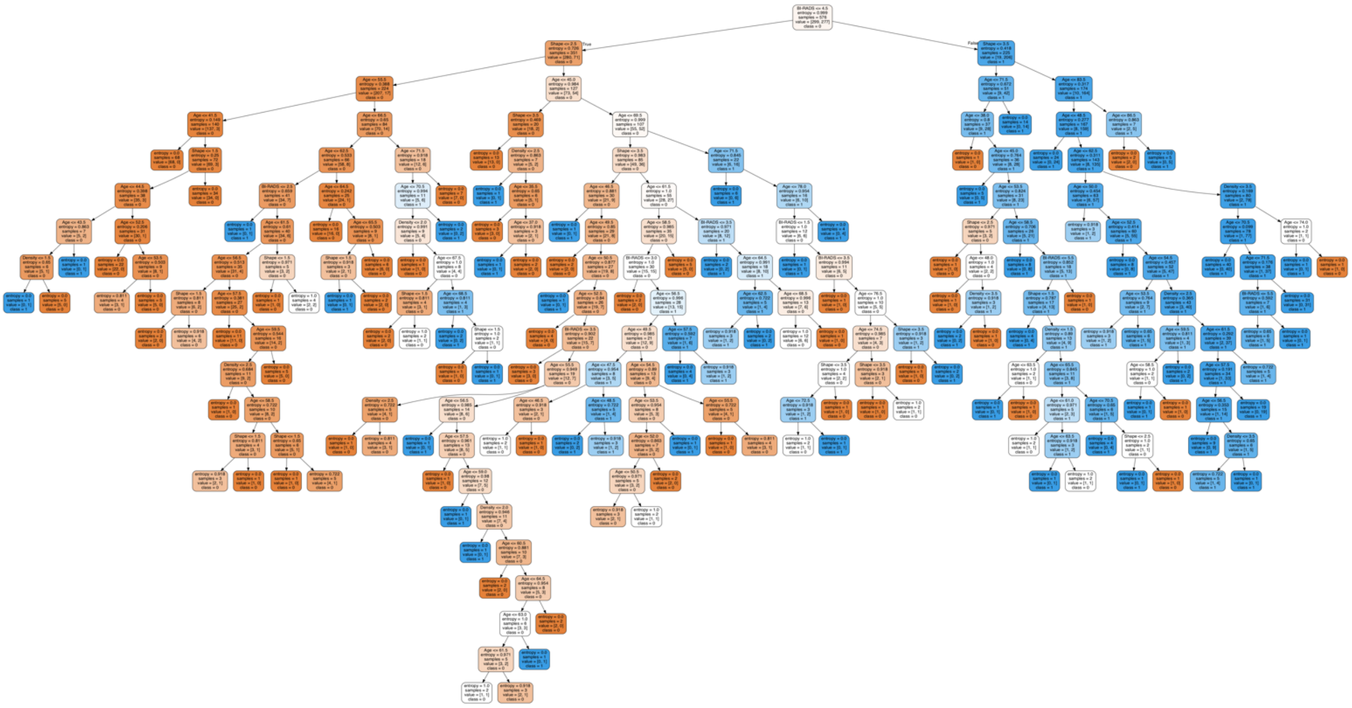
1. Plotted trees



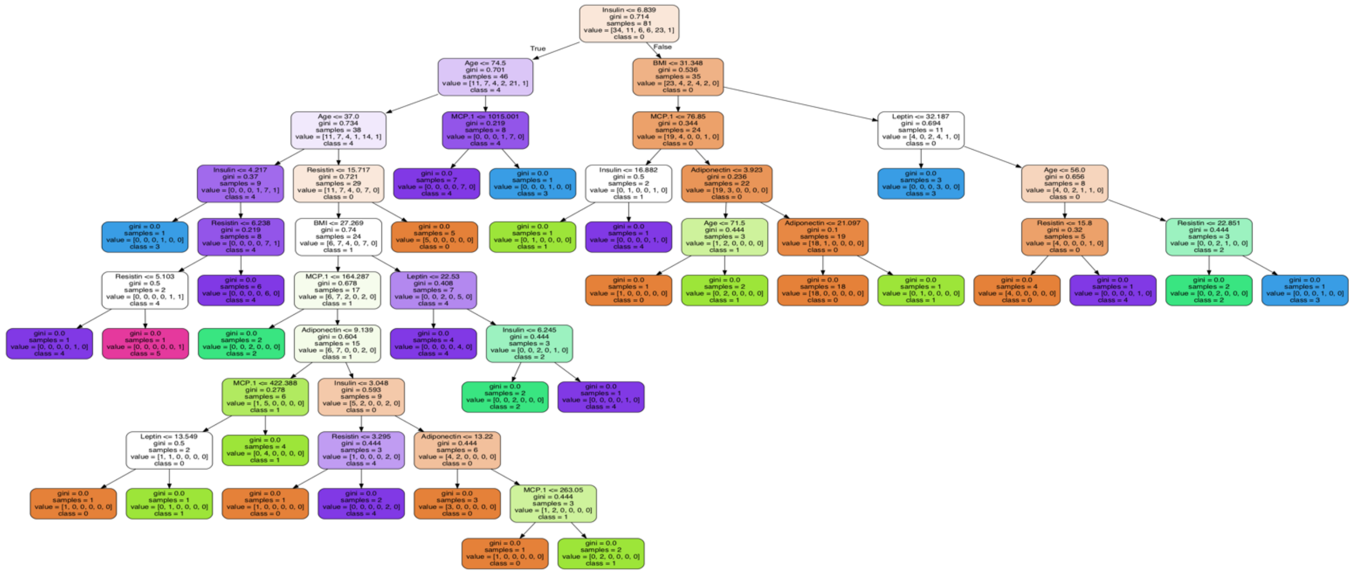
BCC Data Set Binary DT



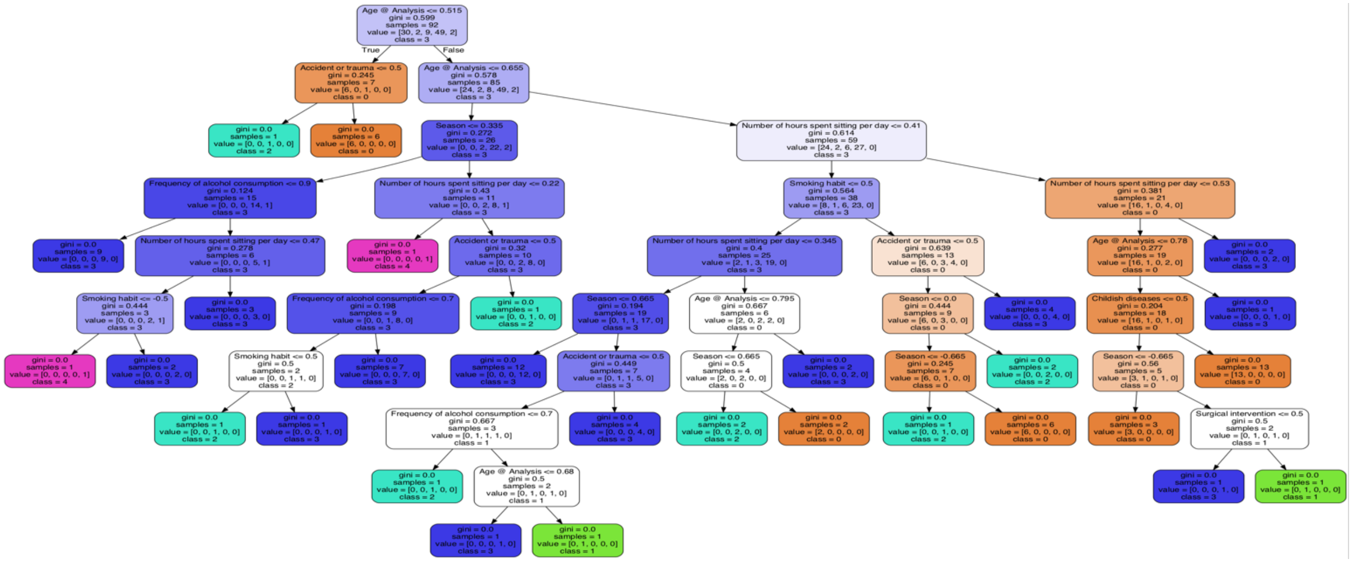
Fertility Data Set Binary DT



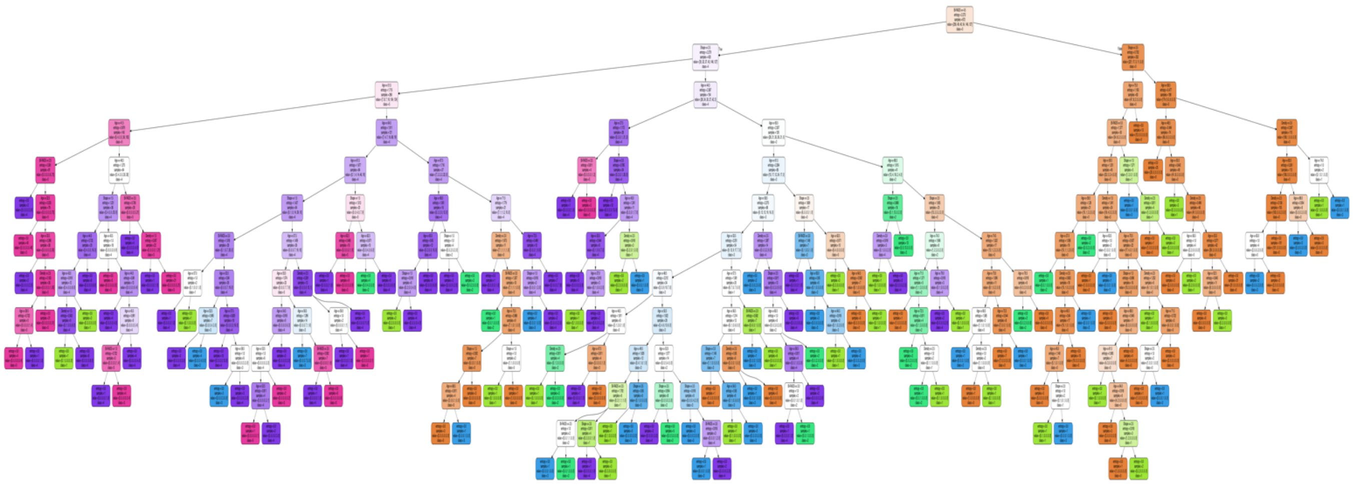
MM Data Set Binary DT


Insulin <= 6.839 gini = 0.714 samples = 81 value = [34, 11, 6, 6, 23, 1] class = 0
True
False
Age <= 74.5 gini = 0.701 samples = 46 value = [11, 7, 4, 2, 21, 1] class = 4
BMI <= 31.348 gini = 0.536 samples = 35 value = [23, 4, 2, 4, 2, 0] class = 0
Age <= 37.0 gini = 0.734 samples = 38 value = [11, 7, 4, 1, 14, 1] class = 4
MCP.1 <= 1015.001 gini = 0.219 samples = 8 value = [0, 0, 0, 1, 7, 0] class = 4
MCP.1 <= 76.85 gini = 0.344 samples = 24 value = [19, 4, 0, 0, 1, 0] class = 0
Leptin <= 32.187 gini = 0.694 samples = 11 value = [4, 0, 2, 4, 1, 0] class = 0
Insulin <= 4.217 gini = 0.37 samples = 9 value = [0, 0, 0, 1, 7, 1] class = 4
Resistin <= 15.717 gini = 0.721 samples = 29 value = [11, 7, 4, 0, 7, 0] class = 0
gini = 0.0 samples = 7 value = [0, 0, 0, 0, 7, 0] class = 4
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0, 0] class = 3
Insulin <= 16.882 gini = 0.5 samples = 2 value = [0, 1, 0, 0, 1, 0] class = 1
Adiponectin <= 3.923 gini = 0.236 samples = 22 value = [19, 3, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 3 value = [0, 0, 0, 3, 0, 0] class = 3
Age <= 56.0 gini = 0.656 samples = 8 value = [4, 0, 2, 1, 1, 0] class = 0
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0, 0] class = 3
Resistin <= 6.238 gini = 0.219 samples = 8 value = [0, 0, 0, 0, 7, 1] class = 4
BMI <= 27.269 gini = 0.74 samples = 24 value = [6, 7, 4, 0, 7, 0] class = 1
gini = 0.0 samples = 5 value = [5, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [0, 1, 0, 0, 0, 0] class = 1
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 1, 0] class = 4
Age <= 71.5 gini = 0.444 samples = 3 value = [1, 2, 0, 0, 0, 0] class = 1
Adiponectin <= 21.097 gini = 0.1 samples = 19 value = [18, 1, 0, 0, 0, 0] class = 0
Resistin <= 15.8 gini = 0.32 samples = 5 value = [4, 0, 0, 0, 1, 0] class = 0
Resistin <= 22.851 gini = 0.444 samples = 3 value = [0, 0, 2, 1, 0, 0] class = 2
Resistin <= 5.103 gini = 0.5 samples = 2 value = [0, 0, 0, 0, 1, 1] class = 4
gini = 0.0 samples = 6 value = [0, 0, 0, 0, 6, 0] class = 4
MCP.1 <= 164.287 gini = 0.678 samples = 17 value = [6, 7, 2, 0, 2, 0] class = 1
Leptin <= 22.53 gini = 0.408 samples = 7 value = [0, 0, 2, 0, 5, 0] class = 4
gini = 0.0 samples = 1 value = [1, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 2 value = [0, 2, 0, 0, 0, 0] class = 1
gini = 0.0 samples = 18 value = [18, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [0, 1, 0, 0, 0, 0] class = 1
gini = 0.0 samples = 4 value = [4, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 1, 0] class = 4
gini = 0.0 samples = 2 value = [0, 0, 2, 0, 0, 0] class = 2
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0, 0] class = 3
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 1, 0] class = 4
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 0, 1] class = 5
gini = 0.0 samples = 2 value = [0, 0, 2, 0, 0, 0] class = 2
Adiponectin <= 9.139 gini = 0.604 samples = 15 value = [6, 7, 0, 0, 2, 0] class = 1
gini = 0.0 samples = 4 value = [0, 0, 0, 0, 4, 0] class = 4
Insulin <= 6.245 gini = 0.444 samples = 3 value = [0, 0, 2, 0, 1, 0] class = 2
MCP.1 <= 422.388 gini = 0.278 samples = 6 value = [1, 5, 0, 0, 0, 0] class = 1
Insulin <= 3.048 gini = 0.593 samples = 9 value = [5, 2, 0, 0, 2, 0] class = 0
gini = 0.0 samples = 2 value = [0, 0, 2, 0, 0, 0] class = 2
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 1, 0] class = 4
Leptin <= 13.549 gini = 0.5 samples = 2 value = [1, 1, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 4 value = [0, 4, 0, 0, 0, 0] class = 1
Resistin <= 3.295 gini = 0.444 samples = 3 value = [1, 0, 0, 0, 2, 0] class = 4
Adiponectin <= 13.22 gini = 0.444 samples = 6 value = [4, 2, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [1, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [0, 1, 0, 0, 0, 0] class = 1
gini = 0.0 samples = 1 value = [1, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 2 value = [0, 0, 0, 0, 2, 0] class = 4
gini = 0.0 samples = 3 value = [3, 0, 0, 0, 0, 0] class = 0
MCP.1 <= 263.05 gini = 0.444 samples = 3 value = [1, 2, 0, 0, 0, 0] class = 1
gini = 0.0 samples = 1 value = [1, 0, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 2 value = [0, 2, 0, 0, 0, 0] class = 1

BCC Data Set Distilled DT (Multiclass)


Age @ Analysis <= 0.515 gini = 0.599 samples = 92 value = [30, 2, 9, 49, 2] class = 3
True
False
Accident or trauma <= 0.5 gini = 0.245 samples = 7 value = [6, 0, 1, 0, 0] class = 0
Age @ Analysis <= 0.655 gini = 0.578 samples = 85 value = [24, 2, 8, 49, 2] class = 3
gini = 0.0 samples = 1 value = [0, 0, 1, 0, 0] class = 2
gini = 0.0 samples = 6 value = [6, 0, 0, 0, 0] class = 0
Season <= 0.335 gini = 0.272 samples = 26 value = [0, 0, 2, 22, 2] class = 3
Number of hours spent sitting per day <= 0.41 gini = 0.614 samples = 59 value = [24, 2, 6, 27, 0] class = 3
Frequency of alcohol consumption <= 0.9 gini = 0.124 samples = 15 value = [0, 0, 0, 14, 1] class = 3
Number of hours spent sitting per day <= 0.22 gini = 0.43 samples = 11 value = [0, 0, 2, 8, 1] class = 3
Smoking habit <= 0.5 gini = 0.564 samples = 38 value = [8, 1, 6, 23, 0] class = 3
Number of hours spent sitting per day <= 0.53 gini = 0.381 samples = 21 value = [16, 1, 0, 4, 0] class = 0
gini = 0.0 samples = 9 value = [0, 0, 0, 9, 0] class = 3
Number of hours spent sitting per day <= 0.47 gini = 0.278 samples = 6 value = [0, 0, 0, 5, 1] class = 3
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 1] class = 4
Accident or trauma <= 0.5 gini = 0.32 samples = 10 value = [0, 0, 2, 8, 0] class = 3
Number of hours spent sitting per day <= 0.345 gini = 0.4 samples = 25 value = [2, 1, 3, 19, 0] class = 3
Accident or trauma <= 0.5 gini = 0.639 samples = 13 value = [6, 0, 3, 4, 0] class = 0
Age @ Analysis <= 0.78 gini = 0.277 samples = 19 value = [16, 1, 0, 2, 0] class = 0
gini = 0.0 samples = 2 value = [0, 0, 0, 2, 0] class = 3
Smoking habit <= -0.5 gini = 0.444 samples = 3 value = [0, 0, 0, 2, 1] class = 3
gini = 0.0 samples = 3 value = [0, 0, 0, 3, 0] class = 3
Frequency of alcohol consumption <= 0.7 gini = 0.198 samples = 9 value = [0, 0, 1, 8, 0] class = 3
gini = 0.0 samples = 1 value = [0, 0, 1, 0, 0] class = 2
Season <= 0.665 gini = 0.194 samples = 19 value = [0, 1, 1, 17, 0] class = 3
Age @ Analysis <= 0.795 gini = 0.667 samples = 6 value = [2, 0, 2, 2, 0] class = 0
Season <= 0.0 gini = 0.444 samples = 9 value = [6, 0, 3, 0, 0] class = 0
gini = 0.0 samples = 4 value = [0, 0, 0, 4, 0] class = 3
Childish diseases <= 0.5 gini = 0.204 samples = 18 value = [16, 1, 0, 1, 0] class = 0
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0] class = 3
gini = 0.0 samples = 1 value = [0, 0, 0, 0, 1] class = 4
gini = 0.0 samples = 2 value = [0, 0, 0, 2, 0] class = 3
Smoking habit <= 0.5 gini = 0.5 samples = 2 value = [0, 0, 1, 1, 0] class = 2
gini = 0.0 samples = 7 value = [0, 0, 0, 7, 0] class = 3
gini = 0.0 samples = 12 value = [0, 0, 0, 12, 0] class = 3
Accident or trauma <= 0.5 gini = 0.449 samples = 7 value = [0, 1, 1, 5, 0] class = 3
Season <= 0.665 gini = 0.5 samples = 4 value = [2, 0, 2, 0, 0] class = 0
gini = 0.0 samples = 2 value = [0, 0, 0, 2, 0] class = 3
Season <= -0.665 gini = 0.245 samples = 7 value = [6, 0, 1, 0, 0] class = 0
gini = 0.0 samples = 2 value = [0, 0, 2, 0, 0] class = 2
Season <= -0.665 gini = 0.56 samples = 5 value = [3, 1, 0, 1, 0] class = 0
gini = 0.0 samples = 13 value = [13, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [0, 0, 1, 0, 0] class = 2
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0] class = 3
Frequency of alcohol consumption <= 0.7 gini = 0.667 samples = 3 value = [0, 1, 1, 1, 0] class = 1
gini = 0.0 samples = 4 value = [0, 0, 0, 4, 0] class = 3
gini = 0.0 samples = 2 value = [0, 0, 2, 0, 0] class = 2
gini = 0.0 samples = 2 value = [2, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 1 value = [0, 0, 1, 0, 0] class = 2
gini = 0.0 samples = 6 value = [6, 0, 0, 0, 0] class = 0
gini = 0.0 samples = 3 value = [3, 0, 0, 0, 0] class = 0
Surgical intervention <= 0.5 gini = 0.5 samples = 2 value = [0, 1, 0, 1, 0] class = 1
gini = 0.0 samples = 1 value = [0, 0, 1, 0, 0] class = 2
Age @ Analysis <= 0.68 gini = 0.5 samples = 2 value = [0, 1, 0, 1, 0] class = 1
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0] class = 3
gini = 0.0 samples = 1 value = [0, 1, 0, 0, 0] class = 1
gini = 0.0 samples = 1 value = [0, 0, 0, 1, 0] class = 3
gini = 0.0 samples = 1 value = [0, 1, 0, 0, 0] class = 1

Fertility Data Set Distilled DT (Multiclass)



MM Data Set Distilled DT (Multiclass)

2. Classification reports

A. BCC Data Set

BCC Data Set – Random Forest

	Precision	Recall	F1-Score	Support
0- Healthy	0.90	0.75	0.82	12
1- Unhealthy	0.79	0.92	0.85	12
Micro Avg	0.83	0.83	0.83	24
Macro Avg	0.84	0.83	0.83	24
Weighted Avg	0.84	0.83	0.83	24

BCC Data Set – Binry Decision Tree

	Precision	Recall	F1-Score	Support
0- Healthy	0.75	0.50	0.60	12
1- Unhealthy	0.62	0.83	0.71	12
Micro Avg	0.67	0.67	0.67	24
Macro Avg	0.69	0.67	0.66	24
Weighted Avg	0.69	0.67	0.66	24

BCC Data Set – Binarized Multiclass Decision Tree (Distilled)

	Precision	Recall	F1-Score	Support
0- Healthy	0.82	0.75	0.78	12
1- Unhealthy	0.77	0.83	0.80	12
Micro Avg	0.79	0.79	0.79	24
Macro Avg	0.79	0.79	0.79	24
Weighted Avg	0.79	0.79	0.79	24

BCC Data Set – SVM

	Precision	Recall	F1-Score	Support
0- Healthy	0.80	0.33	0.47	12
1- Unhealthy	0.58	0.92	0.71	12
Micro Avg	0.62	0.62	0.62	24
Macro Avg	0.69	0.62	0.59	24
Weighted Avg	0.69	0.62	0.59	24

BCC Data Set – K-NN

	Precision	Recall	F1-Score	Support
0- Healthy	0.80	0.33	0.47	12
1- Unhealthy	0.58	0.92	0.71	12
Micro Avg	0.62	0.62	0.62	24
Macro Avg	0.69	0.62	0.59	24
Weighted Avg	0.69	0.62	0.59	24

B. Fertility Data Set

Fertility Data Set – Random Forest

	Precision	Recall	F1-Score	Support
0- Healthy	0.91	0.95	0.93	21
1- Unhealthy	0.80	0.67	0.73	6
Micro Avg	0.89	0.89	0.89	27
Macro Avg	0.85	0.81	0.83	27
Weighted Avg	0.88	0.89	0.89	27

Fertility Data Set – Binry Decision Tree

	Precision	Recall	F1-Score	Support
0- Healthy	0.90	0.90	0.90	21
1- Unhealthy	0.67	0.67	0.67	6
Micro Avg	0.85	0.85	0.85	27
Macro Avg	0.79	0.79	0.79	27
Weighted Avg	0.85	0.85	0.85	27

Fertility Data Set – Binarized Multiclass Decision Tree (Distilled)

	Precision	Recall	F1-Score	Support
0- Healthy	0.91	0.95	0.93	21
1- Unhealthy	0.80	0.67	0.73	6
Micro Avg	0.89	0.89	0.89	27
Macro Avg	0.85	0.81	0.83	27
Weighted Avg	0.88	0.89	0.89	27

Fertility Data Set – SVM

	Precision	Recall	F1-Score	Support
0- Healthy	0.88	0.71	0.79	21
1- Unhealthy	0.40	0.67	0.50	6
Micro Avg	0.70	0.70	0.70	27
Macro Avg	0.64	0.69	0.64	27
Weighted Avg	0.78	0.70	0.73	27

Fertility Data Set – K-NN

	Precision	Recall	F1-Score	Support
0- Healthy	0.86	0.86	0.86	21
1- Unhealthy	0.50	0.50	0.50	6
Micro Avg	0.78	0.78	0.78	27
Macro Avg	0.68	0.68	0.68	27
Weighted Avg	0.78	0.78	0.78	27

C. Mammoraphic Mass Data Set

MM Data Set – Random Forest

	Precision	Recall	F1-Score	Support
0- Healthy	0.84	0.77	0.80	114
1- Unhealthy	0.70	0.78	0.74	79
Micro Avg	0.78	0.78	0.78	193
Macro Avg	0.77	0.78	0.77	193
Weighted Avg	0.78	0.78	0.78	193

MM Data Set – Binry Decision Tree

	Precision	Recall	F1-Score	Support
0- Healthy	0.80	0.79	0.79	114
1- Unhealthy	0.70	0.71	0.70	79
Micro Avg	0.76	0.76	0.76	193
Macro Avg	0.75	0.75	0.75	193
Weighted Avg	0.76	0.76	0.76	193

MM Data Set – Binarized Multiclass Decision Tree (Distilled)

	Precision	Recall	F1-Score	Support
0- Healthy	0.83	0.75	0.79	114
1- Unhealthy	0.69	0.77	0.73	79
Micro Avg	0.76	0.76	0.76	193
Macro Avg	0.76	0.76	0.76	193
Weighted Avg	0.77	0.76	0.76	193

MM Data Set – SVM

	Precision	Recall	F1-Score	Support
0- Healthy	0.82	0.87	0.84	114
1- Unhealthy	0.79	0.72	0.75	79
Micro Avg	0.81	0.81	0.81	193
Macro Avg	0.80	0.79	0.80	193
Weighted Avg	0.81	0.81	0.81	193

MM Data Set – K-NN

	Precision	Recall	F1-Score	Support
0- Healthy	0.82	0.82	0.82	114
1- Unhealthy	0.74	0.75	0.74	79
Micro Avg	0.79	0.79	0.79	193
Macro Avg	0.78	0.78	0.78	193
Weighted Avg	0.79	0.79	0.79	193