# Application of ML and AI to the medical field, classification of patients with lung diseases

C.D. Nguyen[a], L Pisano[a]

[a]*Department of Data Science, EURECOM, Sophia Antipolis 06410 Biot.*

## 1. Introduction

In the medical field Artificial Intelligence (AI) is the use of Machine Learning (ML) models to search through medical data and uncover insights to help improve health outcomes and patient experiences [1]. The most common roles for AI in the medical field are clinical decision support[1] and imaging analysis[2]. Some are also trying to generate prediction models for early detection and treatment of diseases.

When it comes to diagnosis of pulmonary diseases there are some common tests that usually are done; some are simple like Spirometry, where you breath through a tube, while others are quite bothersome and intrusive, such as CT scans or Bronchoscopy. However, before all these modern methods of testing for pulmonary diseases were introduced, doctors usually relied on analog stethoscopes to assess the hearts, lungs and bowels of patients. In modern medicine, analog stethoscopes has become obsolete and replaced by digital stethoscopes and imaging tools. The digital stethoscope is able to convert an acoustic signal to electronic signals which can be processed, enhanced and digitalized.

The goal of this project is to explore the possibilities of detecting pulmonary diseases by analysing a respiratory sound database provided on Kaggle [2].

In order to do that, we divide our problem into two tasks; first we explore the ability to classify if a patient has a pulmonary disease or not by analysing the demographic information and whether or not wheezels and crackles are present. Afterwards we explore the multi-class diagnostic capabilities using the sound recordings. Thus, we have a binary-class and multi-class classification problems.

For our studies, we are analyzing different classification algorithms such as the decisional tree model and random forest model for the binary classification problem. Furthermore, we are using a soft voting classifier model based on the support vector machine (SVM) model, random forest (RF) model, and the K nearest neighbor (KNN) model. Finally, we are going to compare their result in terms of accuracy and error and to discuss their behavior with the data given in input.

## 2. Related work

As the respiration sound database has been retrieved from Kaggle there are plenty of work done on the related problem. Some try to classify healthy and unhealthy lungs, while others are trying to tackle the multi-class classification problem. The approach is also dissimilar as the use of features are varying from using the annotated data, to use of the audio recordings. We will review their work systematically by first going through their approach of preprocessing, choice of model, and then interpret the results.

The challenge with the dataset is that it is heavily imbalanced. Out of 126 patients who have participated towards the database, 63 have COPD[3], one has Asthma, and three have LRTI[4]. Although there are only 126 participants, the total amount of recordings are up at 920 because a patient's recording can be recorded with different digital stethoscope models.

To counter the imbalance of the dataset, there is a concensus among the contributors [3] that removing the heavily underrepresented classes is desirable. Thus, LRTI and Asthma has been cut from the dataset, which we agree with. Furthermore, each recording is of a patient's respiratory cycle, and it has been suggested by contributor *Dana Elisa Nicolas* [4] to create subslices of each sound based on the respiratory cycle interval. Doing this will increase the number of audio files from 920 to 6860. This method will allow us to gain a much bigger dataset, however, it does not solve the imbalance problem as the number of subslices of COPD increases to 5746.

The feature extraction of audio signals are plenty. *Trends in audio signal feature extraction methods* [5] explains all the state-of-the-art feature extraction methods for their respective area of application. For *Nicolas'* case she is using the STFT method, while another contributor, *DHoa* [6], uses the Mel Spectrogram method. We've decided to

[1]Help making decisions about treatments and medications by providing quick access to information.

[2]Analyse CT scans, X-rays, MRIs for anomalies that a human radiologist might miss.

[3]Chronic Obstructive Pulmonary Disease.

[4]Lower Respiratory Tract Infections.

experiment with MFCC, Chroma CSTFT and Mel Spectrograms as these are the most commonly used features. The features are technically equivalent to images which means it is also possible to apply computer vision techniques to them, such as measurements to counter imbalance.

An article from Medium [7] explains different methods of solving the imbalanced class problem for image classification. The mentioned and most popular approaches are: Weighted class (which is what Mr. Truong and Ms. Nicolas are doing), under-sampling, data augmentation for minority-class, and synthetic minority over-sampling technique (SMOTE). None of the work we have reviewed have chosen to down-sample the data, which we think would be a mistake. This could lead to an accuracy on testing data which would only represent the model performance on COPD classification. An example of such behaviour comes from contributor *ChrisBurlacu* [8] who have gotten an accuracy of 95% by using CNN[5]. A quick look at his work and confusion matrix shows that 165 samples of the test data belongs to the COPD class while the 15 remaining samples are unevenly distributed among the 5 other classes. Thus, the average accuracy[6] would be 95%, while the overall accuracy[7] would be 63%.

We will be using the under-sampling and SMOTE approach to solve our imbalanced class problem. To avoid similar problems as mentioned in the previous paragraph, we will use the *stratify* option of train_test_split()[8], which makes a split so that the proportion of values in the sample produced will be the same as the proportion of values provided.

For the choice of model, we have already seen promising results with a CNN model made by AaronHuang [9], who has managed to get an average and overall accuracy of 96%. This leads us to rather explore the performance of simpler models provided by the sci-kit learn Pytho library. We will experiment and tune classifiers such as: Decision Trees, Support Vector Machines (SVM), Random Forest (RF), Logistic Regression, and K Nearest Neighbour (KNN). From these models we will pick the most prominent ones and run them through a Voting Classifier in order to potentially resolve the errors made by each other.

## 3. Dataset and Features

The dataset we're working with has been provided by two research teams in Portugal and Greece and was published 3 years ago on Kaggle. It includes 920 annotated recordings of varying length - 10s to 90s. These recordings were taken from 126 patients. There are a total of 5.5 hours of recordings containing 6898 respiratory cycles - 1864 contain crackles, 886 contain wheezes and 506 contain both crackles and wheezes. The data includes both clean respiratory sounds as well as noisy recordings that simulate real life conditions. The patients span all age groups - children, adults and the elderly. The Kaggle dataset includes [2]: 920 .wav sound files, 920 annotation .txt files, a text file listing the diagnosis for each patient, a text file explaining the file naming format, a text file containing demographic information for each patient.

As we have a binary- and multiclass classification problem, we will be using the annotated .txt files and patient's demographic for the binary-class classification, and audio files for the multi-class classification.

### *3.1. Binary-class classification*

In the patient's demographic file we have the columns: Patient ID, age, sex, adult BMI, child weight and child height. As the BMI for a child does not exist, the weight and height will be used as explained in the literature [? ].

In the annotation .txt files we have the columns: Beginning of respiratory cycle(s), end of respiratory cycle(s), presence/absence of crackles, and presence/absence of wheezes.

For our study we applied the one-hot encoding on the sex column. In order to make our model as precise as possible with the given data, we counted the number of occurences of wheezes and crackles for each patient. We did this in order to understand if there was a correlation between these values and the respiratory problems.

### *3.2. Multi-class classification*

The reasoning for most of the pre-processing has been discussed and mentioned in Section 1.

As we are dealing with audio signals, there are alot of pre-processing that needs to be done before it is ready to be used for the classification models. Our audio files are .wav with different lengths depending on the amount of respiration cycles. We are using contributor Shivam Sharma's [? ] pre-processing approach of making sub-slices of each .wav file given their average respiration cycle. As the patients respiration cycle varies in length, we had to cut the length of some, as well as pad others with zeros in order to standardize them for our feature extracture method. We also removed the files of patients with Asthma and LRTI as the number of samples was too small to get anything of significance.

As our dataset is heavily imbalanced, we counter the possibility of overfitting for the majority class by pruning the COPD samples down to 600 to take into account that the accuracy can "suffer like third world countrymen"[10].

The data is then split into train/validation/test with the ratio 0.60/0.25/0.20. As described earlier, our features are MFCC, CSTFT and Mel Spectrograms. Following this, we've also normalized our training, validation and test data using the Cepstral Mean and Variance Normalization

[5]Convolutional Neural Networks.

[6]Average accuracy is calculated as the sum of the accuracy figures in column accuracy divided by the number of classes in the test set.

[7]overall accuracy is calculated as the total number of correctly classified pixels (diagonal elements) divided by the total number of test pixels.

[8]Function that splits arrays or matrices into random train and test subsets.

(CVMN) [11] technique. The mean and variance used is from our training data.

Our final approach to handle the imbalance problem is SMOTE. SMOTE is used on our training set in order for our classification models to have a more balanced data to train on.

## 4. Methods

In the first step, we want to explore the ability to classify if a patient has a pulmonary disease or not by analyzing the demographic information and the number of times wheezes and crackles have been detected for each patient.

To predict these values, we used two classifier models which are the decision tree and random forest.

The decision tree algorithm is a non-parametric supervised learning method that is used to predict a value based on decision rules gathered from a data structure. The decisional tree aims to minimize the impurity function, which measures the label purity of the children of a given leaf's tree. In this study we will use the library *sklearn*, more specifically, we will use the class *DecisionTreeClassifier*. From the documentation [12] we know that the impurity function used by default is the *Gini* one.

$$G(S) = \sum_k p_k(1 - p_k) \tag{1}$$

It is well known that the decision trees tend to overfit data with a large number of features. It is really important to understand clearly the tradeoff between the right ratio of samples to the number of features so we do not fall into the overfit trap. To understand which are the most important features in our dataset, we used the library *sklearn.feature_selection*.

Another model we would like to test our dataset with is the random forest. This classifier should increase the accuracy of the prediction since it is composed of several decisional trees. Theoretically, the more trees we have in the forest, the more robust it is. On the other hand, we do not want or need always to use huge tree numbers, but we wish to find a good trade-off based on the accuracy we get and the complexity of the model. It is possible to sum up the classification process of the random forest for an unseen data in the following way:

**Algorithm 1** Random forest prediction pseudocode

```
for each unseen data of the test set do
    for each tree in the random forest do
        - get prediction of x based on tree
        - pick next tree in the random forest
    end
    - make majority voting
    - assign new value to label
    - pick next unseen data
end
```

Let C_b(x) be the class prediction of the bth random-forest tree

$$\begin{gathered} C_r^B(x) = majority\ voting\ C_b(x)_1^B \\ where\ B = 1, ..n\ with\ n \neq 0 \end{gathered} \tag{2}$$

After getting if the patient understudy has a pulmonary disease or not, we would like to try developing multi-classification models to understand what kind of disease the patient has. In this case of study, multiple diseases per patient are not taken into consideration, since the dataset we are using is providing only one (if it is the case) illness per patient.

In order to predict the different diseases, we used three different models, which are KNN, SVM, and Logistic Regression.

In the first case, we apply one a models whhich is based on the nearest neighbors. This classifier, in fact, is generating all the distance between the different data in the graphs using euclidian distance (typically), to retrieve which are the k (settled during design time) closest neighbors starting from unseen data. Even in this case, we define the final label based on the majority voting of the closest points. The known issue of this classifier is the expensiveness of calculating the distance between each data from every point and the problem regarding the assignation of K. When this parameter is too high, we risk to underfit our model, leading it to a generalization issue, meanwhile, when it is too high, the model becomes too specific and fails to generalize well.

The second classifier we took into consideration for our case study is SVM. It is a supervised learning method. In our case, we used the SVM with a linear kernel, so, it has a behavior which is really close to the linear classifier's one. SVM is usually used in the case of high dimensional spaces and it requires to have in input separable data. We chose this model because it is supposed to produce good results even with a low number of samples, as our dataset is providing. In the case of non-linear data, it is possible to invoke the 'poly' kernel of the SVM using *sklearn* library [13] or to use the kernel trick [14]. For this model, we have used the default regularization parameter C.

Lastly, we used is the logistic regression model. The reason why we chose this classifier is just written in the documentation of this class [15]. It is mentioned that 'It can handle both dense and sparse input', so it was good to do not worry about this data and to concentrate our effort for classifying correctly in the parameters given in input such as the number of maximum iterations needed to try converging the solution. For this model, we have used the default norm of the penalty given by *sklearn*, which is equal to L2.

Finally, in order to unify all the work we have done in this phase, we implemented a Voting classification model with all the three previous multi-classification models developed. Even in this case, we used the library *sklearn*, more in deep, the class voting classifier [16]. The voting classification, as the name suggests, allows all the classifiers to vote and the majority is deciding which label will be given as output for the unseen data x.

## 5. Experiments/Results/Discussion

### *5.1. Binary-class classification*

Appling the KBest method, we wanted to remove all the features except the highest-scoring ones. Furthermore, by the documentation [17] we saw that we could specify the number of 'top' features we would like to take into consideration for our case study. As result, we got that the *bestK*, following this algorithm logic, is the child weight. In the second and third place we find the child height and adult BMI, exactly the three features we discarded during the pre-processing phase. Following this data, we got the feature regarding the age of the patient, number of crackles, number of wheezes, and finally the value corresponding to the sex of the patient. The importance of adult BMI value has been confirmed even from the K-fold cross-validation. Based on the entire dataset, we got the ranking of value's importance, if the *optimal* model should include or not this parameter, and the Cross-validation score. Here we attach the results we got from this feature selection algorithm.

**Table 1:** Feature selection using k-fold cross validation

| Column ID | Rank | Optimal | Grid score |
|---|---|---|---|
| PID | 6 | False | 0.83 |
| Age | 3 | False | 0.81 |
| Sex | 2 | False | 0.96 |
| Adult BMI | 1 | True | 0.95 |
| Child Weight | 4 | False | 0.95 |
| Child Height | 5 | False | 0.95 |
| N° wheezes | 1 | True | 0.94 |
| N° crackles | 1 | True | 0.94 |

The optimal value of features to use, suggested by the k-fold cross-validation, is equal to three. In the third column, it is possible to observe which features we should take into consideration based on the k-fold cross validation.

Once obtained this information, we decided to split the study process in two, the first one is following the aim described in chapter 3 (Dataset and Features), so we will use the entire dataset and filter the features, using only the age, sex, number of crackles and number of wheezes to predict the presence or absence of pulmonary diseases. In the second one, we changed direction and, predicting the diagnosis of the patient based on the bestK and k-fold cross validation result we got.

In this report, we started analysing the first one. Having the dataset, we wanted to apply a decisional tree model. To better understand the accuracy, given the different tree depths, we designed a for loop where we saved the error and the accuracy of the decisional tree, to observe its behaviour.

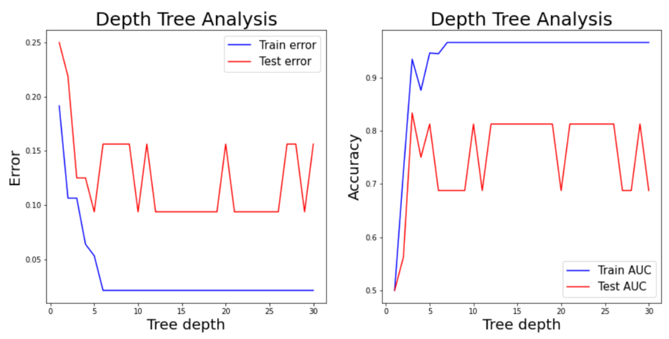


**Figure 5.1:** Decisional tree error and accuracy for different depth.

From the graph above we can see the behavior of the decisional tree with different depth trees. At first sight, it is possible to define that it seems a good model from (around) depth tree 8 up, because, in these cases, we will have the training set (75% of the total dataset available) predicted with no mistakes made by the decisional tree. On the other hand, analyzing figure 4.1, it is possible to see a clear sign of overfitting behavior. This is visible at depth three equals 6. Meanwhile, the training error keeps decreasing slightly, the test error rises up from 0.090 to 0.16. Given these results, we chose to select the decisional tree with a depth equal to 5 with accuracy for the train set of 94.6% and 90.6% for the test.

Since we got pretty good results with this mode, we wanted to check if this accuracy on the test set could have increased using the random forest. From theory we know the random forest can generalize over the data in a better way. This is because the decision tree model gives high importance to a particular set of features meanwhile the random forest chooses features randomly during the training process.This randomized feature selection makes random forest much more accurate than a decision tree [18].

From the study result, we got to notice that the accuracy with respects to the decisional tree one, did not increase as we were expecting. In fact, we got 91.03% accuracy for the testing set, and 96.32% So we conclude that, eventhough the results shown are pretty good, the random forest is not suitable for our study case.

Regarding the bestK features selection we have introduced in the previous chapter, we did some study following the result given by the algorithm presented. Before training the model was necessary to split the dataset into dataset_children and dataset_adults, to properly benefit from the adult_BMI, child_weight, and child_height columns. Once divided the two datasets, we noticed that all the patients who are adults (age $>$ 18) are detected as unhealthy in the original dataset, so it was impossible to run a classification model on these data since there was no chance to predict the wrong label. However, we ran some tests on the children dataset, which, taking into consideration the best features suggested from the features selection algorithms, had lower accuracy than the best one shown in Figure 5.1. Finally, we re-ran again the features selection algorithms and the attributes chosen by them changed radically. So we concluded that the features selected at the start were not optimal once again in the final split dataset.

### *5.2. Multi-class classification*

To decide which models we wanted to include in our Voting Classifier we used the validation set to gain the highest accuracy possible for each model and compared them. We are using overall and average accuracy as a primary metric as we wish to compare our results with existing solutions.

The parameters that have been chosen is a result of iterating through different values of the parameters to gain the highest validation accuracy or until convergence. A visualization of such experiments can be seen in Figure ??

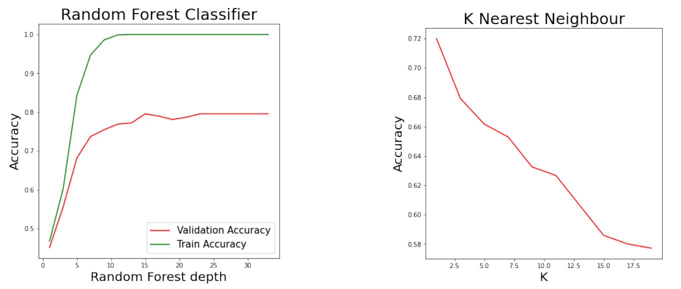


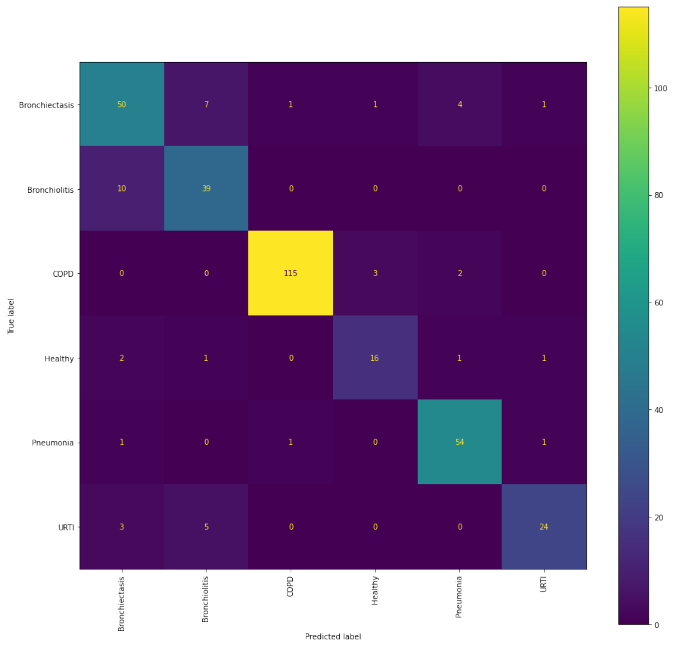
**Figure 5.2:** Analysis of model performance by tuning parameters.

Table 2 shows the complete overview of the models performance on our validation set after training on our training set.

**Table 2:** Highest accuracy gained from tuning of parameters for different models

| **Model** | **Parameters** | **Accuracy** |
|---|---|---|
| Random Forest (RF) | max depth = 27 | 80% |
| Decision Tree | max depth = 100 | 57% |
| SVM | kernel = linear | 78% |
| Logistic Regression | – | 76 |
| K-NN | N=1 | 72% |
| Voting Classifier | SVM, 3-NN, RF | 81% |

Figure 5.3 shows the Voting Classifiers confusion matrix on the unseen test dataset.



| True label | Bronchiectasis | Bronchiolitis | COPD | Healthy | Pneumonia | URTI |
|---|---|---|---|---|---|---|
| Bronchiectasis | 50 | 7 | 1 | 1 | 4 | 1 |
| Bronchiolitis | 10 | 39 | 0 | 0 | 0 | 0 |
| COPD | 0 | 0 | 115 | 3 | 2 | 0 |
| Healthy | 2 | 1 | 0 | 16 | 1 | 1 |
| Pneumonia | 1 | 0 | 1 | 0 | 54 | 1 |
| URTI | 3 | 5 | 0 | 0 | 0 | 24 |

**Figure 5.3:** Voting Classifier model with SVM, RF and 3-NN as estimators.

The results are quite promising and shows that the model works much better on the test set than on the validation. The average accuracy is 83.2% while the overall accuracy is 86.9%. This means that we have managed to fight the imbalanced problem of our dataset. The ability to classify individual classes is quite much better than what we've seen from the other contributors.

## 6. Conclusion

We have realized that high accuracy does not neccessarily mean good results when it comes to a heavily imbalanced dataset. We've tried to perform a binary-class classification using only the annotated and demographic data and gotten great results of 90% accuracy. However, as we know the accuracy could be due to the imbalance problem.

We have used the Voting Classifier with SVM, RF and K-NN to perform our multi-class classification as those models were the ones who performed the best in our final model. The results were promising and showed better results than other work done in our field with regards to average (83.2%) and overall (86.9%) accuracy.

Future work in this field would be to try other feature extraction methods. It would also be interesting to add more features to our model, as in combine all the demographic data with our audio features to see how the performance would be affected.

The code can be accessed from this list: [[19], [20], [21], [22], [23]]

## 7. Contribution

### *7.1. Christian*

Pre-processing of wav files, study in deep of MFCC and features extraction algorithms. Focus on audio files. Implementation of multi-classification algorithm based on audio files. Development of synthetic minority over-sampling technique. Implementation of the voting classification algorithm.

### *7.2. Lorenzo*

Pre-processing of wav files, study in deep of MFCC and features extraction algorithms. Focus on demographic data. Feature selections and implementation of models to predict binary classification (Healthy/Unhealthy) based on annotated data. Implementation of the voting classification algorithm.

## References

[1] IBM: *How is artificial intelligence used in medicine?* https://www.ibm.com/topics/artificial-intelligence-medicine.

[2] Wijayasingha, Lahiru Nuwan: *Respiratory sound database*, 2019. https://www.kaggle.com/vbookshelf/respiratory-sound-database.

[3] *Contributors*, 2019. https://www.kaggle.com/vbookshelf/respiratory-sound-database/code.

[4] *Cnn (part 1): Create subslices for each sound*, 2019. https://www.kaggle.com/danaelisanicolas/cnn-part-1-create-subslices-for-each-sound.

[5] *Trends in audio signal feature extraction methods*, January 2020. https://www.sciencedirect.com/science/article/abs/pii/S0003682X19308795?casa_token=8hRkavweyycAAAAA:v0Sm5lDJ1Reb40IOIs9v1tb6WRXxbZkPu8kw31vk-OvN_4-quWj4LUbzse74LHgMRpEgcrIGTlX3.

[6] *Healthy lung classification spectrogram fast.ai*, 2021. https://www.kaggle.com/dienhoa/healthy-lung-classification-spectrogram-fast-ai.

[7] *How to use smote for dealing with imbalanced image dataset for solving classification problems*, February 2020. https://medium.com/swlh/how-to-use-smote-for-dealing-with-imbalanced-image-dataset-for-solving-classification-problems-3aba7d2b9cad.

[8] *Cnn: Disease classification, linked features (95%)*, 2020. https://www.kaggle.com/markdenton/cnn-disease-classification-linked-features-95.

[9] *Disease classified (acc=96%) by audio variables*, 2020. https://www.kaggle.com/chingchiehhuang/disease-classified-acc-96-by-audio-variables/notebook.

[10] *Funny comment*, 2021. https://www.kaggle.com/shivam316/part-3-feature-extraction-modeling-95-acc/comments.

[11] *Cepstral mean and variance normalization.* https://www.semanticscholar.org/topic/Cepstral-Mean-and-Variance-Normalization/551159.

[12] *Documentation decisional tree sklearn.* https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html.

[13] *Documentation svm sklearn.* https://scikit-learn.org/stable/modules/svm.html.

[14] *Kernel trick svm.* https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f.

[15] *Documentation linear regression sklearn.* https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

[16] *Documentation voting classifier sklearn.* https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html.

[17] *Documentation selectk best sklearn.* https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html.

[18] *Decisional tree vs random forest.* https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/.

[19] *Feature selection notebook.* https://www.kaggle.com/lorenzopisan/feature-selection-database.

[20] *Decisional tree notebook.* https://www.kaggle.com/lorenzopisan/decisional-tree-analysis.

[21] *Random forest notebook.* https://www.kaggle.com/lorenzopisan/random-forest-analysis.

[22] *Multi classification notebook.* https://colab.research.google.com/drive/1kMmwvLSddBNhfl_-Xts1SWemP2FyUeyj?usp=sharing.

[23] *Multi classification notebook.* https://github.com/christng98/MALIS_Project.