**Introduction**

The data is dedicated to classification problem related to the post-operative life expectancy in the lung cancer patients. In particular this dataset presents data of patients, attributes and whether they survive within one year of the thoracic operation. The goal is to understand if there is a way to determine the 1-year postoperative survival of lung cancer patients using patient attributes in the dataset. This could help patients and doctors assess the risks of the surgery and, therefore, decide whether to proceed or evaluate other alternatives.

**Dataset description**

The dataset is available at <http://archive.ics.uci.edu/ml/datasets/Thoracic+Surgery+Data>. According to the main repository site the data was collected retrospectively at Wroclaw Thoracic Surgery Centre for patients who underwent major lung resections for primary lung cancer in the years 2007-2011. The Centre is associated with the Department of Thoracic Surgery of the Medical University of Wroclaw and Lower-Silesian Centre for Pulmonary Diseases, Poland, while the research database constitutes a part of the National Lung Cancer Registry, administered by the Institute of Tuberculosis and Pulmonary Diseases in Warsaw, Poland.

The original dataset was in the form of a Weka ARFF file, so we decided to convert it to a CSV file.

**Attribute information**

There are 11 binary attribute containing object string for T and F value, 3 categorical attributes, containing data in the form of a combination of a string and an int value, and other 3 continuous attributes. The corresponding column descriptions found on the UCI machine learning repository site is shown below:

* **DGN**: Diagnosis - specific combination of ICD-10 codes for primary and secondary as well multiple tumours if any (DGN3, DGN2, DGN4, DGN6, DGN5, DGN8, DGN1)
* **PRE4**: Forced vital capacity - FVC (numeric). Amount of air which can be forcibly exhaled from the lungs after taking the deepest breath possible
* **PRE5**: Volume that has been exhaled at the end of the first second of forced expiration - FEV1 (numeric)
* **PRE6**: Performance status - Zubrod scale (PRZ2, PRZ1, PRZ0)
* **PRE7**: Pain before surgery (T, F)
* **PRE8**: Haemoptysis before surgery (T, F)
* **PRE9**: Dyspnoea before surgery (T, F)
* **PRE10**: Cough before surgery (T, F)
* **PRE11**: Weakness before surgery (T, F)
* **PRE14**: T in clinical TNM - size of the original tumour, from OC11 (smallest) to OC14 (largest) (OC11, OC14, OC12, OC13)
* **PRE17**: Type 2 DM - diabetes mellitus (T, F)
* **PRE19**: MI - Myocardial infarction (Heart Attack) up to 6 months before surgery (T,F)
* **PRE25**: PAD - peripheral arterial diseases (T, F)
* **PRE30**: Smoking (T, F)
* **PRE32**: Asthma (T, F)
* **AGE**: Age at surgery (numeric)
* **Risk1Y**: 1 year survival period - (T) value if died (T, F)

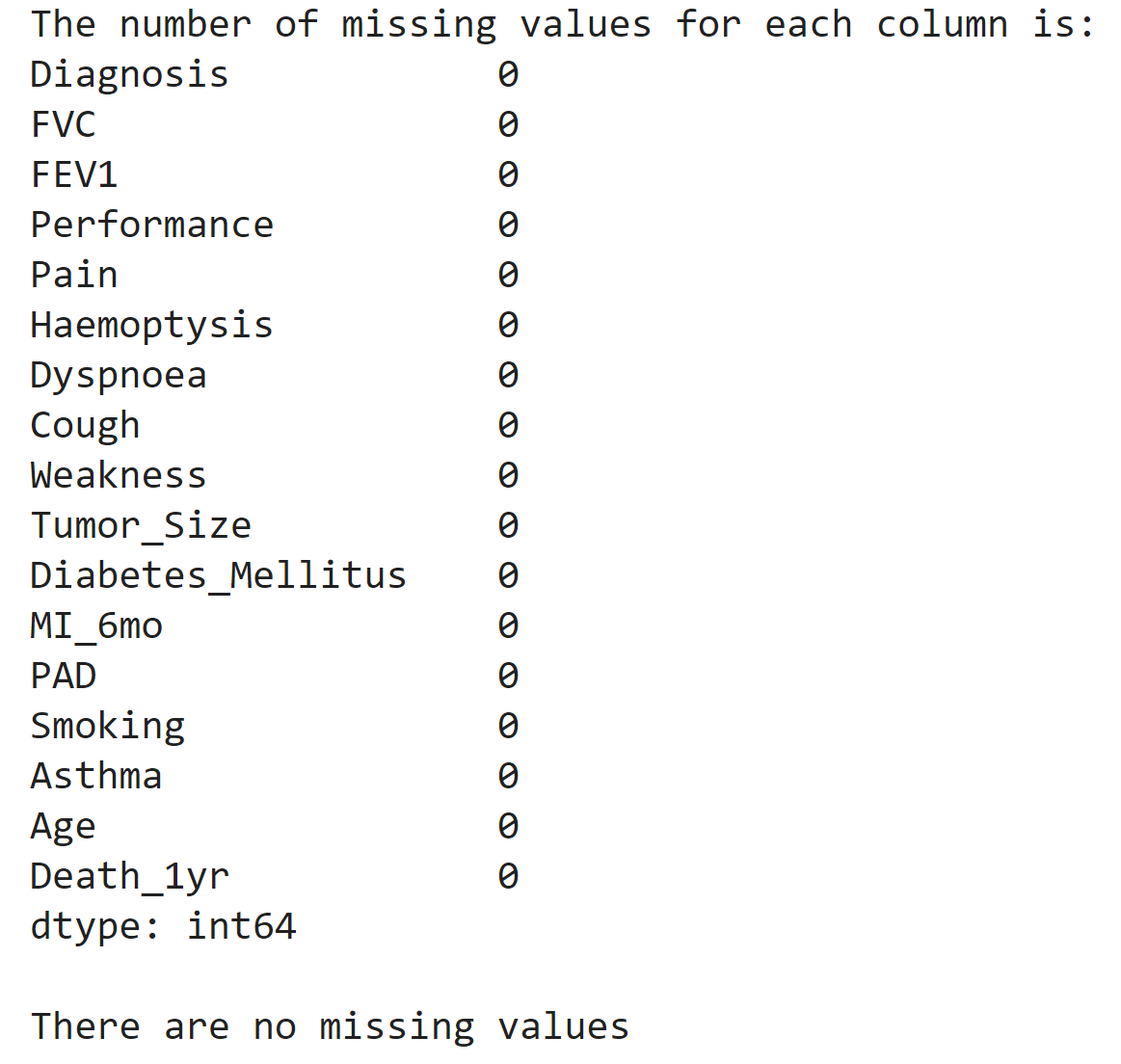
**Dataset cleaning**

We decided to make changes to the original dataset in order to make it more understandable and easier to use. In particular we mapped T-F values in 0-1 values, we removed the id column, because it was not necessary and useless in the description of the patients, and finally we renamed the attributes with more understandable names. The mapping is shown below:

* 'DGN': 'Diagnosis',
* 'PRE4': 'FVC',
* 'PRE5': 'FEV1',
* 'PRE6': 'Performance',
* 'PRE7': 'Pain',
* 'PRE8': 'Haemoptysis',
* 'PRE9': 'Dyspnoea',
* 'PRE10': 'Cough',
* 'PRE11': 'Weakness',
* 'PRE14': 'Tumor\_Size',
* 'PRE17': 'Diabetes\_Mellitus',
* 'PRE19': 'MI\_6mo',
* 'PRE25': 'PAD',
* 'PRE30': 'Smoking',
* 'PRE32': 'Asthma',
* 'AGE': 'Age',
* 'Risk1Yr': 'Death\_1yr'

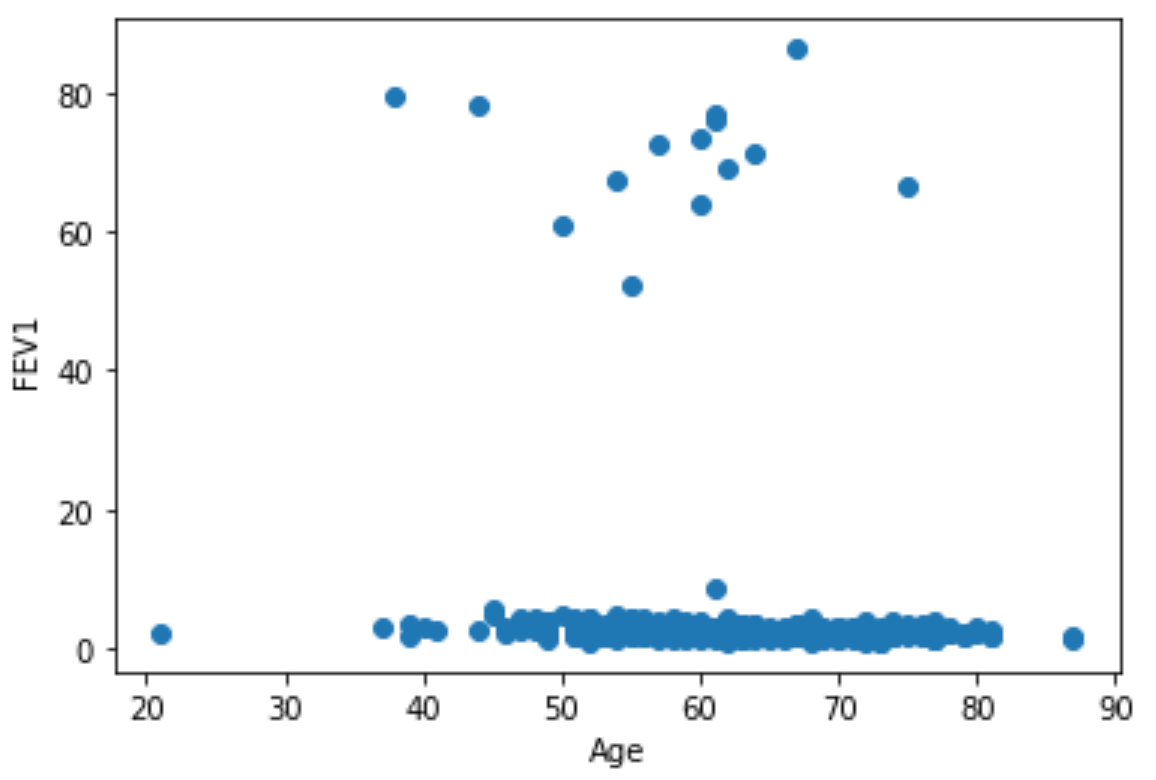
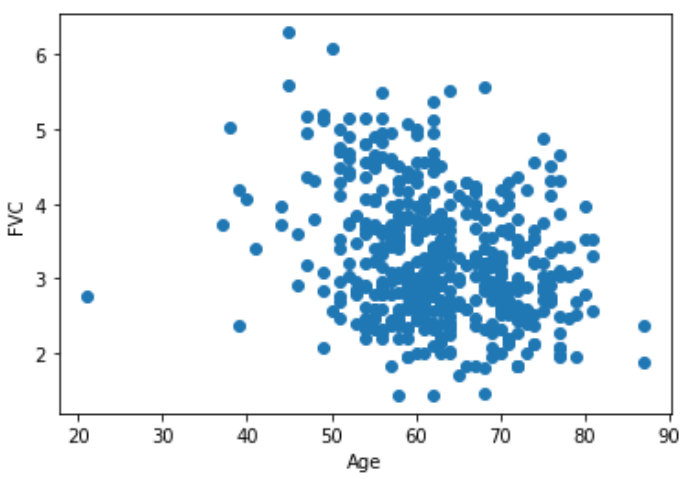
**Missing Values**

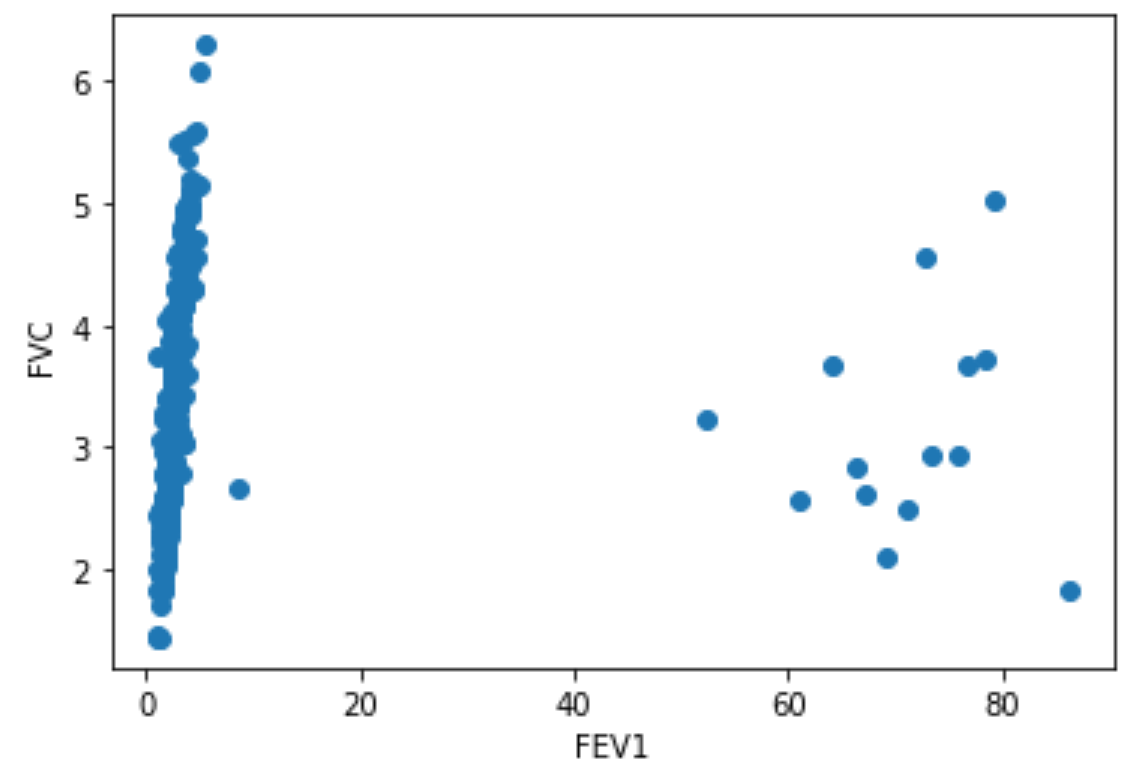
Verifying the presence of miss values, we found that there are none.



**Outliers detection**

For the outlier detection, there are only three continuous attributes to be considered: FVC, FEV1 and Age. The scatter plots of those attributes show the presence of some outliers:





So we decided to remove all data with a FEV1 value grater than 7. In this way we removed 15 outliers. Observing the Age attribute it is possible to notice the presence of a possible outlier with a value of about 20. However, we decided to keep it as we think it may contain important information for the purposes of classification.

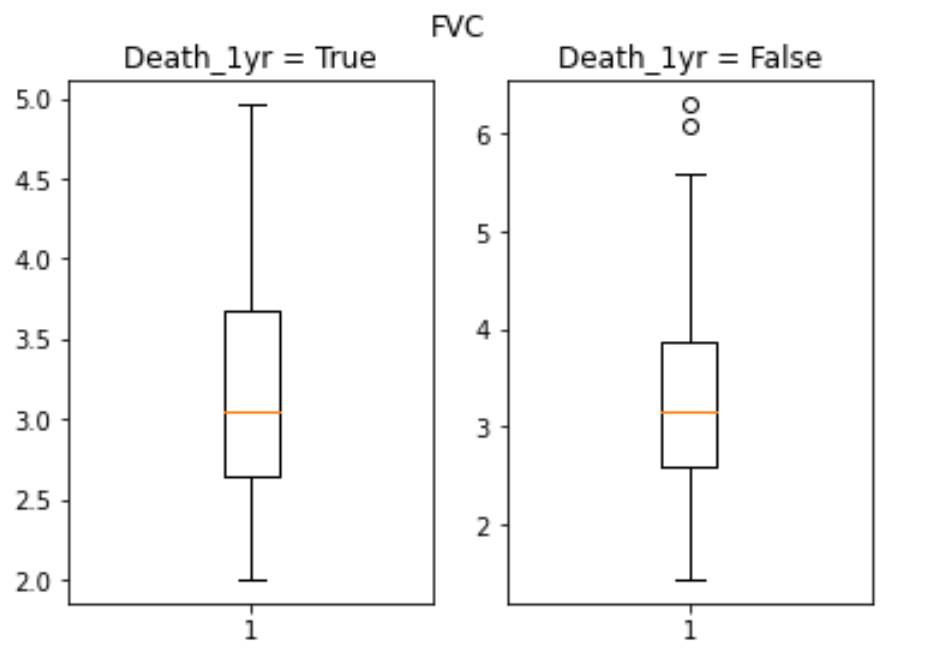
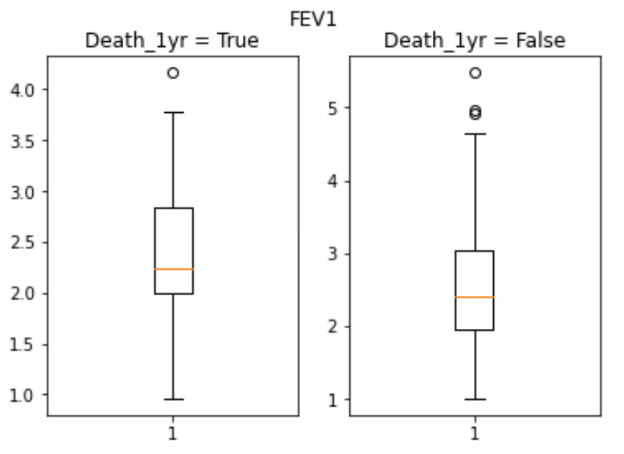
After the outliers removal, the dataset contains 455 instances against the starting 470. This quantity seems sufficient compared to the original number of instances, without remove too much information from the original data.

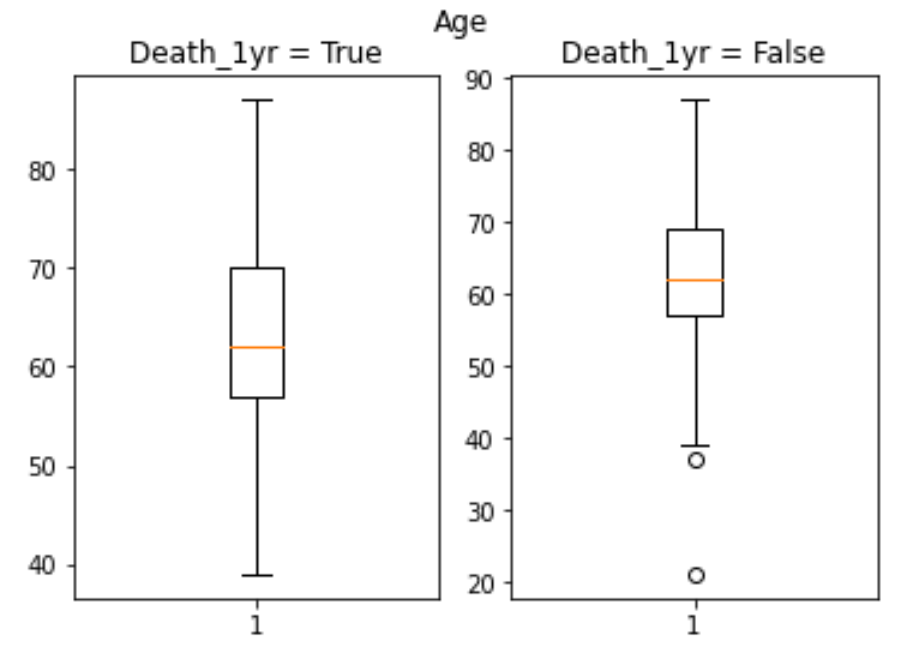
The code for the above work can be found at:

<https://github.com/GiuseppeMoscarelli/Thoracic-Surgery/blob/main/src/0_clean_dataset.ipynb>

**Data analysis**

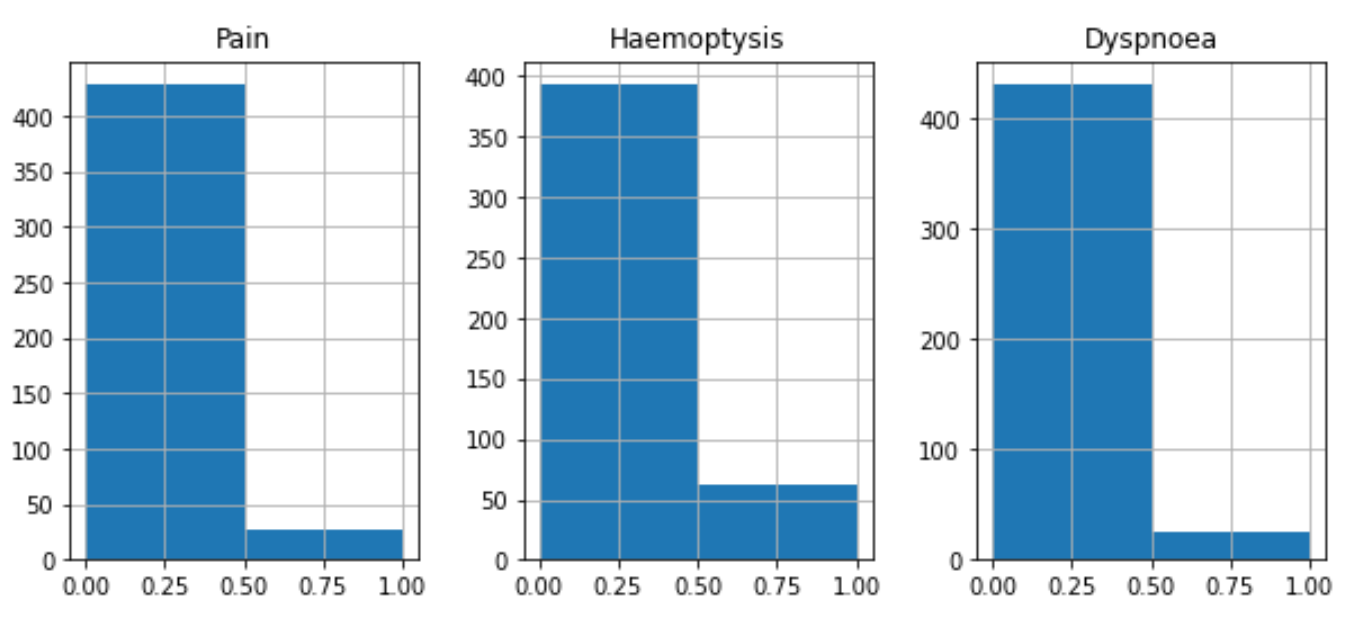
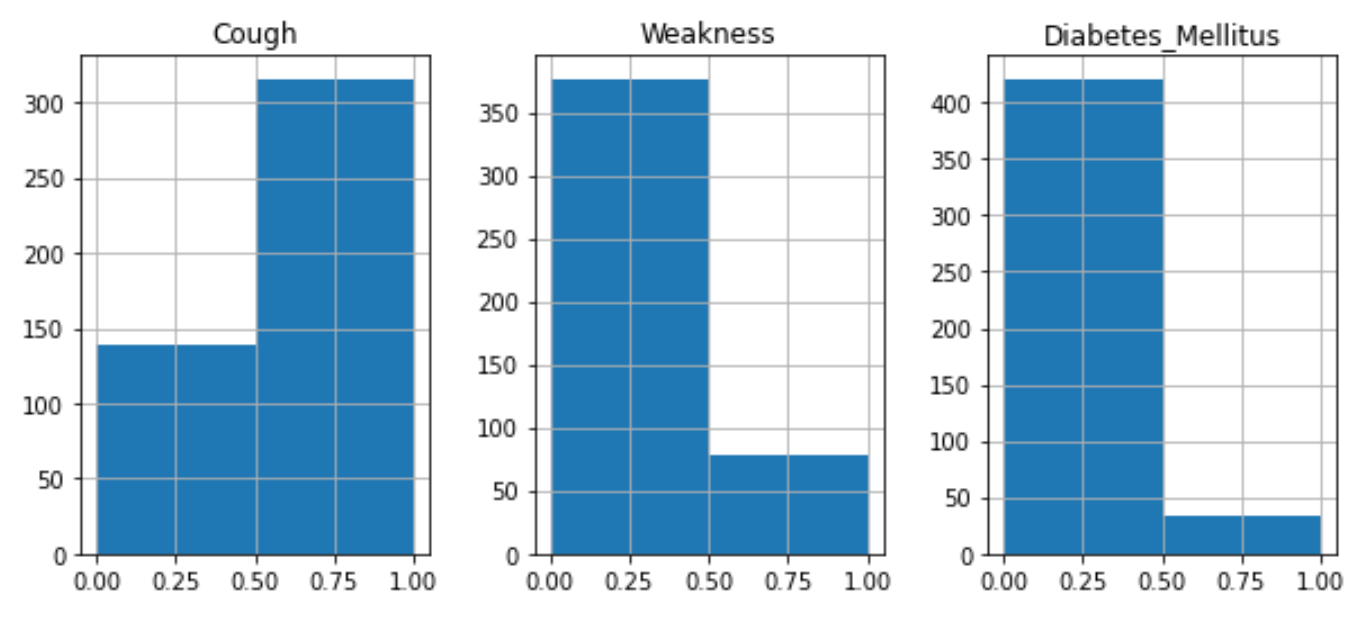
In order to retrieve some information on how the attributes influence the decision, we plotted the boxplots of the continuous variables considering separately the cases in which the class label "Death\_1yr" was positive (True) and the cases in which it was negative (False). The results are shown below:

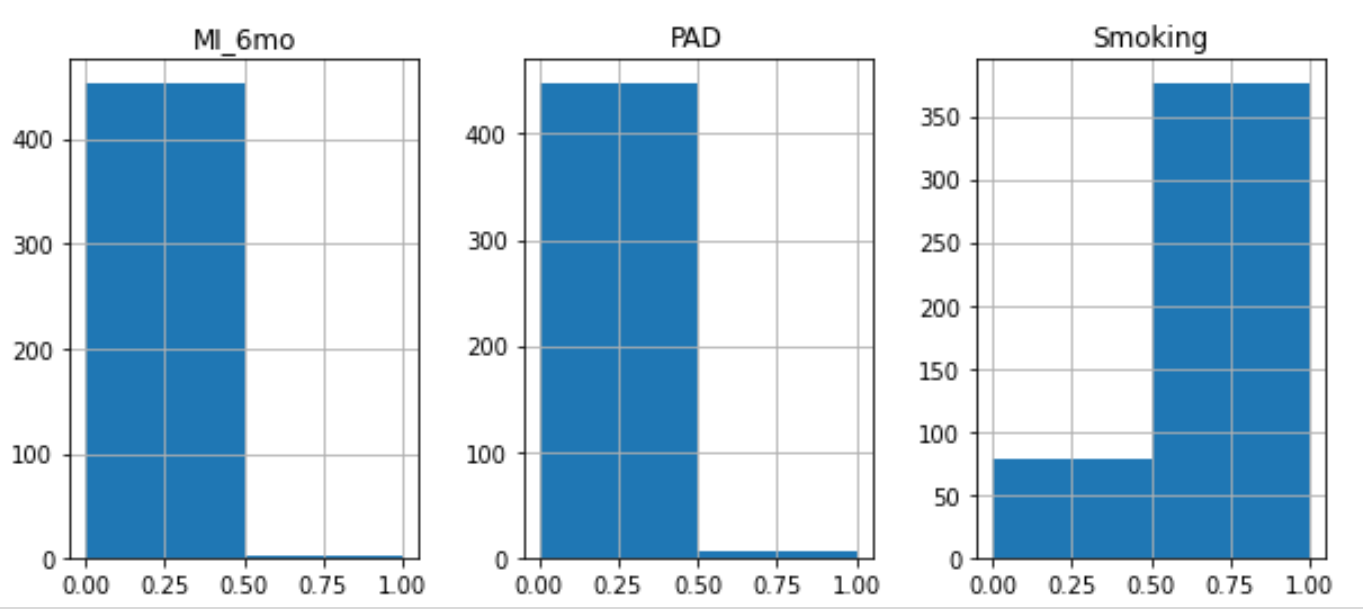
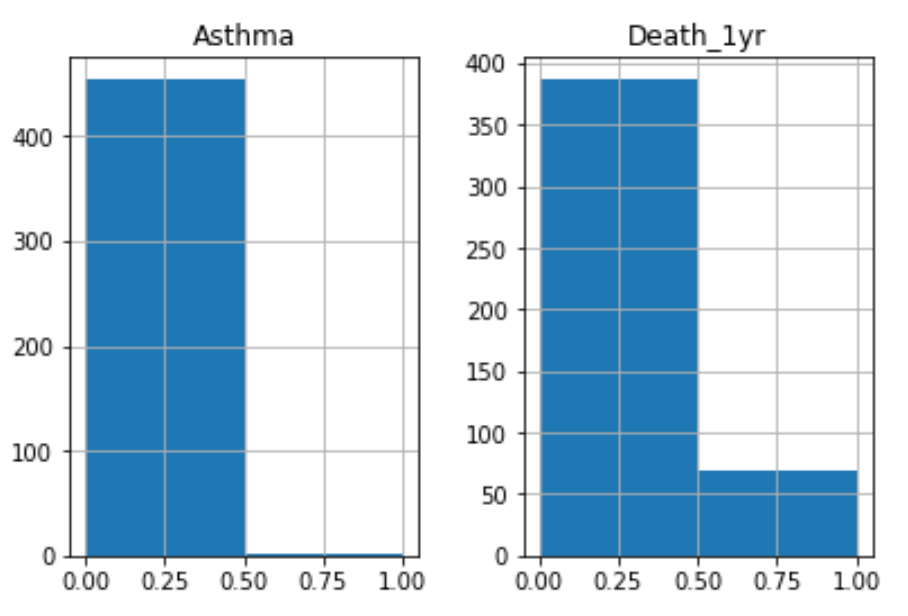


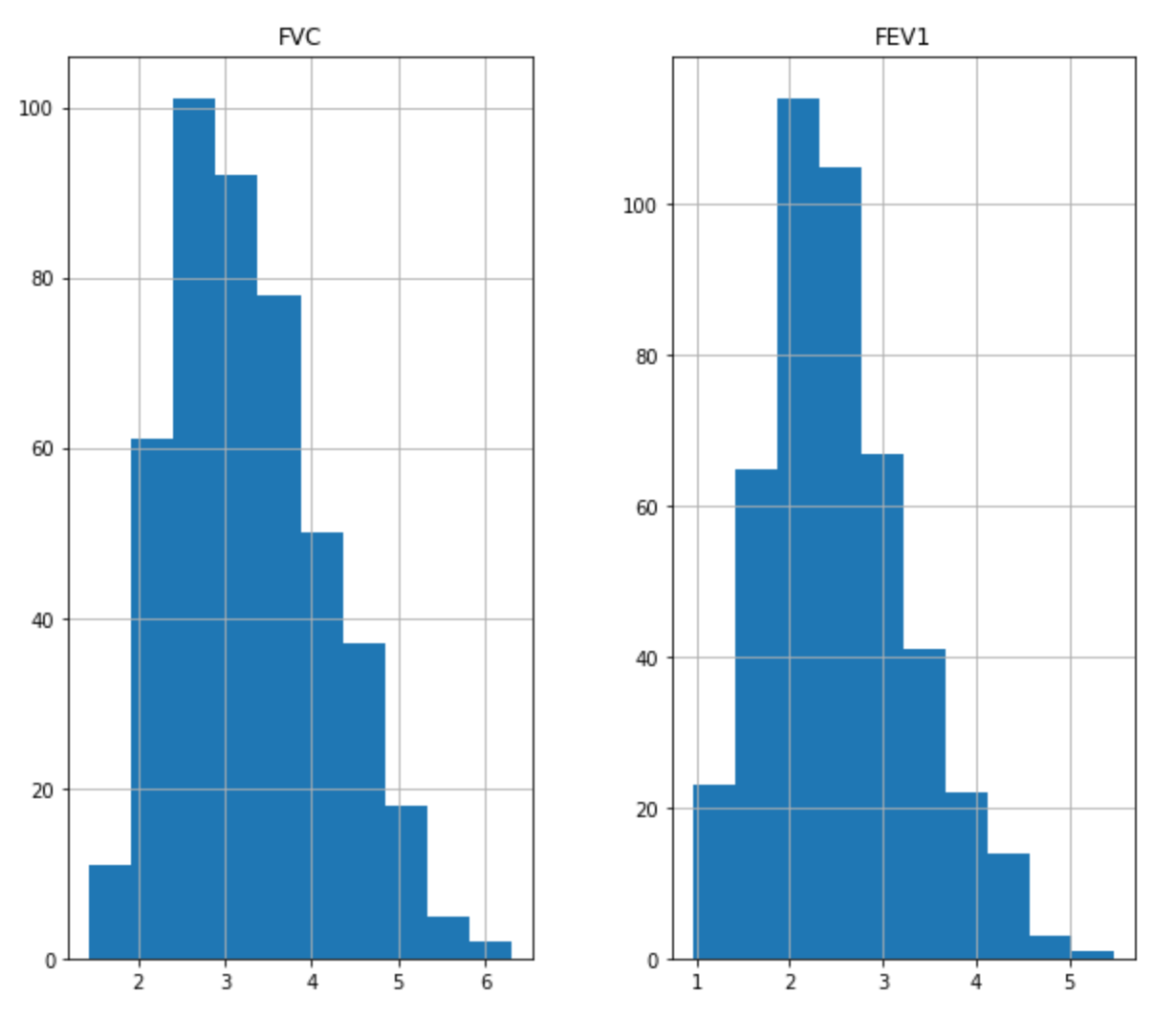
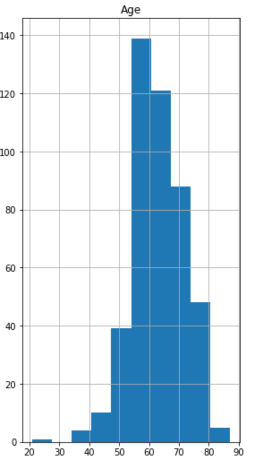


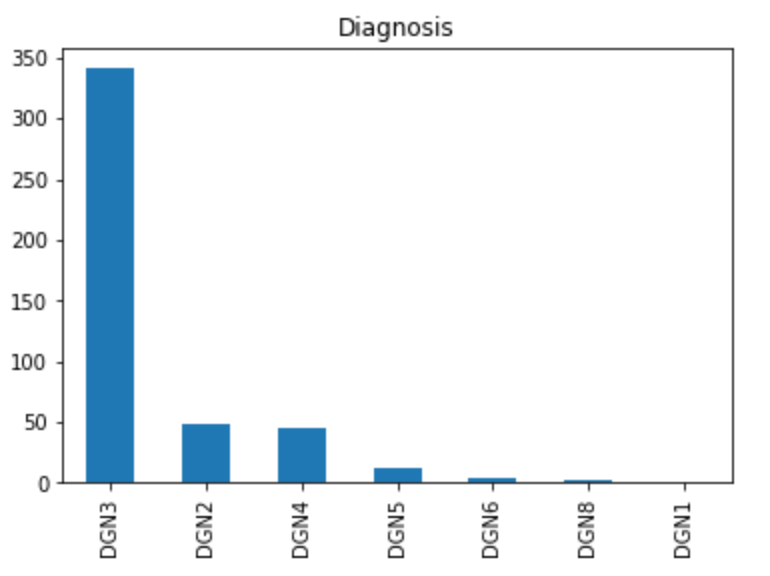
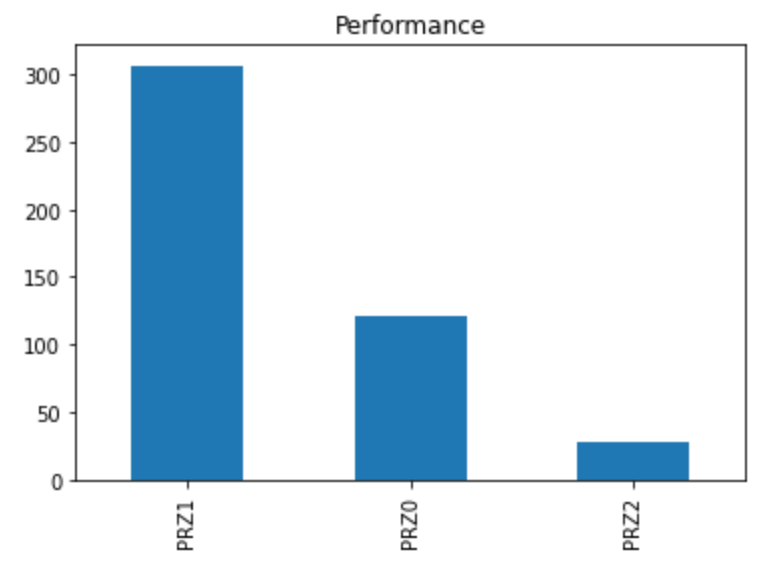
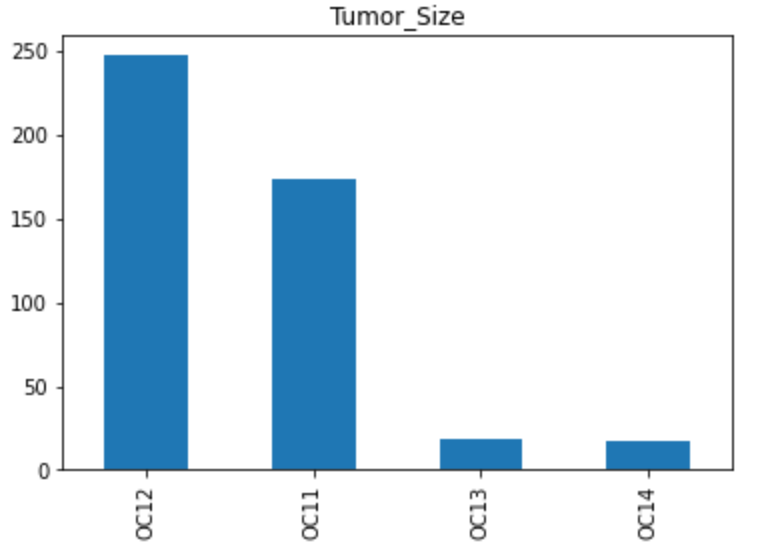
The results show that the Age attribute plays a more decisive role in the decision, as there is a more marked difference between the assumed values with respect to the other two attributes. This denote the fact that there is a correlation between Age attribute and the class label.

After that, in order to better the distribution of all attributes, we plotted various histogram and bar graph:

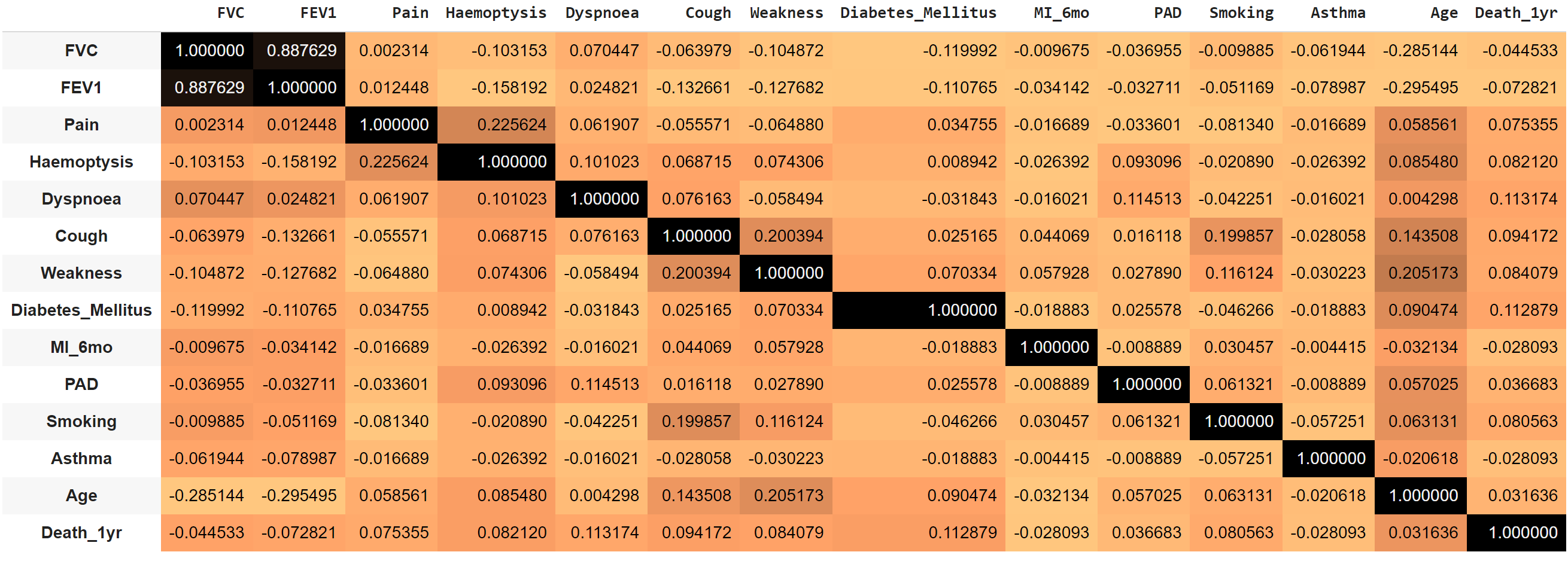
 



Focusing on the label class “Deth\_1yr” we can note that the dataset is very unbalanced.

In order to better understand the correlation between the different attributes, we obtained the correlation matrix:



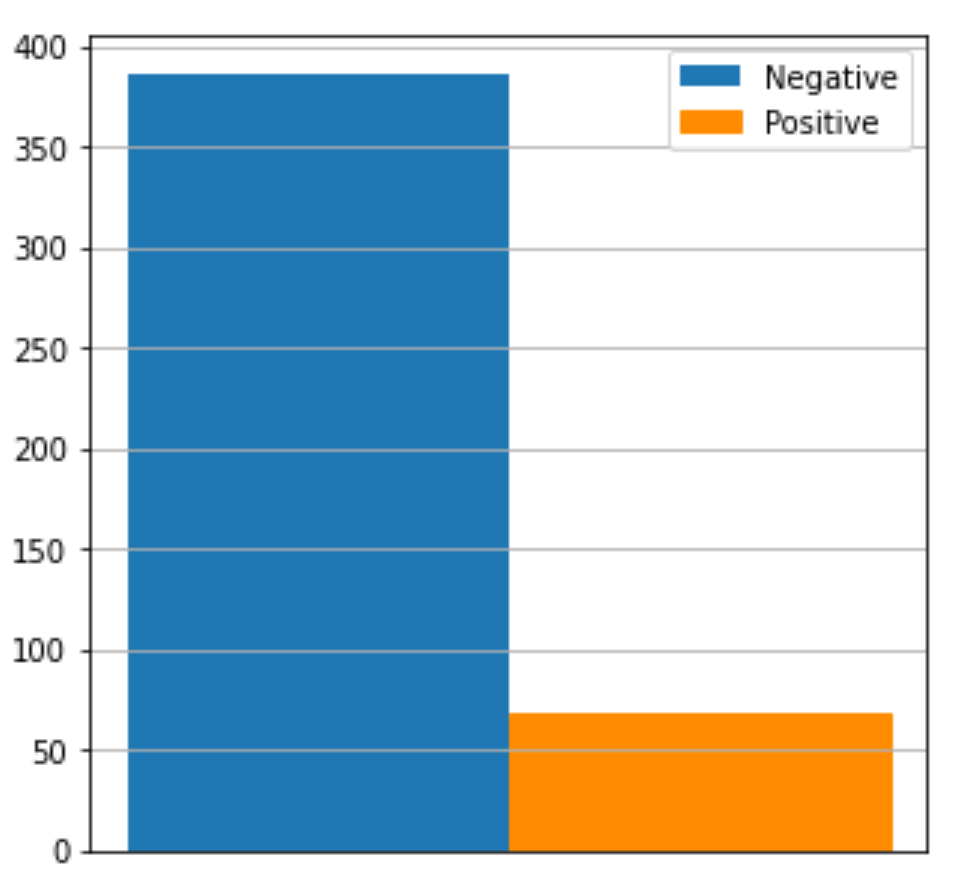
The obtained correlation matrix shows that there is a very strong correlation between FEV1 and FVC but also a mild negative correlation between Age and FVC and FEV1. The latter makes intuitive sense as it would be expected that as you get older, your lung capacity decreases.

The code for the above work can be found at:

<https://github.com/GiuseppeMoscarelli/Thoracic-Surgery/blob/main/src/1_data_analysis.ipynb>.

**Dataset rebalancing**

As previously mentioned, the dataset is very unbalanced. Indeed, there are 69 positive samples and 386 negative sample.



So, we needed to rebalance our dataset before start on it, in order to improve the performance of the different algorithms.

First of all, we split the entire dataset in training set and test set, assigning 70% of data points to the former and the remaining 30% to the latter, in a stratified way. Therefore, we trained the models using the training set and then applied the model to the test set. In this way, we could evaluate the performance of our model using different metrics.

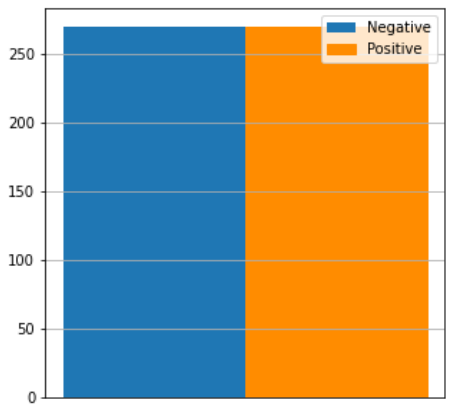
So, after the splitting, we obtain a training set composed of 318 samples and a test set of 137 samples.

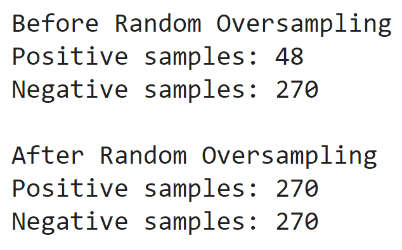
As a second step, we rebalanced only the training set, leaving intact the test set. We decided to use two different techniques: SMOTE and Random Oversampling.

**Random Oversampling**

Random Oversampling is one of the main approaches to randomly resampling an imbalanced dataset. It consists in duplicating examples from the minority class, selecting them with replacement, and adding them to the training dataset. This means that examples from the minority class can be chosen and added to the new “more balanced” training dataset multiple times. This approach is repeated until the desired class distribution is achieved in the training dataset, such as an equal split across the classes (in our case Negative and Positive).

So, starting with 270 negative samples and 48 positive samples in the training set, we obtained a balanced one containing exactly 270 negative and 270 positive samples.





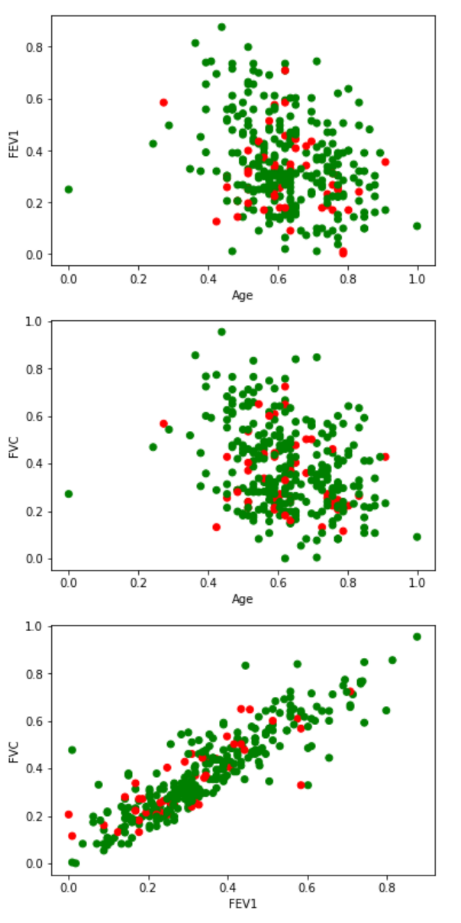
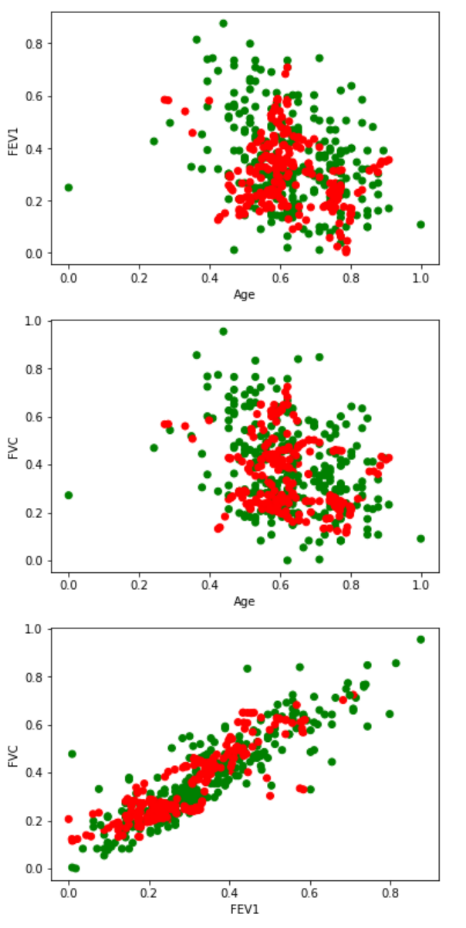
**SMOTE**

The Random Oversampling technique balance the class distribution but does not provide any additional information or variation to the model. An improvement on duplicating examples from the minority class is to synthesize new examples from the minority class that were not present in the original dataset. This is a type of data augmentation for tabular data and can be very effective. The most widely used approach to synthesizing new examples is called the **Synthetic Minority Oversampling Technique** **(SMOTE).**

SMOTE works by utilizing a k-nearest neighbour algorithm to create synthetic data. First it starts by choosing random data from the minority class, then k-nearest neighbours from the data are set. A randomly selected neighbour is chosen and a synthetic example is created at a randomly selected point between the two examples in feature space. In other words, SMOTE first selects a minority class instance **a** at random and finds its k nearest minority class neighbours. The synthetic instance is then created by choosing one of the k nearest neighbours **b** at random and connecting **a** and **b** to form a line segment in the feature space. The synthetic instances are generated as a convex combination of the two chosen instances **a** and **b**.

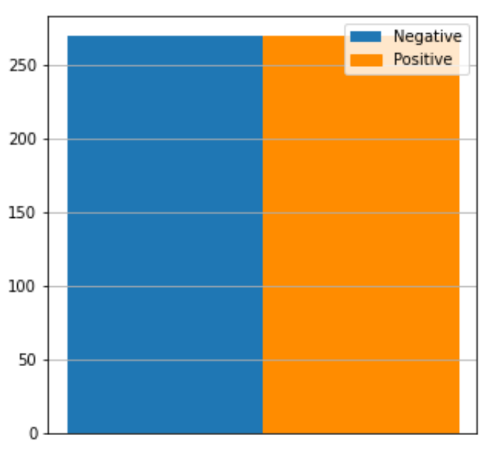
The procedure is repeated enough times until the minority class has the same proportion as the majority class.

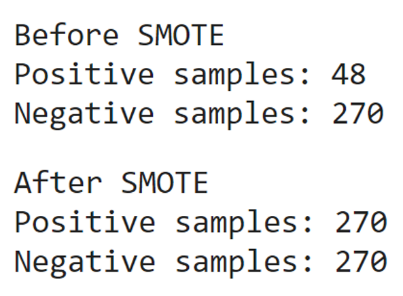
Because in our dataset there are categorical variables, as well as continuous variables, we used a variation of SMOTE, called **SMOTE-NC.** Indeed, if we had oversampled categorical variables using standard SMOTE, we would have end up with oversampled data which would not make sense at all.

Some examples of the dataset before and after SMOTE\_NC application are shown below:

On the left there are three plots of the data before SMOTE-NC application, while on the right there are three plots of the data after SMOTE-NC application. Green points are those belonging to the negative class, while red points are those belonging to the positive class (minority class). As we can see on the right are there are many new generated samples belonging to the positive class which were note present before.

The number of positive and negative samples of the obtained new balanced training set, are obviously the same as of balanced training set obtained from Random Oversampling:





The code for the above work can be found at:

<https://github.com/GiuseppeMoscarelli/Thoracic-Surgery/blob/main/src/2_rebalancing_dataset.ipynb>.

**Principal Component Analysis**

PCA is a technique commonly used for dimensionality reduction by projecting each data point onto a lower-dimensional space, through a linear mapping, while preserving as much of the data's information as possible. This compressing procedure is called **data encoding**, while the reverse procedure is called **data decoding**. So PCA produce as a solution a linear mapping such that the distance between the original data and the reconstruction (encoding and then decoding) of the data is as small as possible. In mathematical terms it is expressed in the following way:

[Parlare di U, W e A?]

In term of variance, the first principal component can be equivalently defined as a vector lying on the direction that maximizes the variance of the projected data. In other words, given a set of features X1, X2, …, Xp, the first principal component is the normalized linear combination of the features:

PC1 = α11X1+ α21X2 + …. + αd1Xd =

and has the direction of largest variance on the feature space. After the first principal component PC1 has been determined, we can find the second principal component PC2 which is itself a normalized linear combination of the p features and has maximal variance among all linear combinations that are uncorrelated with PC1.

It turns out that constraining PC2 to be uncorrelated with PC1 is equivalent to constraining the directions of the two principal components to be orthogonal. We can continue in this way until we find *p*-th principal component.

In mathematical terms, the problem of PCA in term of variance can be formulated as:

So, in general, we can say that the *i*-th principal component can be taken as a direction orthogonal to the first *(i-1)*-th principal components that maximizes the variance of the projected data.

In order to estimate the variance explained by each component and to understand the importance of each component we have to compute the **PVE (Proportional Variance Explained)**.

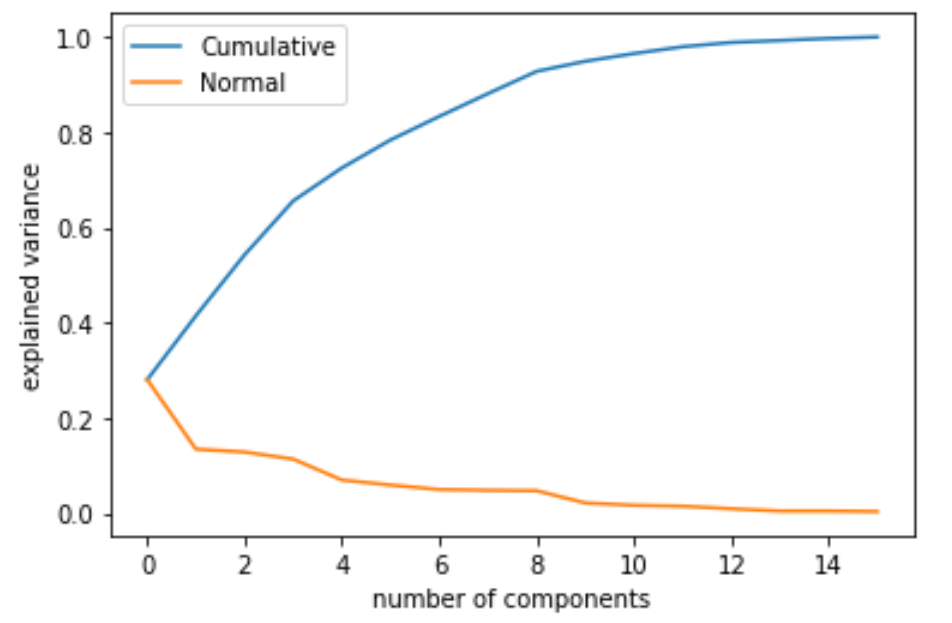
Given a centered dataset X composed by n observations, the total variance is defined as:

and the variance explained by the *m*-th principal component is:

[completare la parte di teoria]

To perform PCA on our dataset we removed prefix DGN, PRZ and OC from categorical variables and then we normalized them.

After plotting the values of cumulative and normal explained variance, we decided to take the first 9 principal component that explain a cumulative variance of almost 90% as showed by following figure:



The code for the above work can be found at:

<https://github.com/GiuseppeMoscarelli/Thoracic-Surgery/blob/main/src/3_PCA.ipynb>.