Politecnico
di Torino

Heart Failure Data Analysis

Mathematics in Machine Learning

Salvatore Junior Curello
s268066

A.A
2021/2022

Introduction

This study was focused on survival analysis of heart failure patients who were admitted to Institute of Cardiology and Allied hospital Faisalabad-Pakistan between April and December (2015). The dataset used to perform the analysis is the e "Heart failure clinical records Dataset"

Goal:

The goal of this study is to produce predictions taking into account only the clinical features of the patients and to show which model performs better among all the models that have been tested.

Dataset Overview

It is constitute of 299 patients of heart failure comprising of 105 women and 194 men which their age is in the range [40, ..., 95]. It contains 11 features on which the analysis are based, then the time (which indicates the follow-up period) and the DEATH_EVENT which will be the target variable during the analisys.

Feature	Explaination	Measurement	Range
Age	Age of the patient	*Years*	[40, ..., 95]
Anemia	Decrease of red blood cells or hemoglobin	*Boolean*	0, 1
High Blood Pressure	If the patient has hypertension	*Boolean*	0, 1
Creatinine phosphokinase	Level of the CPK enzyme in the blood	*mcg/L*	[23, ..., 7861]
Diabetes	If the patient has diabetes	*Boolean*	0, 1
Ejection fraction	Percentage of blood leaving the heart at each contraction	*Percentage*	[14, ..., 80]
Platelets	Platelets in the blood	*kiloplatelets/mL*	[25.01, ..., 850.00]
Sex	Woman or man	*Binary*	0, 1
Serum creatinine	Level of serum creatinine in the blood	*mg/dL*	[0.50, ..., 9.40]
Serum sodium	Level of serum sodium in the blood	*mEq/L*	[114, ..., 148]
Smoking	If the patient smokes or not	*Boolean*	0, 1
Time	Follow-up period	*Days*	[4, ..., 285]
Death Event (Target)	If the patient deceased during the follow-up period	*Boolean*	0, 1

Dataset Visualization & Missing Values

The dataset is constructed as follows:

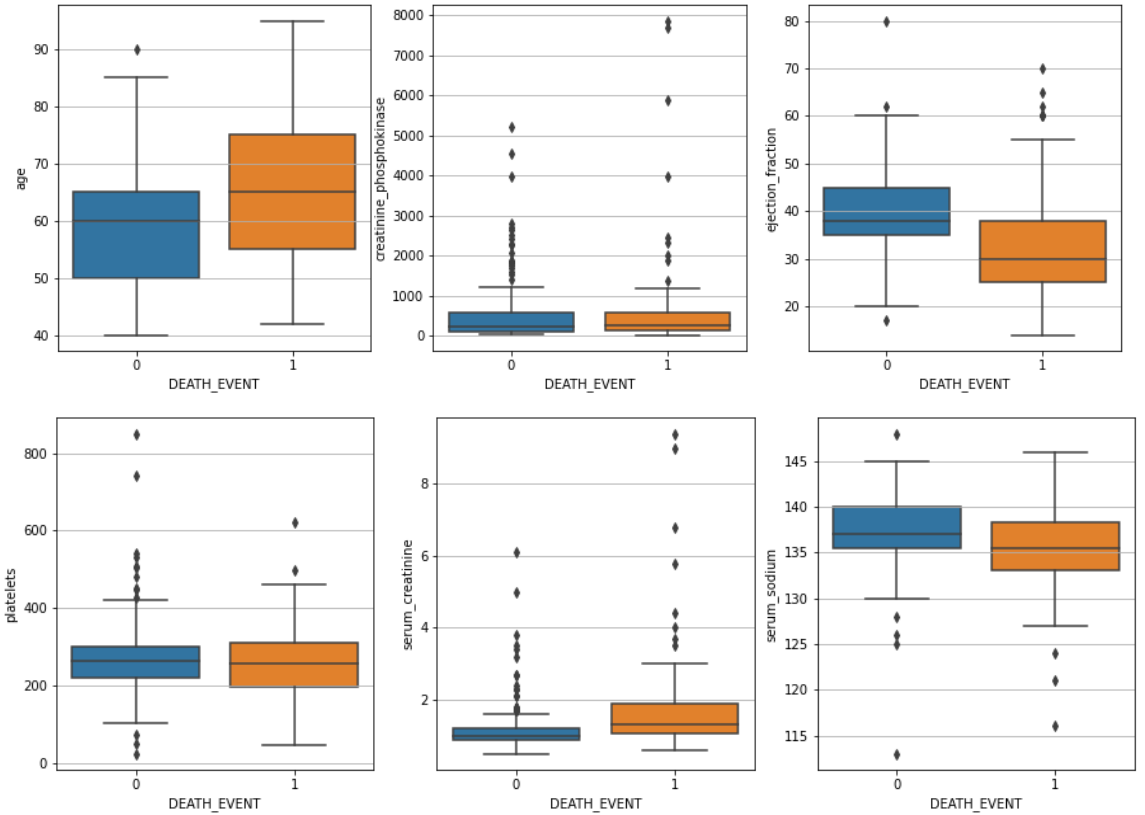
	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
0	75.0	0	582	0	20	1	265.00000	1.9	130	1	0	4	1
1	55.0	0	7861	0	38	0	263.35803	1.1	136	1	0	6	1
2	65.0	0	146	0	20	0	162.00000	1.3	129	1	1	7	1
3	50.0	1	111	0	20	0	210.00000	1.9	137	1	0	7	1
4	65.0	1	160	1	20	0	327.00000	2.7	116	0	0	8	1

A null-value check is performed and no missing values were detected:

```
age                         0
anaemia                     0
creatinine_phosphokinase    0
diabetes                    0
ejection_fraction           0
high_blood_pressure         0
platelets                   0
serum_creatinine            0
serum_sodium                0
sex                         0
smoking                     0
time                        0
DEATH_EVENT                 0
dtype: int64
```

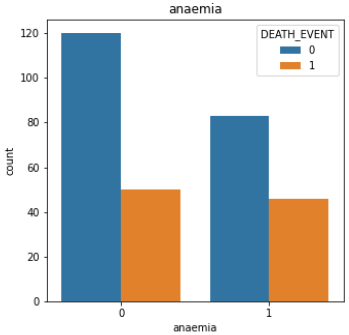
Numerical Features

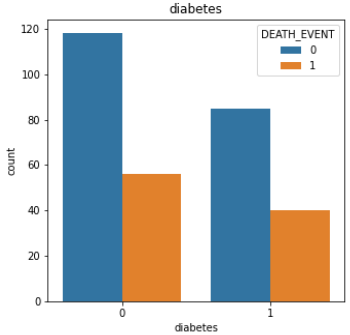
Numerical Features: age, creatinine_phosphokinase, ejection_fraction, platelets, serum_creatinine, serum_sodium

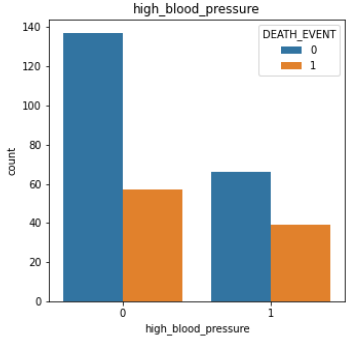


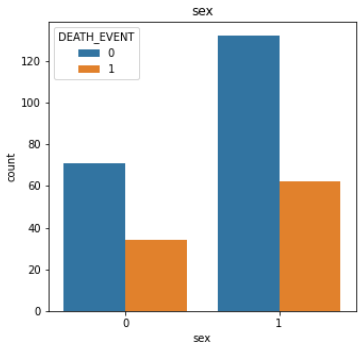
Categorical Features

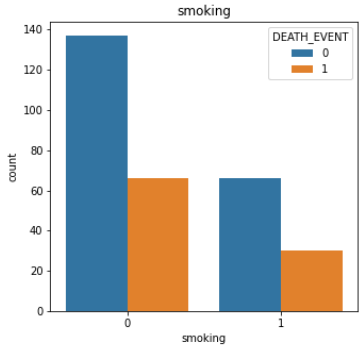
Categorical Features: anaemia, diabetes, high_blood_pressure, sex and smoking









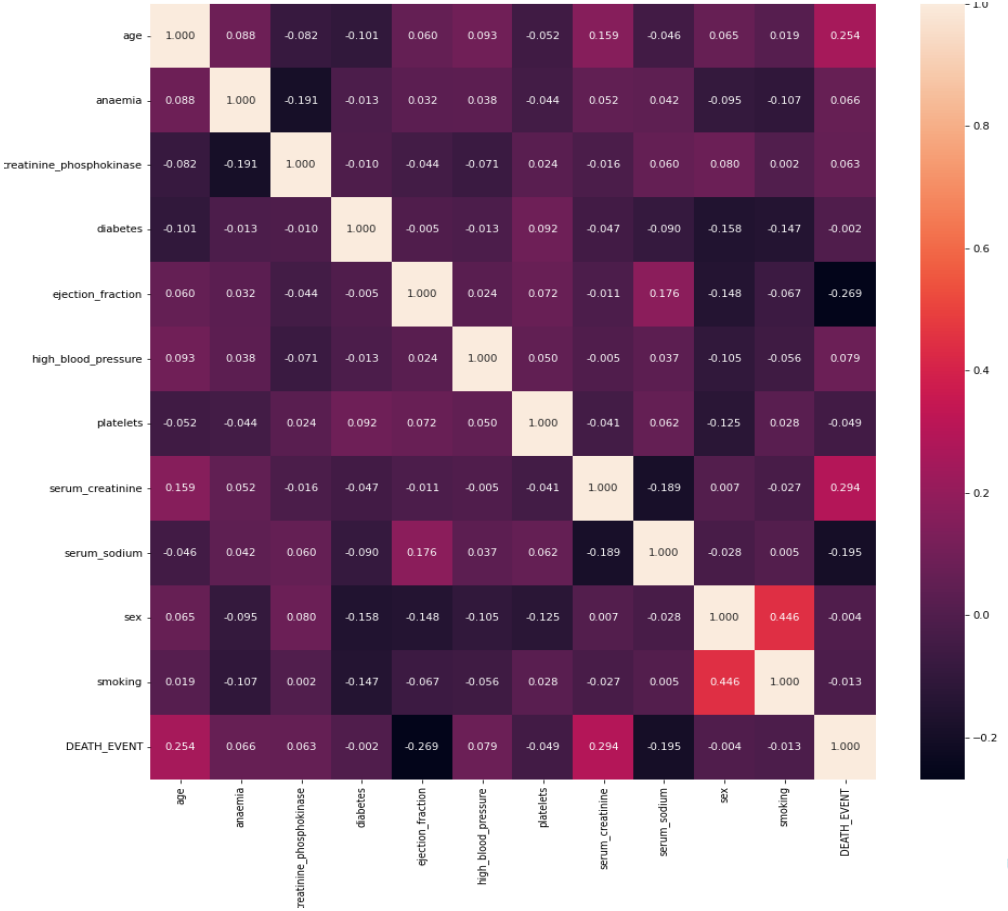


Correlation Matrix

The correlation coefficient of two variables can be computed by dividing the covariance of these variables by the product of the standard deviations of the same values. Mathematically:

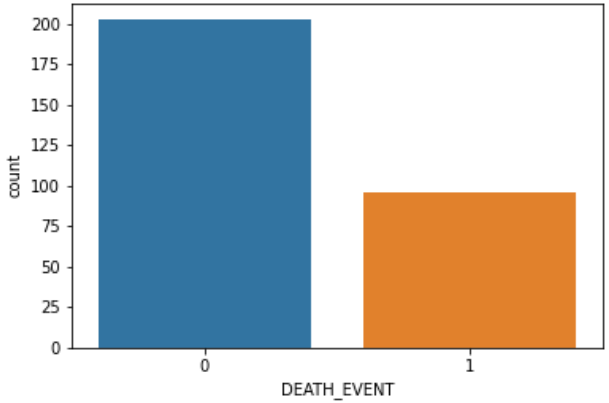
$$\rho(x,y) = \frac{COV(x,y)}{S_x S_y}$$

The features are quite uncorrelated with the exception of *sex* and *smoking* that seems to be slightly positively correlated.



Class Unbalance

In order to avoid biased results it is important to inspect some class imbalance.

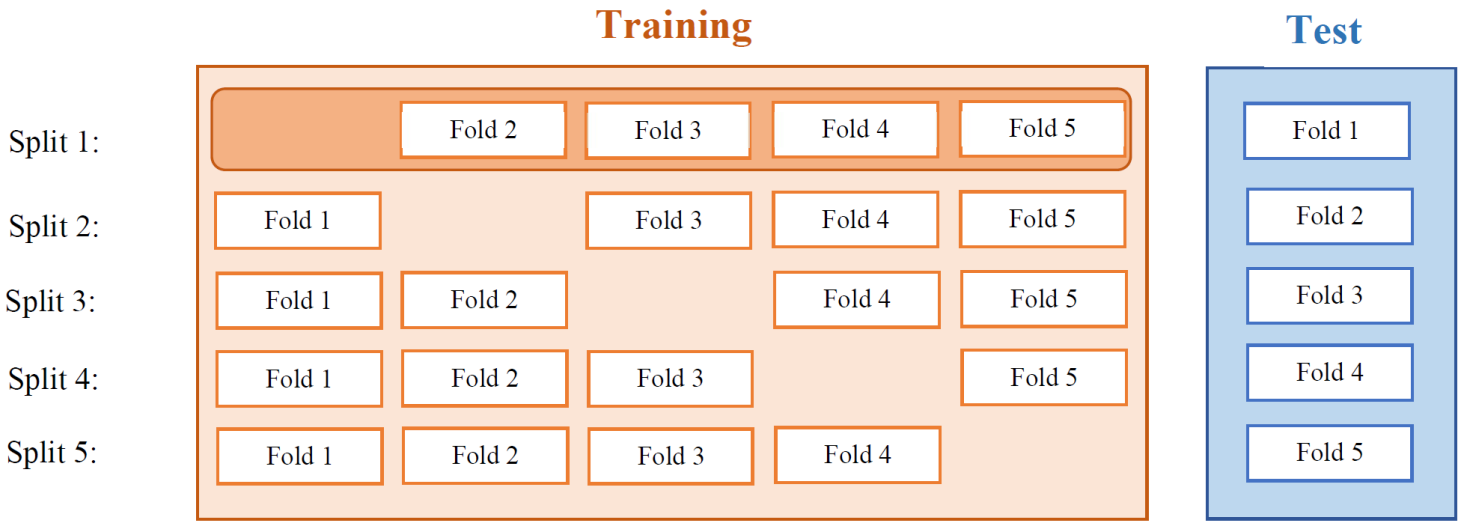


Class	Value Counts	Percentage (%)
0 (survived)	203	67.89
1 (deceased)	96	32.11

Even if not so strong the dataset is slightly imbalanced with 67.89% (203) survival cases & 32.11% (96) death events. In this situation the model could become more inclined towards learning and predicting the negative examples than the positive (death) cases.
To overcome this issue three different technique of re-balancing are applied.

Data Preparation

- **Holdout:** the Holdout method is the most simple validation technique and it consists in splitting the data set in three set: training, validation and testing. The training set is used for training the model, the validation set to evaluate the model in order to find the best parameters and finally the test set is used for testing the model performances on data which was never seen by the algorithm.
- **K-fold Cross Validation (5-fold):** a stratified cross validation is applied with 5 folds.



Data standardization

Features measured at different scales don't contribute equally to the analysis and might end up creating bias.
In order to avoid this, a standardization process is applied.

Standardize data is calculated by first to determining the distribution mean and standard deviation for each feature and then by subtracting the mean for each of them. Then the values are divided by its standard deviation.

The following formula summarize it:

$$z = \frac{(x - \bar{x})}{\sigma}$$

where $\boldsymbol{x}$ is the original feature vector, $\overline{\boldsymbol{x}}$ is the mean of that feature vector, and $\boldsymbol{\sigma}$ is its standard deviation.

	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium	anaemia	diabetes	high_blood_pressure	sex	smoking
0	1.164202	-0.350370	-2.000867	-1.439568	-0.188705	0.131259	1	0	0	1	0
1	1.164202	-0.505933	-0.022672	-0.408476	1.120602	-0.545811	0	0	1	1	0
2	-0.032819	-0.500642	-0.710740	1.345442	0.113442	-0.094431	1	1	0	1	0
3	-0.756645	-0.471011	-0.710740	-0.472255	-0.692285	-0.094431	0	0	0	1	1
4	2.750989	0.005203	-0.022672	0.009892	0.445805	-0.545811	0	1	1	1	0

Feature Selection

- **Mutual Information:** it is a non-negative value, which measures the dependency between the variables where higher values indicate stronger dependence. For Discrete Distributions, the mutual information of two jointly discrete random variables X and Y is calculated as a double sum:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p_{X,Y}(x,y) log\left(\frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}\right)$$

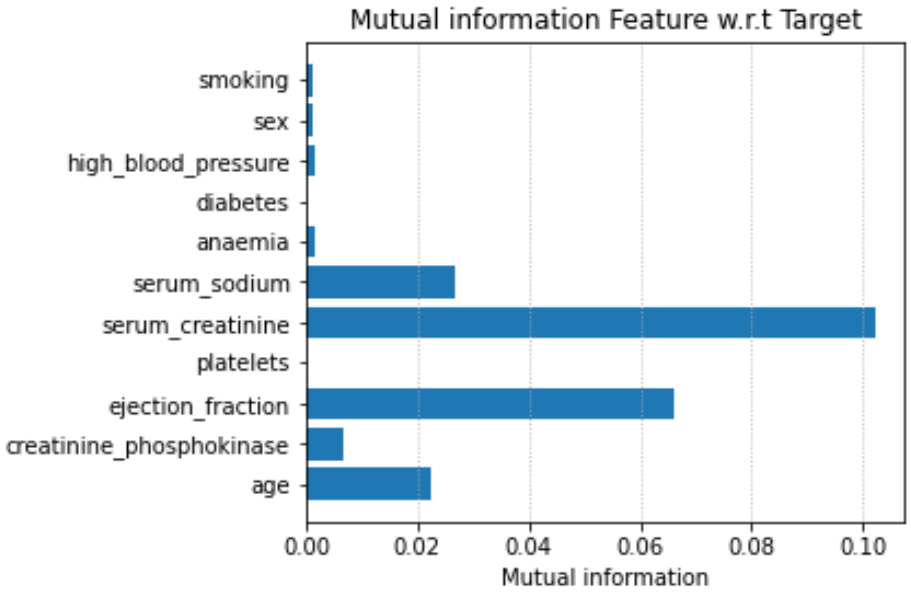
where, $P_{X,Y}$ is the joint probability mass function of X and Y, and p_X and p_y are the marginal probability mass functions of X and Y. For continuous random variables, the double sum is replaced by a double integral.

- **Chi-Square Test:** it is used to determine the relationship between the independent category feature (predictor) and dependent category feature (response). The formula for Chi-Square is:

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

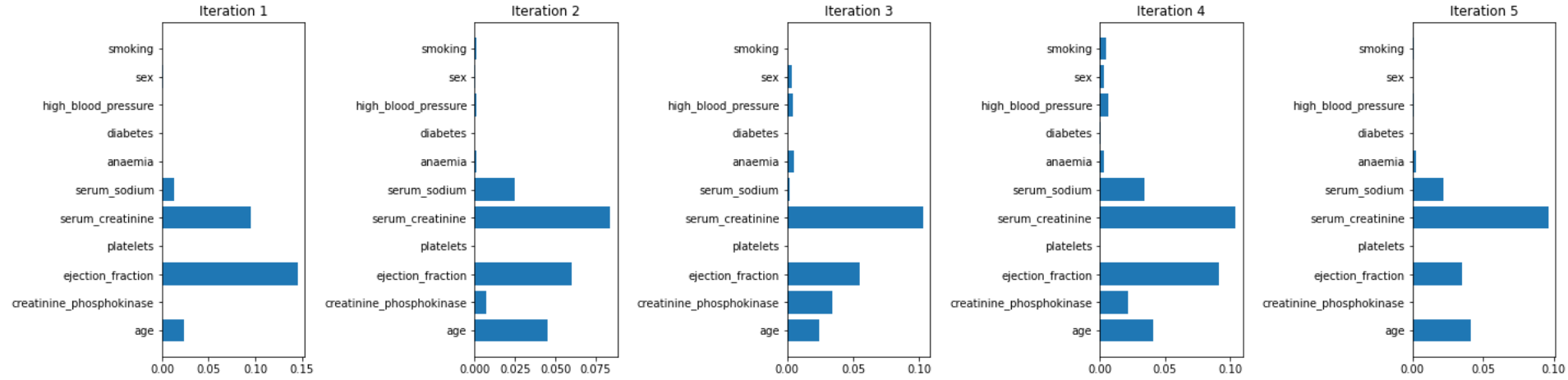
where *c* are the degrees of freedom of the distribution, O_i is the observed value and E_i is the expected value.

Mutual Information – Holdout



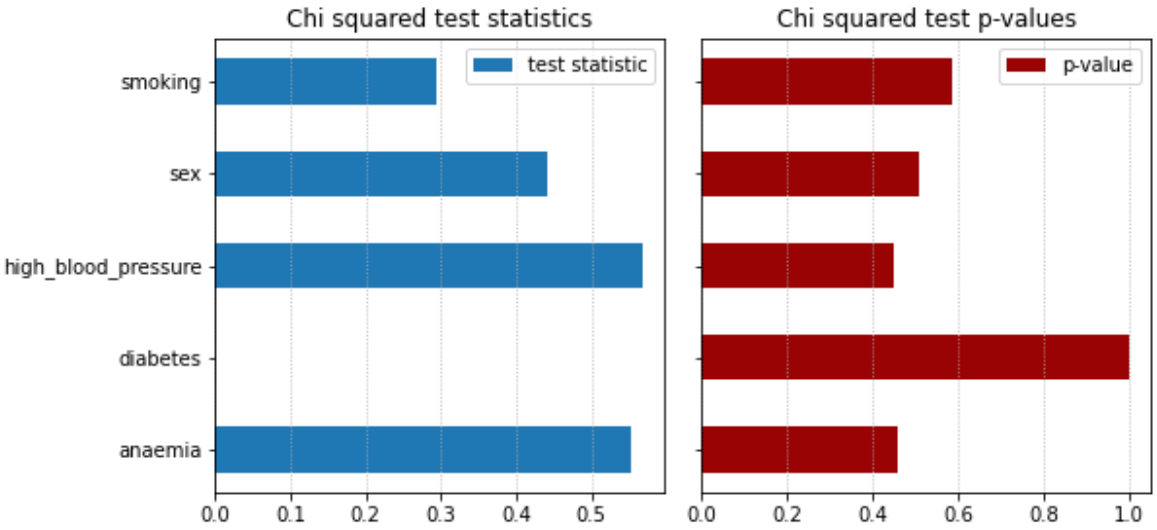
The only continous feauture that could be avoided durign the analisys is "platelets".

Mutual Information – K-Fold



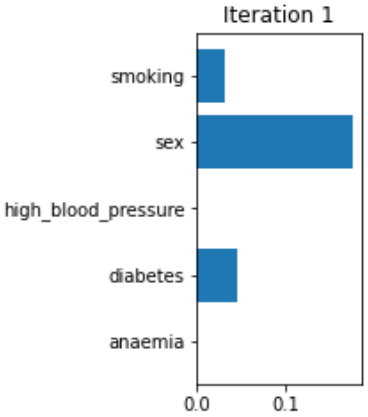
We have almost the same behevior of the holdout method so we keep all the features except "platelets“.

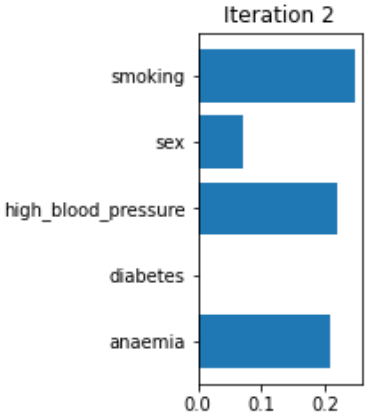
Chi-Squared Test - Holdout

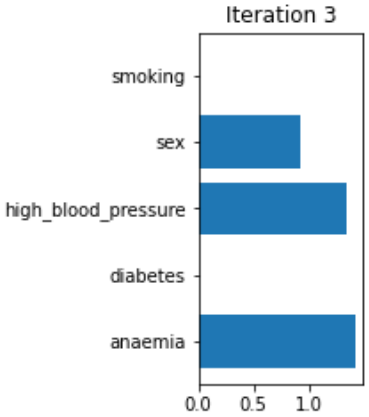


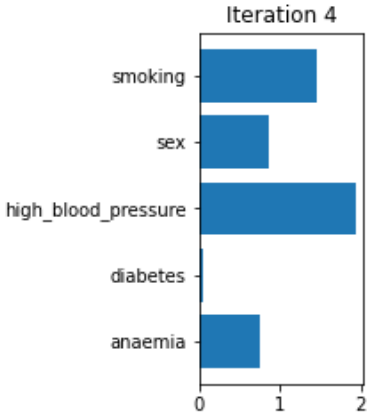
Their contribution is not so high but we still decied to consider the features "high_blood_pressure" and "anaemia"

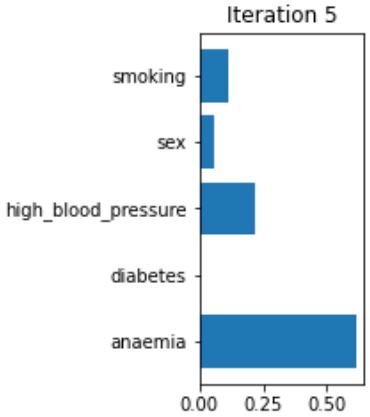
Chi-Squared Test - Kfold









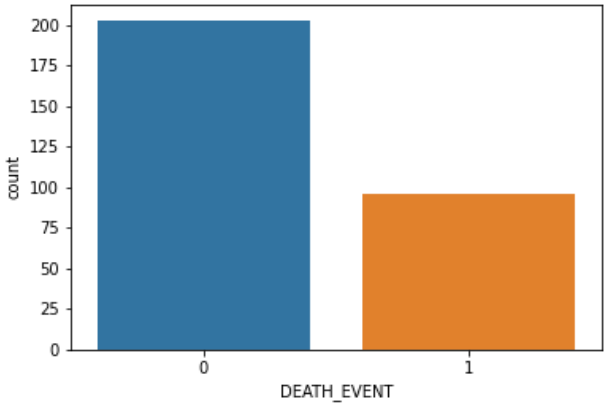


There is a different situation here w.r.t the holdout so only the top 2 features for each iteration are considered.

Class Rebalancing – Random Oversampling

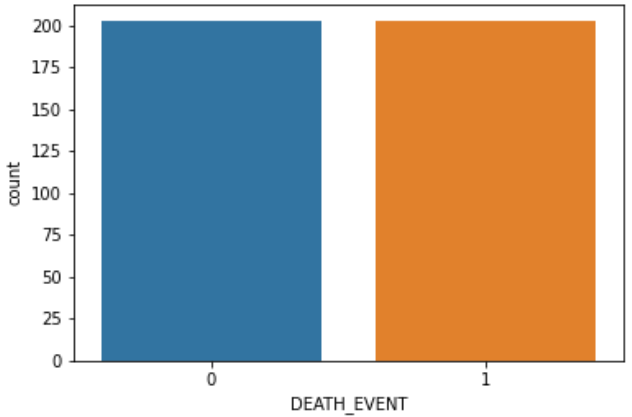
Random oversampling is the process of randomly duplicating examples from the minority class and adding them to the training dataset.

BEFORE RANDOM OVERSAMPLING



Class	Value Counts	Percentage (%)
0 (survived)	203	67.89
1 (deceased)	96	32.11

AFTER RANDOM OVERSAMPLING



Class	Value Counts	Percentage (%)
0 (survived)	203	50
1 (deceased)	203	50

Class Rebalancing – SMOTE

In SMOTE, the training set is altered by adding synthetically generated minority class instances, causing the class distribution to become more balanced. It can be summarized in three steps:

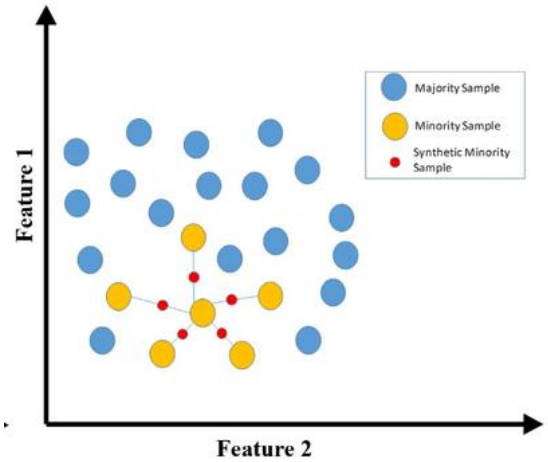
- Considering the minority class as A, for each $x \in A$, the the k-nearest neighbors of x are obtained by calculating the Euclidean distance between x and every other sample in set A.

- For each $x \in A$, N examples (i.e x1, x2, …xn) are randomly selected from its k-nearest neighbors, and they construct the set A_1.

- For each example $x_k \in A_1$ (k=1, 2, 3…N), the following formula is used to generate a new example:

$$x' = x + rand(0,1) * |x - x_k|$$

where rand(0, 1) denotes a random number between 0 and 1.



Model Selection - Metrics

Each model will be evaluated according different metrics.

- **Accuracy:** It is defined as the number of correct predictions divided by the total number of predictions

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

- **Precision:** It is defined as the number of true predicted positives over the total number of positives(is the fraction of relevant instances among the retrieved instances)

$$precision = \frac{TP}{TP+FP}$$

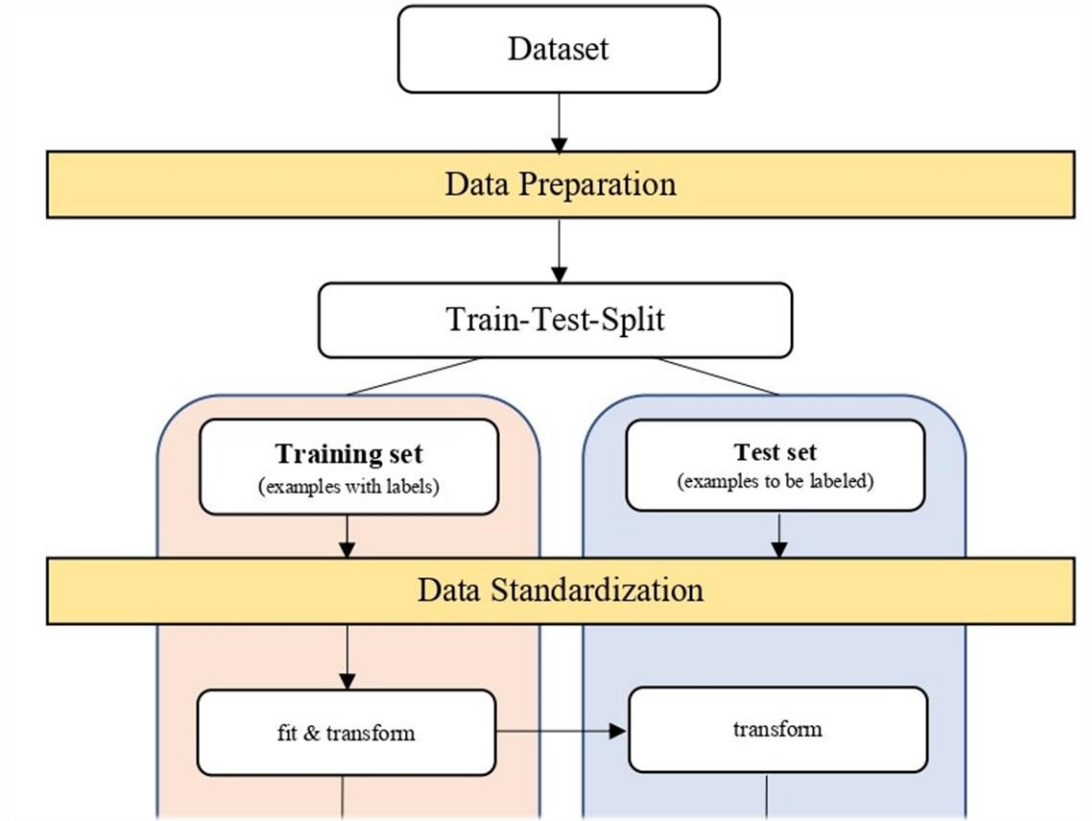
- **Recall:** It is defined as the number of true predicted positives over the total number of positives(is the fraction of relevant instances that were retrieved)

$$recall = \frac{TP}{TP+FN}$$

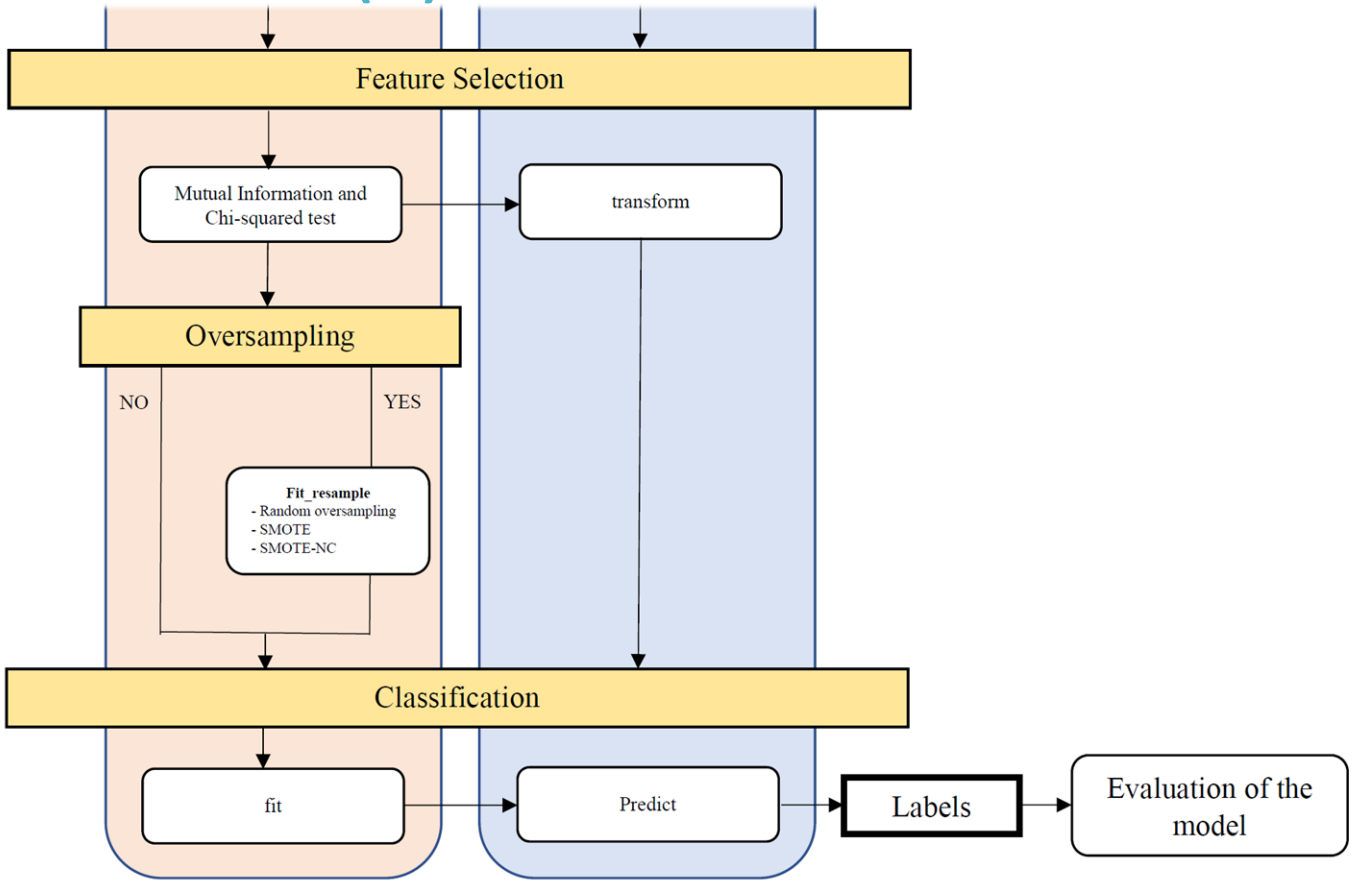
- **F1 score:** It is the harmonic mean between precision and recall. Its range is [0,1] and its value indicates how precise and robust the classifier is.

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$

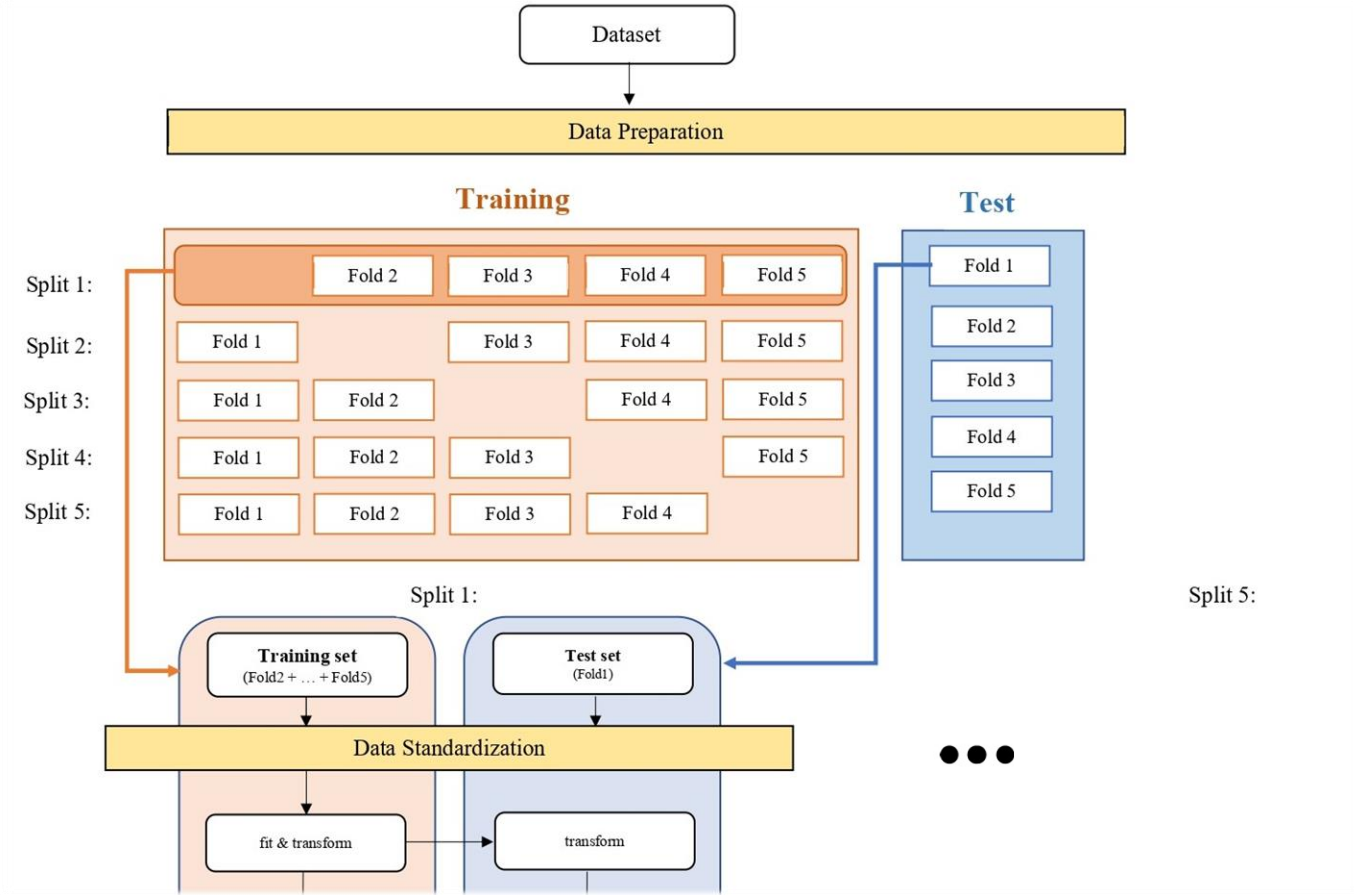
Pipeline HOLDOUT (1)



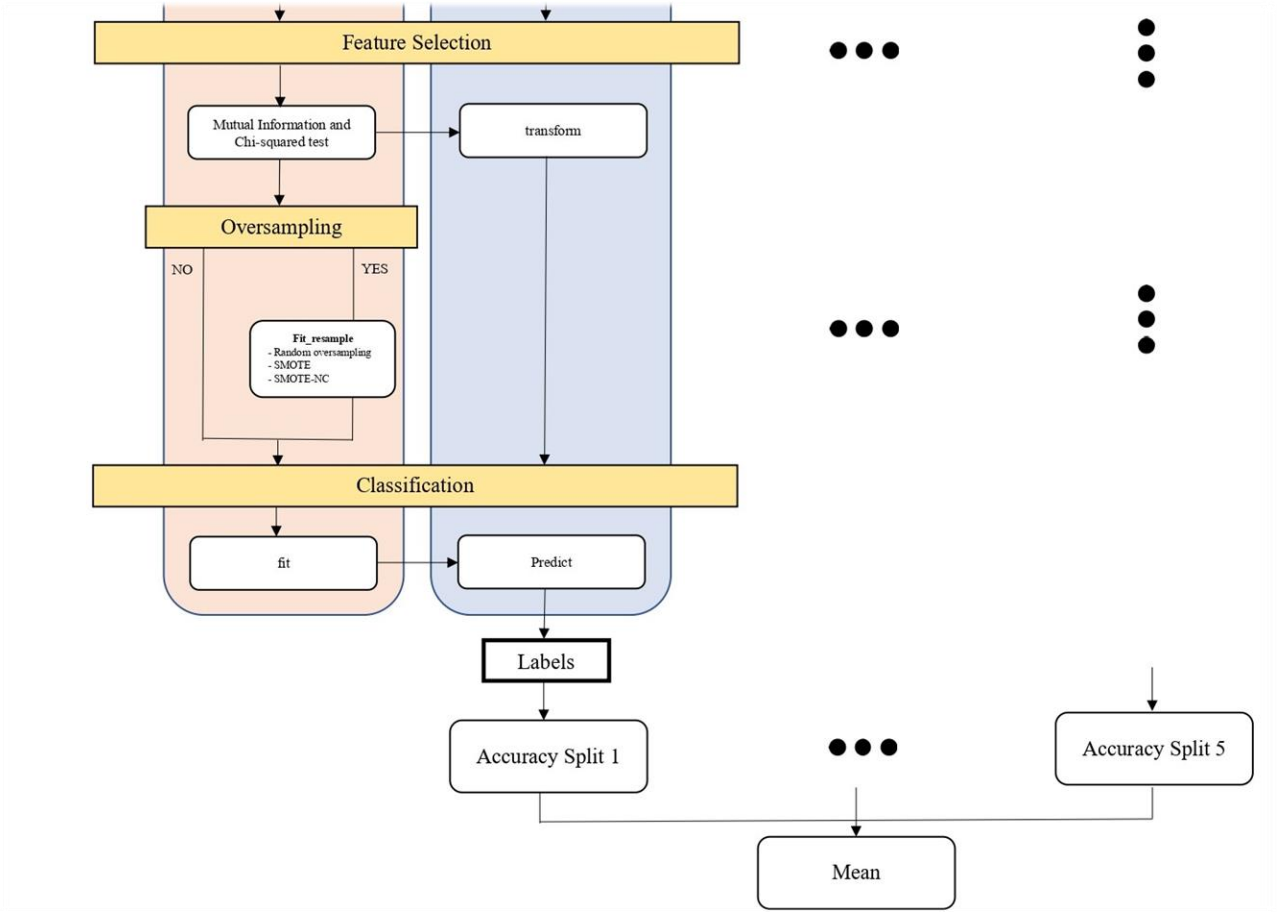
Pipeline HOLDOUT (2)



Pipeline K-FOLD (1)



Pipeline K-FOLD (2)



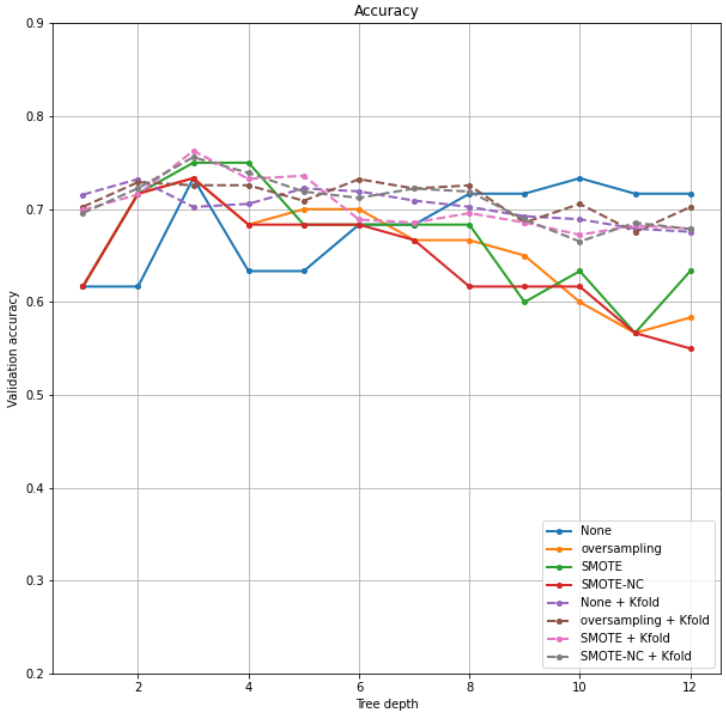
Decision Tree

Decision Trees are a supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Basically, a process of dividing up the input space is applied. A greedy approach is used to divide the space called recursive binary splitting.

The measure that we used to do so is the **Gini Index** which can be formalized as follow:

$$G = \sum_k p_k(1 - p_k)$$

where $\boldsymbol{p_k}$ is the ratio between number of samples of class $\boldsymbol{k}$ and total number of samples(the proportion of training instances with class k).



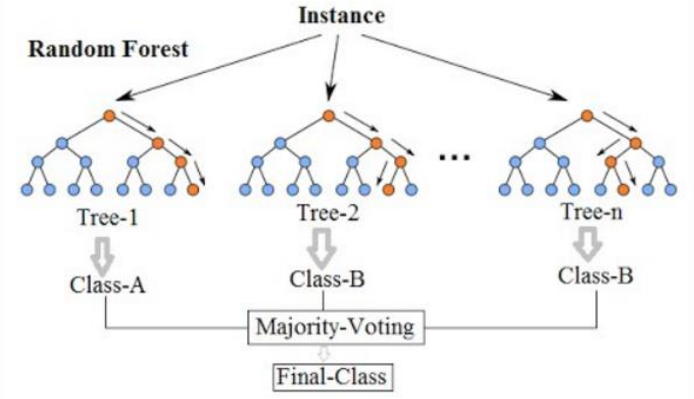
Hyperparameters: max_depth=[1,2,3,4,5,6,7,8,9,10,11,12]

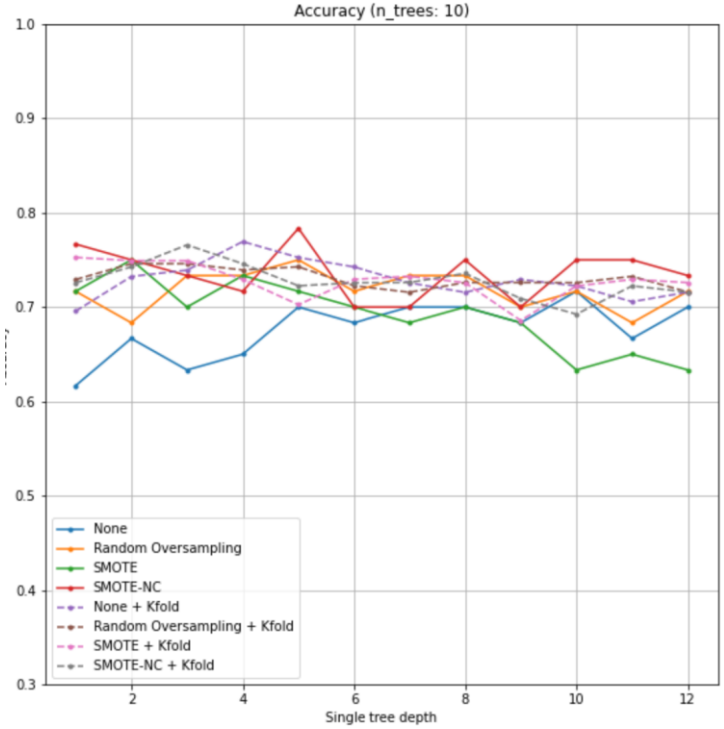
Best configuration:

Model	Kfold	Oversampling Method	Max_depth	Accuracy
DecisionTree	YES	smote	3	0.762712

Random Forest

Random Forest is an esemble of decision trees. Basically, we build a number of decision trees on bootstrapped training sample. But when building these decisione trees, each time a split in a tree is performed, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. Typically, we choose $\boldsymbol{m \approx \sqrt{p}}$.





Hyperparameters:

max_depth=[1,2,3,4,5,6,7,8,9,10,11,12]

n_estimators = [5,10,20,50,100]

Best configuration:

Model	Kfold	Oversampling Method	Max_depth	N_estimators	Accuracy
RandomForest	NO	smoteNC	5	10	0.783333

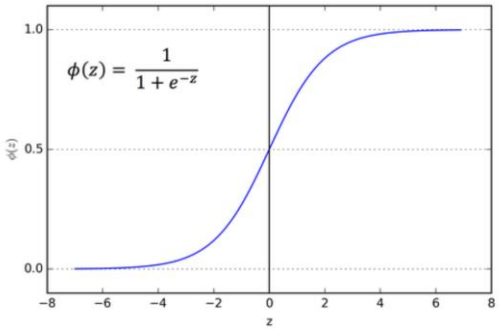
Logistic Regression (1)

Considering our dataset our response (DEATH_EVENT) falls into two category 0 or 1. Logistic Regression models the probability that this response belongs to a particular category. In our analysis, since our target variable is binary we will use "Binomial Logistic regression". We can define:

$$p(X) = P(Y = 1|X)$$

as the probability that X belongs to class 1. In logistic regression, we use the *logistic function*:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$



After a bit of manipulation we will obtain:

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0+\beta_1 X}$$

The quantity $\frac{p(X)}{1-p(X)}$ is called odds (and can take any values between 0 and infinite). By taking the logarithm of both sides, we have:

$$\log(\frac{p(X)}{1 - p(X)}) = \beta_0 + \beta_1 X$$

where the left-hand side is called the log-odds or logit.

In order to estimate parameters $\beta 0$ and $\beta 1$, we use the maximum likelihood:

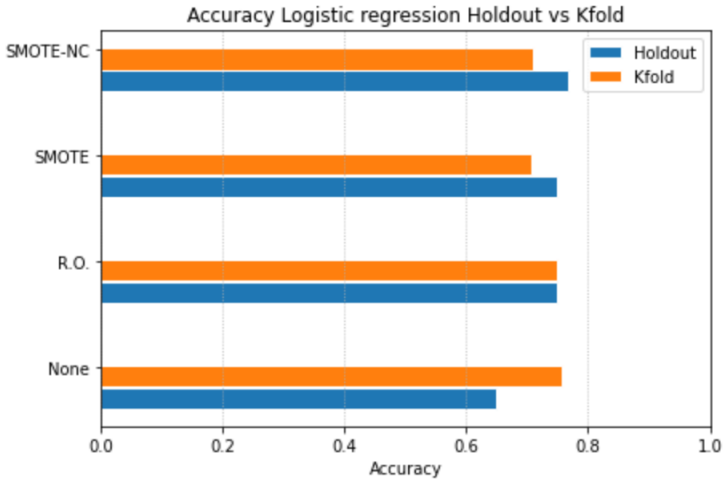
$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i))$$

which gives the probability of observed zeros and ones in data and parameters are chosen in order to maximize it.

Logistic Regression (2)

Best configuration:
C=1

Model	Kfold	Oversampling Method	Accuracy
LogisticRegression	NO	None	0.650000
LogisticRegression	NO	oversampling	0.750000
LogisticRegression	NO	smote	0.750000
LogisticRegression	NO	smoteNC	0.766667
LogisticRegression	YES	None	0.755989
LogisticRegression	YES	oversampling	0.749266
LogisticRegression	YES	smote	0.705876
LogisticRegression	YES	smoteNC	0.709209



Support Vector Machine (1)

It consists of finding a Hyperplane that separates the training set with the largest possible margin (meaning the smallest distance between the hyerplane and a training sample).

Hard-SVM

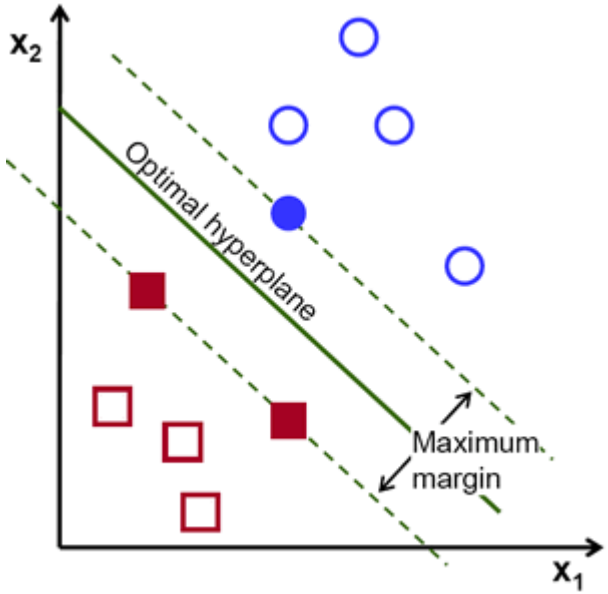
$$min_{w,b}\frac{1}{2}\|w\|^2 \;\; s.t. \;\; \forall i, \;\; y_i\big(\langle w, x_i\rangle + b\big) > 1$$

Soft-SVM

$$min_{w,b}\Big(\frac{1}{2}\|W\|^2 + C\sum_{i=1}^{m}\xi_i\Big) \;\; s.t. \;\; \forall i, \;\; y_i\big(\langle w, x_i\rangle + b\big) \geq 1 - \xi_i \;\; and \;\; \xi_i \geq 0$$

Kernel Trick

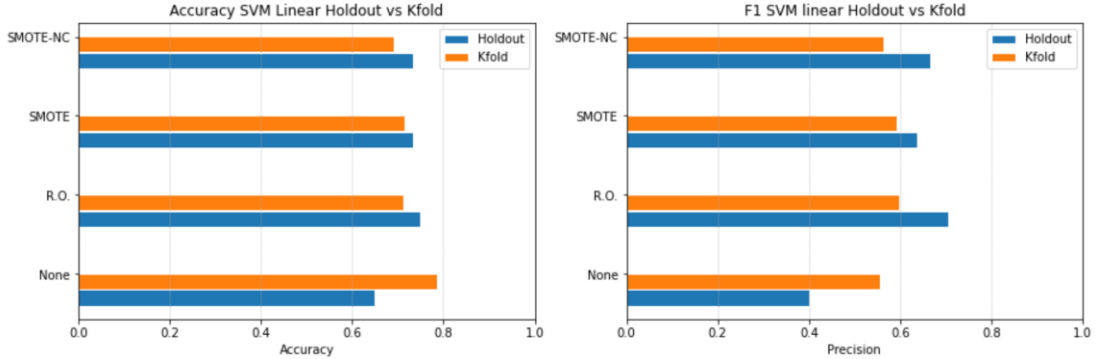
$$K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}')\rangle$$



Support Vector Machine (2)

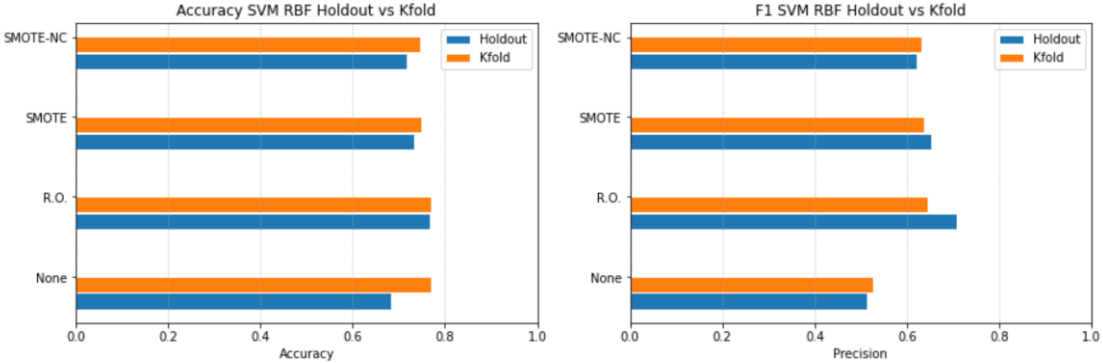
Hyperparameters (Linear SVM): C=[0.01,0.1,1,10]
Best Configuration: C=1

Linear SVM



RBF

$$K(x, x') = exp(-\gamma \cdot ||x - x'||^2)$$



Results (1)

Accuracy

Oversampling	Decision Tree Accuracy	Random Forest Accuracy	Logistic Regression Accuracy	SVM Linear Accuracy	SVM RBF Accuracy
None (Holdout)	0.733	0.750	0.650	0.650	0.683
Random Oversampling (Holdout)	0.733	0.751	0.750	0.750	0.767
SMOTE (Holdout)	0.750	0.766	0.750	0.733	0.733
SMOTE-NC (Holdout)	0.734	0.783	0.767	0.733	0.717
None + Kfold	0.769	0.786	**0.779**	**0.782**	0.758
Random Oversampling + Kfold	**0.773**	**0.809**	0.719	0.708	**0.775**
SMOTE + Kfold	0.723	0.752	0.710	0.720	0.741
SMOTE-NC + Kfold	0.771	0.789	0.744	0.690	0.748

Results (2)

F1

Oversampling	Decision Tree F1	Random Forest F1	Logistic Regression F1	SVM Linear F1	SVM RBF F1
None (Holdout)	0.636	0.667	0.363	0.400	0.512
Random Oversampling (Holdout)	0.667	0.693	0.694	0.706	0.708
SMOTE (Holdout)	0.681	0.730	0.681	0.636	0.652
SMOTE-NC (Holdout)	0.667	0.745	0.708	0.667	0.622
None + Kfold	0.511	0.600	0.525	0.555	0.527
Random Oversampling + Kfold	0.696	0.701	0.620	0.601	0.666
SMOTE + Kfold	0.550	0.645	0.580	0.600	0.634
SMOTE-NC + Kfold	0.590	0.668	0.648	0.555	0.629

Thank you for your attention