POLITECNICO DI TORINO

Master's Degree
in Data Science and Engineering

Chronic Kidney Disease Analysis

Professors
prof. FRANCESCO VACCARINO
prof. MAURO GASPARINI

Candidate
Rostislav Timuta
(280238)

Table of contents

INTRODUCTION

The burden of chronic diseases has increased significantly in recent decades. Historically, chronic diseases were considered to be a health problem only occurring in the developed world. However, four out of five chronic disease deaths occur in low- and middle-income countries today. The estimated number of deaths caused by chronic diseases in India raised from 3.78 million in 1990 (40.4% of all deaths) to 7.63 million in 2020 (66.7% of all deaths).[1]

The current analysis is based on real data collected by the Indian Apollo Hospitals during the month of July, 2015.

DATA OVERVIEW AND DESCRIPTION

The dataset in analysis contains 400 distinct instances with 24 + class attributes and the target is the ‘classification’, which is either 'ckd' or 'notckd'. Ckd stands for chronic kidney disease. Missing values are denoted by ‘?’ and there are several of them.

NUMERICAL FEATURES

- Age - age in years
- Blood Pressure - bp - in mm/Hg
- Specific Gravity - sg - (1.005,1.010,1.015,1.020,1.025)
- Albumin - al - (0,1,2,3,4,5)
- Sugar - su - (0,1,2,3,4,5)
- Blood Glucose Random - bgr - in mgs/dl
- Blood Urea - bu - in mgs/dl
- Serum Creatinine - sc - in mgs/dl
- Sodium - sod - in mEq/L
- Potassium - pot - in mEq/L
- Hemoglobin - hemo - in gms
- Packed Cell Volume
- White Blood Cell Count - wc - in cells/cumm
- Red Blood Cell Count - in - millions/cmm

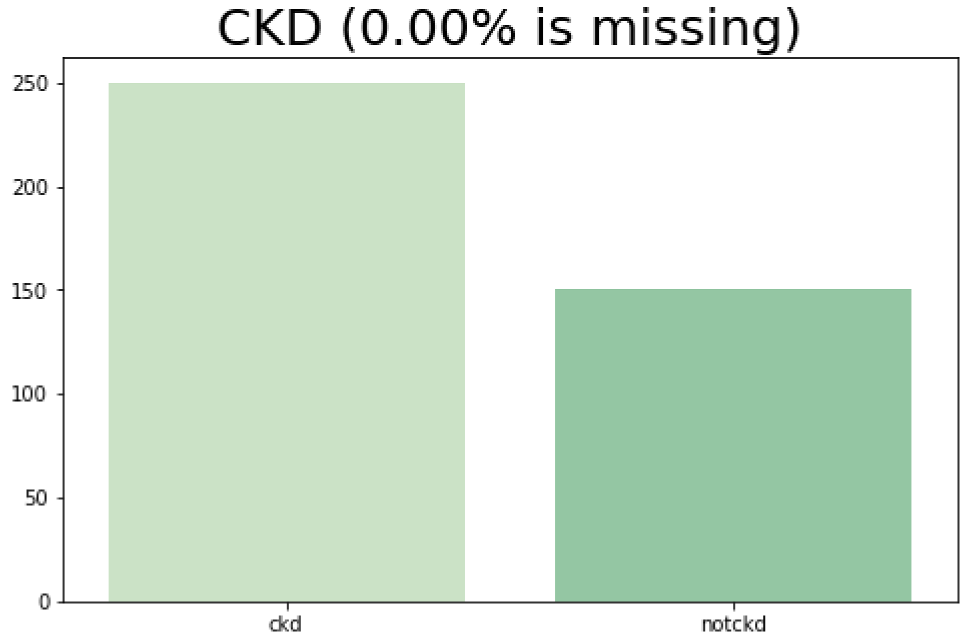
CATEGORICAL FEATURES

- Red Blood Cells - rbc - (normal,abnormal)
- Pus Cell - pc - (normal,abnormal)
- Pus Cell clumps - pcc - (present,notpresent)
- Bacteria - ba - (present,notpresent)
- Hypertension - htn - (yes,no)
- Diabetes Mellitus - dm - (yes,no)
- Coronary Artery Disease - cad - (yes,no)
- Appetite - appet - (good,poor)
- Pedal Edema - pe - (yes,no)
- Anemia - ane - (yes,no)
- Class - class - (ckd,notckd)

DATA EXPLORATION

BALANCE

Exploration of the data is the first step of the analysis, and is a necessary step in gaining a comprehensive understanding of the data and their distributions. In this sense, the class label distribution is of paramount importance, since it determines the balance or imbalance of the dataset. An "unbalanced" dataset is one in which samples from one class outnumber those from the other, as in the current analysis.

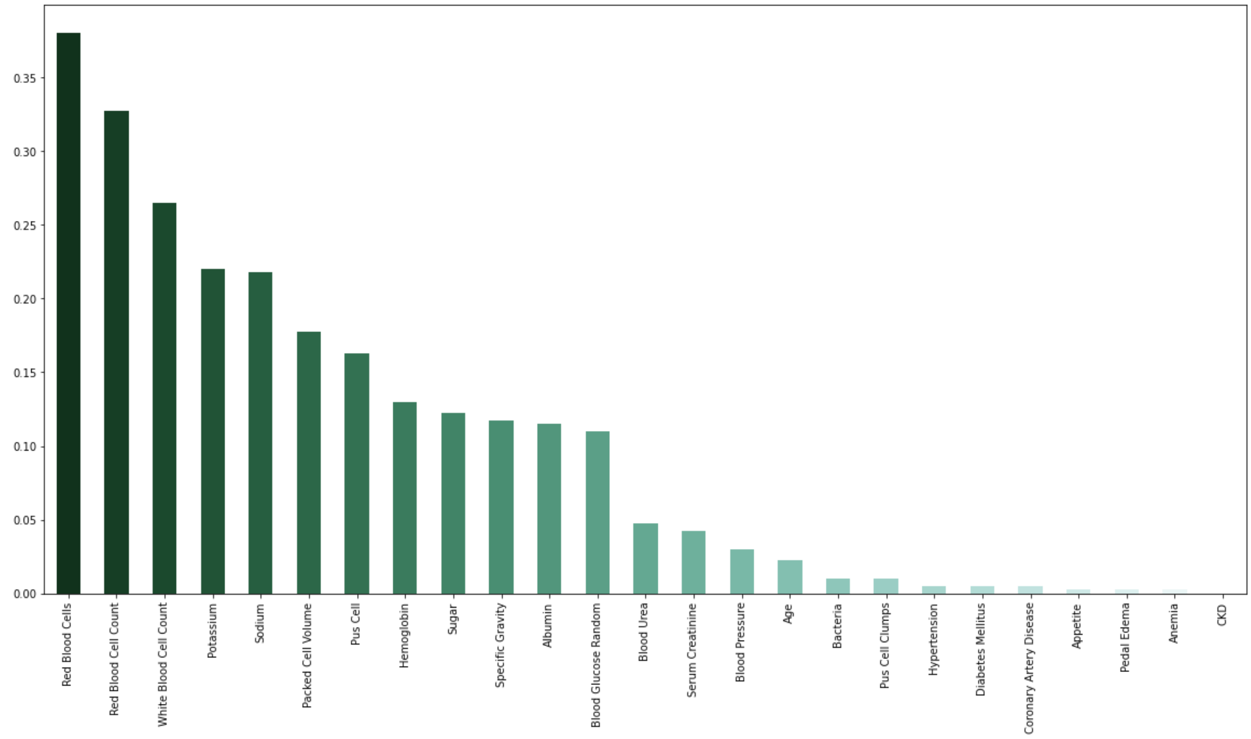


As we can see in the figure, the date is slightly skewed in favor of one class and this can cause issues in the training. For example Random Forests classifiers are sensitive to the proportion of classes. These and other machine learning classifiers that cope with unbalanced training datasets, tend to favor the class with the largest proportion of observations.

MISSING VALUES

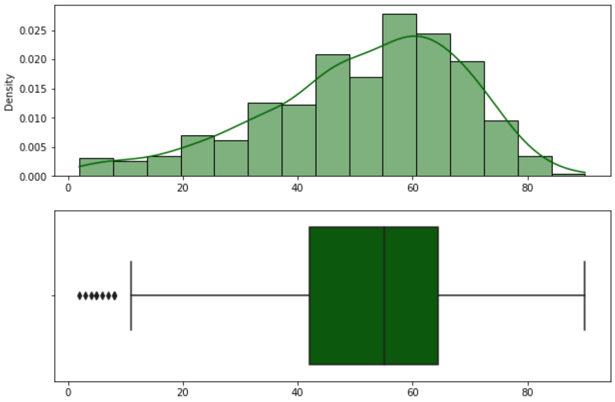
As part of data exploration, it is important to look for the existence of Nan / Null values since they have to be handled, whether by replacing them with a default value, by replacing them with a custom value (such as the most present class value, the mean, the median, etc.), or by dropping all records containing them.

This dataset contains several missing values with the following proportions.

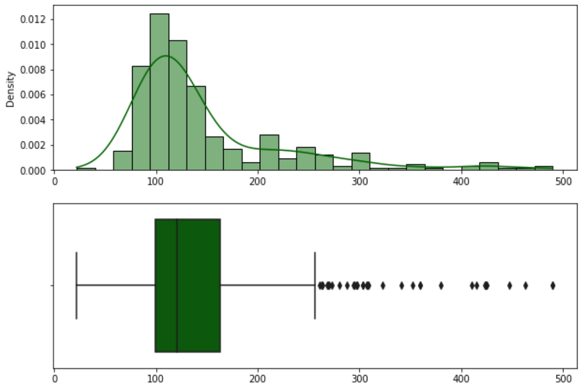


Proportions of Missing Values

DISTRIBUTION OF NUMERICAL FEATURES

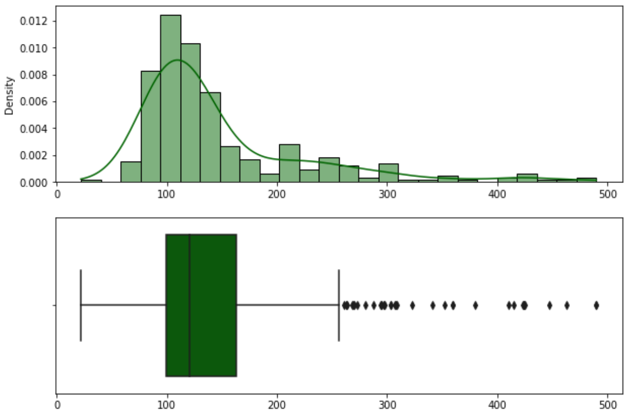


Age (2.25% is missing)

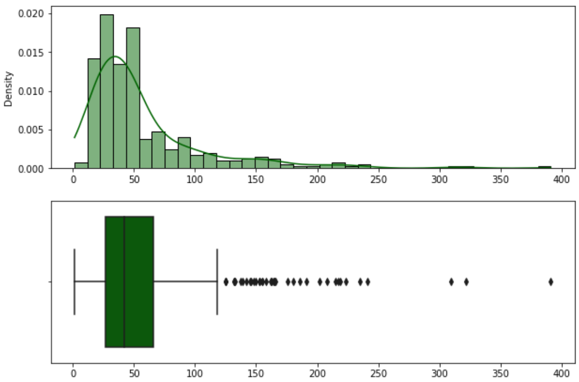


Blood Glucose Random (11.00% is missing)

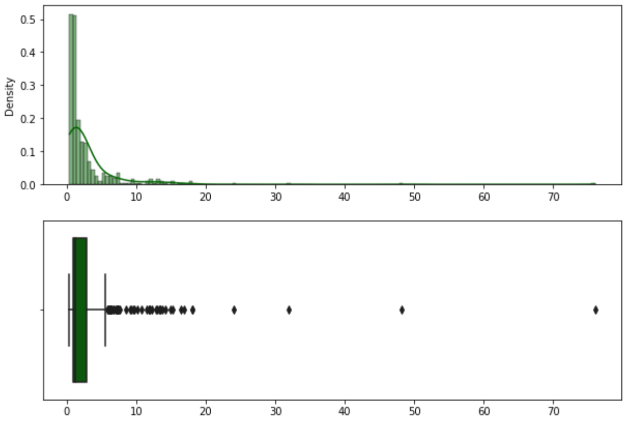
Blood Glucose Random (11.00% is missing)



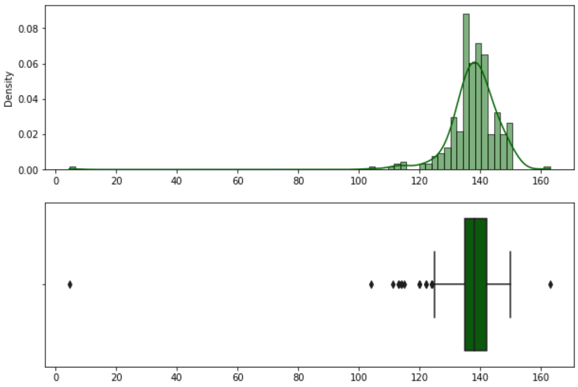
Blood Urea (4.75% is missing)



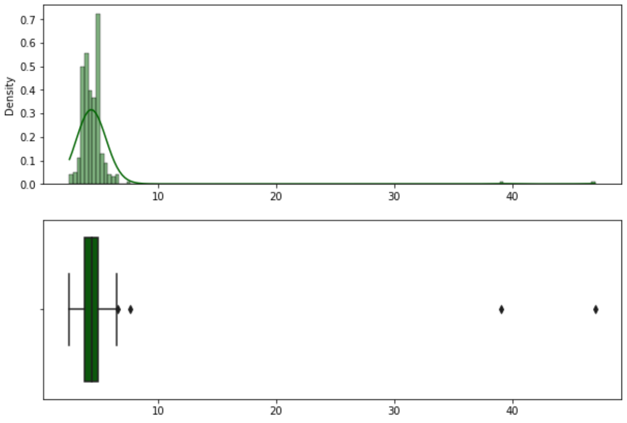
Serum Creatinine (4.25% is missing)



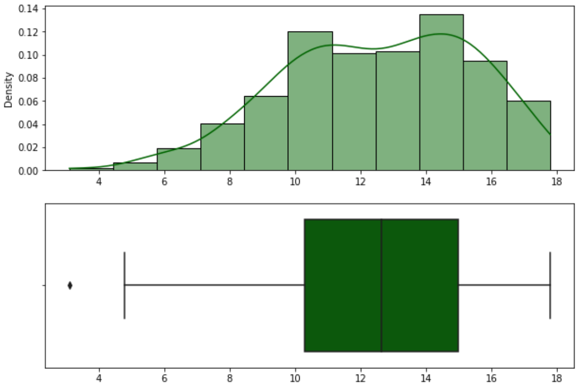
Sodium (21.75% is missing)



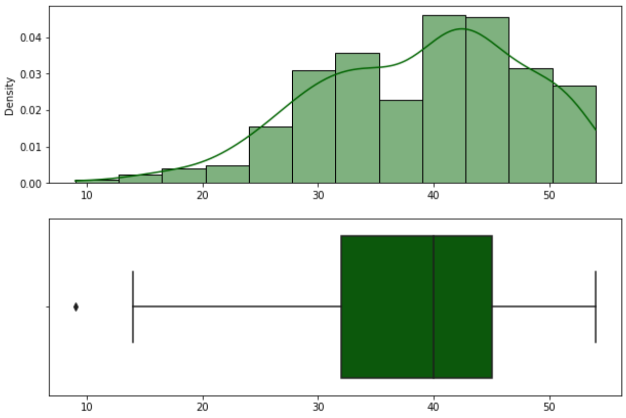
Potassium (22.00% is missing)



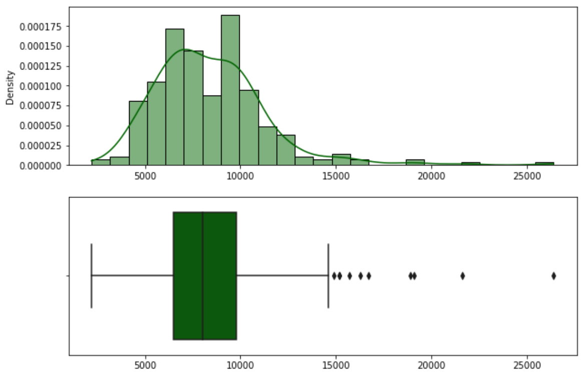
Hemoglobin (13.00% is missing)



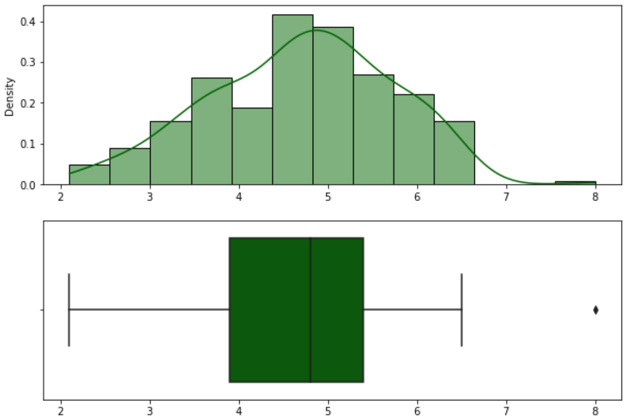
Packed Cell Volume (17.75% is missing)



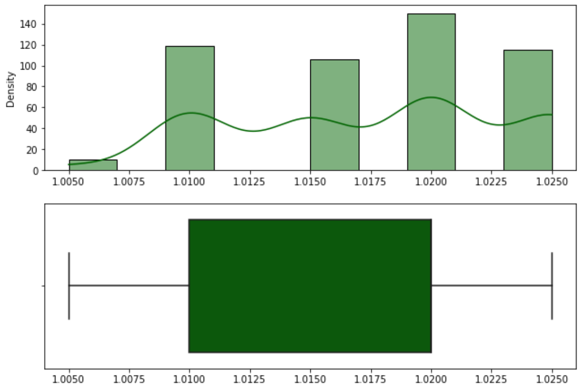
White Blood Cell Count (26.50% is missing)



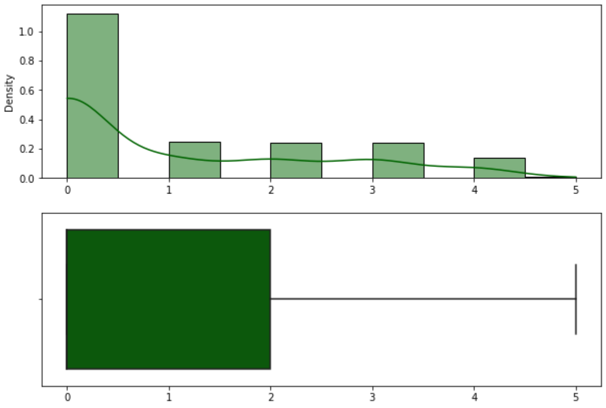
Red Blood Cell Count (32.75% is missing)



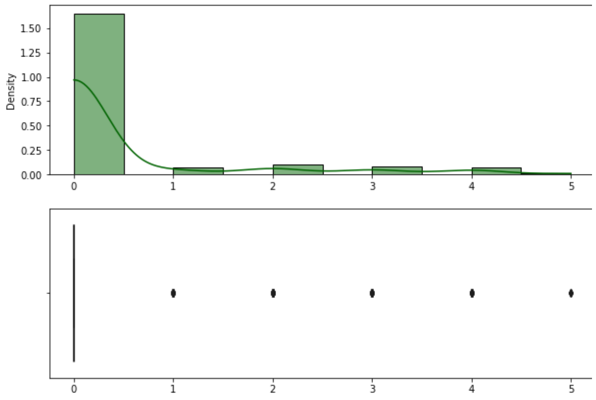
Specific Gravity (11.75% is missing)



Albumin (11.50% is missing)

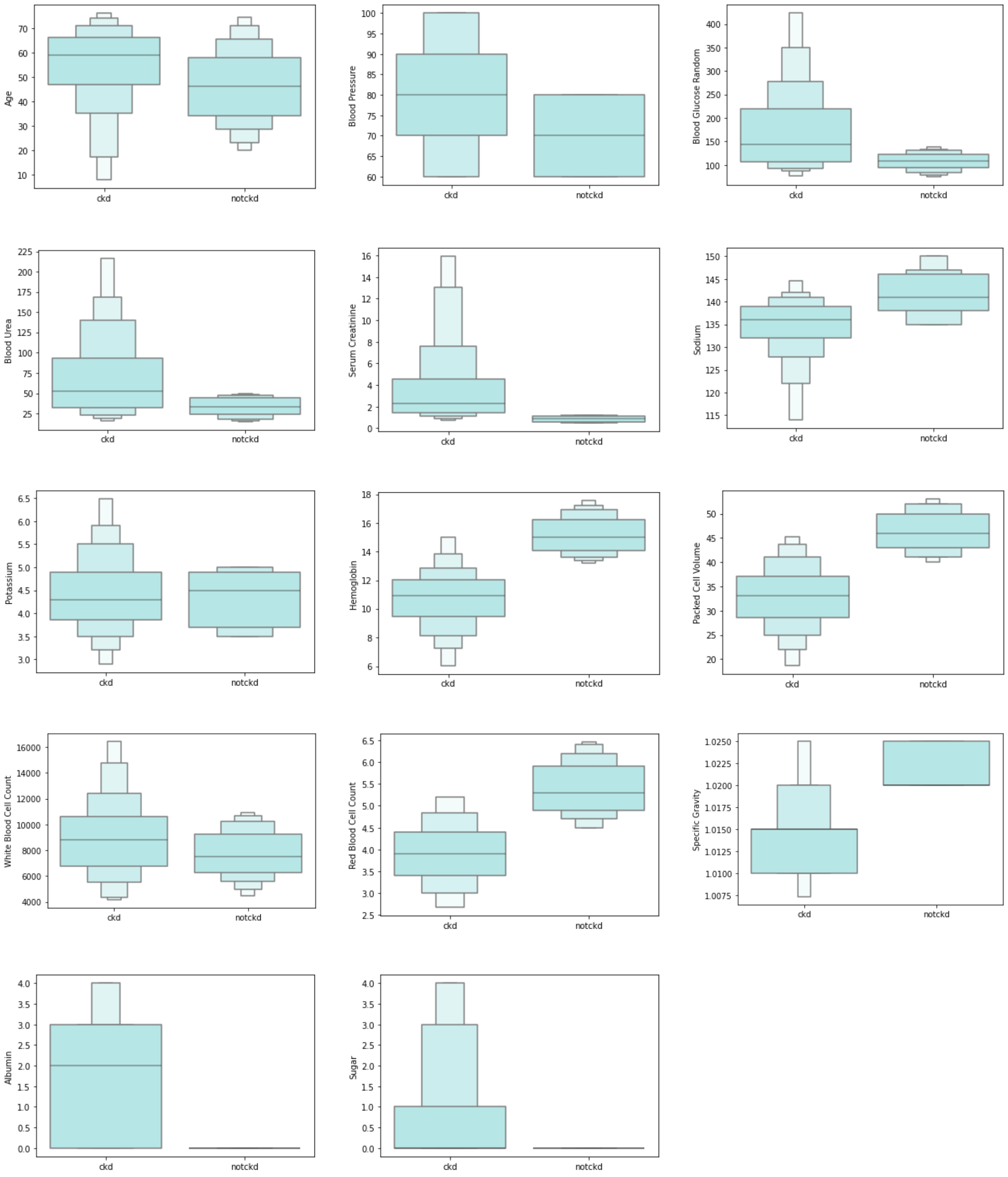


Sugar (12.25% is missing)

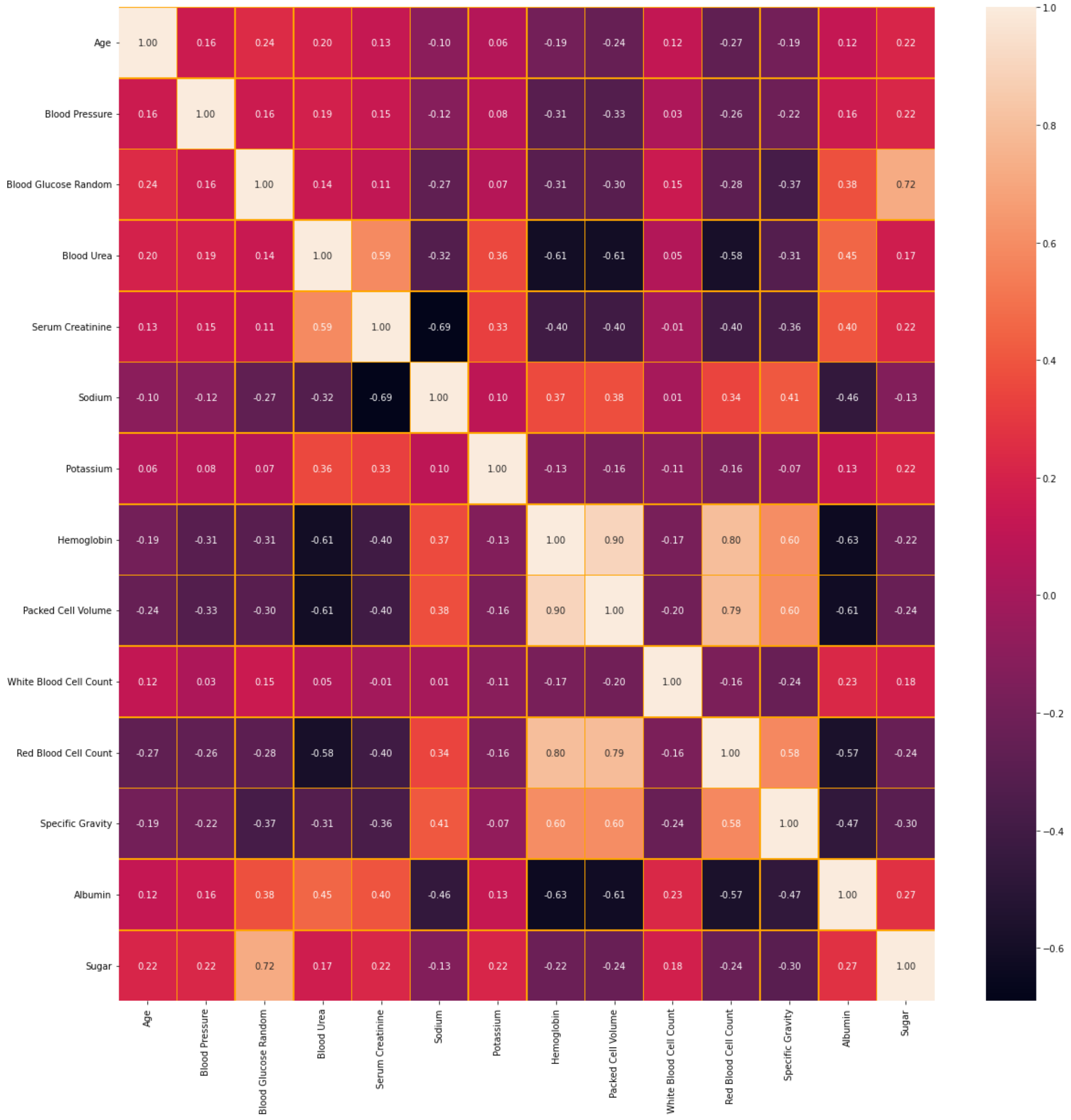


Some features show some very distant outliers.
Note: Specific Gravity, Albumin and Sugar have discrete values but they're going to be treated as continuous because since these are biological variables, in reality they are continuous.
It is also easy to notice that some features have high proportions of missing values.

NUMERICAL FEATURES VS TARGET VARIABLE



CORRELATION MATRIX



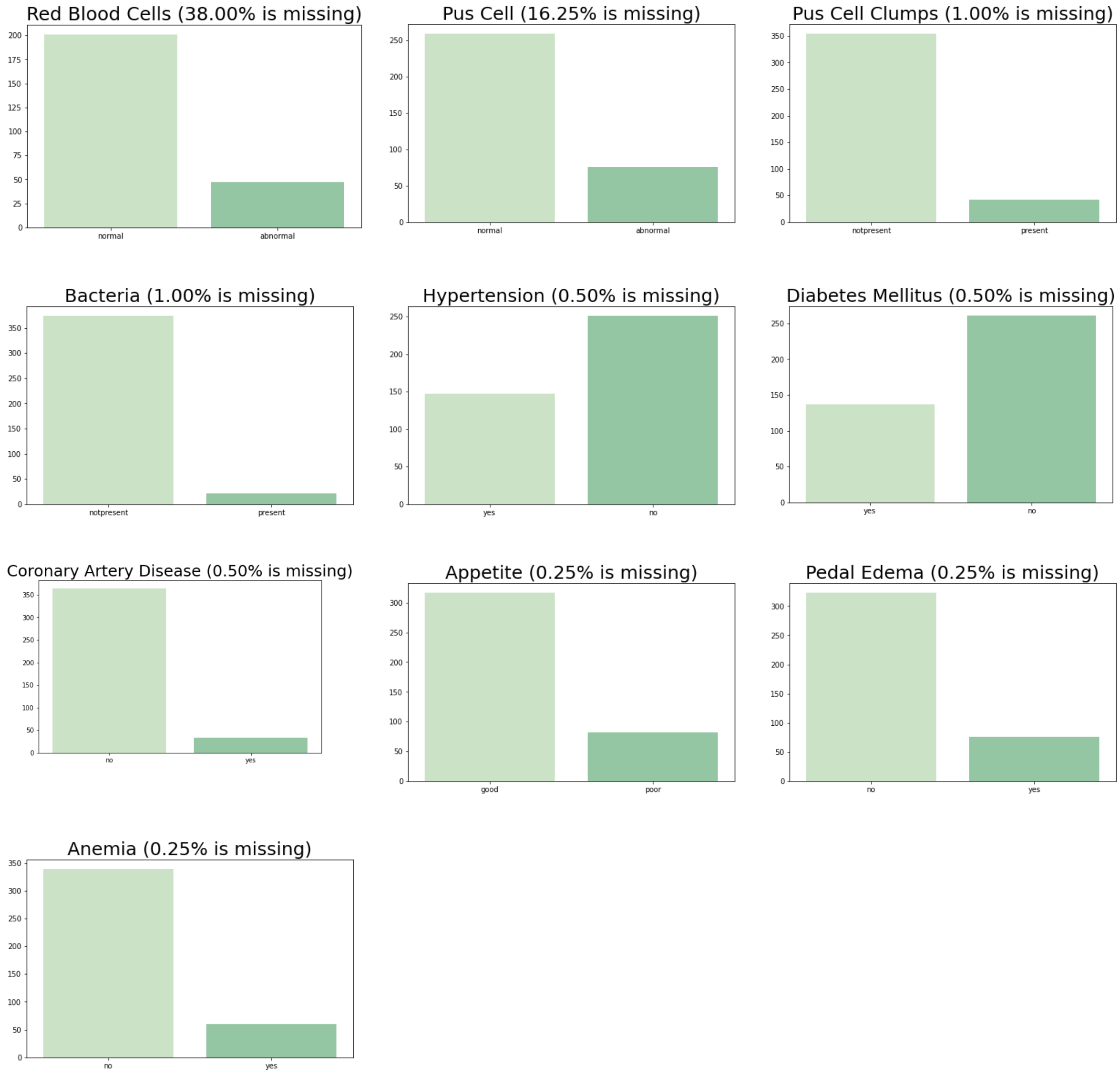
Positive Correlation:

- Blood Glucose Random and Sugar
- Blood Urea and Serum Creatinine
- Hemoglobin and Red Blood Cell Count
- Packed Cell Volume and Red Blood Cell Count

Negative Correlation:

- Blood Urea and [Red Blood Cell Count, Packed Cell Volume, Hemoglobine]
- Serum Creatinine and Sodium
- Albumin and [Hemoglobin, Packed Cell Volume, Red Blood Cell Count]

DISTRIBUTION OF CATEGORICAL FEATURES



Red Blood Cells (38.00% is missing)

Pus Cell (16.25% is missing)

Pus Cell Clumps (1.00% is missing)

Bacteria (1.00% is missing)

Hypertension (0.50% is missing)

Diabetes Mellitus (0.50% is missing)

Coronary Artery Disease (0.50% is missing)

Appetite (0.25% is missing)

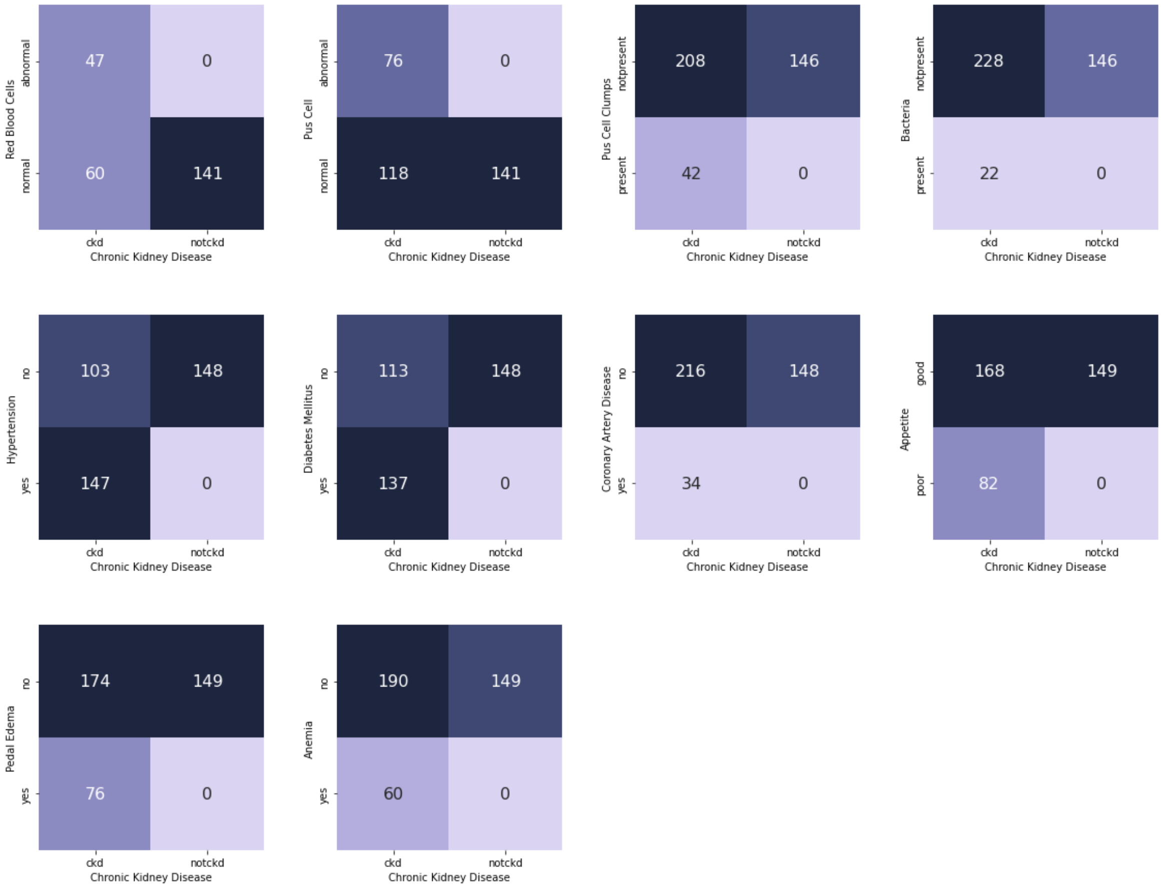
Pedal Edema (0.25% is missing)

Anemia (0.25% is missing)

As for the numerical features, also here we can notice a high percentage of missing values for some numerical ones. On the other hand some of them have almost none.

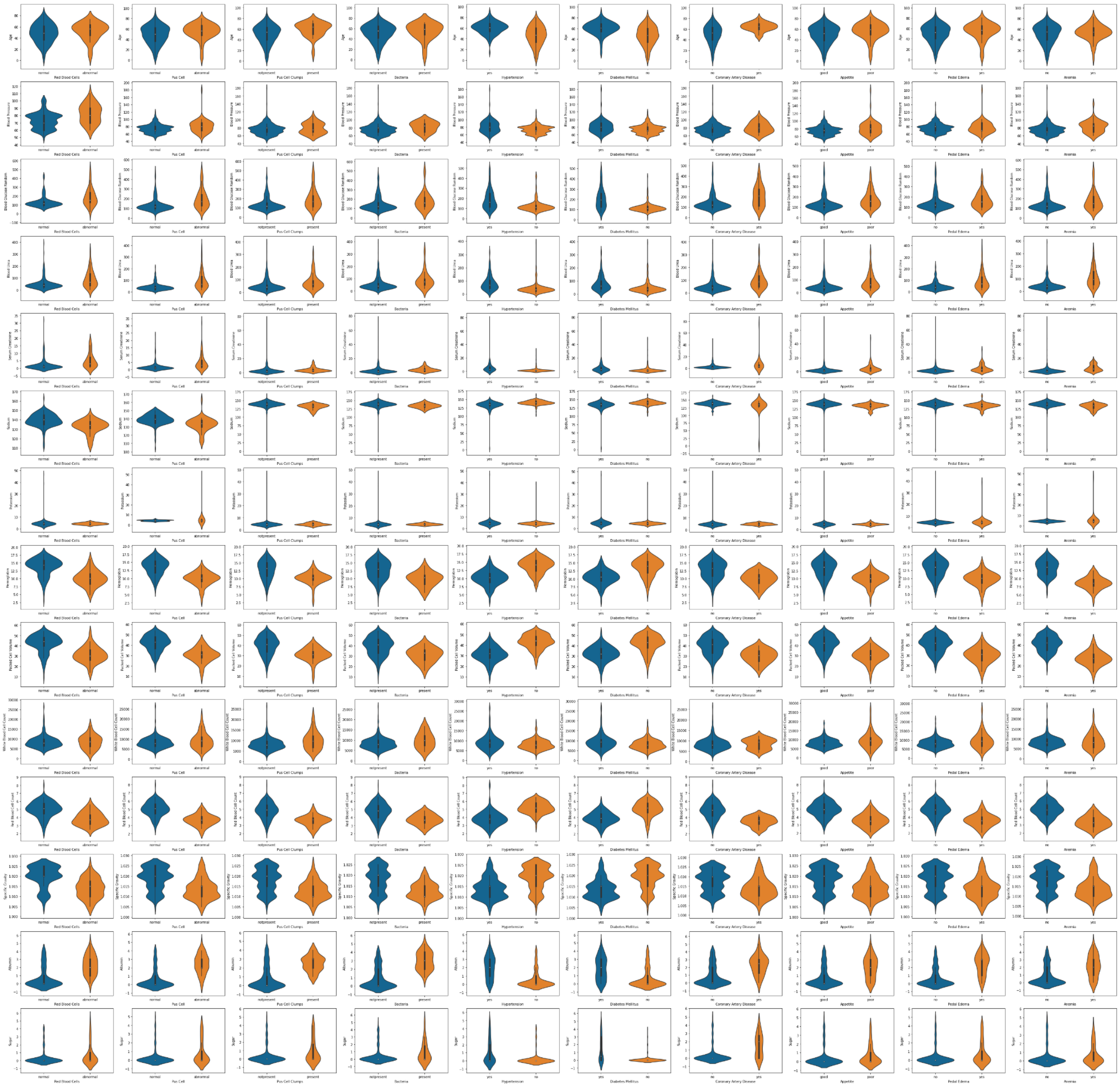
Some conditions or diseases such as diabetes and hypertension seem to be commun.

CATEGORICAL FEATURES VS TARGET VARIABLE



NUMERICAL & CATEGORICAL FEATURES

DISTRIBUTIONS AND CORRELATIONS



The importance of data exploration is relevant since by plotting all the density distributions, all the possible correlation configurations we are able to visually distinguish and interpret some interesting characteristics.

We’ve seen how some diseases are potentially linked to CKD and how for some others it is hard to establish a connection.

DATA PREPARATION

In order to feed machine learning models, the second step involves preparing data, which means we must go through selection to ensure the data we maintain is relevant, Nan values management, possible feature engineering, outlier management, categorical data encoding (not always required, but we will do this once for all), normalization, under/oversampling, possible reductions in dimensionality, and dividing the dataset into training, validation, and test sets.

FILLING NaN VALUES

Numerical missing values have been filled with their medians.
Categorical missing values have been filled with their mode.

LABEL ENCODING

Machine learning models generally require all input and output variables to be numeric. Consequently all the numerical features in the dataset have been numerically encoded.

FEATURES SCALING

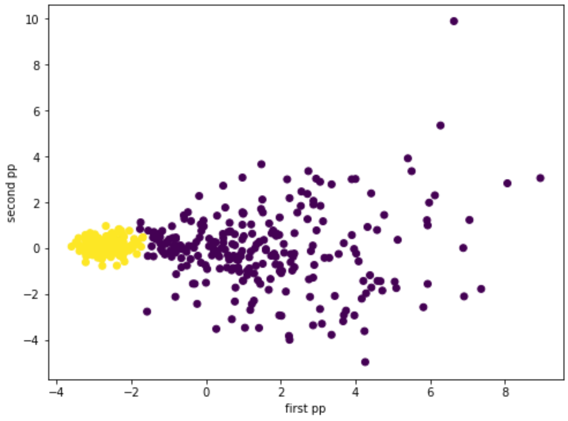
Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. In our case, StandardScaler has been applied. It standardizes features by removing the mean and scaling to unit variance.
The standard score of a sample x is calculated as: $z = (x - u)/s$

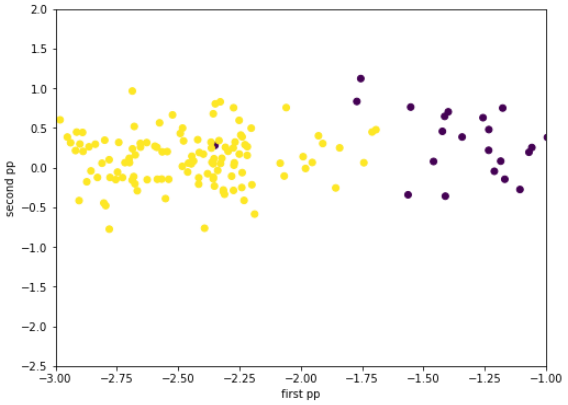
PCA

In order to perform dimensionality reduction, is it possible to use Principal Component Analysis (PCA). This method creates new uncorrelated variables that successively maximize the variance finding the most accurate data representation in a lower dimensional space.

Considering our dataset's number of instances and features, making use of the PCA doesn't make a significant difference.

Below the representation on PCA2.





SETTINGS AND METRICS

SMOTE

As previously mentioned, it is challenging working with imbalanced datasets because most machine learning techniques will ignore the minority class having as a result a poor performance on it.

One approach to addressing imbalanced datasets is to oversample the minority class. This can be done using the Synthetic Minority Oversampling Technique, or SMOTE.

It works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line.

SMOTE first selects a minority class instance at random and finds its k nearest minority class neighbors. The synthetic instance is then created by choosing one of the k nearest neighbors 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.[2]

METRICS

Performance metrics are a part of every machine learning pipeline.They are used to monitor and measure the performance of a model (during training and testing), and don't need to be differentiable.

For this project 4 metrics have been taken into consideration:

- **Accuracy**: States how many predictions are actually corrected.

- **Precision**: Proportion of Positive correctly predicted.
- **Recall**: Proportion of Positive correctly detected.
- **F1 Score**: The harmonic mean of precision and recall, indicating how precise the classifier is

MODEL SELECTION

A really brief explanation:

KNN

The K-Nearest Neighbor algorithm is one of the simplest ML algorithms and it assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories. In order to classify a new data point, it stores all available data and compares the similarity of this data to the previous data.

DECISION TREE

In tree-based methods, the predictor space is stratified or segmented into a number of simple, non-overlapping regions. As a result, a local constant is obtained for regions with boundaries parallel to the axes. The majority of the predictions obtained in the training observations are assigned to every observation that falls within a region. Learning simple decision rules from data features is used to build a model that predicts the value of a variable. The Classification embraces a root-to-leaf path: at each node the child is chosen (usually among two) based on a threshold T of a specific feature that characterizes that node. The tree is built through a recursive binary splitting in a supervised setting.

RANDOM FOREST

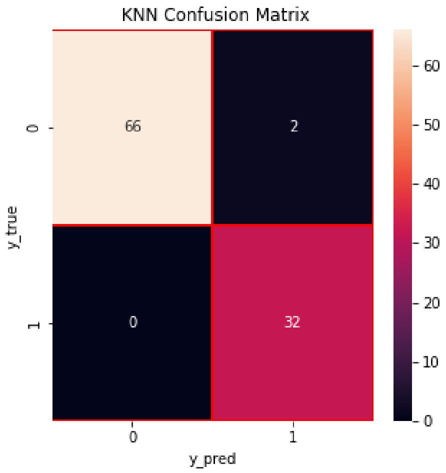
Random Forests provide a huge improvement of the simple decision tree by bagging (building several decision trees on bootstrapped training samples) and by choosing a random subset of all predictors as split candidates from the full set of predictors. This decorrelates the trees from one another and reduces the variance when computing averages, resulting in more accurate and robust models. The final prediction is then reached by majority voting among all trees.

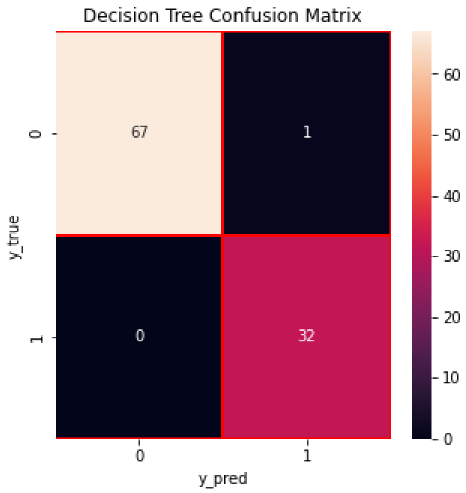
SVM

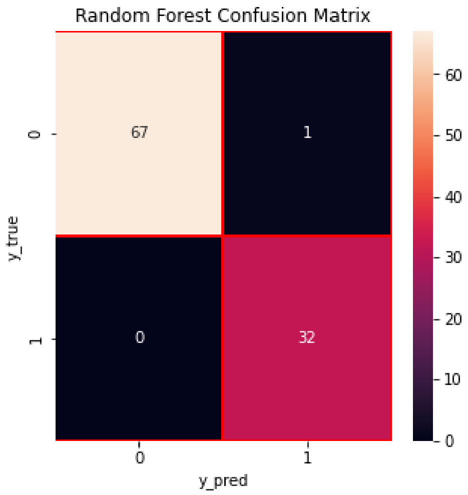
Support Vector Machine is able to find a hyperplane that maximizes the distance between two classes in a feature space. A good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class.
It builds a learning model that assigns new examples to one group or another. By these functions, SVMs are called a non-probabilistic, binary linear classifier.

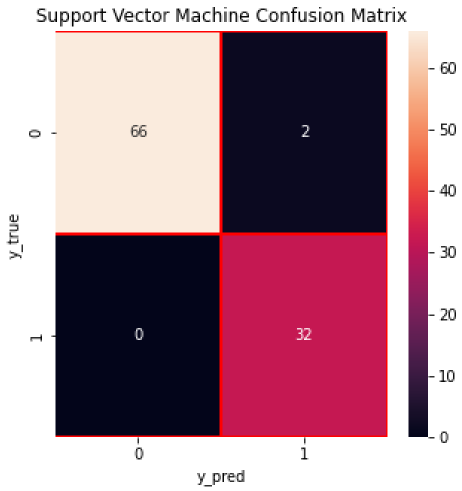
RESULTS

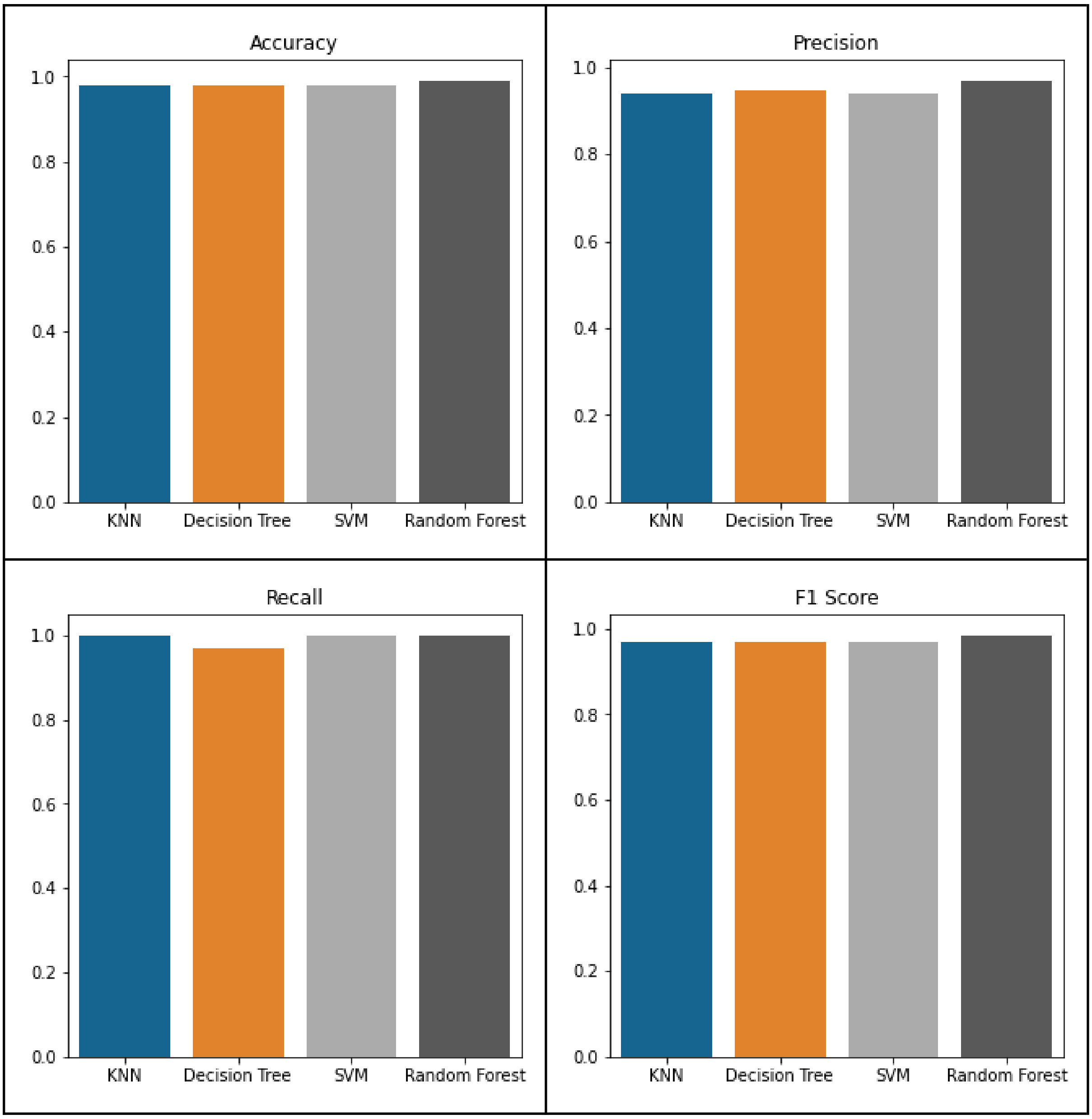
The following Confusion Matrices allow us to better visualize the performance of the algorithms used in this project.











As we can see from the last 2 figures, all the models have high results for all 4 metrics and Random Forest performs the best overall.

Last consideration: Not all metrics are equal. Accuracy is reliable and meaningful in case of a well balanced dataset and/or when the only goal is to have the overall right prediction but not caring about classes subdivision.

In some fields like fraud detection or healthcare, the focus is the right detection of a certain class, usually being in a very small proportion. Predicting cancer or chronic kidney disease (in our case) when it is not true is less worse than not predicting cancer when it actually is. Recall is a useful metric in these cases.

REFERENCES

[1] https://www.karger.com/Article/Fulltext/199460

[2] Imbalanced Learning: Foundations, Algorithms, and Applications, Haibo He