| **Report**  **Machine Learning Project**  **Topic:**  **Chronic Kidney Disease**  **Realised by:**  Safa Chaari  Abir Barouni  Ghofrane Soltani  Mariem Mazouz  Mohamed Aziz Omrani  Aziz Tebassi  **2022 – 2023** |
| --- |

**SUMMARY**

**CHAPTER 0: Project Overview**

[**0.1 Introduction**](#_n3wh4dunhxah) **4**

[**0.2 Presentation of the project**](#_wr69o8ooliws) **4**

[0.2.1 project framework](#_rsr9de4f1ibx) 4

[0.2.2 Context and Issue](#_phocmccbqvf7) 4

[0.2.3 Objectif](#_ofu5ys581wkk) 5

[**0.3 Methodology adopted**](#_5y8xg2brgq3) **5**

[0.3.1 CRISP-DM phases](#_5v7icxqyz32k) 5

[0.3.2 Choice of methodology](#_ujfq2cgbiiys) 7

[**0.4 Conclusion**](#_4nl3s8ww0zme)

**CHAPTER 1:Business understanding 7**

[**1.1 Introduction**](#_1nw1hqsui07z) **7**

[**1.2. Business objectives**](#_3niutnm7yii4) **7**

[**1.3 Project planning**](#_z4uwzuqnuj30) **8**

[**1.4 Used tools**](#_s3782mjnlrav) **8**

[**1.5 Steps**](#_a60ttg2v6wu1) **9**

[**1.6 Conclusion**](#_ipi9ad7ei25w)

**CHAPTER 2: Data understanding 10**

[**2.1 Introduction**](#_a2ufixu6qtir) **10**

[**2.2 Data source**](#_7364orc2n00v) **10**

[**2.3 Features description**](#_3dfcqp6h0u0u) **11**

[**2.4 Visualizing data for categorical features**](#_aiuvbfa8rvb1) **12**

[**2.5 Visualizing data for numerical features**](#_toogr3a898zu) **15**

[**2.6 Conclusion:**](#_e55lbzcstc5r)

**CHAPTER 3: Data Preparation 18**

[**3.1 Introduction**](#_bxvckr4soj7r) **19**

[**3.2 Data Cleaning**](#_l3c28lknyrif) **19**

[**3.3 Feature engineering**](#_dj3nru30umyn) **21**

[**3.4 Feature selection:**](#_j0rshb9fvymy) **27**

[3.4.1 Correlation matrix:](#_babqnzs7kxn6) 27

[3.4.2 Extracting the important features of the dataset:](#_1li6spy0mzlx) 28

[**3.5 Conclusion**](#_r3zqswy55kcx)

**CHAPTER 4: Modeling 28**

[**4.1 Introduction:**](#_4h6x1n6xnvd) **29**

[**4.2 Used methods:**](#_90bry2neqyon) **30**

[**4.3 Logistic regression:**](#_oxldjwna0934) **30**

[4.3.1 Definition:](#_2ize16t0dhvz) 30

[4.3.2 Searching for the best parameters](#_qdte6uvergm0) 31

[**4.4 Adaboost :**](#_q8pr6wdrv1vx) **32**

[4.4.1 Definition:](#_x8mu8gz5nwd6) 32

[4.4.2. Searching for the best parameters](#_nyph6oqxn9gf) 34

[**4.5 K-nearest neighbors (KNN):**](#_jrwj5wsb3nk7) **35**

[4.5.1. Definition:](#_7mkdqrs1frgx) 35

[4.5.2. Searching for the best parameters](#_bxf5em7w414h) 36

[**4.6 Decision Tree Classifier:**](#_ylb8aayud8m) **37**

[4.6.1 Definition](#_bnjxzt2serfd) 37

[4.6.2 Searching for the best parameters](#_muai23wgmiqe) 37

[**4.7 Random Forest:**](#_vudewfi06pe3) **38**

[4.7.1 Definition:](#_qvjljsoq6894) 38

[4.7.2 Searching for the best parameters](#_va8nyot19nqe) 39

[**4.8 SVM:**](#_mr7z58wkiul) **39**

[4.8.1 Definition](#_fakkjplos2xa) 39

[4.8.2 Searching for the best parameters](#_zfy5yl5qudhp) 40

[**4.9 Naive Bayes:**](#_v50aetvrmp5f) **41**

[4.9.1 Definition:](#_7y1oqu5slsvh) 41

[4.9.2 Searching for the best parameters](#_px10n8j308ge) 42

[**4.10 Conclusion:**](#_5rddmo9tlz25)

**CHAPTER 5: Evaluation 42**

[**5.1 Introduction:**](#_174p2f6vqt5y) **43**

[**5.2 Définition:**](#_ungx8s8i6jpu) **43**

[**5.3 Logistic regression**](#_ejynurg5uh10) **46**

[5.3.1 Classification Report :](#_gdg3ood31u39) 46

[5.3.2 Confusion matrix visualization](#_cz3gkrdnh6zh) 47

[**5.4 Decision Tree Classifier:**](#_hoej11c5cpco) **47**

[5.4.1 Classification Report :](#_w771fq62by66) 47

[5.4.2 Confusion matrix visualization](#_906w73ovmy5q) 48

[**5.5 ADaboost:**](#_3gxof2ayld0h) **49**

[5.5.1 Classification Report :](#_97o88vijk23h) 49

[5.5.2 Confusion matrix visualization:](#_2fqtqp5qtbt6) 50

[**5.6 Random Forest Classifier**](#_mnel983l1cff) **51**

[5.6.1 Classification Report:](#_bj9hxxvkfp2q) 51

[5.6.2 Confusion matrix visualization:](#_jm2cutrwvnx6) 52

[**5.7 KNN:**](#_n8g6iqf7s1eb) **53**

[5.7.1 Classification Report :](#_ypykqmlshk) 53

[5.7.2 Confusion matrix visualization:](#_9ecs87sd0zxo) 54

[**5.8 Naive Bayes:**](#_pqv2ydxuvhc0) **54**

[5.8.1 Classification Report :](#_t746f54hc7h2) 54

[5.8.2 Confusion matrix visualization :](#_sd0vp4s4ejlg) 55

[**5.9 SVM:**](#_r18o1nppnld) **56**

[5.9.1 Classification Report :](#_m0fhij65yi3p) 56

[5.9.2 Confusion matrix visualization:](#_32utsffbvmyr) 57

[**5.10 Comparing the performance of the models:**](#_lj1tm41sv2s6) **57**

[**5.11 Selecting the best k-features:**](#_iha5pic9ge4o) **58**

[**5.12 Conclusion:**](#_48fv2l31jivu) **59**

**Table of figures**

Figure 1: Methodology adopted ------------------------------------------------------------------------**7**

Figure 2: Tools -----------------------------------------------------------------------**10**

Figure 3: Description of data -----------------------------------------------------------------------**12**

Figure 4: Percentage of Class Feature -------------------------------------------------------------**14**

Figure 5: Result after replacing ‘?’ with ‘Nan’ -----------------------------------------------------**21**

Figure 6: missing values -----------------------------------------------------------------------**21**

Figure 7: converting features to numerical type --------------------------------------------------**22**

Figure 8: no missing values ------------------------------------------------------------------------**24**

Figure 9: all the categorical values after the use of the ordinal encoding ------------------**25**

Figure 10: Data normalized ------------------------------------------------------------------------**25**

Figure 11: correlation matrix ------------------------------------------------------------------------**29**

Figure 12: Cross validation ------------------------------------------------------------------------**31**

Figure 13: Logistic regression -----------------------------------------------------------------------**32**

Figure 14: adaboost process ------------------------------------------------------------------------**35**

Figure 15: knn process -----------------------------------------------------------------------**36**

Figure 16: Decision tree classifier -------------------------------------------------------------------**38**

Figure 17: Random forest process -------------------------------------------------------------------**39**

Figure 18: SVM -----------------------------------------------------------------------**41**

Figure 19: naive bayes process ----------------------------------------------------------------------**43**

Figure 20: Classification report ----------------------------------------------------------------------**45**

Figure 21: LR classification report --------------------------------------------------------------------**47**

Figure 22: Confusion matrix LR -----------------------------------------------------------------------**48**

Figure 23: Decision tree classifier classification report -----------------------------------------**49**

Figure 24: Confusion matrix of Decision tree classifier -----------------------------------------**49**

Figure 25: adaboost classification report -----------------------------------------------------------**50**

Figure 26: adaboost classification report -----------------------------------------------------------**50**

Figure 27: Classification Report RFC -----------------------------------------------------------**51**

Figure 28: Confusion matrix ----------------------------------------------------------**51**

Figure 29: KNN classification report ----------------------------------------------------------**52**

Figure 30: Confusion matrix ----------------------------------------------------------**53**

Figure 31 :Naive Bayes classification report ------------------------------------------------------**54**

Figure 32: Confusion matrix of Naive Bayes ------------------------------------------------------**55**

Figure 33: SVM classification report ----------------------------------------------------------**55**

Figure 34: Confusion matrix of SVM -----------------------------------------------------------**56**

Figure 35 : Comparaison of models -----------------------------------------------------------------**56**

Figure 36 : Top 9 Features with Adaboost ---------------------------------------------------------**57**

**General Introduction**

Chronic kidney disease (CKD) is a condition characterized by progressive loss of kidney function over time and it is among the top 20 causes of death worldwide and affects approximately 10% of the world's adults.

It describes a clinical entity that causes kidney damage and affects the general health of the human body, it is a serious life-threatening disease, with high rates of morbidity and mortality. Therefore, artificial intelligence techniques are of great importance in the early detection of CKD.

Improper diagnosis and treatment of the disease can eventually lead to endstage renal disease and ultimately lead to the patient’s death.

Moreover, Machine Learning (ML) techniques have acquired an important role in disease prediction and are a useful tool in the field of medical science.

In the present research work, we aim to build efficient tools for predicting CKD occurrence, following an approach which exploits ML techniques.

In this report, we will be going through the Chronic kidney disease modifications we applied on our dataset and doing the complete analysis on the same.

Our main goal will be to predict whether an individual will have chronic kidney disease or not based on the data provided.

For that , on the chapter 0: Project Overview , we are going to introduce the methodology adopted, determine our goals.

Next, We will be planning the steps and the tools we used.

Then we are going to deal with the data understanding in which we will explore the data, verify its quality and will prepare it for the next step which is Data preparation that will be our third chapter.

**CHAPTER 0: Project Overview**

# **0.1 Introduction**

In this chapter , we will present our data ,the objectives of our project and the methodology of the work adopted.

# **0.2 Presentation of the project**

First of all, we will expose the project framework thereafter, we will define the context and the problematic of the subject.

## **0.2.1 project framework**

This project will be delivered 11/12/2022 as a machine learning project at ESPRIT: Private High School of Engineering and Technologies.

We started carrying the project from 28/10/2022.

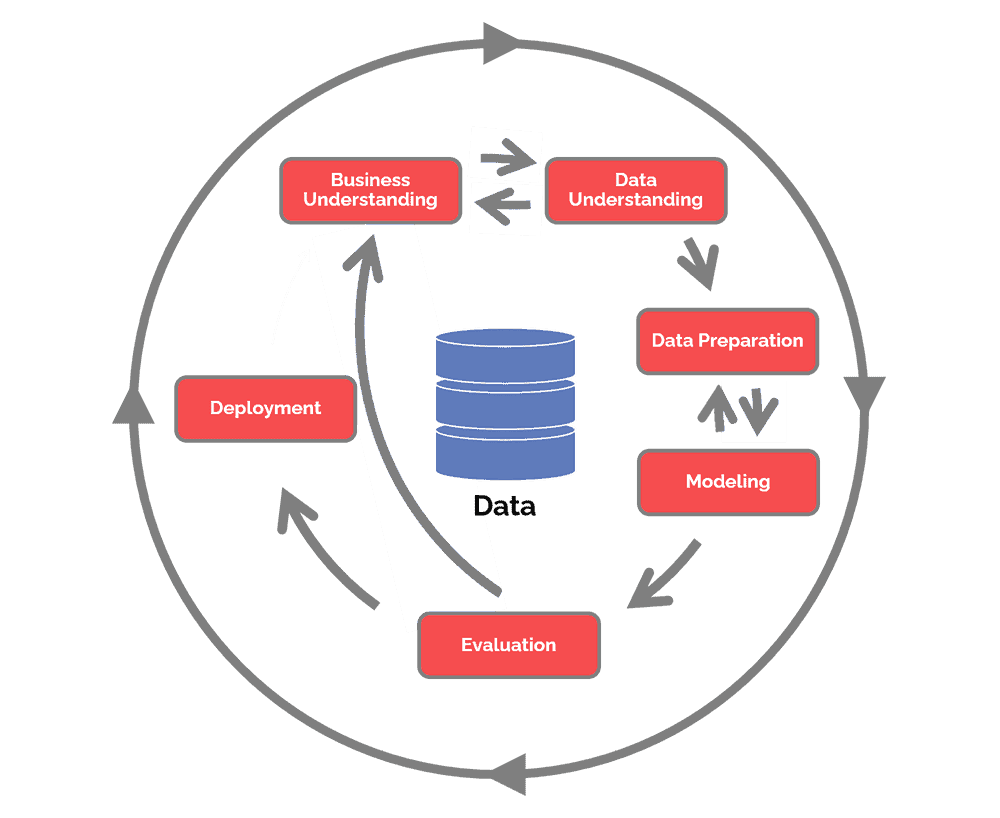
## **0.2.2 Context and Issue**

Chronic kidney disease (CKD) consists of predicting the stages of kidney disease based on 24 attributes which includes symptoms, signs and risk factors of CKD.

## **0.2.3 Objectif**

The purpose of this research is to develop and design an effective and efficient model for CKD prediction.

# **0.3 Methodology adopted**

[****](#D2L_fig_label_figure 1 : Methodology adopted) **Figure 1: Methodology adopted**

The CRoss Industry Standard Process for Data Mining (*CRISP-DM*) is a process model that serves as the base for a [data science process](https://www.datascience-pm.com/data-science-process/). It has six sequential phases:

1. Business understanding – What does the business need?
2. Data understanding – What data do we have / need? Is it clean?
3. Data preparation – How do we organize the data for modeling?
4. Modeling – What modeling techniques should we apply?
5. Evaluation – Which model best meets the business objectives?
6. Deployment – How do we access the results?

## **0.3.1** **CRISP-DM phases**

**1.Business Understanding:** The Business Understanding phase focuses on understanding the objectives and requirements of the project

**2. Data Understanding:** Associate the data with their signification from a business perspective, to determine precisely the data to be analyzed and to determine its quality.

**3. Data Preparation:**The data preparation step groups all activities required to construct, from raw data, a precise set of data to analyze. It thus includes data sorting based on selected criteria, data cleansing, and, most importantly, data recoding to ensure compatibility with any and all algorithms that will be used.

Digital data parametricity and its recoding into categorical data are extremely important and must be executed with the greatest care in order to ensure that used algorithms do not produce inaccurate results during the next step. All data must be centralized in a structured database known as a Data Hub.

**4. Modeling:** This is the actual Data Science step. Modeling includes selecting, configuring and testing various algorithms, as well as deciding on their sequence, which creates a model. The process is initially a descriptive one that generates knowledge and explains why things happened. It then becomes predictive and explains what will happen, and later prescriptive as it helps optimize future situations.

**5. Evaluation:** The aim of the evaluation step is to verify any models or knowledge obtained in order to ensure that they meet the objectives identified at the beginning of the process. The evaluation also informs model deployment decisions, or as required, model improvement ones. At this stage, the robustness and accuracy of developed models are tested.

**6. Deployment:** The final step of the process. It consists in implementing generated models for end users and its aim is to use modeling to format knowledge in such a way that it can be integrated into the decision-making process. Depending on objectives, deployment can thus range from the simple generation of a report describing knowledge obtained to the installation of an application that helps leverage the obtained model to predict unknown values for an element of interest.

## 

## **0.3.2 Choice of methodology**

The CRISP-DM methodology will provide us with the most adequate solution for the needs of customers with a better workflow , it encourages best practices and allows projects to replicate. This methodology provides a uniform framework for planning and managing a project.

# **0.4 Conclusion**

Now that we detailed the overall framework of the system by presenting the problem, the solution that we are going to design and develop, as well as the methodology to follow,

We will begin with the first phase of our methodology, which is business understanding

**CHAPTER 1: Business Understanding**

# **1.1 Introduction**

In this chapter we will introduce the first step of the CRISP-DM which is the business understanding , first of all we will present our business objectives , after that we will invoke the used tools and finally the steps of the project.

# **1.2. Business objectives**

The commercial and Technical issues even before approaching the data or the work tools are important in this phase … We must therefore define what we want to accomplish and define the reasons that push us to achieve this goal.

Building a good accurate model able to predict Chronic kidney disease early to avoid developing kidney failure.

# **1.3** **Project planning**

Describe the intended plan for achieving the data mining goals and thereby achieving the business goals. Our plan should specify the steps to be performed during the rest of the project, including the initial selection of tools and techniques. A good project management will allow us to ensure that the objectives of the project are well aligned with the strategic objectives of the company.

-Importing Data : 1 day

-Exploring & understanding data : 2 days

-Preparing data: 1week

+As Putting together all the data:

+As Cleaning the data :

+[Visualize the data](https://www.simplilearn.com/data-visualization-article):

+Splitting the cleaned data into two sets - a training set and a testing set

-the Modeling phase :1 week

-Evaluate our models and fit our achievement into our client demands: 2days

# **1.4** **Used tools**

For our project , we had used jupyter notebook which is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience. Also we had installed a lot of libraries to facilitate our work such as :

* NumPy is a well known general-purpose array-processing package
* SciPy offers modules for linear algebra, image optimization, integration interpolation, special functions, Fast Fourier transform, signal and image processing
* Scikit-learn has a wide range of supervised and unsupervised learning algorithms that works on a consistent interface in Python
* Pandas are turning up to be the most popular Python library that is used for data analysis with support for fast, flexible, and expressive data structures designed to work on both “relational” or “labeled” data
* Matplotlib helps to generate histograms, plots, error charts, scatter plots, bar charts with just a few lines of code.

[](#D2L_fig_label_Tools)  
**Figure 2: Tools**

# 

# 

# 

# 

# **1.5** **Steps**

First of all, we will import the data that we will use in our project. Then, we have to explore and understand it. Next we will prepare our data by:

* Putting together all the data we have and randomizing it. This helps make sure that data is evenly distributed, and the ordering does not affect the learning process.
* Cleaning the data to remove what is unwanted, missing values, rows and columns, duplicate values, data type conversion, etc. we might even have to restructure the dataset and change the rows and columns or index of rows and columns.
* [Visualize the data](https://www.simplilearn.com/data-visualization-article) to understand how it is structured and understand the relationship between various variables and classes present.
* Splitting the cleaned data into two sets - a training set and a testing set. The training set is the set your model learns from. A testing set is used to check the accuracy of your model after training.

After that, the Modeling phase,we will choose a model which is relevant to the task at hand.

And finally, we have to evaluate our models already built, and fit our achievement into our client demands.

# **1.6** **Conclusion**

So in this chapter , we had exposed the objectives , invoked the steps of work ,and presented the tools that we had used.

For the next chapter, we will see the next step of CRISP which is the data understanding.

**CHAPTER 2: Data Understanding**

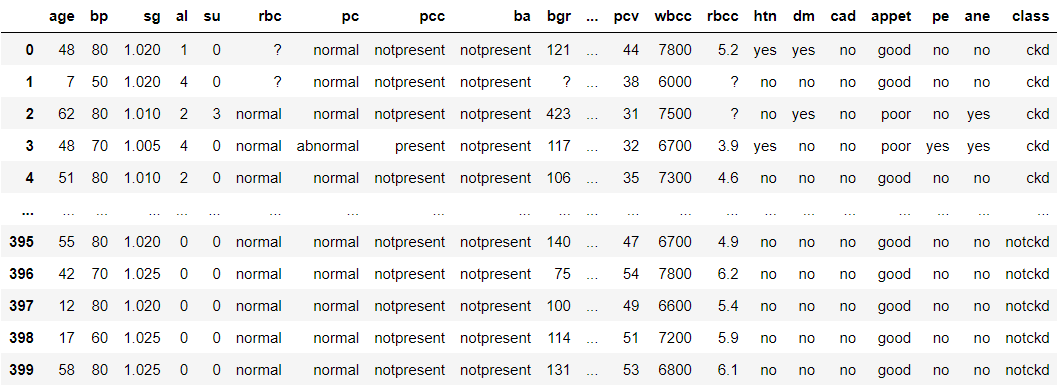
# **2.1 Introduction**

In this chapter we will expose the second step of the CRISP-DM process , So first of all we will explore the data , then we will visualize and understand it.

# **2.2** **Data source**

The dataset comprises 24 features divided into 11 numeric features and 13 categorical features,in addition to the class features, such as “ckd” and “notckd” for classification. Features include age, blood pressure,specific gravity, albumin, sugar, red blood cells, pus cell, pus cell clumps, bacteria, blood glucose random, blood urea,serum creatinine, sodium, potassium, hemoglobin, packed cell volume, white blood cell count, red blood cell count, hypertension, diabetes mellitus, coronary artery disease, appetite, pedal edema, and anemia.

* The diagnostic class contains two values: ckd and notckd.



**Figure 3: Description of data**

# 

# **2.3** **Features description**

Each feature, or column, represents a measurable piece of our data that can be used for analysis:

-age: 'age' numeric

- blood pressure :'bp' numeric

-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}

- red blood cells :'rbc' {normal,abnormal}

-pus cell :'pc' {normal,abnormal}

-pus cell clumps :'pcc' {present,notpresent}

-bacteria : 'ba' {present,notpresent}

-blood glucose random :'bgr' numeric

-blood urea :'bu' numeric

-serum creatinine: 'sc' numeric

-sodium:'sod' numeric

-potassium : 'pot' numeric

-hemoglobin: 'hemo' numeric

-packed :'pcv' numeric

-white blood cell count : 'wbcc' numeric

-red blood cell count :'rbcc' numeric

-hypertension :'htn' {yes,no}

-diabetes mellitus: 'dm' {yes,no}

-coronary artery disease: 'cad' {yes,no}

-appetite: 'appet' {good,poor}

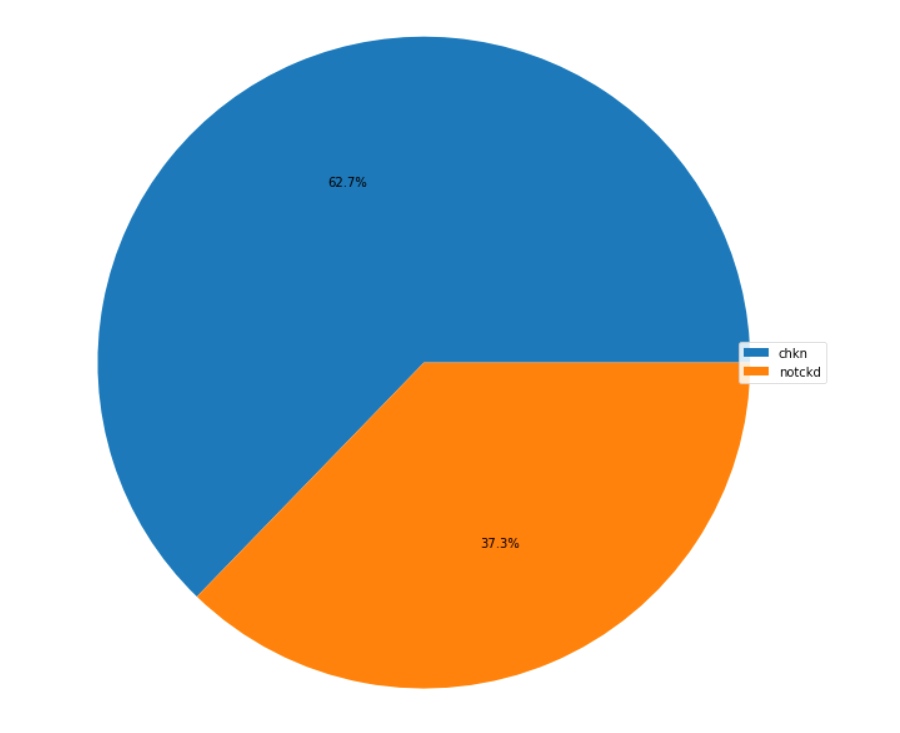
-pedal edema: 'pe' {yes,no}

-anemia : 'ane' {yes,no}

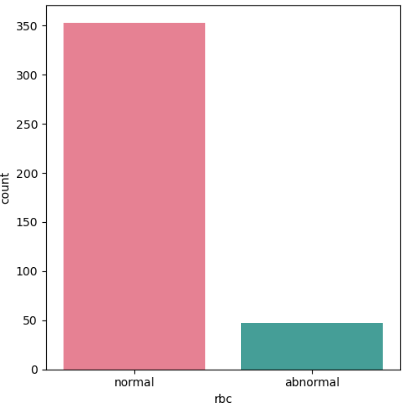
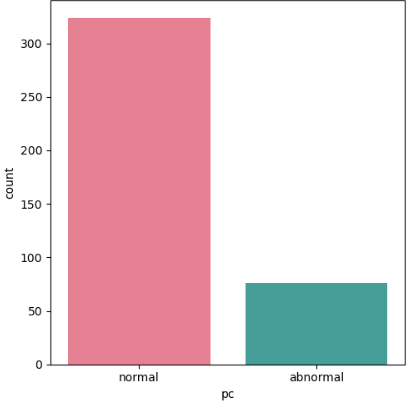
-classification : 'class' {ckd,notckd}

# **2.4** **Visualizing data for categorical features**

In this figure we can notice that 62.7% of people in our data had the chkn(checked) , and 37.3% do not have it (not checked).

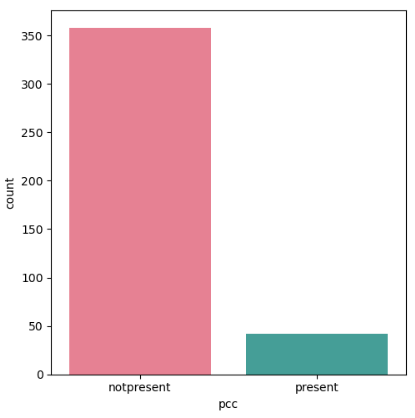
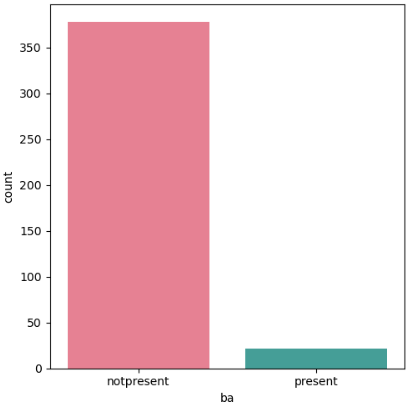
[](#D2L_fig_label_Percentage of Class Feature)  
**Figure 4: Percentage of Class Feature**

In the following barPlots we have displayed for each entity of the categorical variable a bar and the size of the bar represents its numeric values.

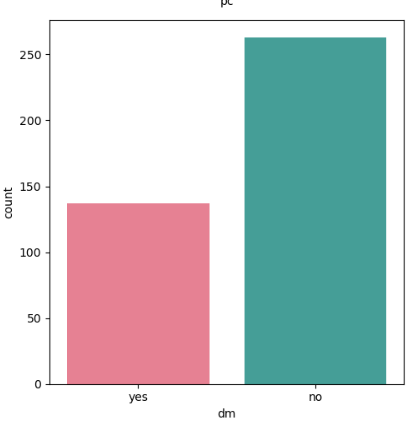
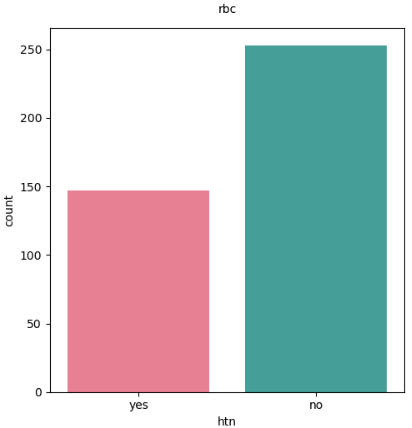
 

=>Here we can notice that 350 red blood cells =>Here we can notice that 350 pus cell are

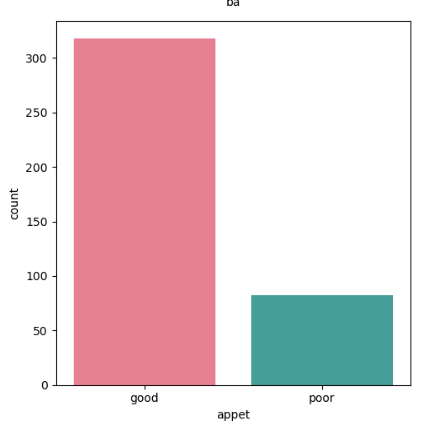
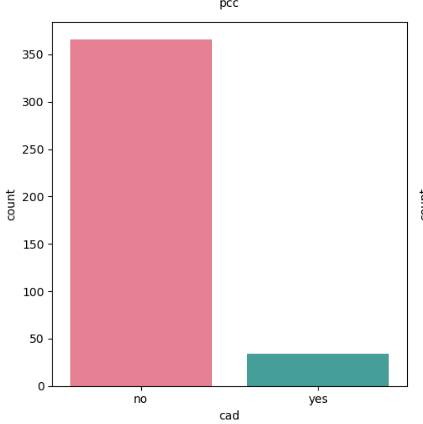
are normal while 50 are abnormal normal while 75 are abnormal

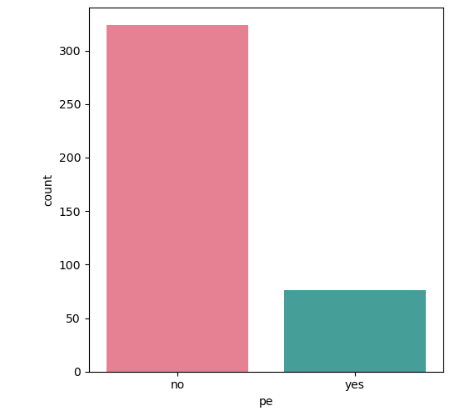
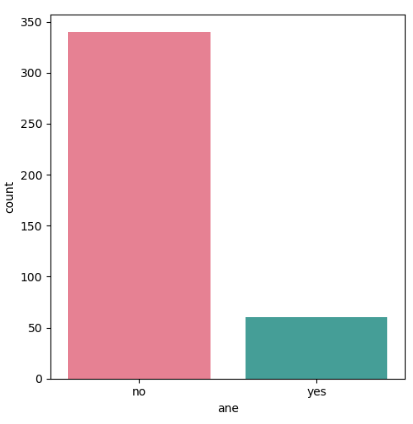
=>Here we can notice that 350 p are not present =>Here we can notice that 375 of bacteri while 50 are present are not present while 25 are present



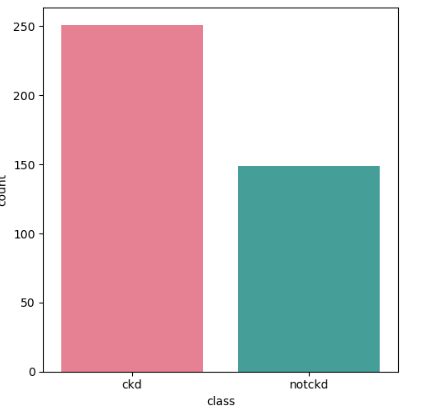
=>Here we can notice that 250 of htn are no =>Here we can notice that 275 of dm are yes while 140 while 150 are yes are yes



=>Here we can notice that 350 of cad are no =>Here we can notice that 330 of appet while while 40 are yes are good while 75 are poor

=>Here we can notice that 350 of pe are no =>Here we can notice that 340 of anemia are while 75 are yes no while 50 are yes



=>Here we can notice that 250 of class are checked

while 150 are not checked

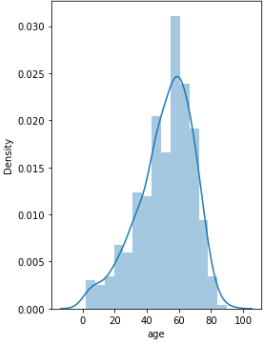
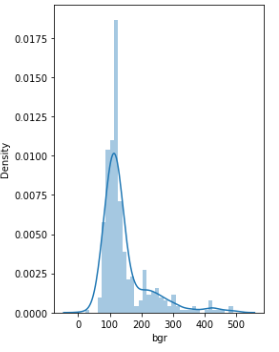
# 

# 

# 

# **2.5** **Visualizing data for numerical features**

These graphical representations illustrate the numerical feature as well as their mean value, their maximum value, minimum value and the percentage of the missing value compared to the valid value.

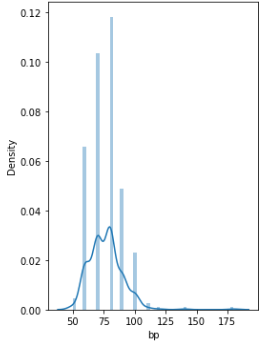
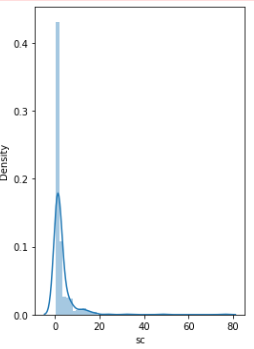
Valide 391 98% Valide 400 100%

Missing 9 2% Missing 0 0%

Mean 51.5 Mean 146

Max 90 Max 490

Min 2 Min 22

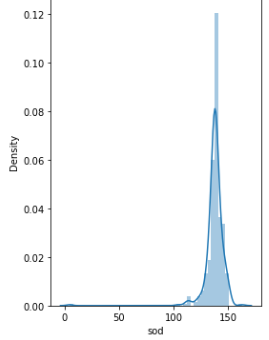
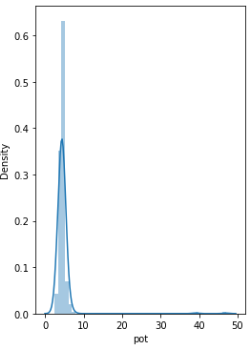
Valide 388 97% Valide 400 100%

Missing 12 3% Missing 0 0%

Mean 67.5 Mean 3.02

Max 180 Max 0.4

Min 50 Min 67

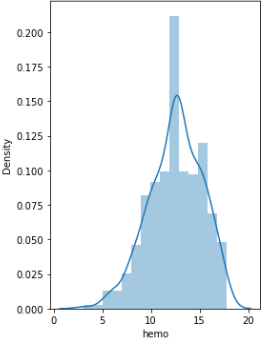
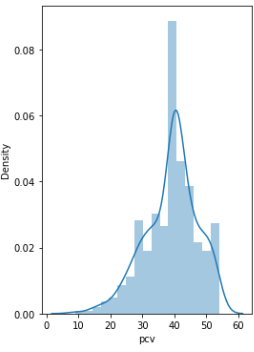
Valide 388 97% Valide 400 100%

Missing 12 3% Missing 0 0%

Mean 67.5 Mean 4.57

Max 180 Max 2.5

Min 50 Min 47

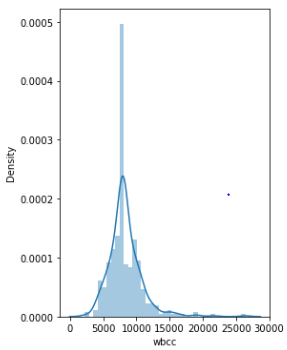
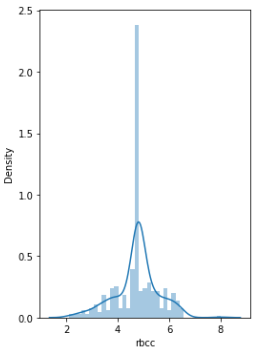
Valide 400 100% Valide 400 100%

Missing 0 0% Missing 0 0%

Mean 12.4 Mean 38.4

Max 3.1 Max 9

Min 17.8 Min 54

Valide 400 100% Valide 400 100%

Missing 0 0% Missing 0 0%

Mean 8.36k Mean 4.59

Max 2.2k Max 2.1

Min 26.4k Min 8

# **2.6 Conclusion:**

In this chapter, we reviewed the process where we ensured the data comprehension stage in which we treated the missing values, we dealt with the outliers and did an overview of the new data.Our data is now ready transformed during the data preparation phase which will be the subject of the next chapter.

**CHAPTER 3: Data Preparation**

# **3.1 Introduction**

Data preparation may be one of the most difficult steps in any machine learning project as it is one of the key players in developing high-quality machine learning models.

It allows us to explore, clean, combine, and format data for sampling and deploying ML models. It is essential as most ML algorithms need data to be in numbers to reduce statistical noise and errors in the data, etc.

In this topic, we will discover how the data will be transformed and we will learn more about the importance of data preparation in building the predictive modeling phase.

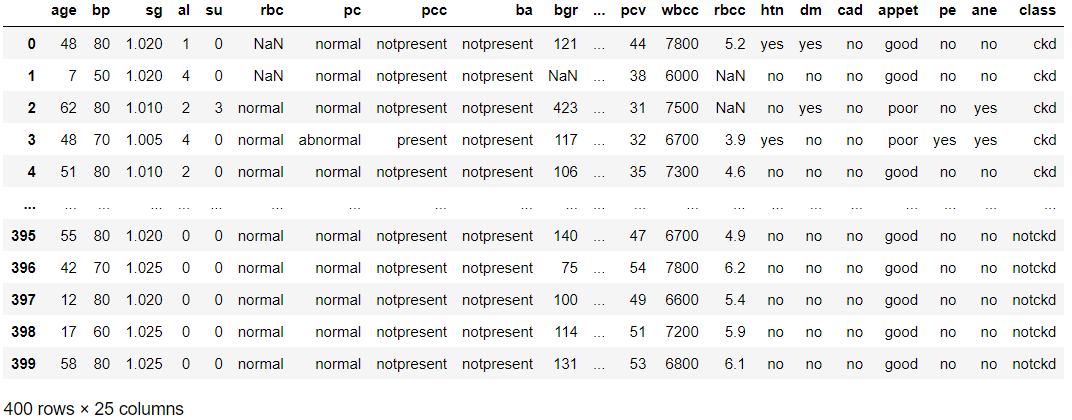
# **3.2 Data Cleaning**

Data cleaning and validation techniques help determine and solve inconsistencies, outliers, anomalies, incomplete data, etc.

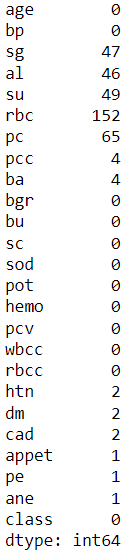
Clean data helps to find valuable patterns and information in data and ignores irrelevant data in the datasets. It is very much essential to build high-quality models, and missing or incomplete data is one of the best examples of poor data.

Since missing data always reduces prediction accuracy and performance of the model, data must be cleaned and validated through various imputation tools to fill incomplete fields with statistically relevant substitutes.

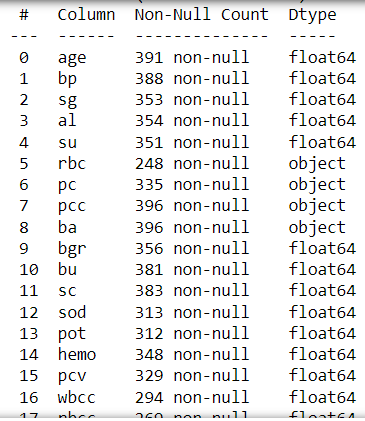
So in our case, we started by replacing all of the ‘?’ existing in our dataset with NaN so that we can discover the missing values in order to eliminate them.

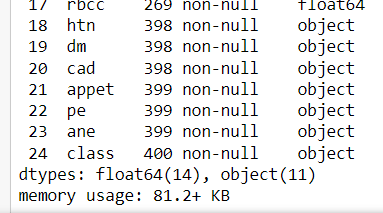
[](#D2L_fig_label_Result after replacing ‘?’ with ‘Nan’)  
**Figure 5: Result after replacing ‘?’ with ‘Nan’**

Rectifying this problem allowed us discover our missing values:

[](#D2L_fig_label_missing values)  
**Figure 6: missing values**

Then comes the phase of converting necessary columns to numerical type since they were all ‘objects’.



[](#D2L_fig_label_converting features to numerical type)  
**Figure 7: converting features to numerical type**

# **3.3 Feature engineering**

Feature engineering is defined as the study of selecting, manipulating, and transforming raw data into valuable features.

Feature engineering enables you to build an enhanced predictive model with accurate predictions.

There are various feature engineering techniques used in machine learning as follows:

* **Imputation:** Feature imputation is the technique to fill incomplete fields in the datasets. It is essential because most machine learning models don't work when there are missing data in the dataset. Although, the missing values problem can be reduced by using techniques such as single value imputation, multiple value imputation, K-Nearest neighbor, deleting the row, etc.
* **Encoding:** Feature encoding is defined as the method to convert string values into numeric form. This is important as all ML models require all values in numeric format. Feature encoding includes label encoding and One Hot Encoding (also known as get\_dummies).

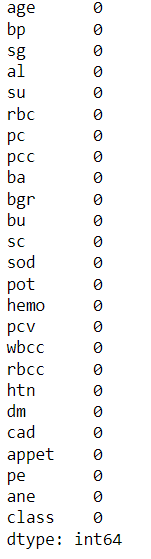
Similarly, feature engineering also includes handling outliers, log transform, scaling, normalization, Standardization, etc.

* **Step 1: Extracting categorical and numerical columns:**
* **Step 2 : imputing missing categorical values:**

A common method of imputation with numeric features is to replace missing values with the mean of the feature’s non-missing values. If the data have outliers, we may want to use the median instead. Either method is easy in Pandas:

With that being said, we applied median() method on these features 'age', ' bp', 'bgr', 'bu', 'sc' ,'sod' ,'pot' ,'hemo' ,'pcv' ,'wbcc' ,'rbcc' and mode() to the following ones ‘al’, ‘su’ ,’rbc’ , ‘pc’ , ‘ba’ ‘htn’ , ‘dm’ ,’cad ,’appet’ , ‘pe’ , ’ane’ , ‘pcc’, ‘sg’.

As we can see, our dataset has been cleaned from its missing values.

[](#D2L_fig_label_no missing values)  
**Figure 8: no missing values**

* **Step 3: replacing incorrect values:**

This phase consists in removing extra whitespace from the dataframe using replace() function.

* There is some ambiguity present in the columns , we have to fix them:

For that we’re going to be using the map method:

**\*\* Map** is a powerful function that not only gives us new ways to transform data, but also aids us in reducing our code to just a few efficient lines in some select cases.

* **Step 4: Encoding Categorical Data:**

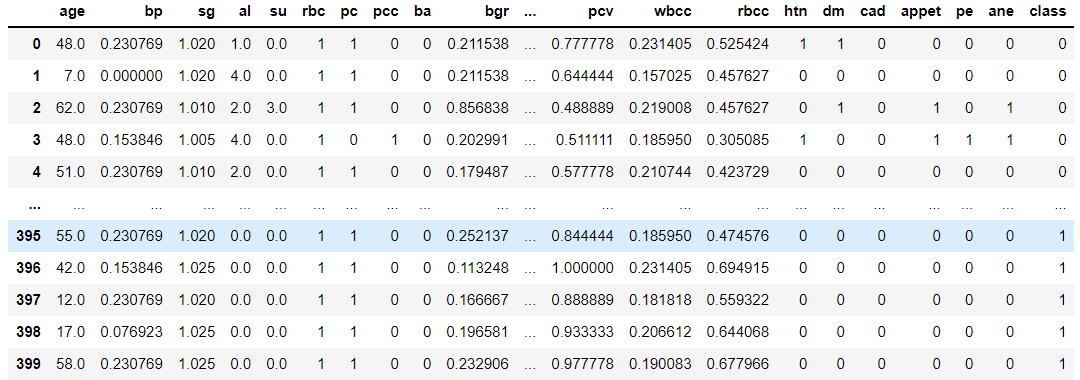
There are three used approaches for converting categorical variables to comparable values. They are:

· Ordinal Encoding

· One-Hot Encoding

· Dummy Variable Encoding

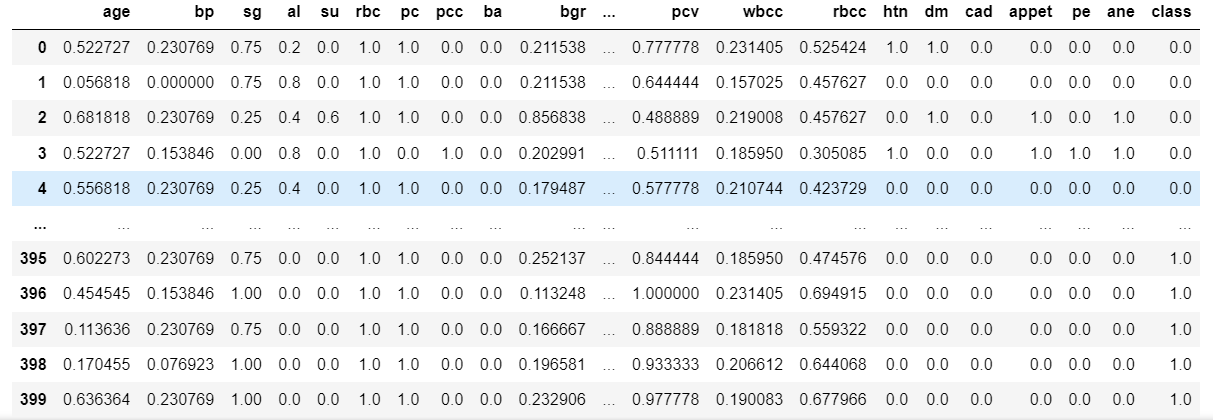
In our project we decide to use Ordinal Encoding, so for each categorical variable it will change to numeric , for example good will be ‘1’ and poor will be ‘0’

[](#D2L_fig_label_all the categorical value after the use of the ordinal encoding.)  
**Figure 9: all the categorical values after the use of the ordinal encoding.**

**Normalisation:**

The goal of normalization is to transform features to be on a similar scale. This improves the performance and training stability of the model.



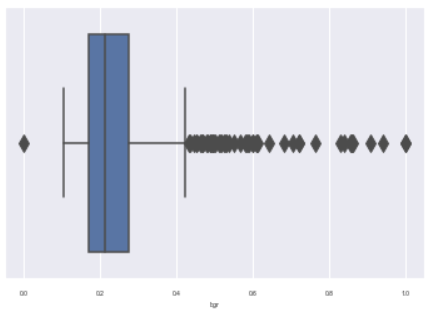
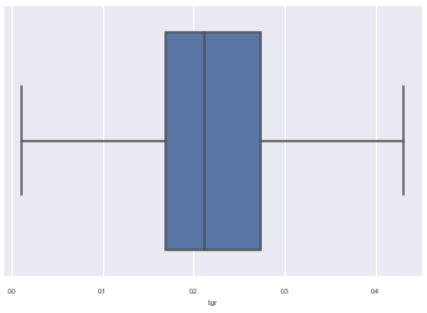
  
**Figure 10: Data normalized**

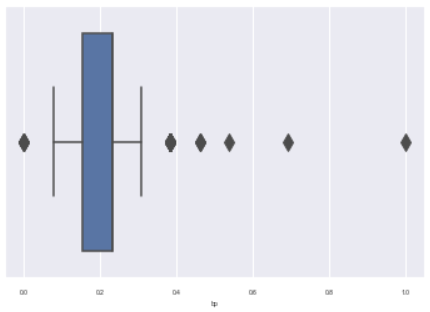
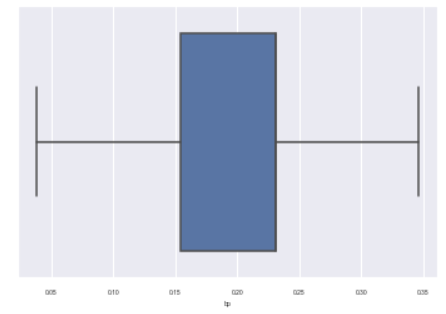
-> We can notice that the values of our dataframe are shifted and rescaled so that they end up ranging between 0 and 1.

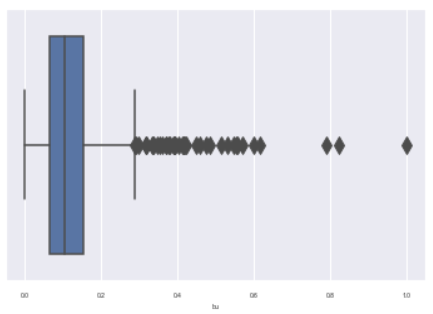
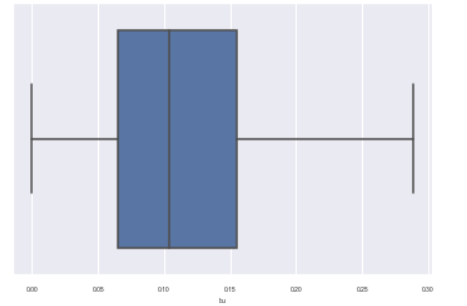
**Fixing and dealing with the outliers**

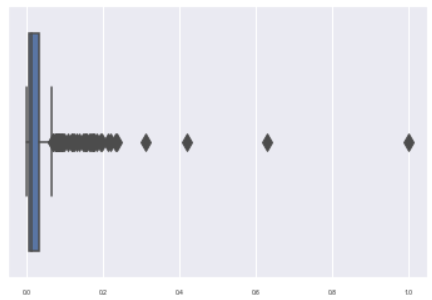
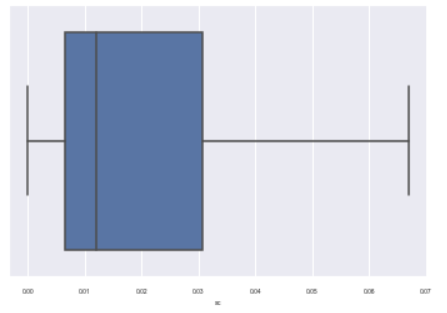
Those boxplots show our dataframe features with outliers on the left , and on the right we find the new features after fixing the outliers by using IQR method and the results before/after are below:

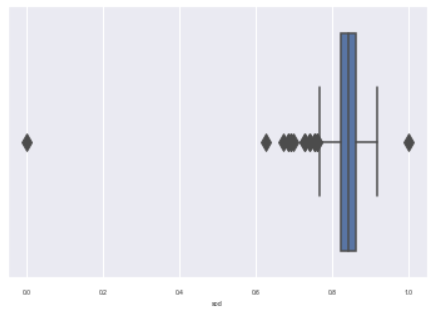
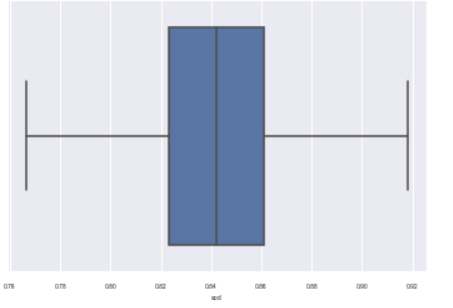
BEFORE AFTER

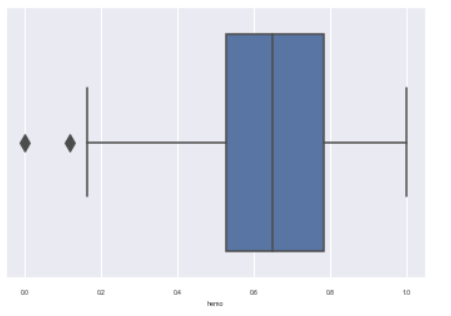
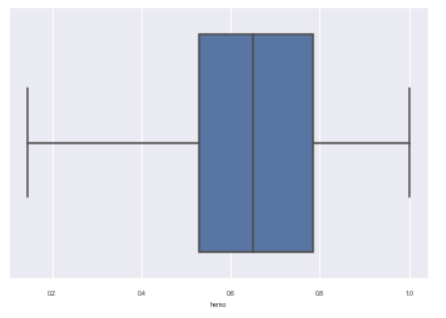
 

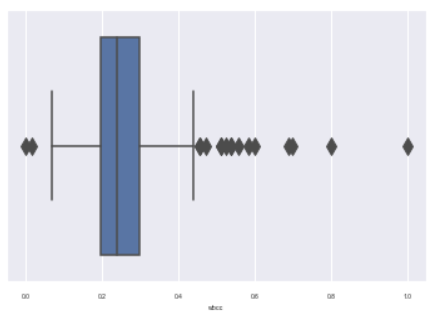
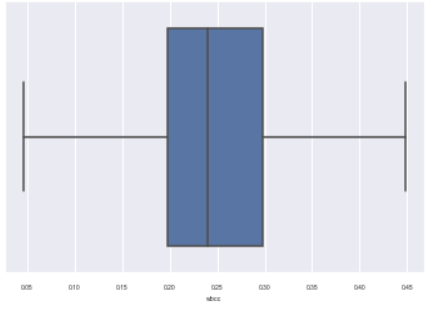
 

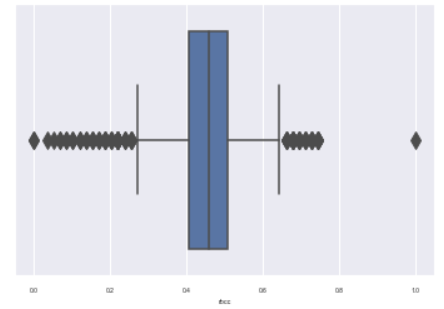
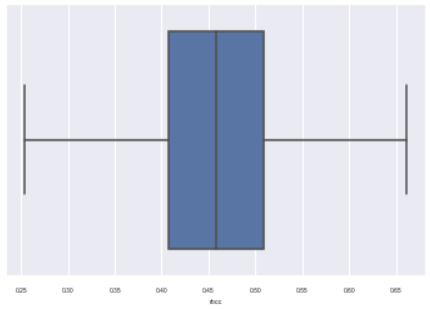
 

** **

** **

** **

** **

** **

First of all we detected the outliers of our data by using boxplot . After that we replaced all the outliers with the IQR method in every column and finally , we visualized the result by reusing the boxplot for every feature.

# **3.4 Feature selection:**

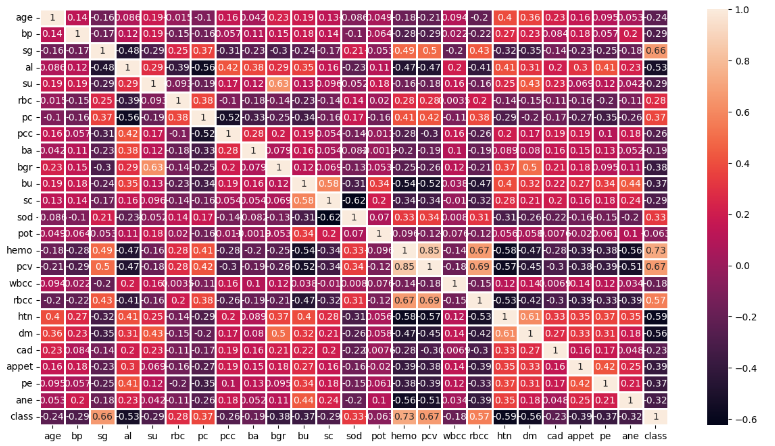
## **3.4.1 Correlation matrix:**

A correlation matrix is a common tool used to compare the coefficients of correlation between different features ( or attributes ) in a dataset.

It allows us to visualize how much ( or how little ) correlation exists between different variables.

This is an important step in pre-processing machine learning pipelines. Since the correlation matrix allows us to identify variables that have high degrees of correlation.

In this is context, we will be showing our correlation matrix:

[](#D2L_fig_label_correlation matrix)  
**Figure 11: correlation matrix**

From Heatmap , we can easily observe that PCV and Hemoglobin are highly correlated with 85%. So we can remove anyone from this column as it is acting like a duplicate of another.

so we choose to remove PCV from our data frame because it has a high correlation with Hemoglobin.

## **3.4.2 Extracting the important features of the dataset:**

Relation among numerical and classification column

We use a function that calculates the p\_value of all the features and the target, and returns a comment if there is a relationship between them or not.

We found that potassium has no correlation with Classification.

So we also removed potassium , so our data contains 20 features and a target.

# **3.5 Conclusion**

In this chapter, We have defined all the methods used for our data preparation , now our data is cleaned and prepared for the next step which is the modeling .

In the next chapter we will define all the algorithms we worked with .

**CHAPTER 4: Modeling**

# **4.1 Introduction:**

For the journey to AI, the most transformational technology of our time, the engine you need is a machine learning model.

Machine Learning models can be understood as a program that has been trained to find patterns within new data and make predictions.

These models are represented as a mathematical function that takes requests in the form of input data, makes predictions on input data, and then provides an output in response.

First, these models are trained over a set of data, and then they are provided an algorithm to reason over data, extract the pattern from feed data and learn from those data.

Once these models get trained, they can be used to predict the unseen dataset.

There are various types of machine learning models available based on different business goals and data sets.

In this phase we are going to use supervised learning algorithms for classification, and We did use a remarquable number of classification algorithms starting by:

AdaBoost, KNN, Random Forest Classifier , Naive Bayes, Decision Tree Classifier, Logistic Regression and SVM.

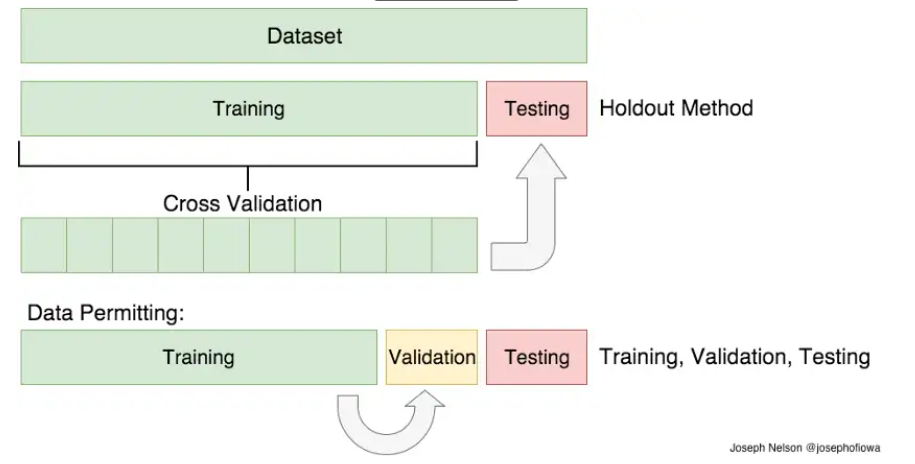
We start by searching the best parameters for each classifier using GridSearch and this is what we are going to do in the following pages

# **4.2 Used methods:**

In this step we will represent the used methods in the modeling:

**Gridsearch**: The Grid Search Method considers several hyperparameter combinations and chooses the one that returns a lower error score. This method is specially useful when there are only a few hyperparameters to optimize, although it is outperformed by other weighted-random search methods when the ML model grows in complexity.

**Cross Validation:** It’s very similar to train/test split, but it’s applied to more subsets. Meaning, we split our data into k subsets, and train on k-1 one of those subset. What we do is to hold the last subset for the test. We’re able to do it for each of the subsets.

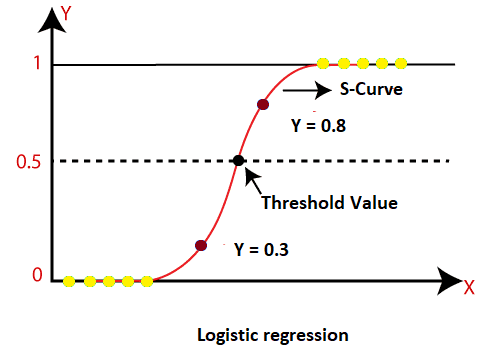


**Figure 12: Cross validation**

# **4.3 Logistic regression:**

## **4.3.1 Definition:**

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.



**Figure 13: Logistic regression**

## **4.3.2 Searching for the best parameters**

We start by searching for the best parameters using Grid search method and the parameters below are the best ones :

'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'

Then we evaluate the model using scoring,confusion matrix and base on the evaluation,

* Results before applying the cross validation:

Train Score=0.996875

Test Score=1.0

accuracy= 1.0

* Results after applying the cross validation:

accuracy= 98.75

# **4.4** **Adaboost :**

## **4.4.1 Definition:**

* **Boosting in Ensemble Methods**

Just as humans learn from their mistakes and try not to repeat them further in life, the Boosting algorithm tries to build a strong learner (predictive model) from the mistakes of several weaker models. You start by creating a model from the training data.

Then, you create a second model from the previous one by trying to reduce the errors from the previous model. Models are added sequentially, each correcting its predecessor, until the training data is predicted perfectly or the maximum number of models have been added.

Boosting basically tries to reduce the bias error which arises when models are not able to identify relevant trends in the data. This happens by evaluating the difference between the predicted value and the actual value.

* **Types of Boosting Algorithms**

1. AdaBoost (Adaptive Boosting)
2. Gradient Tree Boosting
3. XGBoost

In this article, we will be focusing on the details of AdaBoost, which is perhaps the most popular boosting method.

AdaBoost is one of the first boosting algorithms to be adapted in solving practices. Adaboost helps you combine multiple “weak classifiers” into a single “strong classifier”.

→ The weak learners in AdaBoost are decision trees with a single split, called decision stumps.

→ AdaBoost works by putting more weight on difficult to classify instances and less on those already handled well.

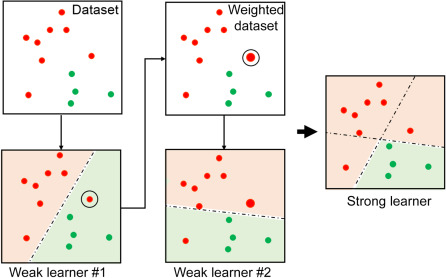
→ AdaBoost algorithms can be used for both classification and regression problems.

**Step 1:** A weak classifier (e.g. a decision stump) is made on top of the training data based on the weighted samples. Here, the weights of each sample indicate how important it is to be correctly classified. Initially, for the first stump, we give all the samples equal weights.

**Step 2**: We create a decision stump for each variable and see how well each stump classifies samples to their target classes. For example, in the diagram below we check for Age, Eating Junk Food, and Exercise. We'd look at how many samples are correctly or incorrectly classified as Fit or Unfit for each individual stump.

**Step 3:** More weight is assigned to the incorrectly classified samples so that they're classified correctly in the next decision stump. Weight is also assigned to each classifier based on the accuracy of the classifier, which means high accuracy = high weight!

**Step 4:** Reiterate from Step 2 until all the data points have been correctly classified, or the maximum iteration level has been reached.



**Figure 14: adaboost process**

## **4.4.2. Searching for the best parameters**

We start by searching for the best parameters with Grid search method and the parameters below are the best:

'learning\_rate': 0.1, 'n\_estimators': 50

Then we evaluate the model using a scoring, confusion matrix and based on the evaluation, we will try to improve the model using feature selection methods. After creating the model, we displayed the train and test score and we obtained:

* Before cross validation:

train score= 0.996875

test score= 1.0

accuracy =1.0

* after cross validation:

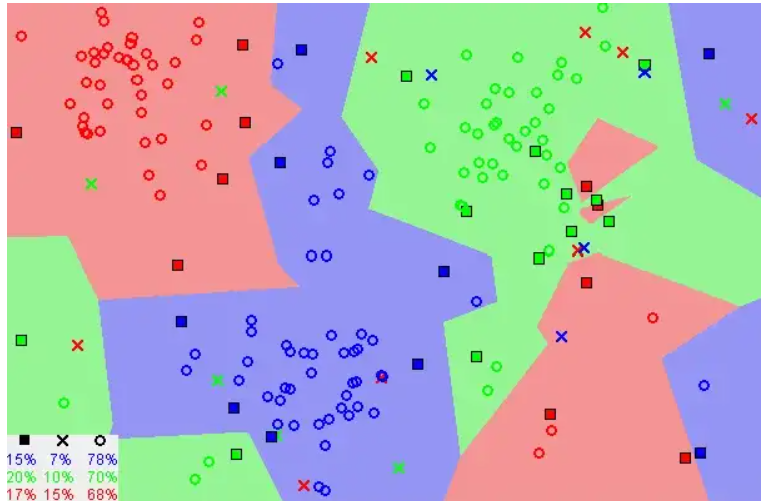
accuracy =0.9900

# **4.5** **K-nearest neighbors (KNN):**

## **4.5.1. Definition:**

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph.



**Figure 15: KNN process**

* **The KNN Algorithm:**

1. Load the data
2. Initialize K to your chosen number of neighbors
3. For each example in the data

3.1 Calculate the distance between the query example and the current example from the data.

3.2 Add the distance and the index of the example to an ordered collection

4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances

5. Pick the first K entries from the sorted collection

6. Get the labels of the selected K entries

7. If regression, return the mean of the K labels

8. If classification, return the mode of the K labels

## **4.5.2. Searching for the best parameters**

We start by searching for the best parameters with Grid search method and

the parameters below are the best:

'n\_neighbors': 2

* Before cross validation:

train score= 0.99375

test score=1.0

* After cross validation:

accuracy =0.985

# **4.6 Decision Tree Classifier:**

## **4.6.1 Definition**

The Decision Tree algorithm belongs to the family of supervised learning algorithms.

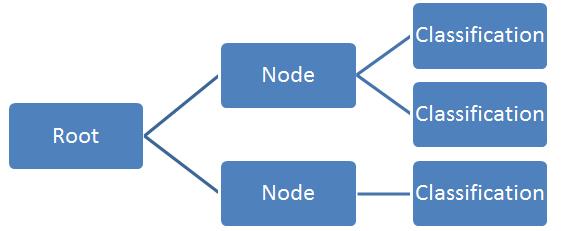
Unlike other supervised learning algorithms, the decision tree algorithm can

be used for solving regression and classification problems too.

The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data).

In Decision Trees, for predicting a class label for a record we start from the root of the tree.

We compare the values of the root attribute with the record’s attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.



**Figure 16: Decision tree classifier**

## **4.6.2 Searching for the best parameters**

We start by searching the best parameters for each classifier using Grid Search and

these are the best ones:

'criterion': 'entropy', 'max\_depth': 13

Then we evaluate the model using scoring,confusion matrix and base on the evaluation,

* Results before applying the cross validation:

Train Score= 1.0

Test Score=1.0

accuracy= 1.0

* Results after applying the cross validation:

accuracy= 0.9700

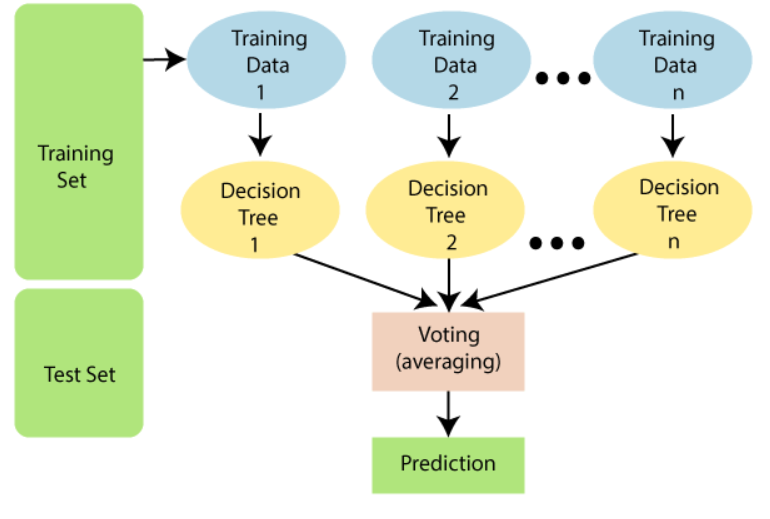
# **4.7 Random Forest:**

## **4.7.1 Definition:**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



**Figure 17: Random forest process**

The Working process can be explained in the below steps and diagram:

**Step-1**: Select random K data points from the training set.

**Step-2**: Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4**: Repeat Step 1 & 2.

**Step-5**: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

## **4.7.2 Searching for the best parameters**

We start by searching the best parameters for each classifier using Grid Search and

these are the best ones:

'criterion': 'gini', 'max\_depth': 7, 'max\_features': 'auto', 'n\_estimators': 70

Then we evaluate the model using scoring,confusion matrix and base on the evaluation,

* Results before applying the cross validation:

Train Score=0.984375

Test Score=1.0

accuracy= 1.0

Results after applying the cross validation:

accuracy= 97.25

# **4.8 SVM:**

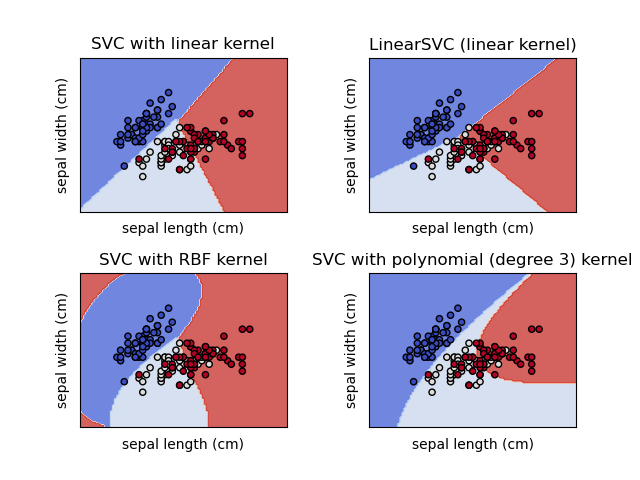
## **4.8.1 Definition**

A support vector machine (SVM) is a machine learning algorithm that analyzes data for classification and regression analysis.

SVM is a supervised learning method that looks at data and sorts it into one of two categories. An SVM outputs a map of the sorted data with the margins between the two as far apart as possible.

SVMs are used in text categorization, image classification, handwriting recognition and in the sciences.

A support vector machine is also known as a support vector network (SVN).

**

**Figure 18: SVM**

## 

## **4.8.2 Searching for the best parameters**

We start by searching the best parameters for each classifier using Grid Search and

these are the best ones:

{gamma=0.8, C=10, kernel='rbf}'

Then we evaluate the model using scoring,confusion matrix and base on the evaluation,

* Results before applying the cross validation:

Train Score= 0.98125

Test Score=1.0

Accuracy =1.0

* Results after applying the cross validation:

Accuracy = 97.00

# **4.9 Naive Bayes:**

## **4.9.1 Definition:**

A Naive Bayes classifier is a probabilistic machine learning model that’s used for classification tasks. The crux of the classifier is based on the Bayes theorem.

* **Multinomial Naive Bayes:**

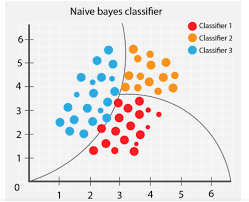
This is mostly used for document classification problems, i.e whether a document belongs to the category of sports, politics, technology etc. The features/predictors used by the classifier are the frequency of the words present in the document.

* **Bernoulli Naive Bayes:**

This is similar to the multinomial naive bayes but the predictors are boolean variables. The parameters that we use to predict the class variable take up only values yes or no, for example if a word occurs in the text or not.

* **Gaussian Naive Bayes:**

When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a gaussian distribution.



**Figure19: naive bayes process**

## **4.9.2 Searching for the best parameters**

We start by searching the best parameters for each classifier using Grid Search on the Bernoulli

Then we evaluate the model using scoring,confusion matrix and base on the evaluation,

* Results before applying the cross validation:

train score=0.934375

test score=0.9625

Accuracy=0.96

* Results after applying the cross validation:

**Accuracy**:

'bernoulli': 0.9875

# **4.10 Conclusion:**

In this chapter, we presented all the algorithms that we tested with our data, and then we had to choose the model with the best accuracy. In the next chapter we will evaluate each algorithm.

**Chapter 5 : Evaluation**

# **5.1 Introduction:**

While [training a model](https://medium.com/r?url=https%3A%2F%2Fheartbeat.fritz.ai%2Ftraining-an-image-classification-convolutional-neural-net-to-detect-plant-disease-using-fast-ai-a9f5315ed59) is a key step, how the model generalizes on unseen data is an equally important aspect that should be considered in every machine learning pipeline. We need to know whether it actually works and, consequently, if we can trust its predictions. Could the model be merely memorizing the data it is fed with, and therefore unable to make good predictions on future samples, or samples that it hasn’t seen before?

The above issues can be handled by evaluating the performance of a machine learning model, which is an integral component of any data science project. Model evaluation aims to estimate the generalization accuracy of a model on future data.

# **5.2 Définition:**

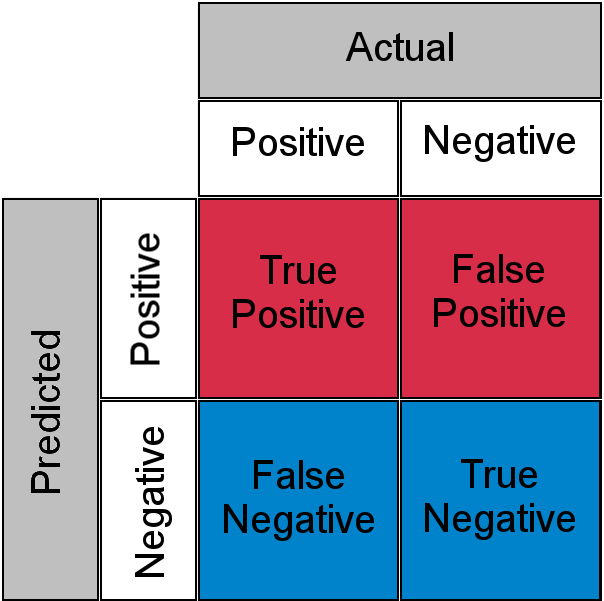
Here some definitions of some concepts that we have used:

**Confusion Matrix**

**-**Confusion matrix evaluates a classification model as it erases ambiguity about the performance of the model on each class. Let’s stay in the example of a binary classifier, which predicts 2 classes: class 0 and class 1.To measure the performance of this classifier, we need to distinguish 4 types of elements:

* **True Positive (TP)**: It refers to the number of predictions where the classifier correctly predicts the positive class as positive (elements of the class 1 correctly predicted).
* **True Negative (TN)**: It refers to the number of predictions where the classifier correctly predicts the negative class as negative (elements of class 0 correctly predicted).
* **False Positive (FP)**: It refers to the number of predictions where the classifier incorrectly predicts the negative class as positive (elements of class 1 wrongly predicted).
* **False Negative (FN)**: It refers to the number of predictions where the classifier incorrectly predicts the positive class as negative (elements of class 0 wrongly predicted).

**This information can be gathered and displayed in tabular form in a confusion matrix. In the case of a binary classifier, we obtain:**



**Figure 20: Classification report**

Classification report

The classification report visualizer displays the precision, recall, F1, and support scores for the model.

· The recall tells you what fraction of all positive samples were correctly predicted as positive by the classifier. It is also known as True Positive Rate (TPR), Sensitivity, Probability of Detection. To calculate Recall, use the following formula:

*TP/(TP+FN)*

· The precision It tells you what fraction of predictions as a positive class were actually positive. To calculate precision, use the following formula:

*TP/(TP+FP)*

· The support is the number of occurrences of the given class in your dataset

· The accuracy: It gives you the overall accuracy of the model, meaning the fraction of the total samples that were correctly classified by the classifier. To calculate accuracy, use the following formula:

*(TP+TN)/(TP+TN+FP+FN)*.

· The f1-score: It combines precision and recall into a single measure. Mathematically it’s the harmonic mean of precision and recall. It can be calculated as follows:

· Specificity: It tells you what fraction of all negative samples are correctly predicted as negative by the classifier. It is also known as True Negative Rate (TNR). To calculate specificity, use the following formula:

*TN/(TN+FP)*

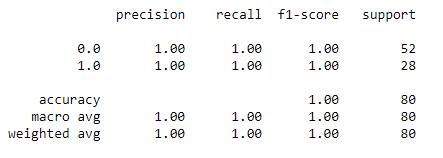
· Misclassification Rate: It tells you what fraction of predictions were incorrect. It is also known as Classification Error. You can calculate it using :

*(FP+FN)/(TP+TN+FP+FN)* or *(1-Accuracy)*.

# **5.3 Logistic regression**

## **5.3.1 Classification Report :**

1. Before Cross-Validation:

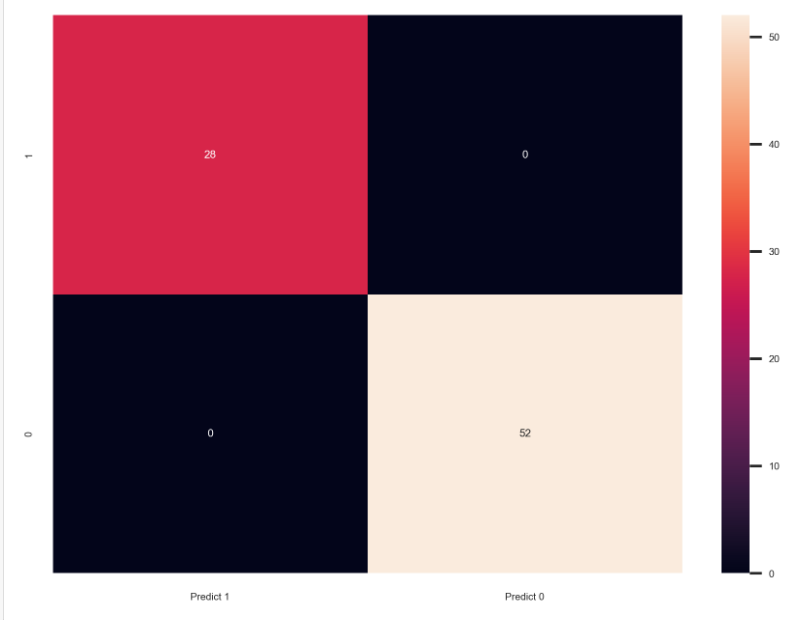


**Figure 21: LR classification report**

1. After Cross-Validation:

After applying the cross-validation method on our Logistic Regression model, we obtained an accuracy = 0.9875

## **5.3.2 Confusion matrix visualization**



**Figure 22: Confusion matrix LR**

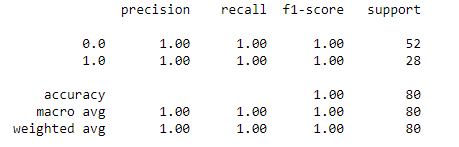
It can be seen in the confusion matrix, the true predicted label for non-checked patients is 52, and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the patients who said that they are checked and they are truly 28, and the rest are 0.

# **5.4 Decision Tree Classifier:**

## **5.4.1 Classification Report :**

1. Before Cross-Validation:

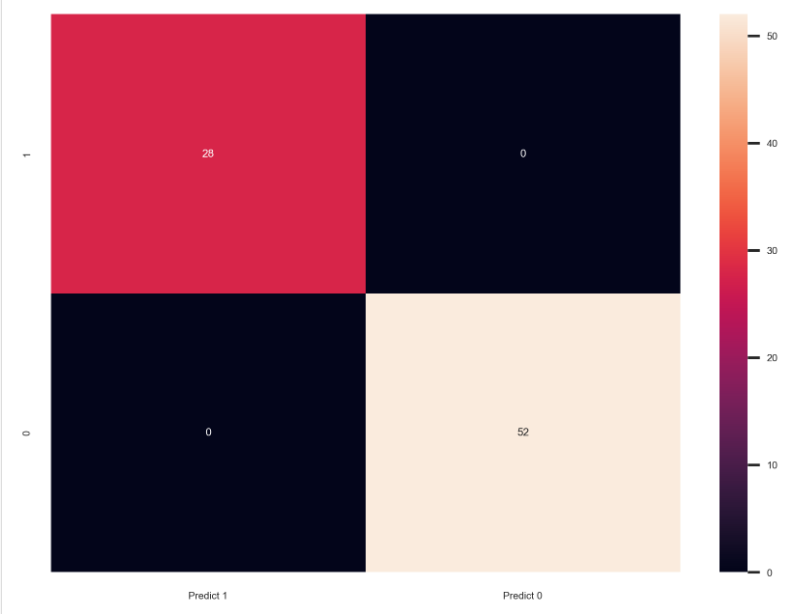


**Figure 23: Decision tree classifier classification report**

1. After Cross-Validation

After applying the cross-validation method on our Decision tree classifier model, we obtained an accuracy = 0.97

## **5.4.2 Confusion matrix visualization**



**Figure 24: Confusion matrix of Decision tree classifier**

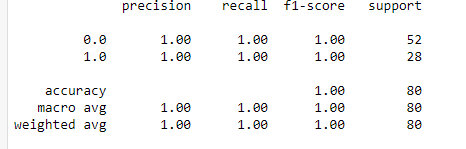
It can be seen in the confusion matrix, the true predicted label for non-checked patients is 52, and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the checked patients who said that they are checked and they are truly ,are 28, and the rest are 0.

# **5.5 ADaboost:**

## **5.5.1 Classification Report :**

1. Before Cross-Validation:

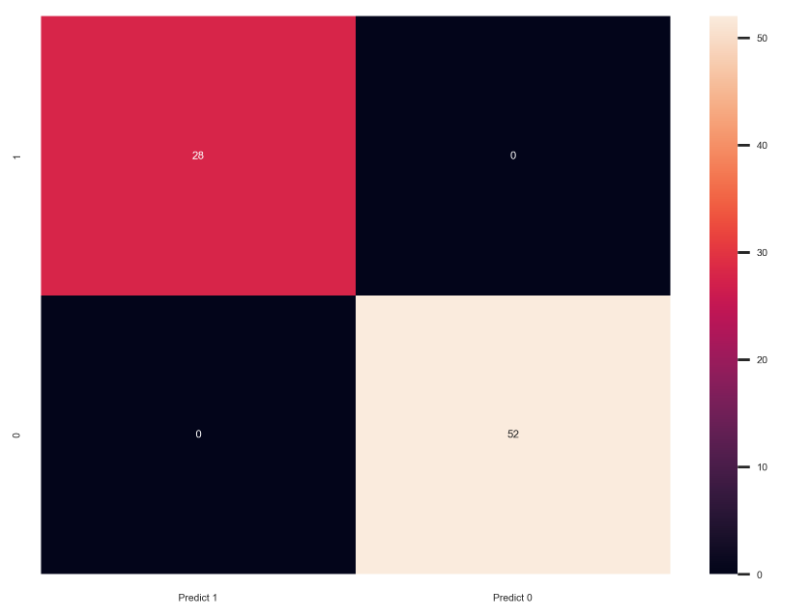


**Figure 25: adaboost classification report**

1. After Cross-Validation

After applying the cross-validation method on our adaboost model, we obtained an accuracy =0.99

## **5.5.2 Confusion matrix visualization:**



**Figure 26: adaboost classification report**

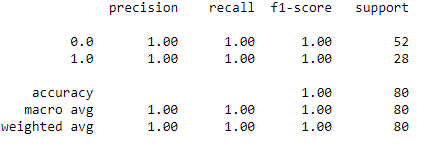
It can be seen in the confusion matrix, the true predicted label for non-checked patients is 52, and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the checked patients who said that they are checked and they are truly, are 28, and the rest are 0.

# **5.6 Random Forest Classifier**

## **5.6.1 Classification Report:**

1. Before Cross-Validation:

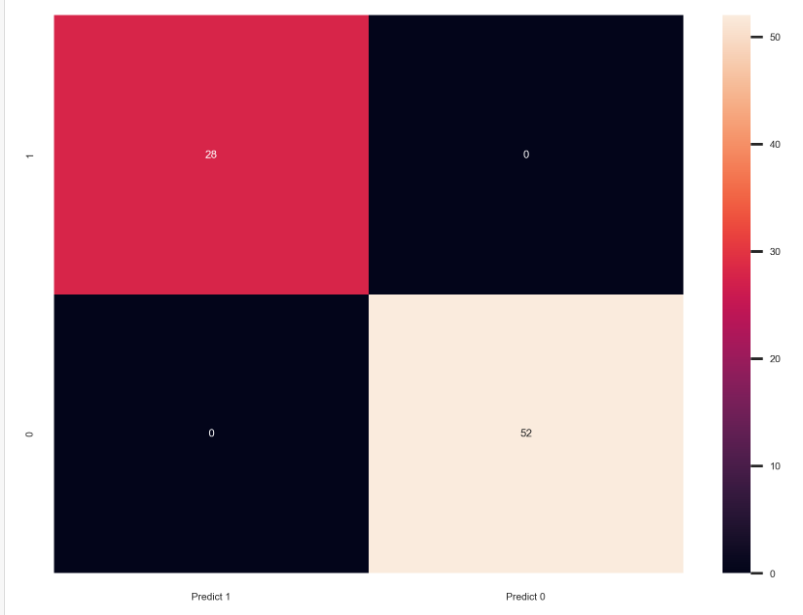


**Figure 27: Classification Report RFC**

1. After Cross-Validation:

After applying the cross-validation method on our Random Forest Classifier model, we obtained an accuracy = 0.9725

## **5.6.2 Confusion matrix visualization:**



**Figure 28: Confusion matrix**

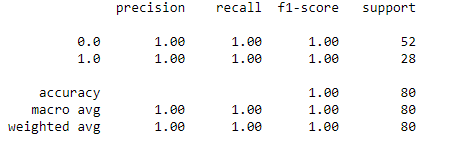
It can be seen in the confusion matrix, the true predicted label for non-checked patients is 52, and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the patients who said that they are checked and they are truly 28, and the rest are 0.

# **5.7 KNN:**

## **5.7.1 Classification Report :**

1. Before Cross-Validation:

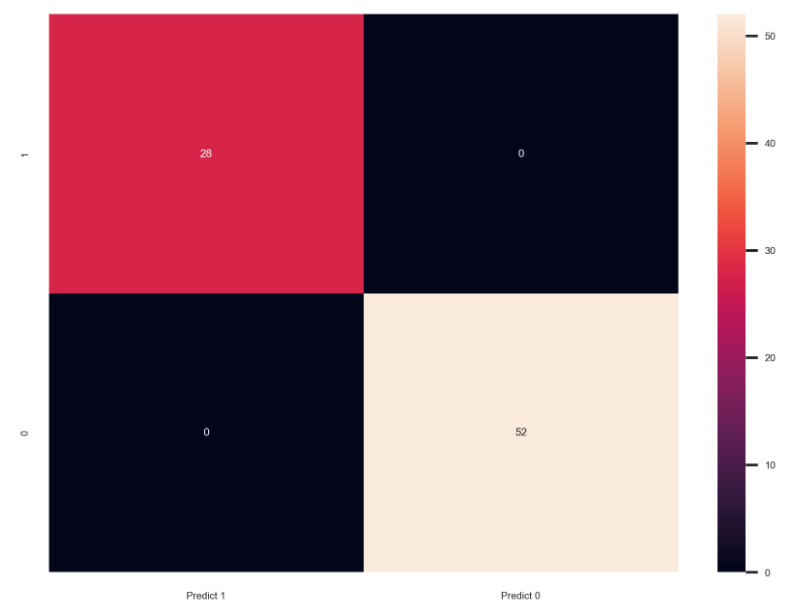


**Figure 29: KNN classification report**

1. After Cross-Validation:

After applying the cross-validation method on our KNN model, we obtained an accuracy = 0.985

## **5.7.2 Confusion matrix visualization:**



**Figure 30: Confusion matrix**

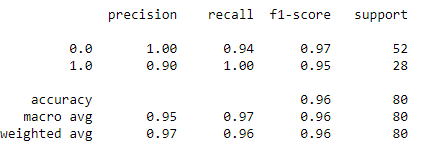
It can be seen in the confusion matrix, the true predicted label for non-checked patients is 52, and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the checked patients who said that they are checked and they are truly, are 28, and the rest are 0.

# **5.8 Naive Bayes:**

## **5.8.1 Classification Report :**

1. Before Cross-Validation:

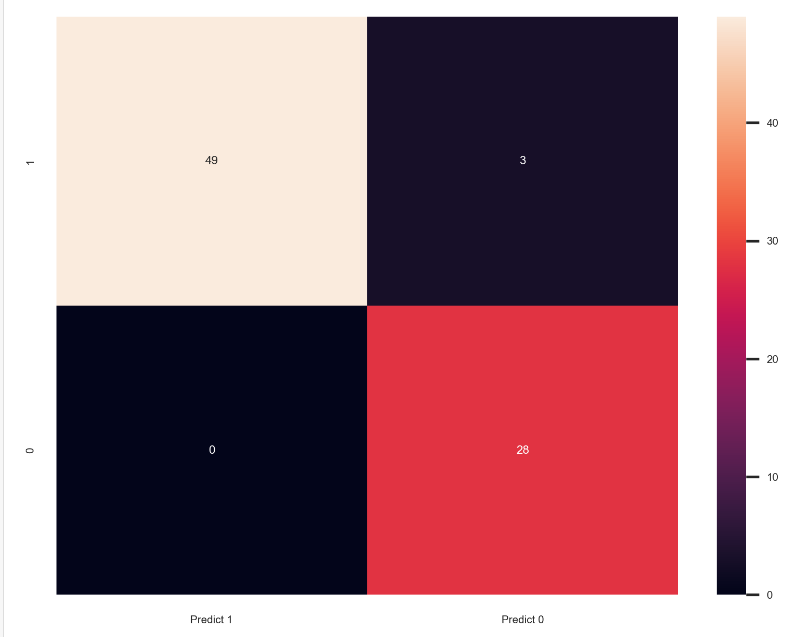


**Figure 31 :Naive Bayes classification report**

1. After Cross-Validation:

After applying the cross-validation method on our Naive Bayes model, we obtained an accuracy = 0.9875

## **5.8.2 Confusion matrix visualization :**



**Figure 32: Confusion matrix of Naive Bayes**

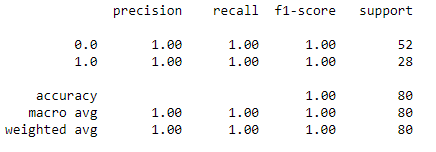
It can be seen in the confusion matrix, the true predicted label for non-checked patients is 28 and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the checked patients who said that they are checked and they are actually checked are 49, and the rest are 3.

# **5.9 SVM:**

## **5.9.1 Classification Report :**

1. Before Cross-Validation:

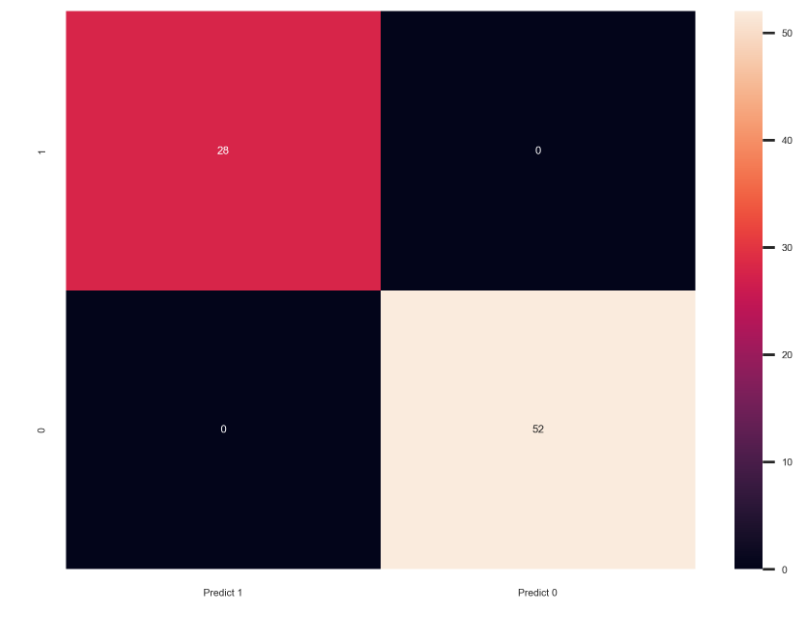


**Figure 33:SVM classification report**

1. After Cross-Validation:

After applying the cross-validation method on our SVM model, we obtained an accuracy = 0.9700

**5.9.2 Confusion matrix visualization:**



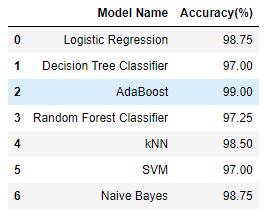
**Figure 34: Confusion matrix of SVM**

It can be seen in the confusion matrix, the true predicted label for non-checked patients is 52, and for those who are not checked but it shows that they could be checked is 0, so we can admit that it is acceptable.

Now for the checked patients who said that they are checked and they are actually checked are 28, and the rest are 0.

# **5.10 Comparing the performance of the models:**

The adaboost has the best accuracy , so we decided to work with it and select the best features.



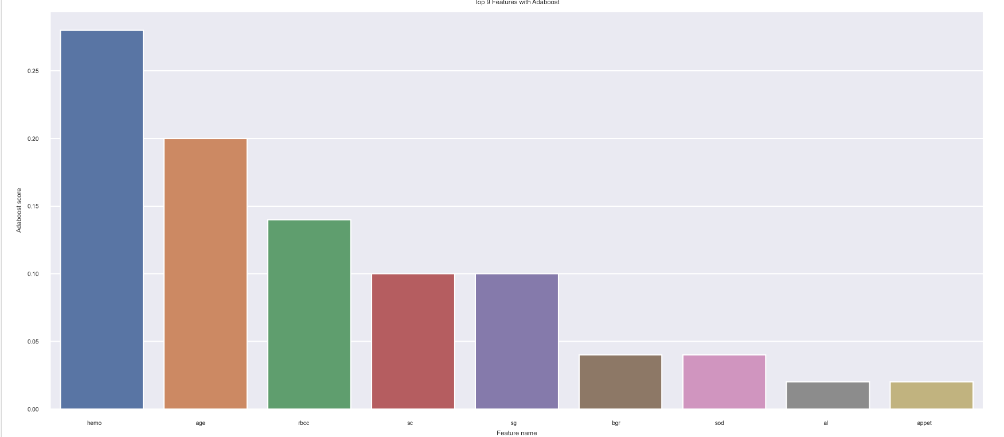
**Figure 35: Comparison of models**

# **5.11 Selecting the best k-features:**

After comparing the models we obtained, we finished by choosing adaboost as the best one.

Based on the adaboost model, we used the “ feature\_importances “ method to extract

the best k-features.



**Figure 36 : Top 9 Features with Adaboost**

As we can see, we got these 9 features:

| age | wbcc | sc | htn | hemo |
| --- | --- | --- | --- | --- |
| sod | al | sg | rbcc | - |

After that, we implemented a method that predicts whether the patient has a chronic kidney disease or not, to make sure that the extraction of the best features went well.

# **5.12 Conclusion:**

In the evaluation phase , we have found that adaboost has the best accuracy after using the cross validation.

So we chose adaboost which is a very popular boosting technique as it controls the depth of decision trees and does multiple predictions and finally predicts output.

As we all know that in boosting, weak classifiers are turned into strong classifiers and if we tell you about Adaboost, with the help of multiple decision trees, it really does so.

Finally we ended up choosing adaboost as the best model to apply the best k-features on.

This phase was treated carefully in the previous parts.

**General Conclusion**

Today, more and more organizations are opening up their doors to big data and unlocking its power\_increasing the value of a [data scientist](https://www.simplilearn.com/a-day-in-the-life-of-a-data-scientist-article) who knows how to tease actionable insights out of gigabytes of [data](https://www.simplilearn.com/what-is-data-article).

It’s become a universal truth that modern businesses are awash with data, this is due to their conscience the fact that whoever holds the information becomes the most powerful and the fact of knowing how to decipher data is an opportunity.

Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies and its [model](https://www.datarobot.com/wiki/model/) predictions allow businesses to make highly accurate guesses as to the likely outcomes of a question based on historical data, which can be about all kinds of things – customer churn likelihood, possible fraudulent activity, and more. These provide the business with [insights](https://www.datarobot.com/wiki/insights/) that result in tangible business value. For example, if a model predicts a customer is likely to churn, the business can target them with specific communications and outreach that will prevent the loss of that customer and in our project machine learning showed us the ability to predict that a particular customer has Chronic Kidney Disease or not.