**“The possible solutions to a given problem emerge as the leaves of a tree, each node representing a point of deliberation and decision.”** - Niklaus Wirth (1934 — ), Programming language designer

# Assignment 3

**Formulate an interesting classification problem and induce a decision tree to solve it.**

Nitesh Yadav(35), Rohit Rana(40), Rohit Shakya(41)
M.Sc. Computer Science
Department of Computer Science,
University of Delhi

## Overview

The target is to classify the given symptoms are for chronic kidney disease or not.

## About Data

- The data was taken over a 2-month period in India with 25 features ( eg, red blood cell count, white blood cell count, etc).

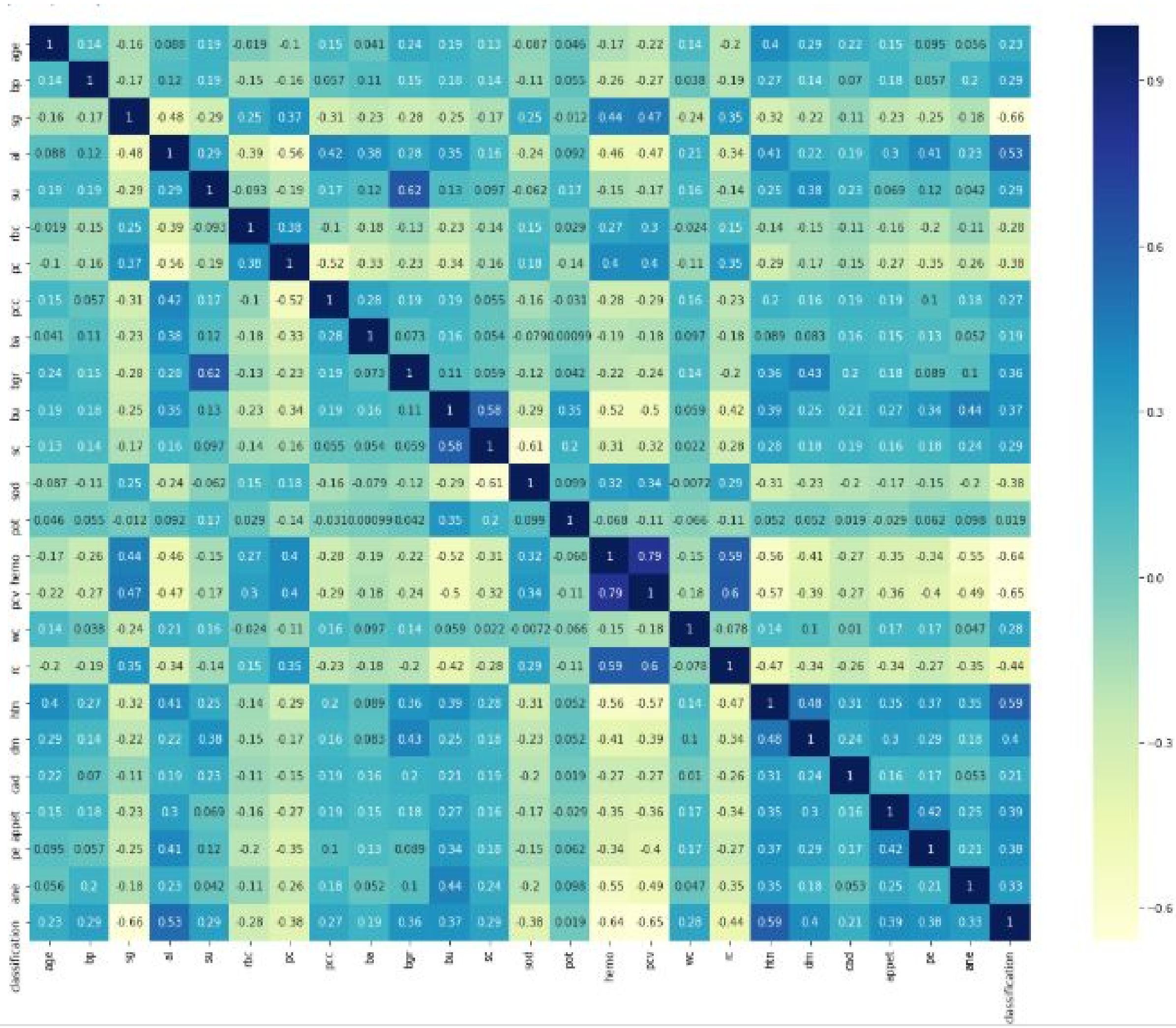
## Specifications

- **Number of Instances:** 400.
- **Attributes :**
    - age - age
    - bp - blood pressure
    - sg - specific gravity
    - al - albumin
    - su - sugar
    - rbc - red blood cells
    - pc - pus cell
    - pcc - pus cell clumps
    - ba - bacteria
    - bgr - blood glucose random
    - bu - blood urea
    - sc - serum creatinine
    - sod - sodium
    - pot - potassium
    - hemo - hemoglobin
    - pcv - packed cell volume
    - wc - white blood cell count
    - rc - red blood cell count
    - htn - hypertension
    - dm - diabetes mellitus
    - cad - coronary artery disease
    - appet - appetite
    - pe - pedal edema
    - ane - anemia
    - class - class

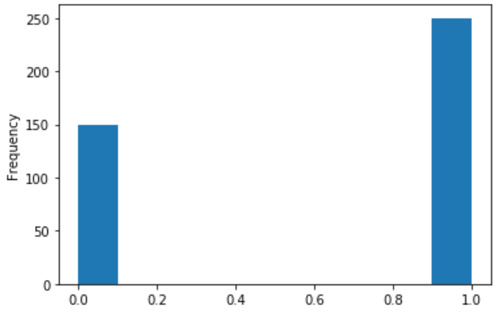
**Used Labelencoder to handle the categorical features.**

**Handle the missing values by replacing it by mode of the corresponding feature.**

## Heat Map



This heatmap shows that many features have a positive linear correlation with target features.



## Distribution of target features :

This histogram of the target shows that the dataset is imbalanced. 1- ckd and 0 - nonckd.

# Methods

## Decision Tree

**Definition:** A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

**Note:** Tree-based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree-based methods empower predictive models with high accuracy, stability and ease of interpretation. Unlike linear models, they map nonlinear relationships quite well. They are adaptable at solving any kind of problem at hand (classification or regression). Decision Tree algorithms are referred to as CART (Classification and Regression Trees).

### Common terms used with Decision Trees

**Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.

**Splitting:** It is a process of dividing a node into two or more sub-nodes.

**Decision Node:** When a sub-node splits into further sub-nodes, then it is called a decision node.

**Leaf or Terminal Node:** Nodes that do not split are called Leaf or Terminal nodes.

**Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.

**Branch or Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.

**Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

### We had used two methods to measure the informativeness of the features and use the feature with the most information as the feature to split the data on. Following methods are :

**Entropy**: It is used to measure the impurity or randomness of a dataset.

**Gini Index:** It is calculated by subtracting the sum of squared probabilities of each class from one. It favors larger partitions and is easy to implement whereas information gain favors smaller partitions with distinct values.

$$Gini = 1 - \sum_{i=1}^{n} p^2(c_i)$$

$$Entropy = \sum_{i=1}^{n} -p(c_i)log_2(p(c_i))$$

**where $p(c_i)$ is the probability/percentage of class $c_i$ in a node.**

## Confusion Matrix

**Definition:** A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

**True Positive:** You predicted positive and it's true.

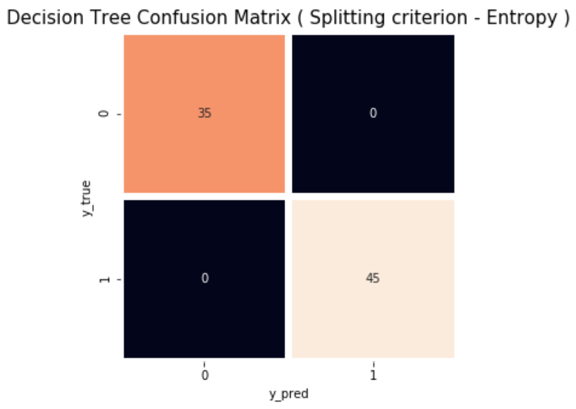
**True Negative:** You predicted negative and it's true.

**False Positive:** You predicted positive and it's false.

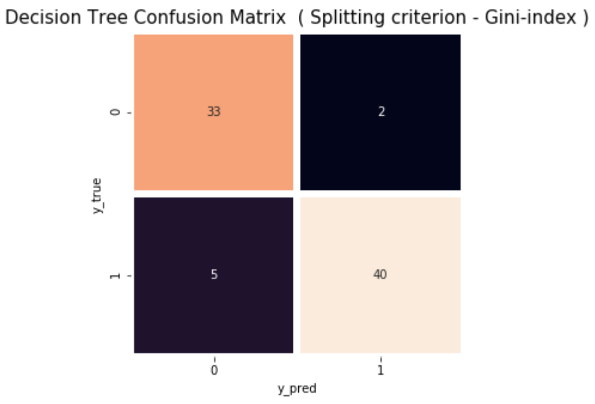
**False Negative:** You predicted negative and it's false.

Just Remember, We describe predicted values as Positive and Negative and actual values as True and False.

### Confusion matrix of our Decision tree mode :

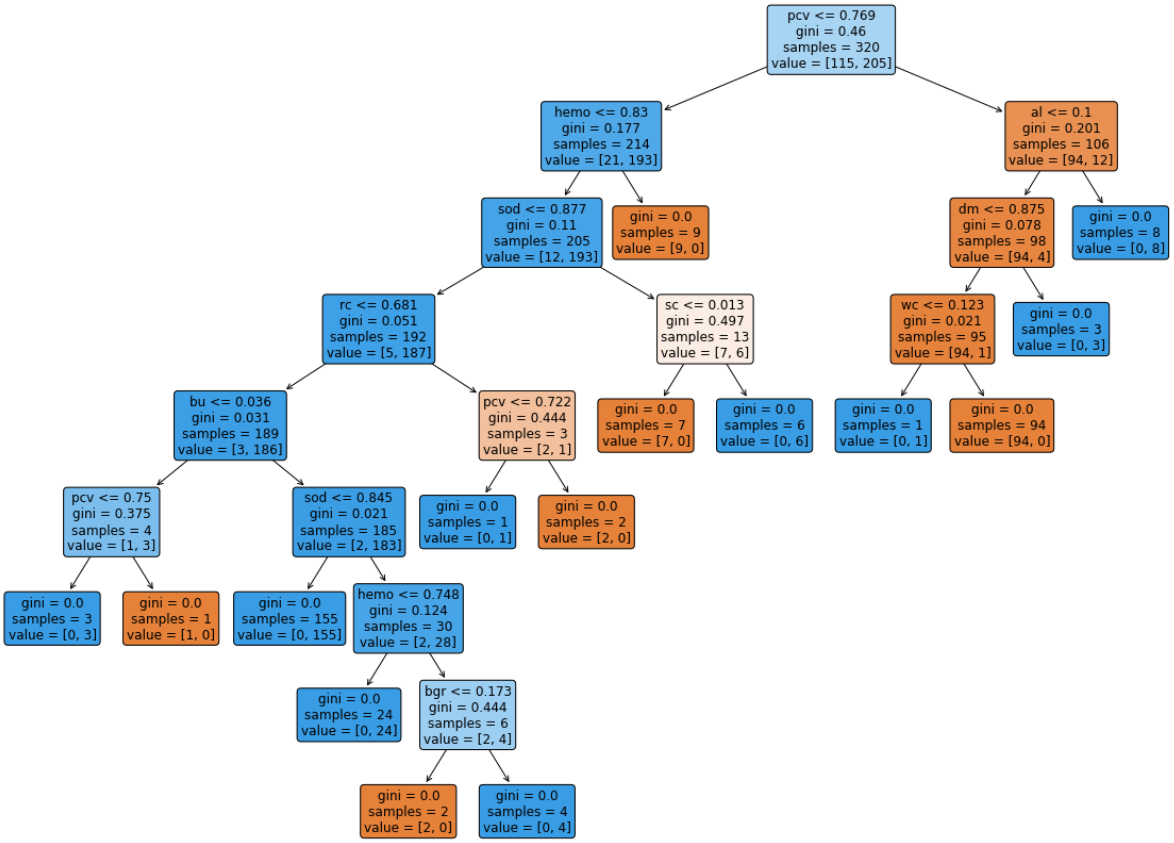


Decision Tree Confusion Matrix ( Splitting criterion - Entropy )

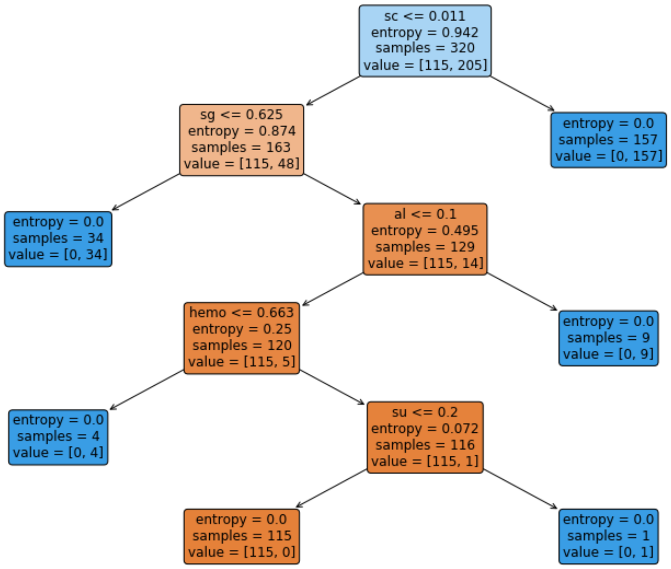


Decision Tree Confusion Matrix ( Splitting criterion - Gini-index )

## Decision tree ( Splitting criterion  - Gini index) :



## Decision tree ( Splitting criterion  - Entropy ) :



## Findings

- The Decision tree formed using gini-index is denser than entropy as splitting metrics.
- The accuracy of a decision tree with entropy is higher than the decision with gini-index.
- As we increase the max_depth of the tree accuracy is also increased.
- The decision is used to find the most important features used to predict the target feature.