# **Task 1 P, C: ML Development Using Python**

## **Introduction**

In this case study we will be using python knowledge with ML libraries to determine and analyze dataset. Dataset that will be used here is of “***Chronic Kidney Disease***”. 400 records have been provided in this dataset.

This dataset contains 25 columns, i.e.,

* **Age**: This column contains numerical data and units is in years with missing values. ColumnName: ‘age’.
* **Blood Pressure**: This column contains numerical data and unit is in mm/Hg with missing values. ColumnName: ‘bp’.
* **Specific Gravity**: This column contains categorical data with unique values as: (1.005,1.010,1.015,1.020,1.025) and some missing values. ColumnName: ‘sg’.
* **Albumin**: This column contains categorical data with unique values as: (0,1,2,3,4,5) and some missing values. ColumnName: ‘al’.
* **Sugar**: This column contains categorical data with unique values as: (0,1,2,3,4,5) and some missing values. ColumnName: ‘su’.
* **Red Blood Cells**: This column contains categorical data with unique values as: (normal, abnormal) and some missing values. ColumnName: ‘rbc’.
* **Pus Cell**: This column contains categorical data with unique values as: (normal, abnormal) and some missing values. ColumnName: ‘pc’.
* **Pus Cell clumps**: This column contains categorical data with unique values as: (present, notpresent) and some missing values. ColumnName: ‘pcc’.
* **Bacteria**: This column contains categorical data with unique values as: (present, notpresent) and some missing values. ColumnName: ‘ba’.
* **Blood Glucose Random**: This column contains numerical data and units is in mgs/dl with missing values. ColumnName: ‘bgr’.
* **Blood Urea**: This column contains numerical data and units is in mgs/dl with missing values. ColumnName: ‘bu’.
* **Serum Creatinine**: This column contains numerical data and units is in mgs/dl with missing values. ColumnName: ‘sc’.
* **Sodium**: This column contains numerical data and units is in mEq/L with missing values. ColumnName: ‘sod’.
* **Potassium**: This column contains numerical data and units is in mEq/L with missing values. ColumnName: ‘pot’.
* **Hemoglobin**: This column contains numerical data and units is in gms with missing values. ColumnName: ‘hemo’.
* **Packed Cell Volume**: This column contains numerical data with missing values. ColumnName: ‘pcv’.
* **White Blood Cell Count**: This column contains numerical data and units is in cells/cmm with missing values. ColumnName: ‘wbcc’.
* **Red Blood Cell Count**: This column contains numerical data and units is in millions/cmm with missing values. ColumnName: ‘rbcc’.
* **Hypertension**: This column contains categorical data with unique values as: (yes, no) and some missing values. ColumnName: ‘htn’.
* **Diabetes Mellitus**: This column contains categorical data with unique values as: (yes, no) and some missing values. ColumnName: ‘dm’.
* **Coronary Artery Disease**: This column contains categorical data with unique values as: (yes, no) and some missing values. ColumnName: ‘cad’.
* **Appetite**: This column contains categorical data with unique values as: (good, poor) and some missing values. ColumnName: ‘appet’.
* **Pedal Edema**: This column contains categorical data with unique values as: (yes, no) and some missing values. ColumnName: ‘pe’.
* **Anemia**: This column contains categorical data with unique values as: (yes, no) and some missing values. ColumnName: ‘ane’.
* **Class**: This column contains categorical data with unique values as: (ckd, notckd) and some missing values. ColumnName: ‘class’.

Brief information about dataset:

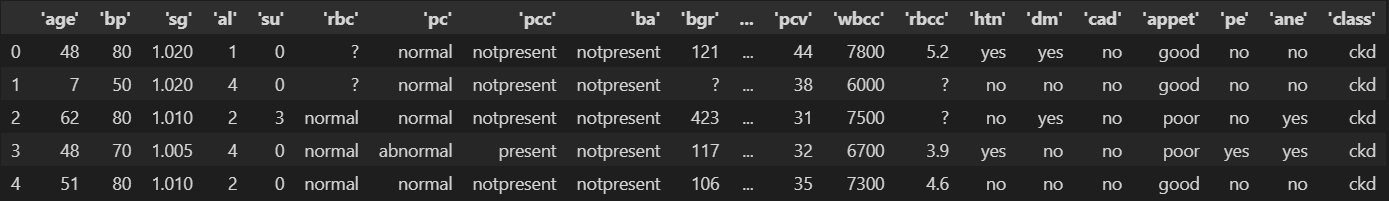
* Dataset contains missing values which has been denoted as ‘?’.
* Dataset have 24 features and 1 target class. These categories have been divided into as 11 numerical and 14 nominal.

## **Data Loading and Data Pre-processing**

* Extracting data into pandas Dataframe.

Command: 

Showing top 5 rows of dataframe:

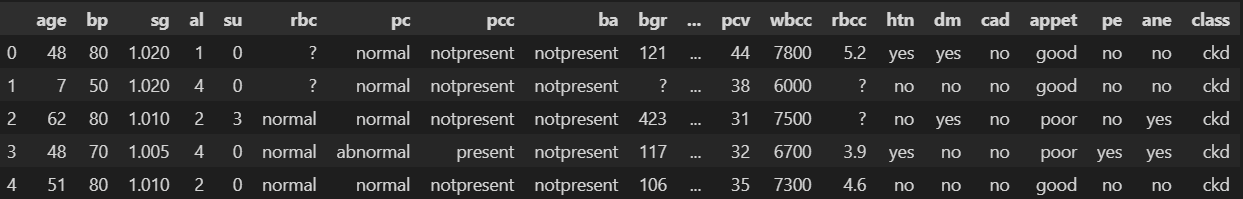


* Rectifying column names, as col names contains quotes, which needs to be removed.

Command:



Showing top 5 rows of dataframe:



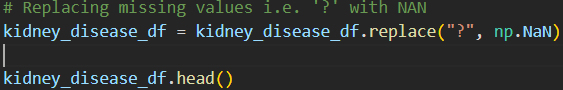
* Checking for duplicate rows, but none is present.

Command:

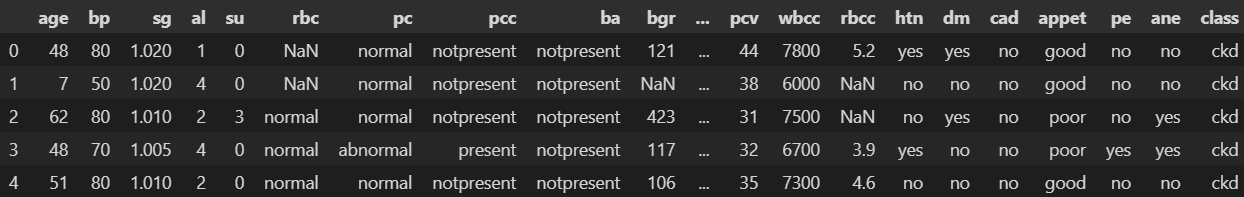


* Replacing missing values i.e. '?' with NAN.

Command:

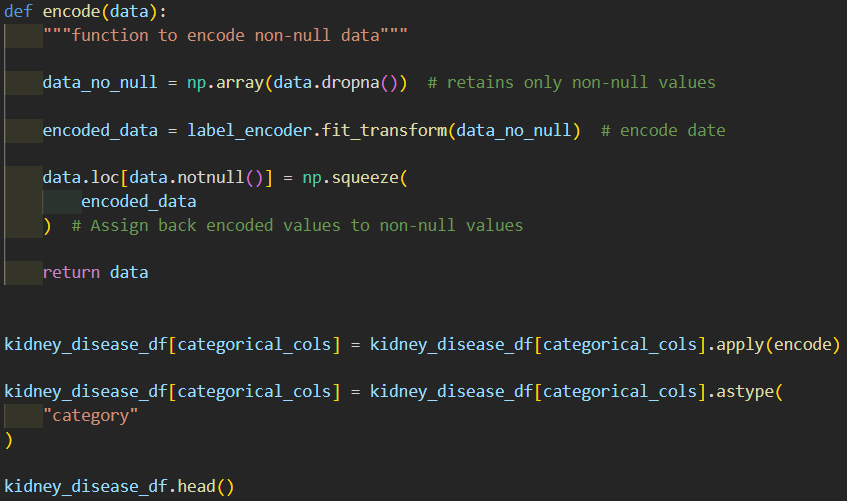


Showing top 5 rows of dataframe:

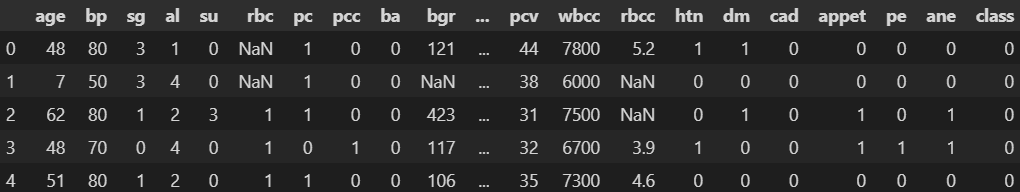


* As we have 14 categorical data that needs to be encoded, so for this we will be using ‘LabelEncoder’ technique from ‘sklearn’ package. This is a technique that converts categorical variables into numerical values.

Command:

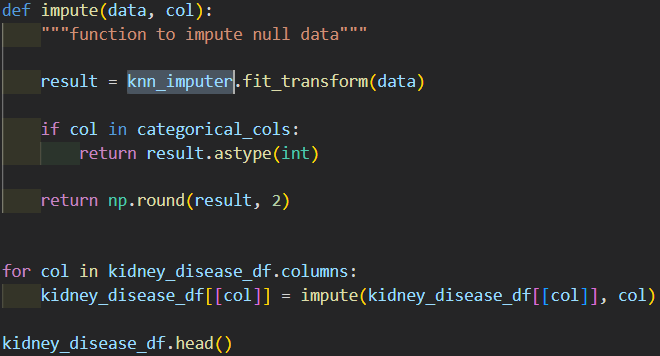


Showing top 5 rows of dataframe:



* As we have ample missing data in every column, so for that we will be using ‘KNNImputer’ technique from ‘sklearn’ package. This is a scikit-learn class that uses the K-Nearest Neighbors (KNN) algorithm to predict or fill in missing values in a dataset. It's a multivariate technique that considers multiple features in the dataset to estimate the missing values.

Command:

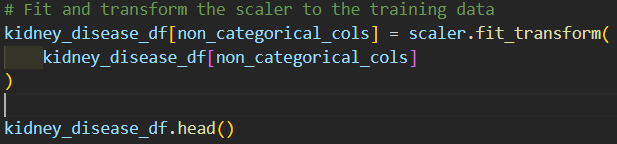


Showing top 5 rows of dataframe:

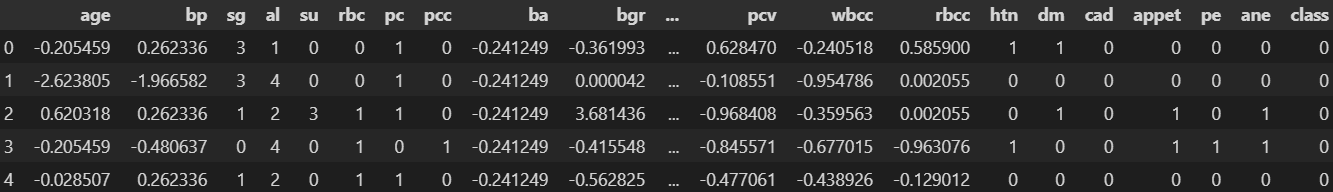


* As we have 11 non categorical data whose values ranges a lot, so to normalize we will be using ‘StandardScaler’ technique from ‘sklearn’ package. It normalizes features by removing the mean and scaling them to unit variance.

Command:



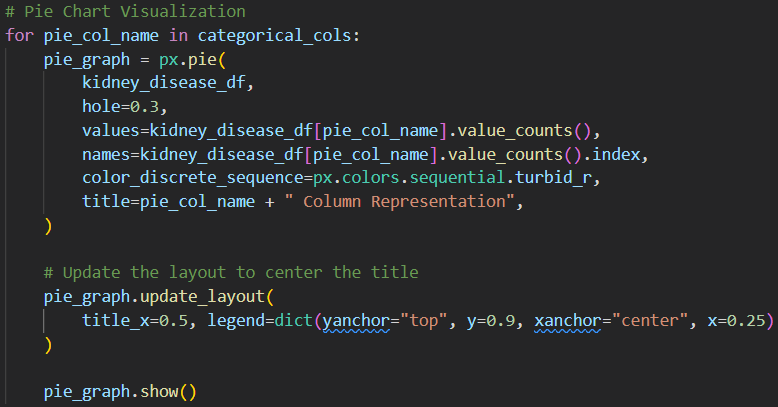
Showing top 5 rows of dataframe:



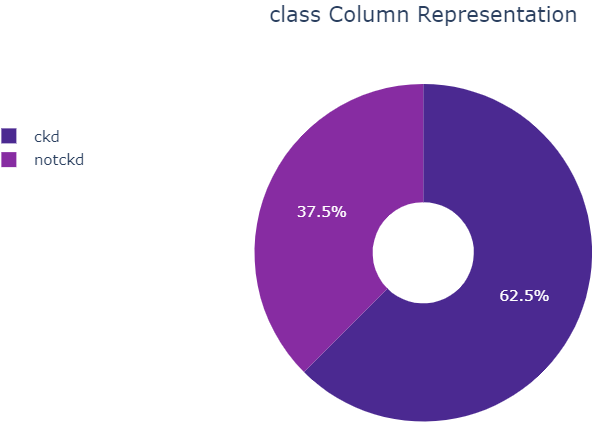
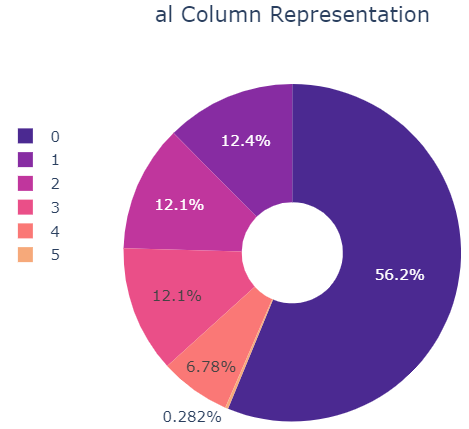
## **Data Visualization**

* Categorical Pie chart visualize

Command:

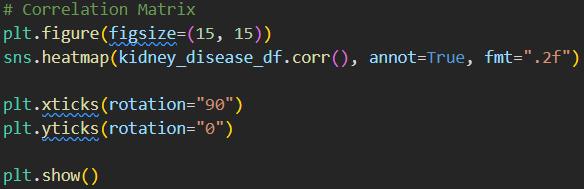


Visualizing sample pie charts:

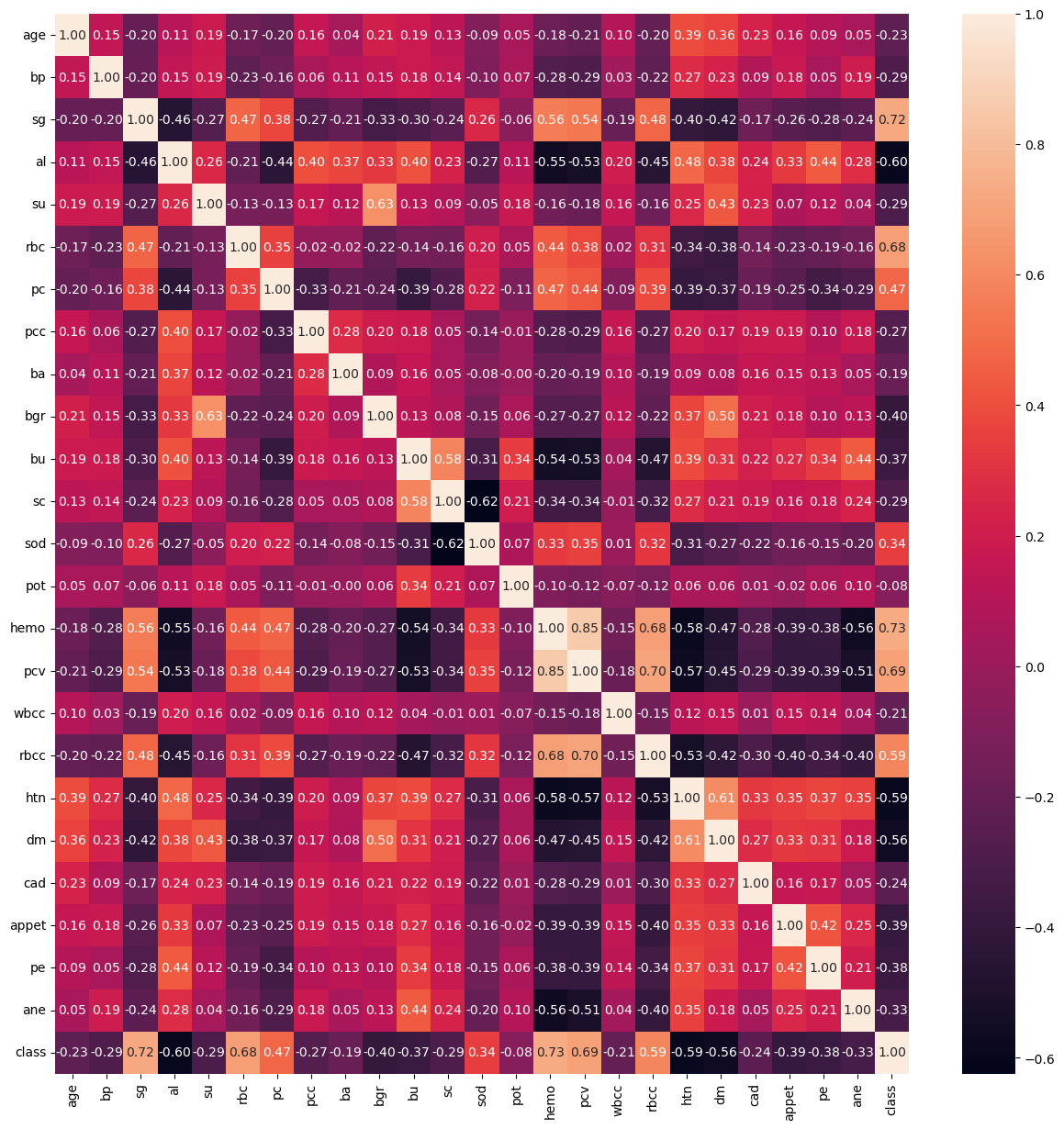
 

* Heatmap visualization

Command:



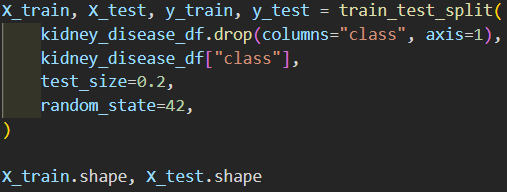
Visualizing heatmap:



## **Data Split**

Now we will be splitting our dataframe into 80% training dataset and 20% as testing dataset. To achieve this, we will be using `train\_test\_split’ from ‘sklearn’ package. This is a function in the scikit-learn library that splits a dataset into two sets: a training set and a test set. The training set is used to fit a machine learning model, while the test set is used to evaluate the model's performance.

Command:



## **Data Modelling**

In this we will be using 2 ML models based on Decision tree and Random Forest.

* DecisionTreeClassifier: It is a machine learning algorithm that predicts the probability of a categorical dependent variable. It's a set of if/else or Yes/No questions that can perform multi-class classification on a dataset.
* RandomForestClassifier: It is a supervised machine learning algorithm that uses decision trees to classify, regress, and perform other tasks. It's a type of ensemble-based learning method that uses a random vector sampled from the input vector to generate each decision tree.

To achieve the best model, we have used ‘GridSearchCV’ technique from ‘sklearn’ package. It is a hyperparameter tuning technique used in machine learning to find the best combination of hyperparameters for a given model. Hyperparameters are variables that are not learned by the model, but rather set by the user before training.

Defining model and param details JSON for our ‘GridSearchCV’.

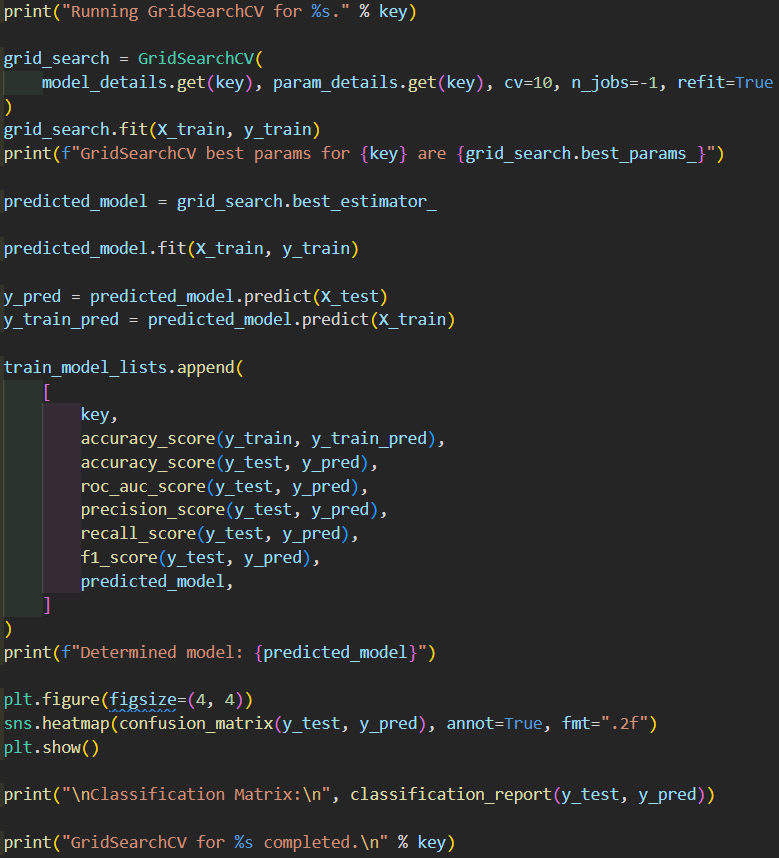
Command:



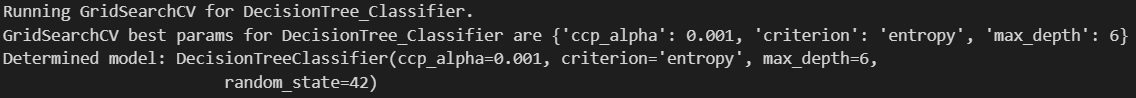
Here we will be defining our ` GridSearchCV’, which will be providing best model that can be used. Once best estimator is extracted, will be displaying performance of our model.

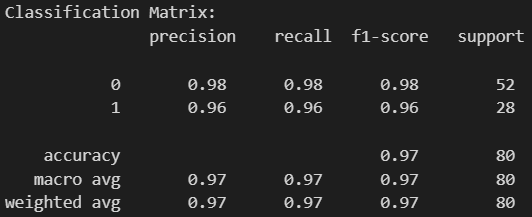
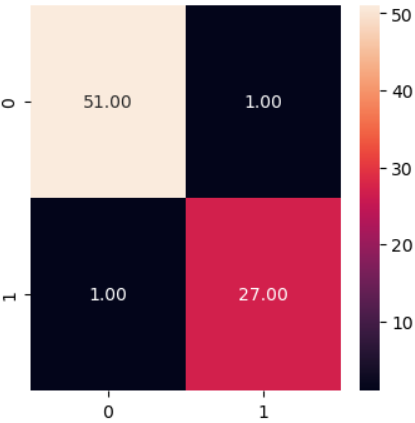
We have used different performance measures like, accuracy, ROC, precision, recall and f1 score.

Command:

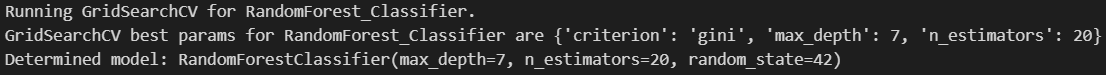


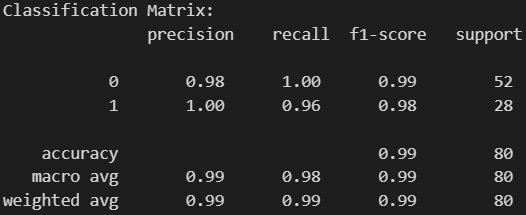
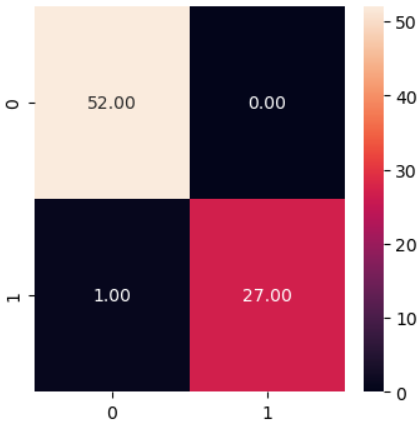
Printing output for Decision tree model:



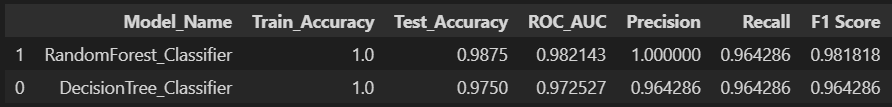
 

Printing output for Random search tree model:



Sorted models with performance ("Recall", "F1 Score").

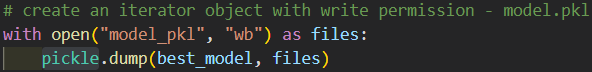


As per the above table, we can conclude that RandomForest\_Classifier has provided best performance and can be used for our further predictions or analysis.

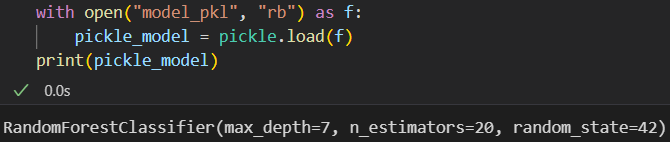
## **Model Save & Load**

Pickling is the process of converting a Python object hierarchy into a byte stream, and "unpickling" is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

* Saving model



* Loading model



## **Limitation**

* In this ‘Chronic Kidney Disease’ dataset have very less records, because of which our model won’t be much helpful.
* We can use SMOTE technique also, to increase the dataset records.
* Dataset have multi null values, for which imputation has been done, but that’s not the real/original values.

## **Conclusion**

In this case study we have compared 2 ML models, from which ‘RandomForestClassifier’ has provided best performance.

For our modelling we can use different ML models also, that can provide much better prediction performance, like Support Vector Machine, K-Nearest Neighbours, Stochastic Gradient Descent, and more.

## **Grade**

Here am targeting for ‘Pass’ and ‘Credit’ grade.

## **References**

* <https://olympus.mygreatlearning.com/dashboard>
* <https://scikit-learn.org/stable/supervised_learning.html#supervised-learning>
* <https://www.geeksforgeeks.org/>
* <https://www.google.com/>