Chronic Kidney Disease

- Shivam Kolhe

Problem Statement

- Identify the factors causing chronic kidney disease.
- Build a model that can help to determine if a patient is suffering from kidney chronic disease or not.

- **NOTE:** THERE CAN BE SOME MISMATCH IN THE READINGS FROM PPT AND THE JUPYTER NOTEBOOK BECAUSE EVERYTIME THE CODE IS RUN, THE DATA IS SHUFFLED.

About The Dataset

- **Dataset Name:** Chronic Kidney Disease¶
- **Abstract:** This dataset can be used to predict chronic kidney disease and it has been collected at a hospital for a period of nearly 2 months.

Data Set Characteristics	Multivariate
Number of Instances	400
Number of Attributes	25
Dataset Type	Classification
Data Separated By	Comma (,)
Dependent Variable Name	target

Data Preparation Approach

- Rename the columns so that the names are readable.

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	target

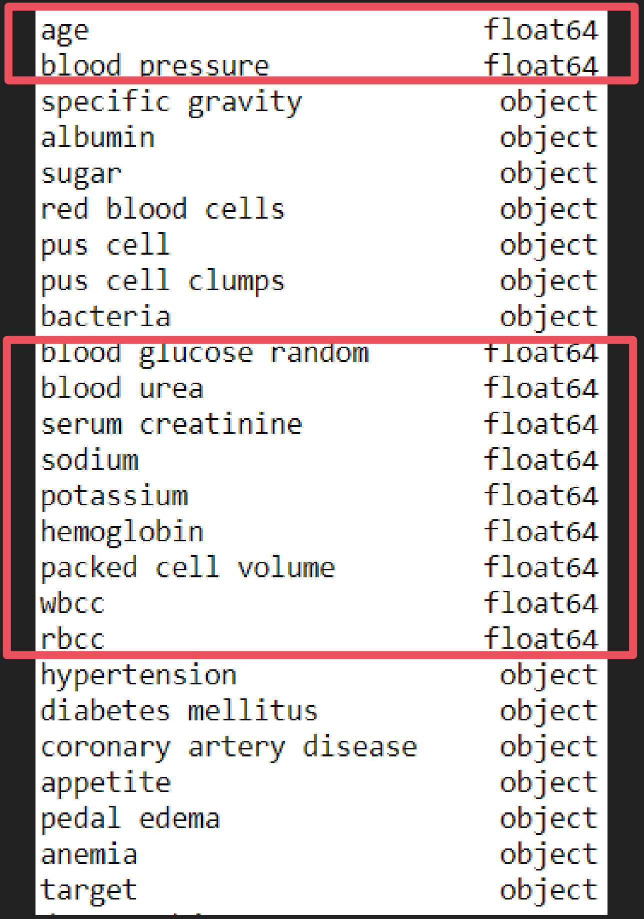
age	age
bp	blood pressure
sg	specific gravity
al	albumin
su	sugar
rbc	red blood cells
pc	pus cell
pcc	pus cell clumps
ba	bacteria
bgr	blood glucose random
bu	blood urea
sc	serum creatinine

Clean Dirty Data From Columns

- It was observed that some columns contained “?” question mark in the data.
- Approach:
  - For numeric columns: Force convert them to numeric using to_numeric with method “coerce”. This will replace the non numeric data with NULL.
  - For Categorical columns: Replace “?” with NULL.

Check the Data Type of Columns

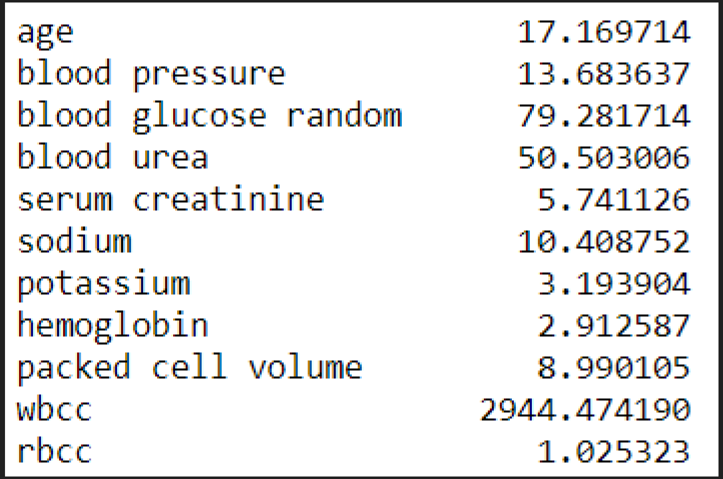
- We noticed that all the columns were 'object/categorical'. But as per the dataset description and basic understanding the data types we were getting were incorrect.
- Thus, all the variables were type casted as per their proper data types



```
age                          float64
blood pressure               float64
specific gravity              object
albumin                       object
sugar                         object
red blood cells               object
pus cell                      object
pus cell clumps               object
bacteria                      object
blood glucose random         float64
blood urea                   float64
serum creatinine             float64
sodium                       float64
potassium                    float64
hemoglobin                   float64
packed cell volume           float64
wbcc                         float64
rbcc                         float64
hypertension                  object
diabetes mellitus             object
coronary artery disease       object
appetite                      object
pedal edema                   object
anemia                        object
target                        object
```

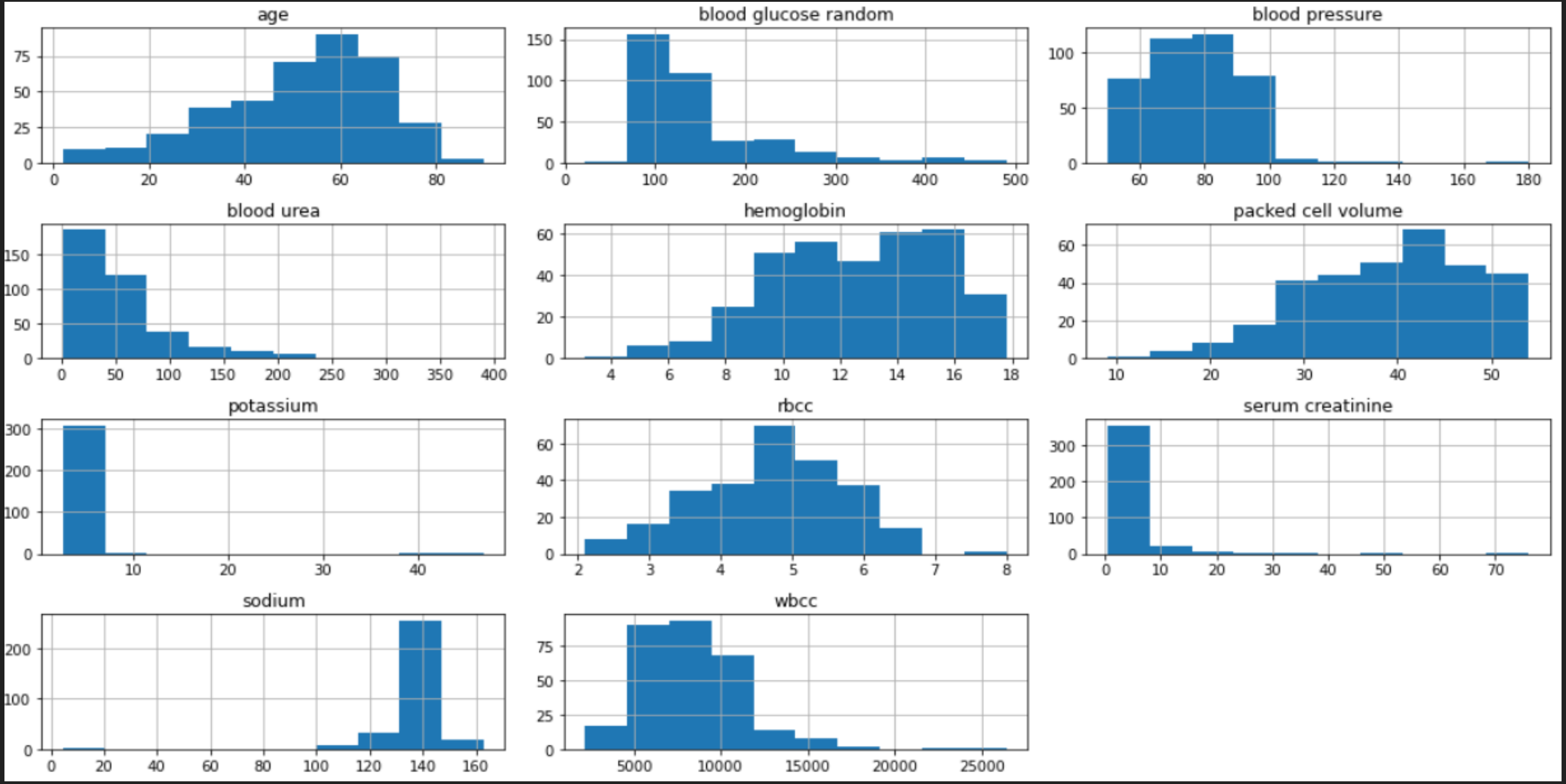
Checking for Insignificant Variables

- Check the standard deviation of the variables. If it is ZERO, it means that the column contains the same data and is useless for modelling. But fortunately, there was no such column.



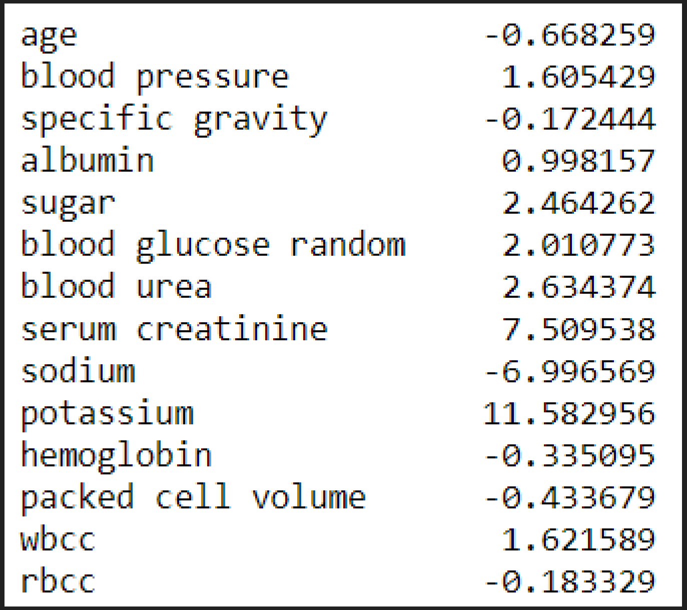
```
age                          17.169714
blood pressure               13.683637
blood glucose random         79.281714
blood urea                   50.503006
serum creatinine              5.741126
sodium                       10.408752
potassium                     3.193904
hemoglobin                    2.912587
packed cell volume            8.990105
wbcc                       2944.474190
rbcc                          1.025323
```

Distribution of numeric variables



Skewness

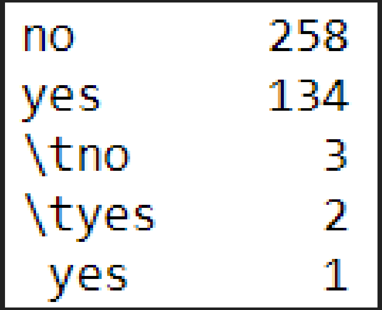
- From the previous distribution chart and the skewness table, we can see that many numerical variables are skewed
  - age looks a bit left skewed but its fine
  - Serum creatinine and potassium are highly right skewed
  - Sodium is highly left skewed
  - Rest of the features are lightly skewed



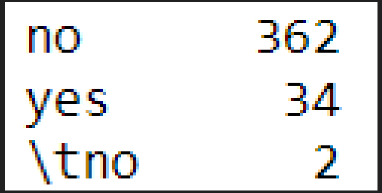
```
age                      -0.668259
blood pressure            1.605429
specific gravity         -0.172444
albumin                   0.998157
sugar                     2.464262
blood glucose random      2.010773
blood urea                2.634374
serum creatinine          7.509538
sodium                   -6.996569
potassium                11.582956
hemoglobin               -0.335095
packed cell volume       -0.433679
wbcc                      1.621589
rbcc                     -0.183329
```

Categorical Variables

- In some of the categorical variables, uncleaned class data was observed. That was treated by replacing the ‘\tno’, ‘\tyes’, ‘   yes’ with no and yes respectively.



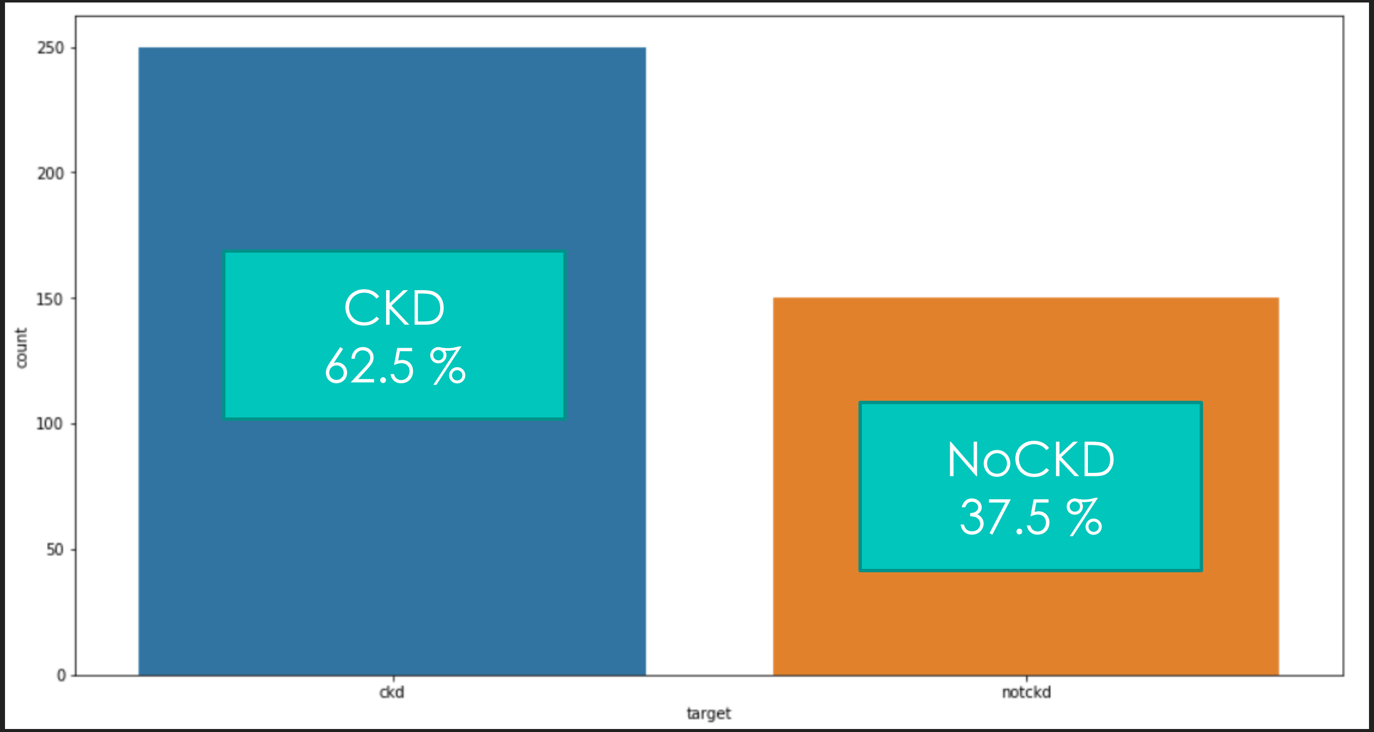
```
no        258
yes       134
\tno        3
\tyes       2
 yes        1
```



```
no        362
yes        34
\tno        2
```

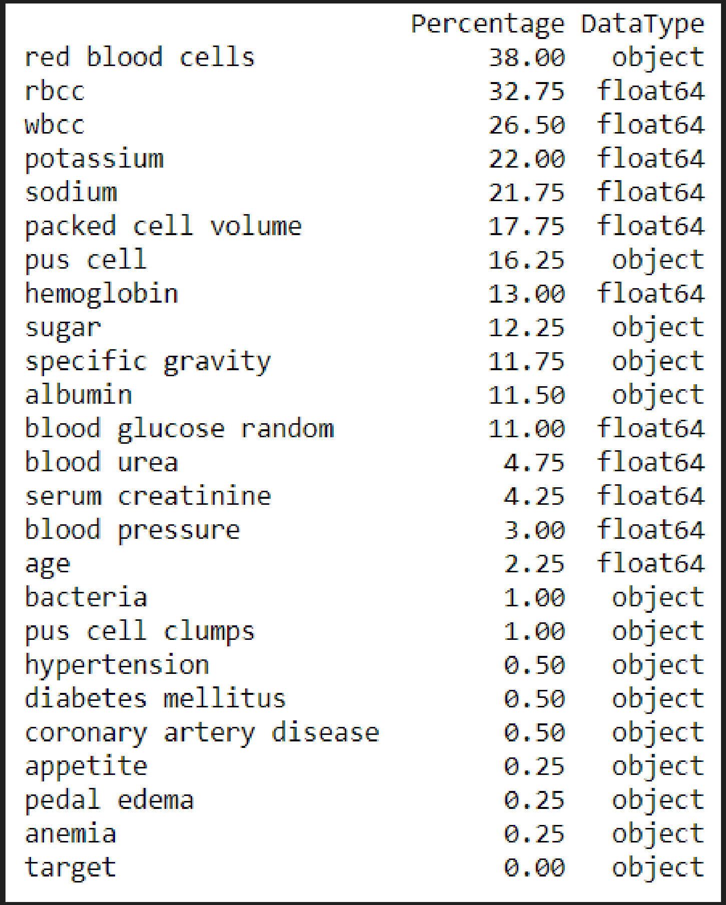
Check the class distribution of target variable

- The distribution looks good and is not that much imbalanced.
- ckd → 62.5 %
- notckd → 37.5 %



Missing Value Treatment

- Missing Data was seen in the data. Thus proper treatment method was required. From the distribution chart we saw that many of the numeric variables are skewed. Thus using MEAN as the imputation method is not preferable. Thus we went with MEDIAN imputation.
- For categorical variables, MODE imputation was used



```
                         Percentage DataType
red blood cells               38.00   object
rbcc                          32.75  float64
wbcc                          26.50  float64
potassium                     22.00  float64
sodium                        21.75  float64
packed cell volume            17.75  float64
pus cell                      16.25   object
hemoglobin                    13.00  float64
sugar                         12.25   object
specific gravity              11.75   object
albumin                       11.50   object
blood glucose random          11.00  float64
blood urea                     4.75  float64
serum creatinine               4.25  float64
blood pressure                 3.00  float64
age                            2.25  float64
bacteria                       1.00   object
pus cell clumps                1.00   object
hypertension                   0.50   object
diabetes mellitus              0.50   object
coronary artery disease        0.50   object
appetite                       0.25   object
pedal edema                    0.25   object
anemia                         0.25   object
target                         0.00   object
```

Exploratory Data Analysis

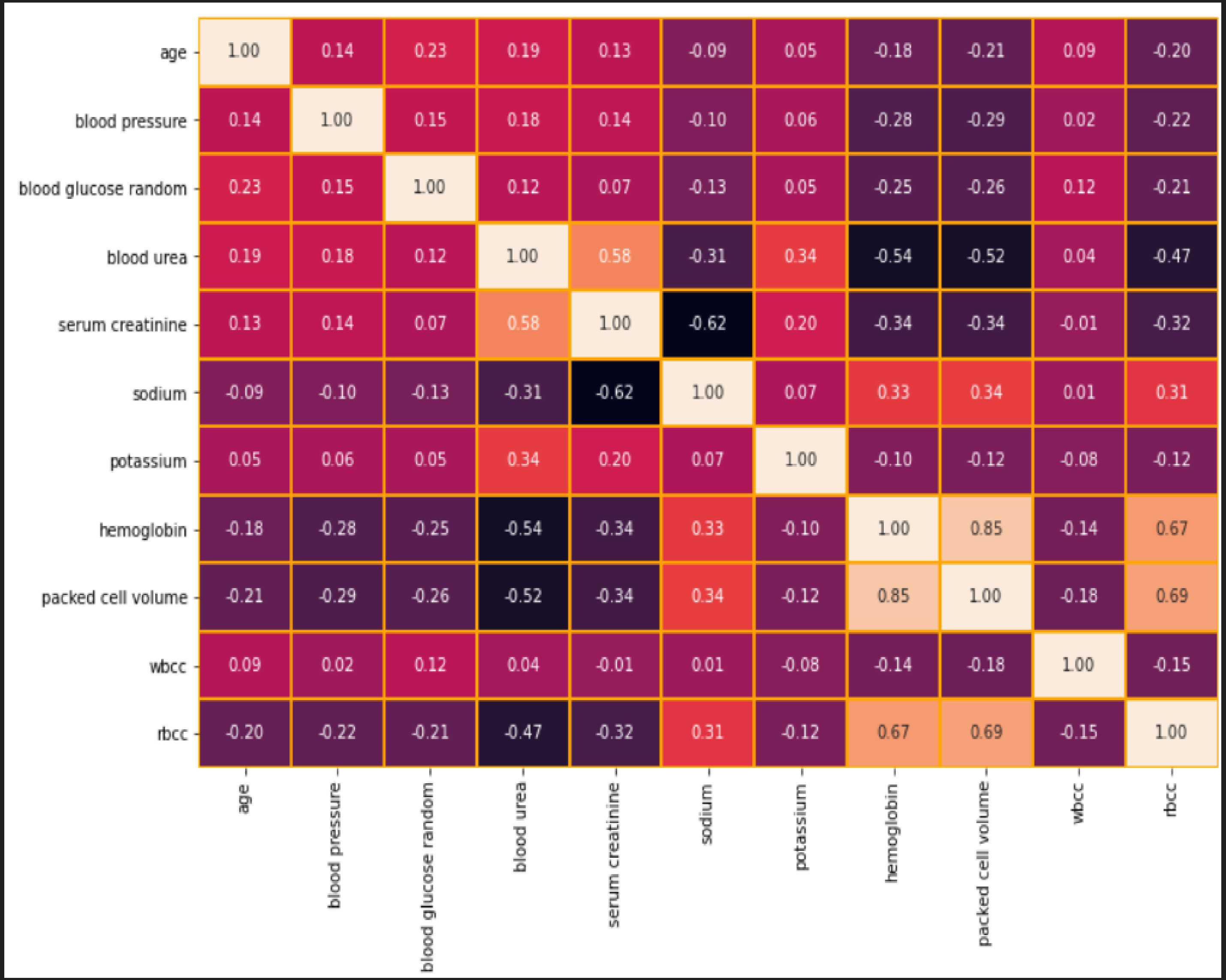
Correlation Matrix and Heatmap

Positive Correlation:

- Red blood cell count → Packed cell volume and Hemoglobin

Negative Correlation:

- Blood urea → Red blood cell count, packed cell volume, Hemoglobin
- Serum creatinine → Sodium
- Serum creatinine → blood urea



	age	blood pressure	blood glucose random	blood urea	serum creatinine	sodium	potassium	hemoglobin	packed cell volume	wbcc	rbcc
age	1.00	0.14	0.23	0.19	0.13	-0.09	0.05	-0.18	-0.21	0.09	-0.20
blood pressure	0.14	1.00	0.15	0.18	0.14	-0.10	0.06	-0.28	-0.29	0.02	-0.22
blood glucose random	0.23	0.15	1.00	0.12	0.07	-0.13	0.05	-0.25	-0.26	0.12	-0.21
blood urea	0.19	0.18	0.12	1.00	0.58	-0.31	0.34	-0.54	-0.52	0.04	-0.47
serum creatinine	0.13	0.14	0.07	0.58	1.00	-0.62	0.20	-0.34	-0.34	-0.01	-0.32
sodium	-0.09	-0.10	-0.13	-0.31	-0.62	1.00	0.07	0.33	0.34	0.01	0.31
potassium	0.05	0.06	0.05	0.34	0.20	0.07	1.00	-0.10	-0.12	-0.08	-0.12
hemoglobin	-0.18	-0.28	-0.25	-0.54	-0.34	0.33	-0.10	1.00	0.85	-0.14	0.67
packed cell volume	-0.21	-0.29	-0.26	-0.52	-0.34	0.34	-0.12	0.85	1.00	-0.18	0.69
wbcc	0.09	0.02	0.12	0.04	-0.01	0.01	-0.08	-0.14	-0.18	1.00	-0.15
rbcc	-0.20	-0.22	-0.21	-0.47	-0.32	0.31	-0.12	0.67	0.69	-0.15	1.00

SCATTER PLOT: Red Blood Cell Count VS. Packed Cell Volume

Approx. General Observation:

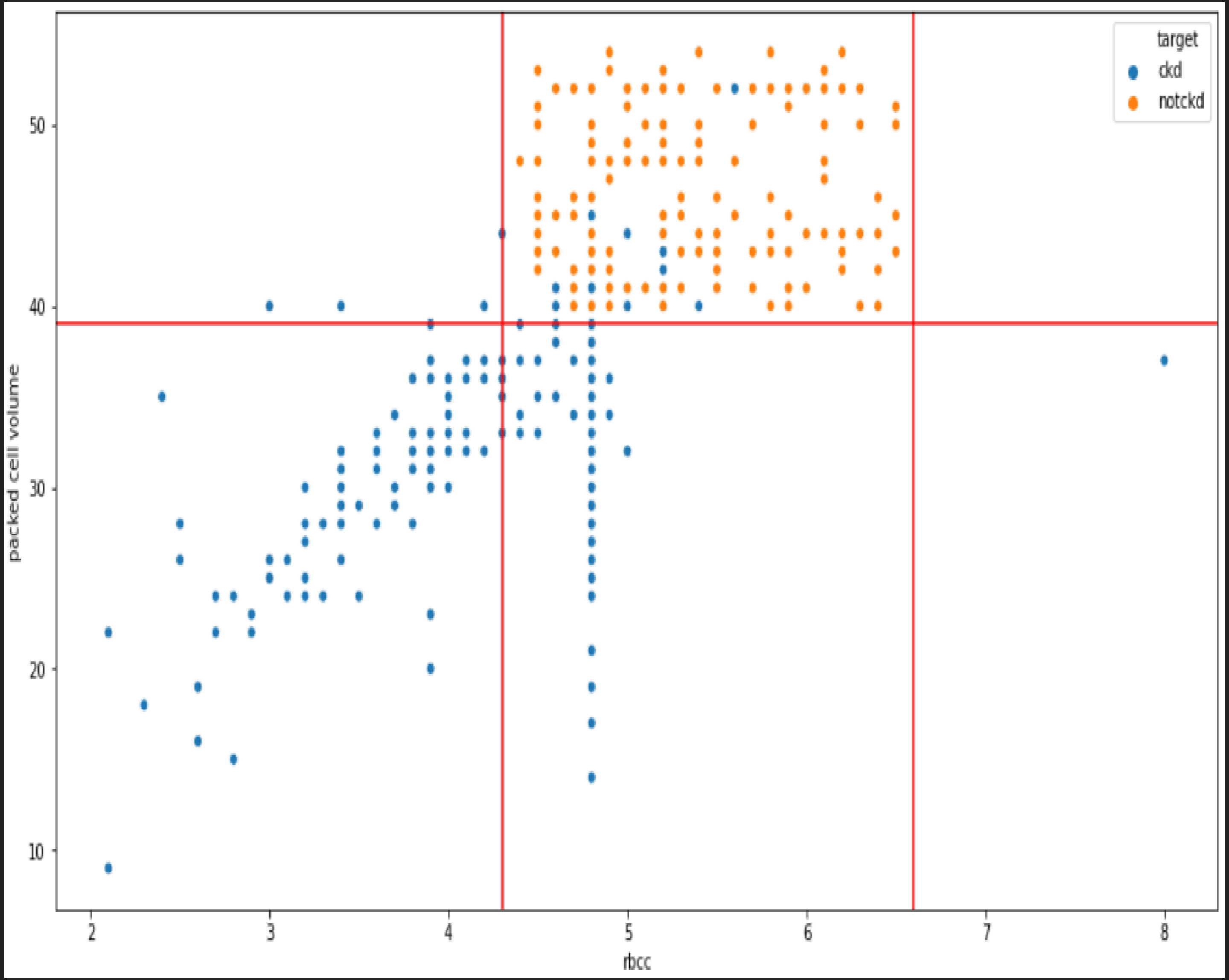
RBCC range

- 2 and < 4.3 and

Packed Cell Volume range

- > 15 and <40

are mostly classified as having chronic kidney disease



SCATTER PLOT: Hemoglobin VS. Packed Cell Volume

Approx. General Observation:

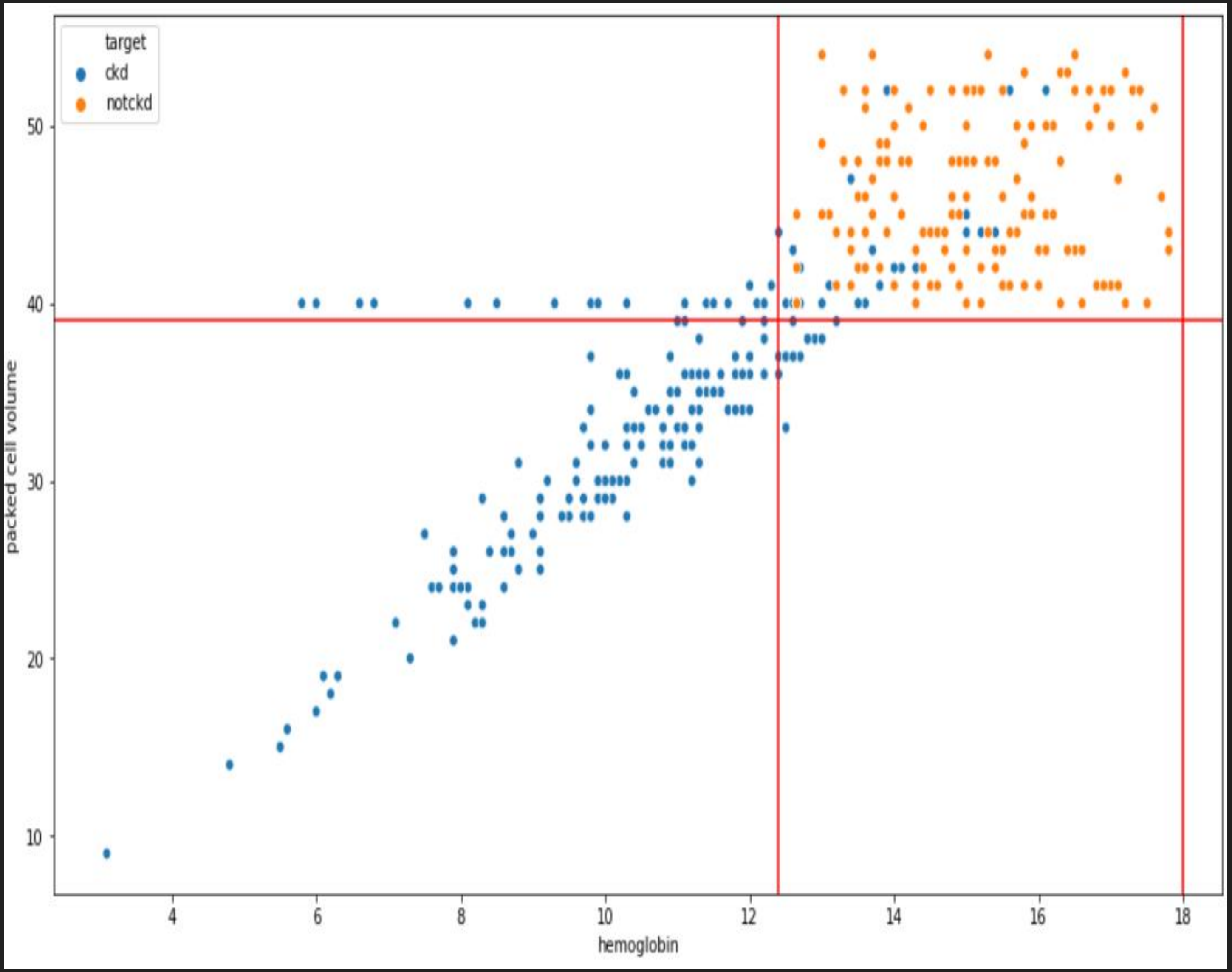
Hemoglobin range

- > 12.5 and

Packed Cell Volume range

- > 40

are mostly classified as NOT having chronic kidney disease



SCATTER PLOT: Red Blood Cell Count VS. Hemoglobin

Approx. General Observation:

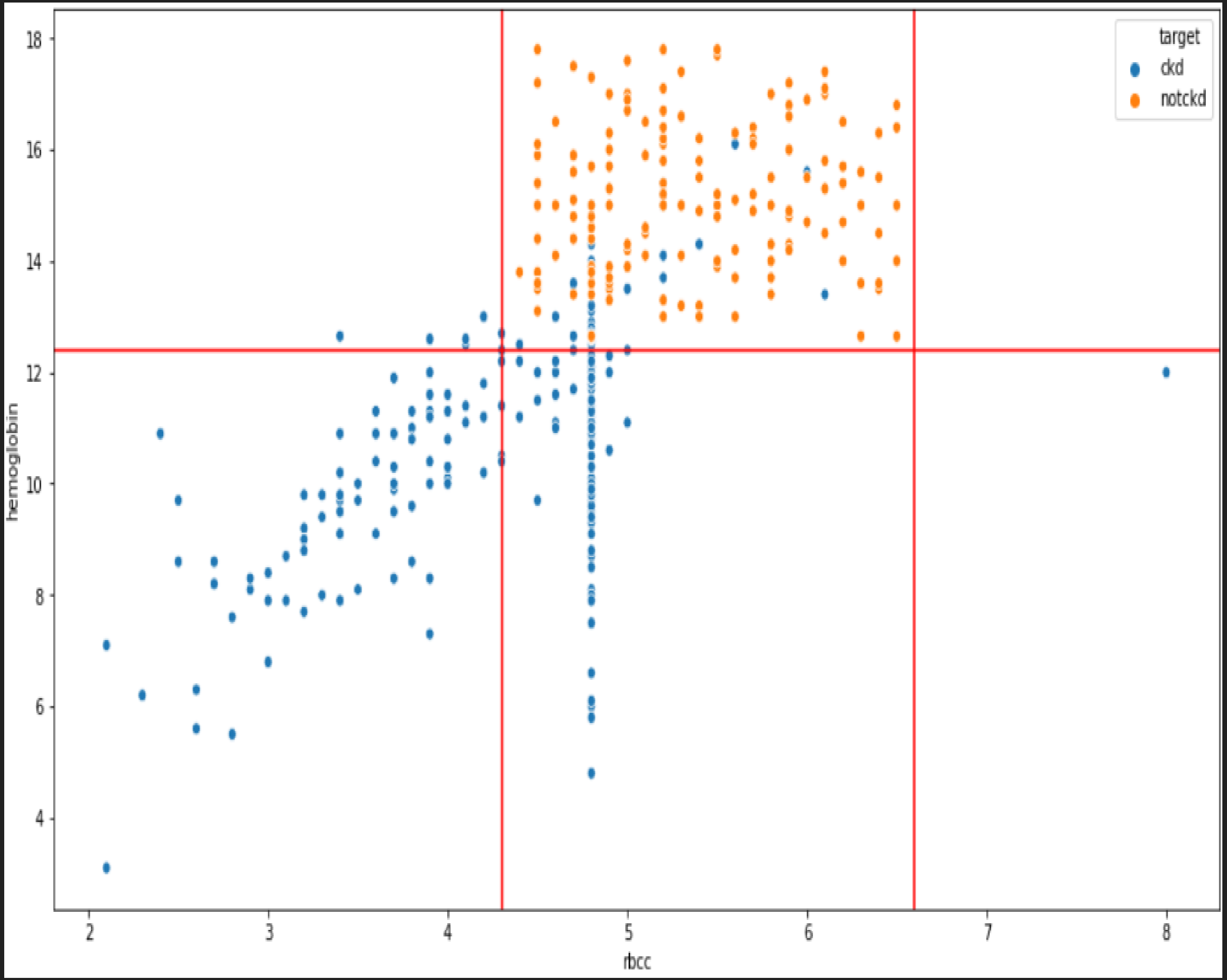
RBCC range

- > 4.2 and < 6.6 and

Hemoglobin range

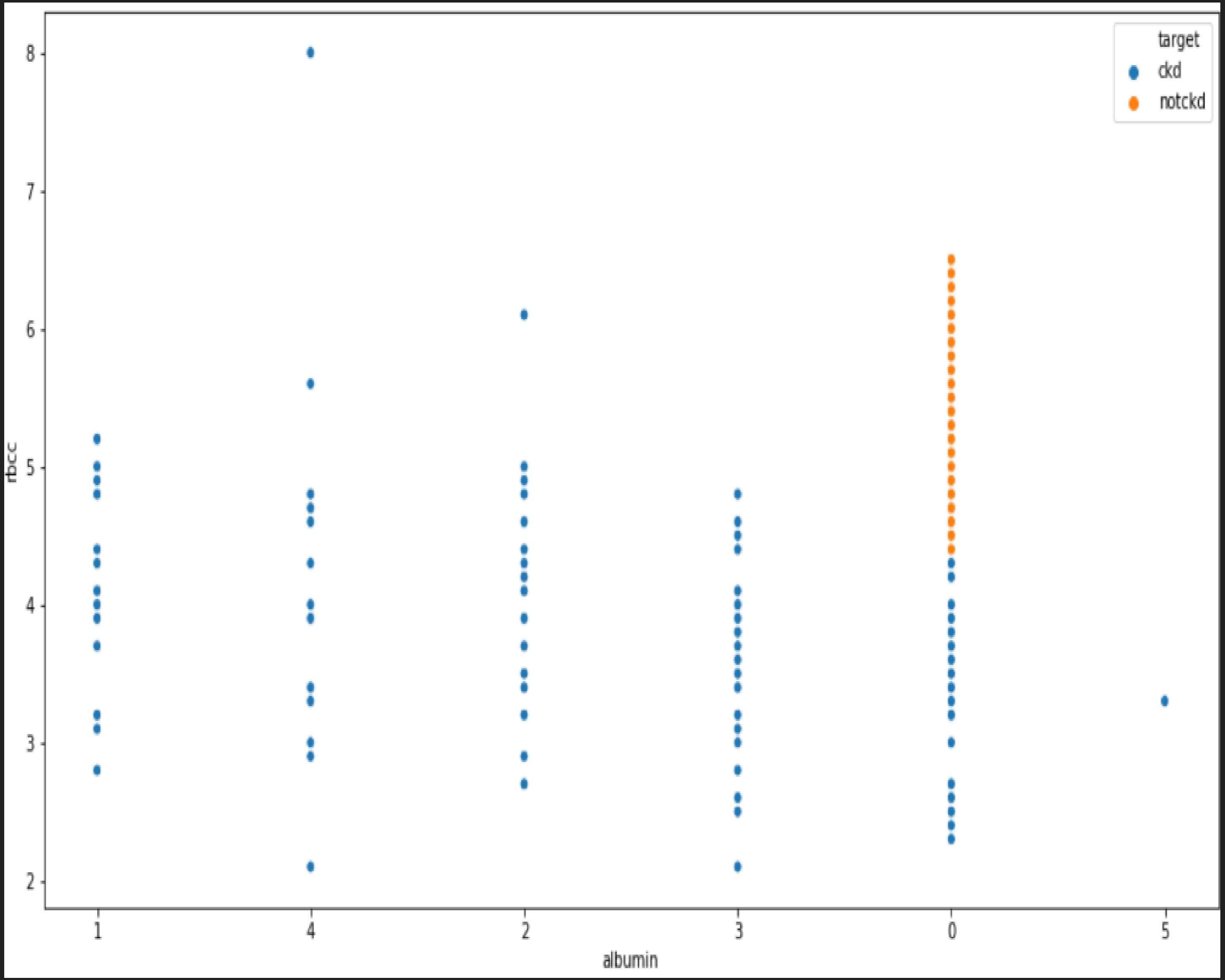
- > 13

are mostly classified as NOT having chronic kidney disease



SCATTER PLOT: Red Blood Cell Count VS. Albumin

From the chart it is clearly seen that Albumin levels greater than 0 affect CKD highly.

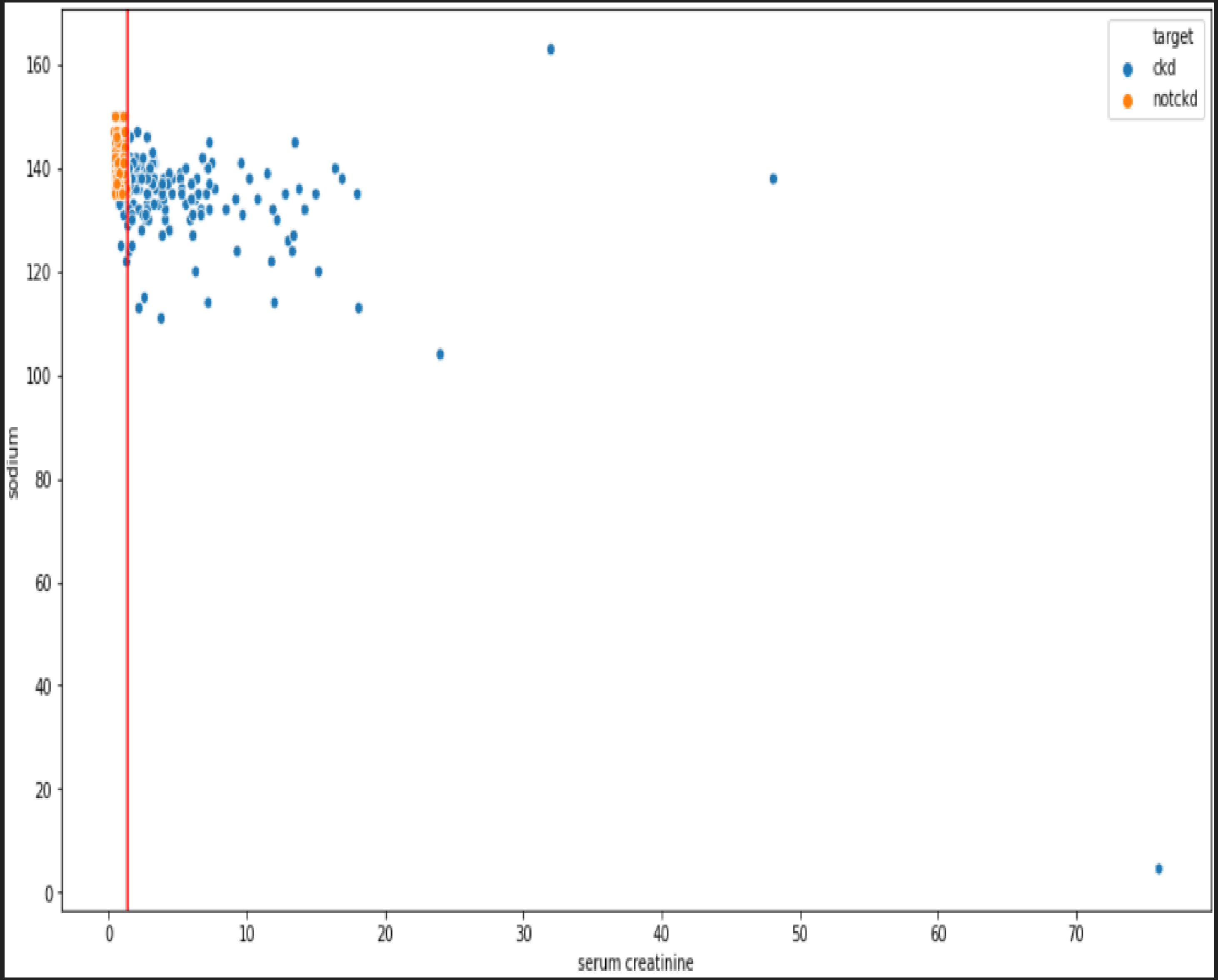


SCATTER PLOT: Sodium VS. Serum Creatinine

Approx. General Observation:

Serum Creatinine value greater than

~ 1.1 shows signs of having CKD



SCATTER PLOT: Blood Urea VS. Hemoglobin

Approx. General Observation:

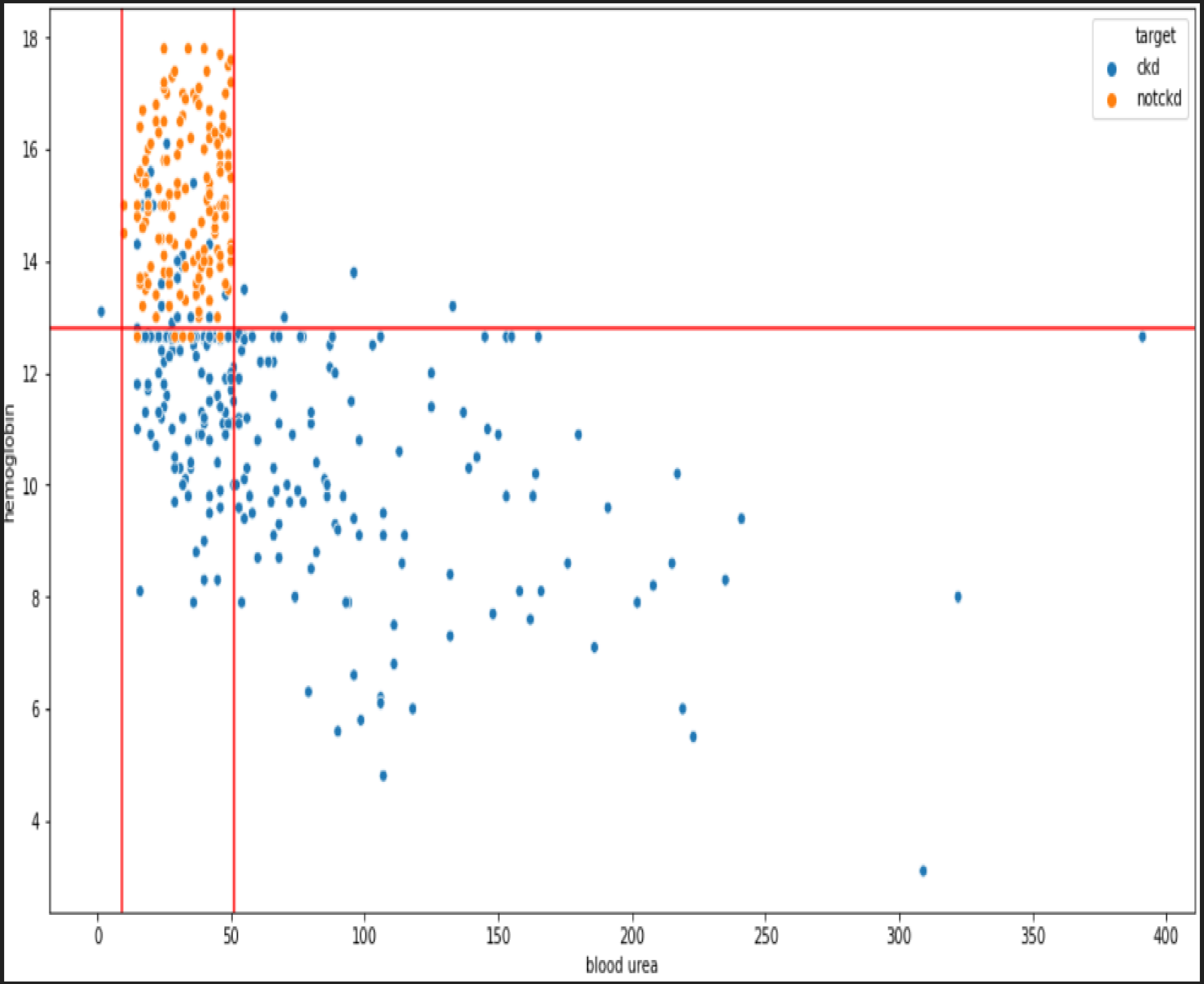
Hemoglobin range

- > 12.5

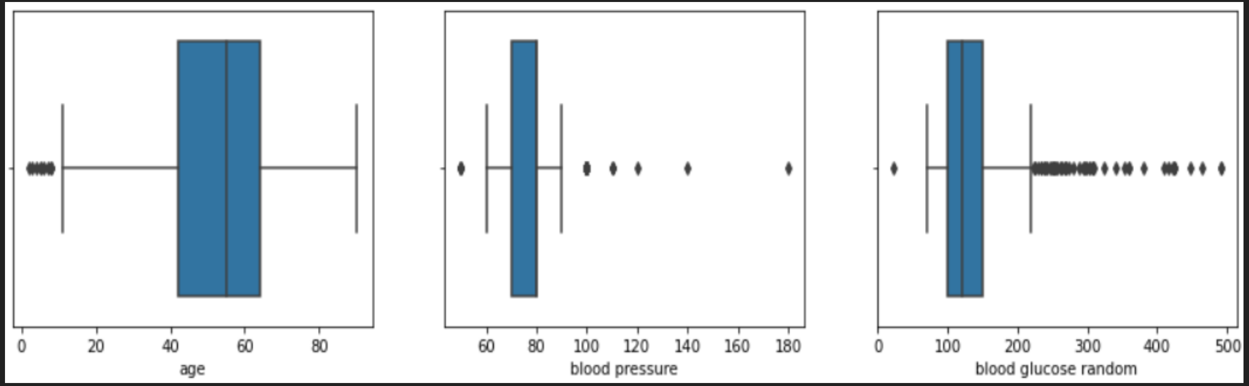
Blood Urea range

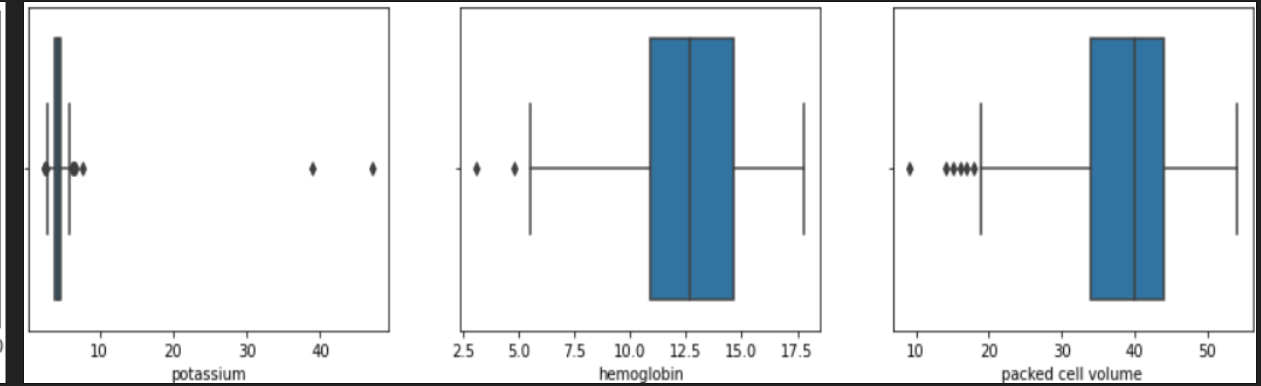
- < 50

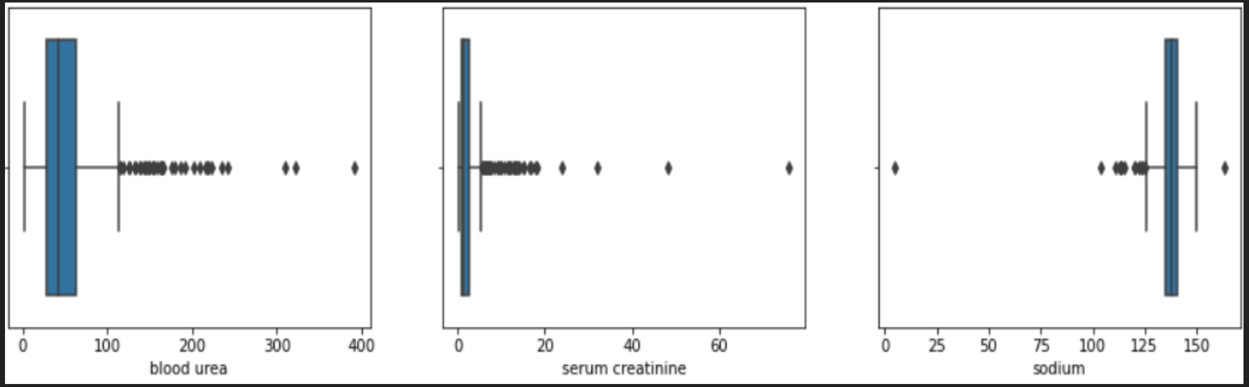
are mostly classified as NOT having chronic kidney disease

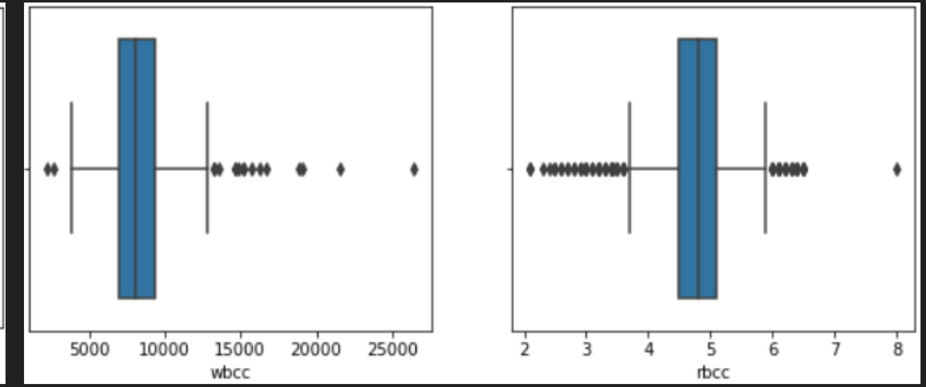


Check for Outliers / Draw Boxplot



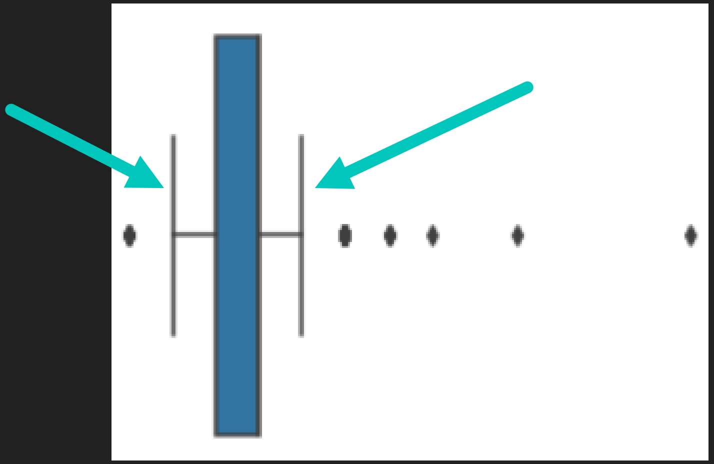






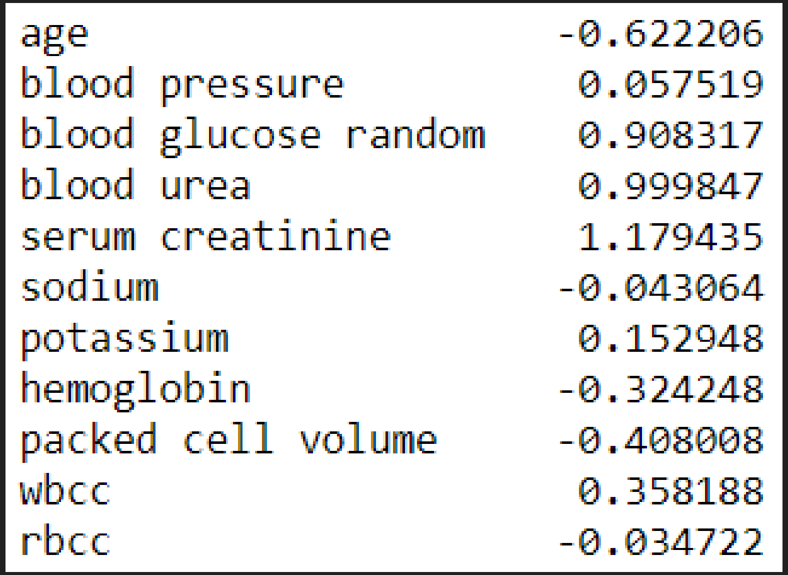
Treat Outliers

- Actually these data cannot be called unnecessary outliers. And we already have very less data i.e. 400 observations. Thus, dropping the outliers is a bad choice. Thus our approach was to replace the positive and negative outliers with upper and lower bound respectively.



Treat Outliers

- After treating outliers, the skewness decreased and the data was good to use.



```
age                     -0.622206
blood pressure           0.057519
blood glucose random     0.908317
blood urea               0.999847
serum creatinine         1.179435
sodium                  -0.043064
potassium                0.152948
hemoglobin              -0.324248
packed cell volume      -0.408008
wbcc                     0.358188
rbcc                    -0.034722
```

Shuffle The Dataset

- The data was sorted on the basis of target variable. Thus, we have shuffled the data for unbiased splitting

Dummy Encode Categorical Variables

- Categorical variables were dummy encoded and the first class was dropped

Encode Target Variable

- Our target variable has string classes. Thus converting them:
  - Ckd → 1
  - Nockd → 0

Why is Scaling needed?

- KNN and Logistic regression are highly affected by variation in data ranges as well as outliers.
- That’s the reason we treated outliers and will do scaling too.
- In the upcoming slides we will see the difference in performance of the models Before and After scaling the data.

Train – Test Split the data

- Training Data : 80 %
- Test Data : 20 %

- This splitting numbers are not fixed. Any pair can be taken 60-40, 80-20, 75-25

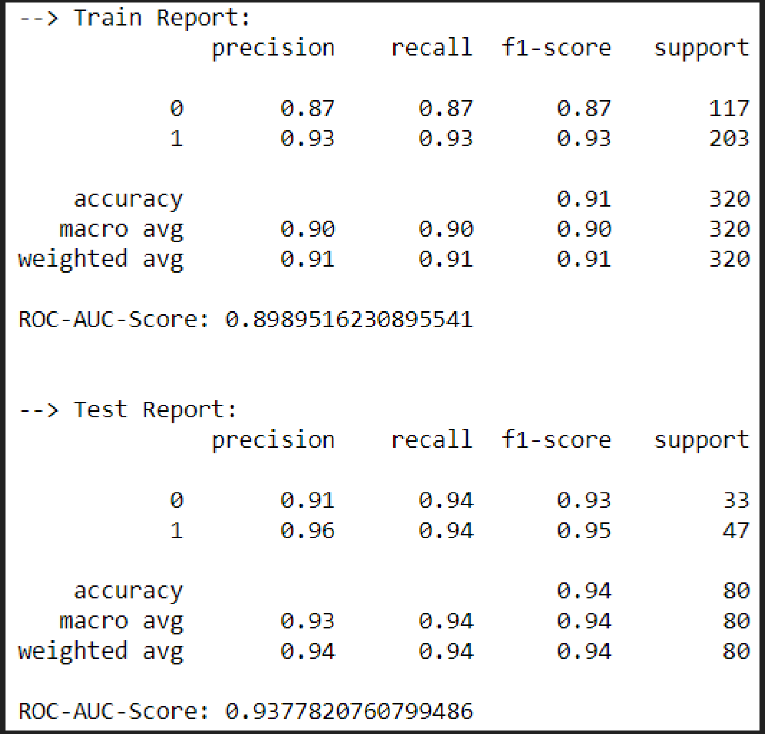
Machine Learning Algorithms Used

- For our project we have use simple yet powerful ML algorithms like:
  - Decision Trees
  - Random Forest
  - Logistic Regression
  - K-Nearest Neighbors

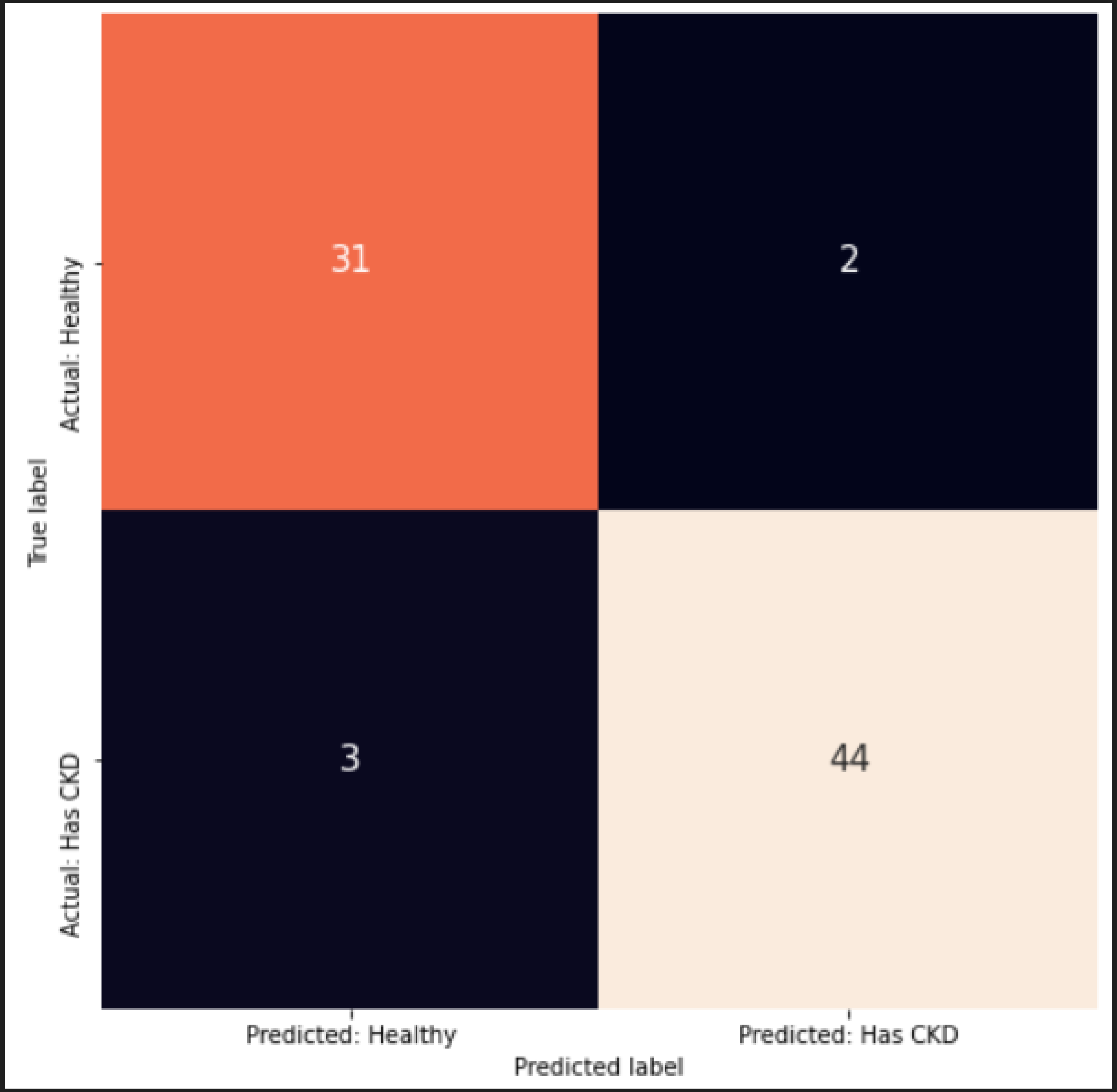
- We would have also used advanced algorithms like Neural Nets, Gradient Boosting, SVM, etc. but why to use them if we are getting a better solution by using simple algorithms.

Model Building (Before Scaling)

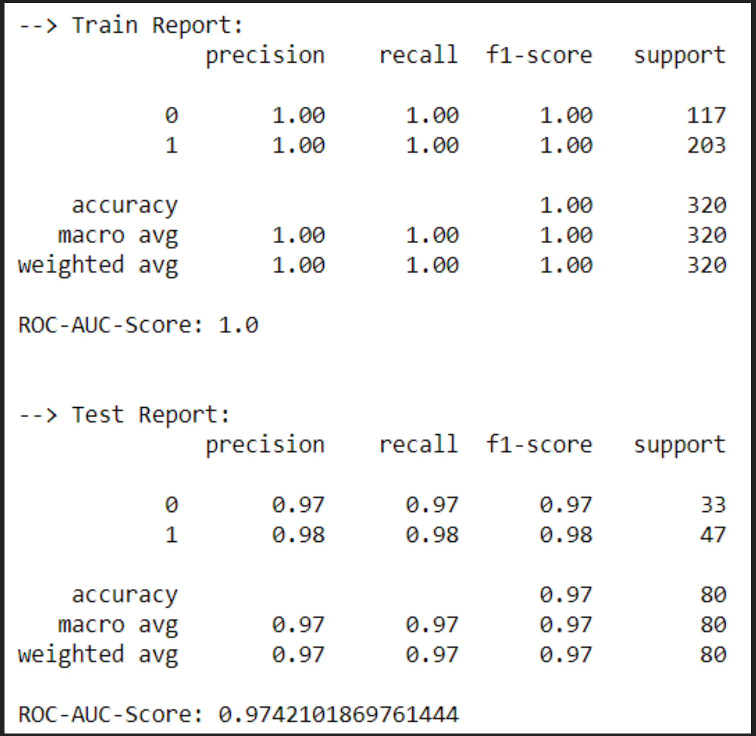
Logistic Regression



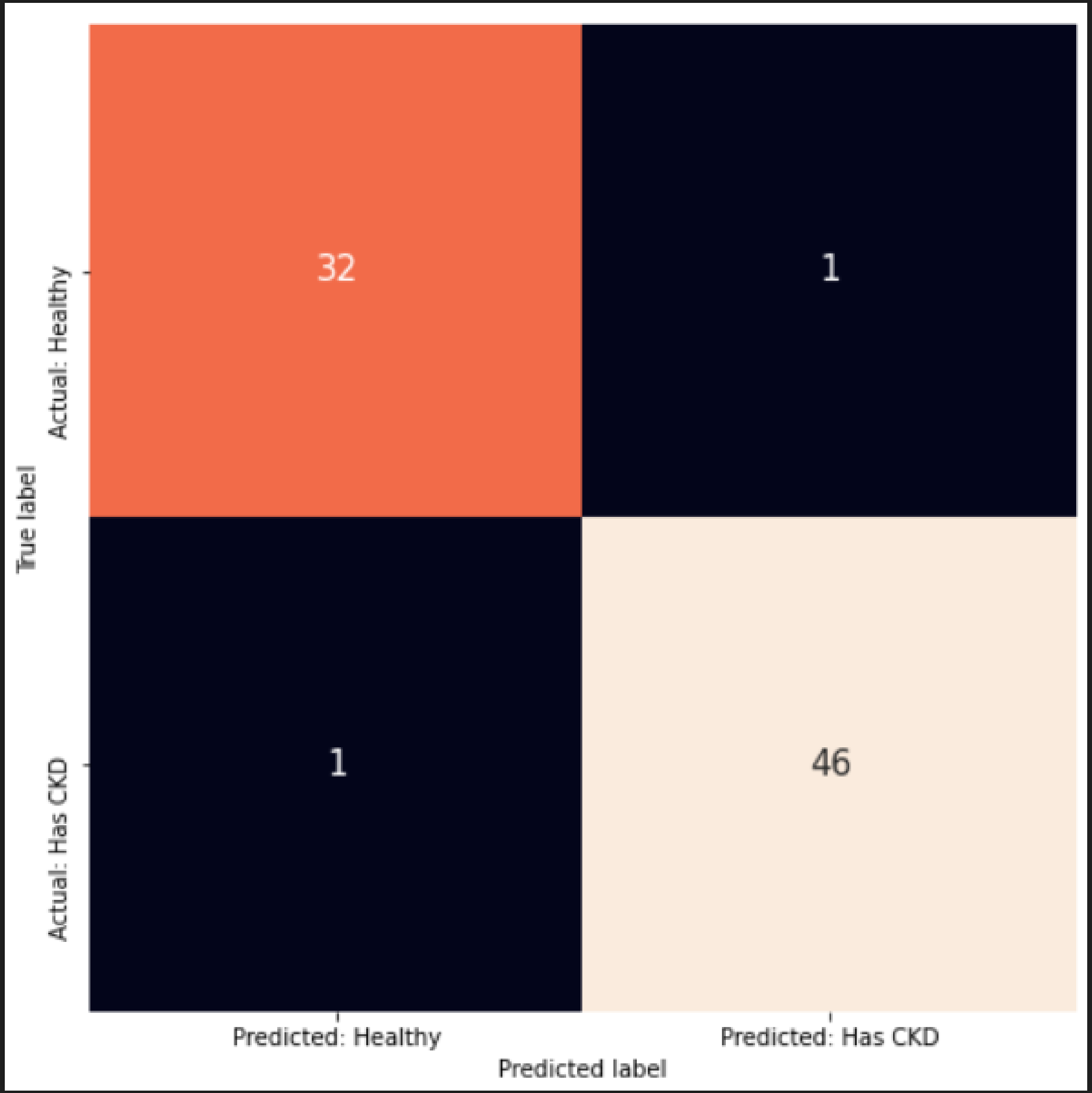
```
--> Train Report:
              precision    recall  f1-score   support

           0       0.87      0.87      0.87       117
           1       0.93      0.93      0.93       203

    accuracy                           0.91       320
   macro avg       0.90      0.90      0.90       320
weighted avg       0.91      0.91      0.91       320

ROC-AUC-Score: 0.8989516230895541


--> Test Report:
              precision    recall  f1-score   support

           0       0.91      0.94      0.93        33
           1       0.96      0.94      0.95        47

    accuracy                           0.94        80
   macro avg       0.93      0.94      0.94        80
weighted avg       0.94      0.94      0.94        80

ROC-AUC-Score: 0.9377820760799486
```



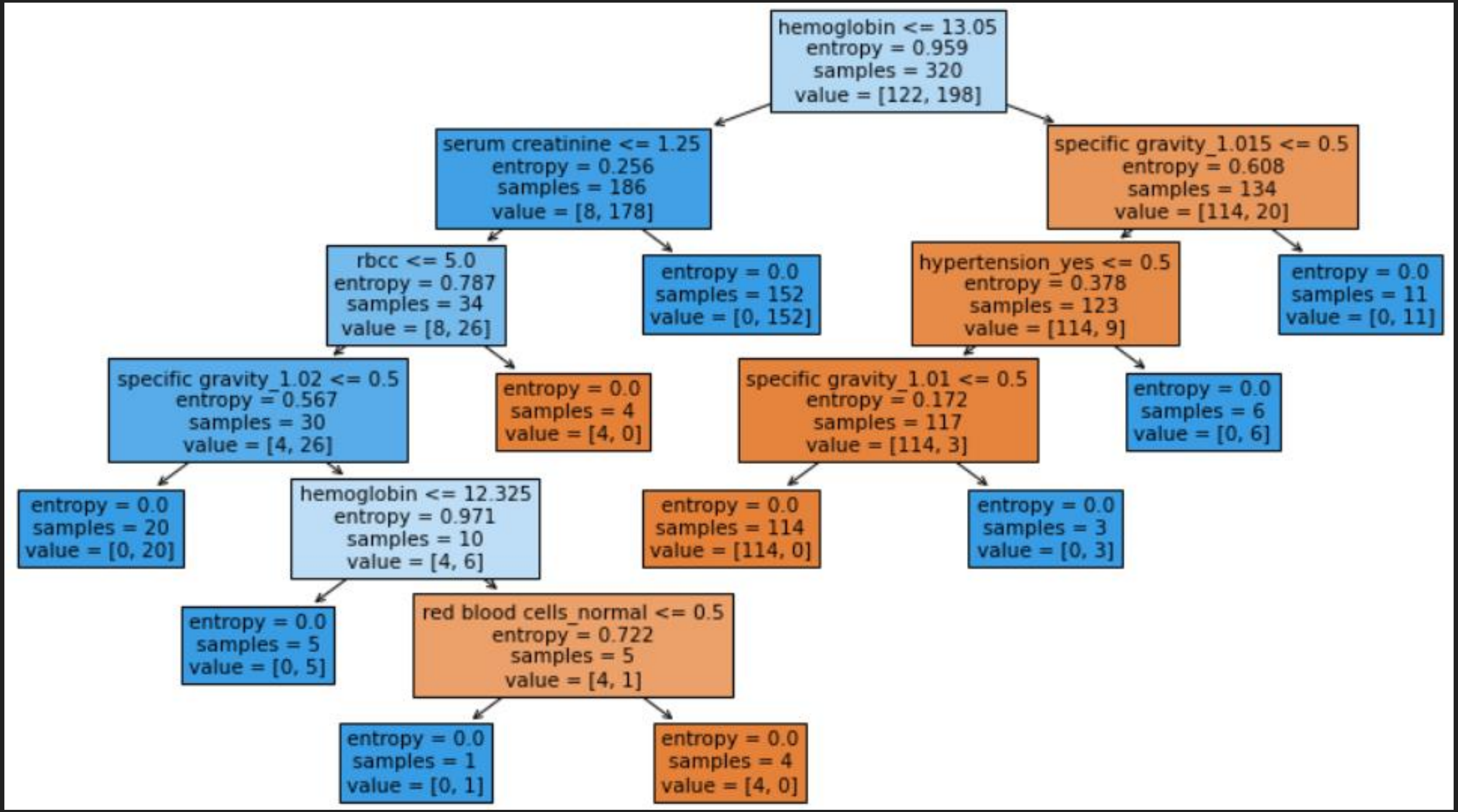
Decision Tree



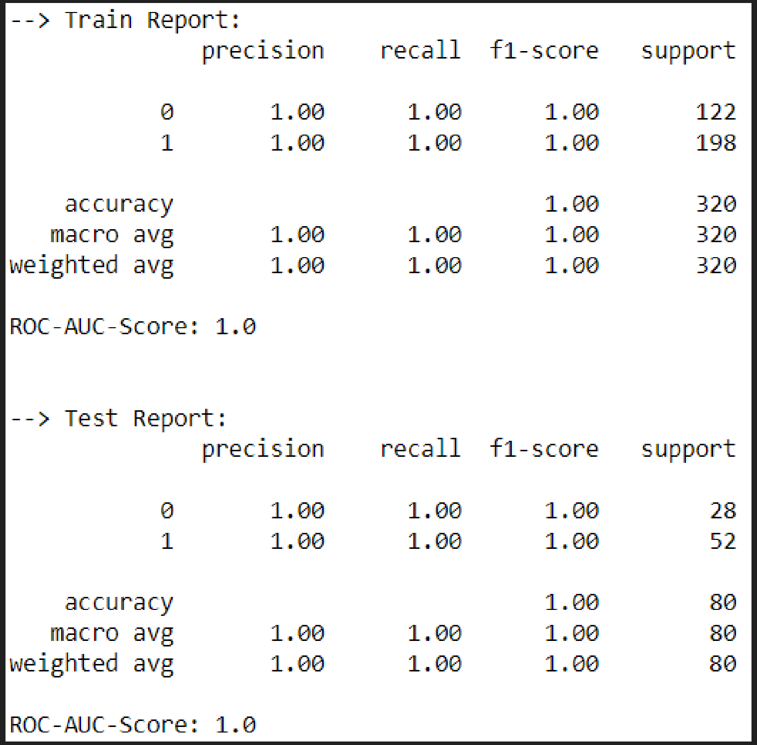
```
--> Train Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       117
           1       1.00      1.00      1.00       203

    accuracy                           1.00       320
   macro avg       1.00      1.00      1.00       320
weighted avg       1.00      1.00      1.00       320

ROC-AUC-Score: 1.0


--> Test Report:
              precision    recall  f1-score   support

           0       0.97      0.97      0.97        33
           1       0.98      0.98      0.98        47

    accuracy                           0.97        80
   macro avg       0.97      0.97      0.97        80
weighted avg       0.97      0.97      0.97        80

ROC-AUC-Score: 0.9742101869761444
```



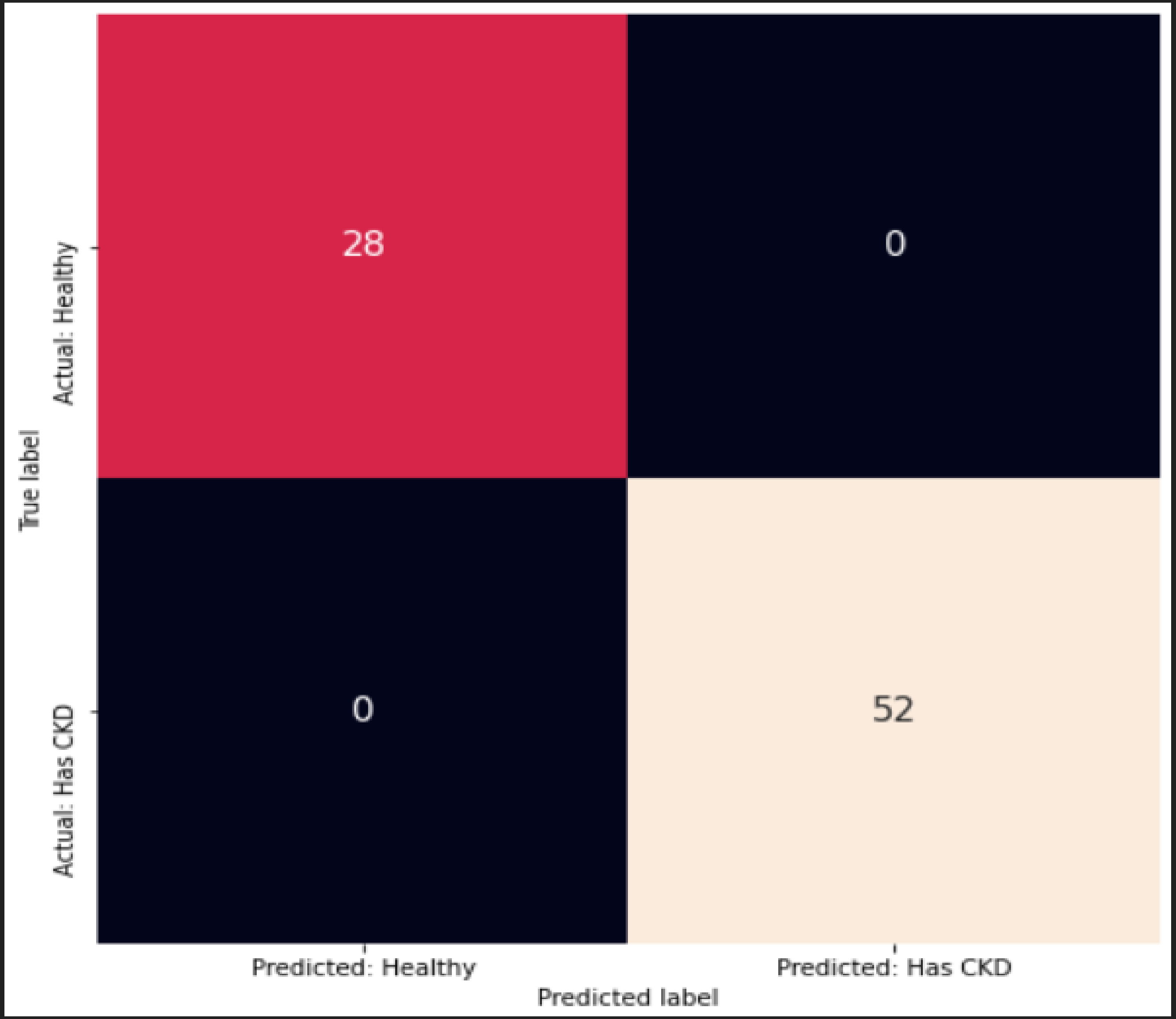
Decision Tree Plot

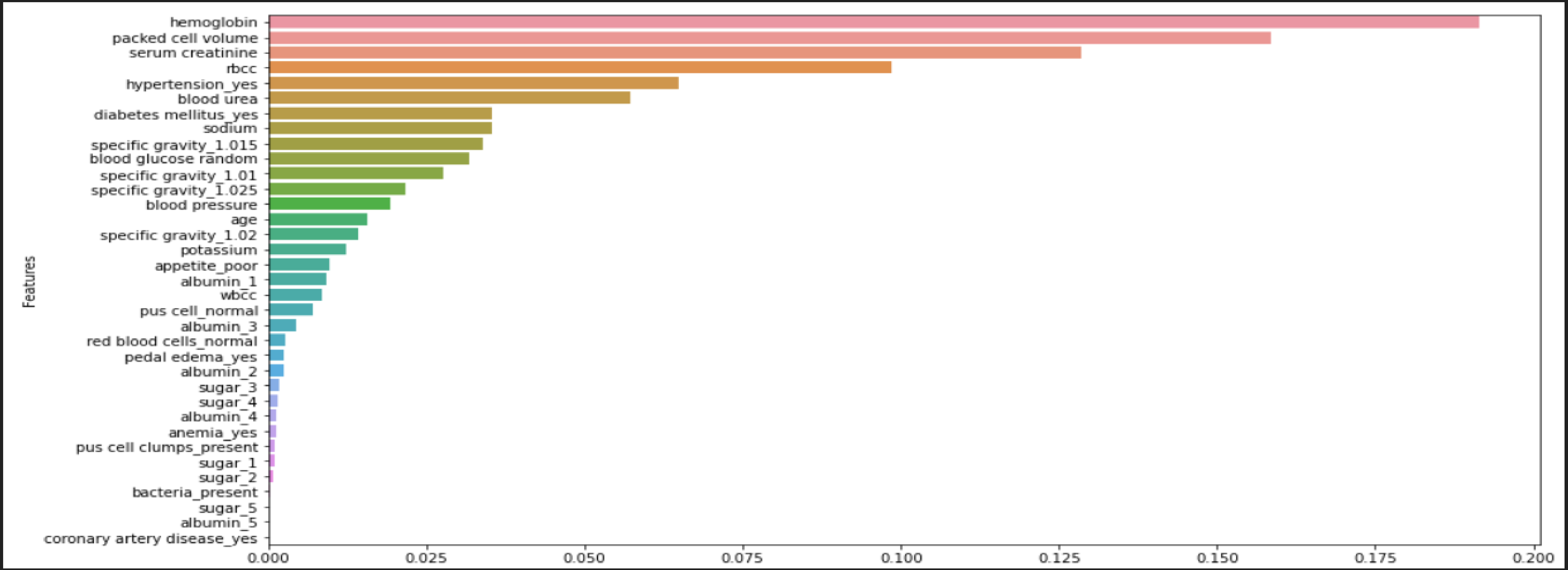


Random Forest



```
--> Train Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       122
           1       1.00      1.00      1.00       198

    accuracy                           1.00       320
   macro avg       1.00      1.00      1.00       320
weighted avg       1.00      1.00      1.00       320

ROC-AUC-Score: 1.0


--> Test Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        28
           1       1.00      1.00      1.00        52

    accuracy                           1.00        80
   macro avg       1.00      1.00      1.00        80
weighted avg       1.00      1.00      1.00        80

ROC-AUC-Score: 1.0
```

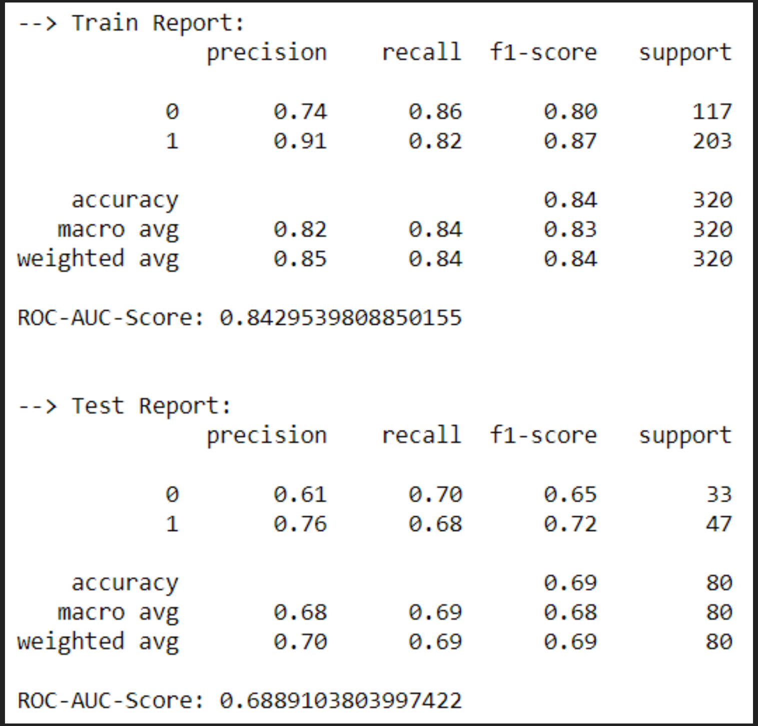




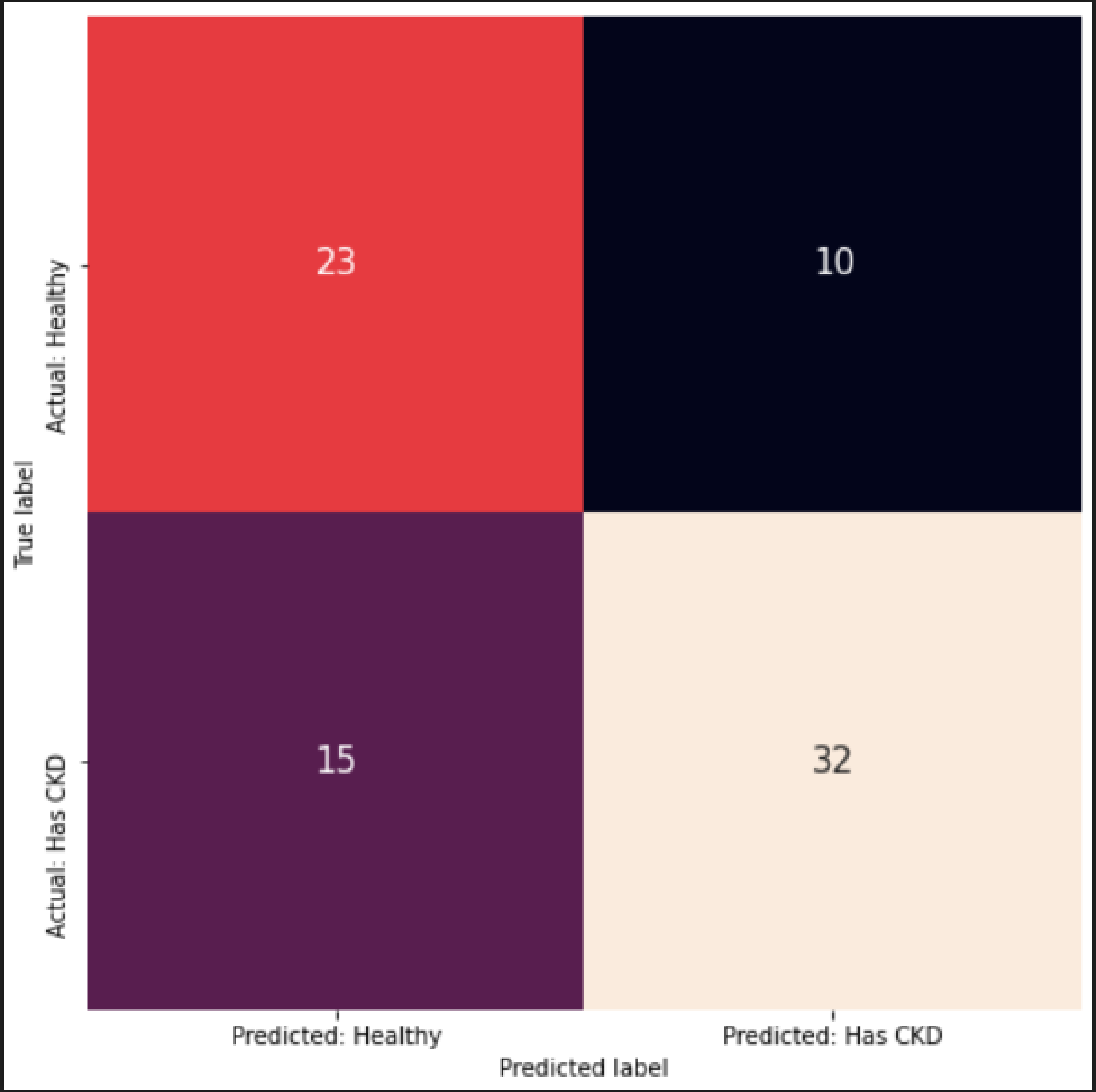
Feature Importance Chart

Hemoglobin, Packed Cell Volume, Serum Creatinine, RBCC and Hypertension are the top 5 important features to deduce if a person has CKD or not.

K-Nearest Neighbors

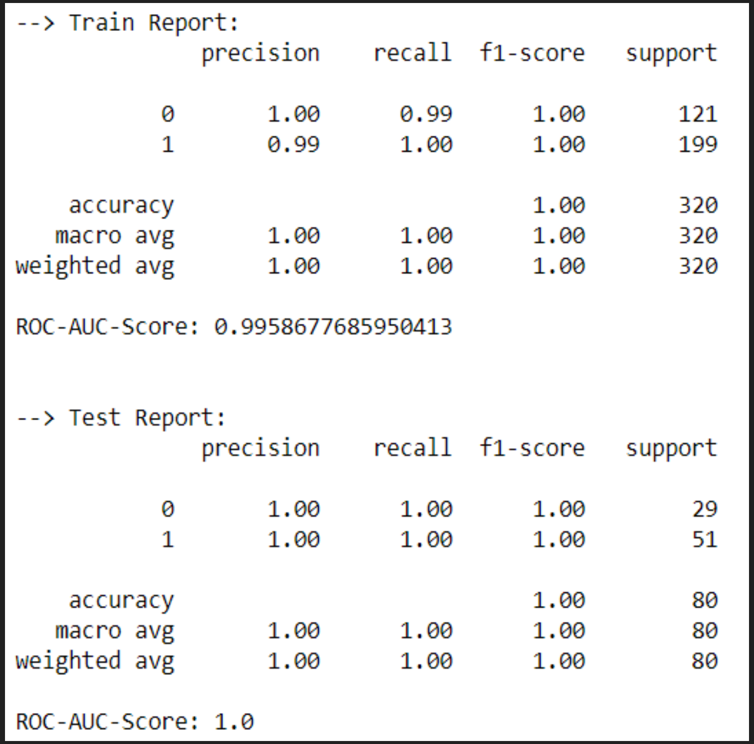


```
--> Train Report:
              precision    recall  f1-score   support

           0       0.74      0.86      0.80       117
           1       0.91      0.82      0.87       203

    accuracy                           0.84       320
   macro avg       0.82      0.84      0.83       320
weighted avg       0.85      0.84      0.84       320

ROC-AUC-Score: 0.8429539808850155


--> Test Report:
              precision    recall  f1-score   support

           0       0.61      0.70      0.65        33
           1       0.76      0.68      0.72        47

    accuracy                           0.69        80
   macro avg       0.68      0.69      0.68        80
weighted avg       0.70      0.69      0.69        80

ROC-AUC-Score: 0.6889103803997422
```

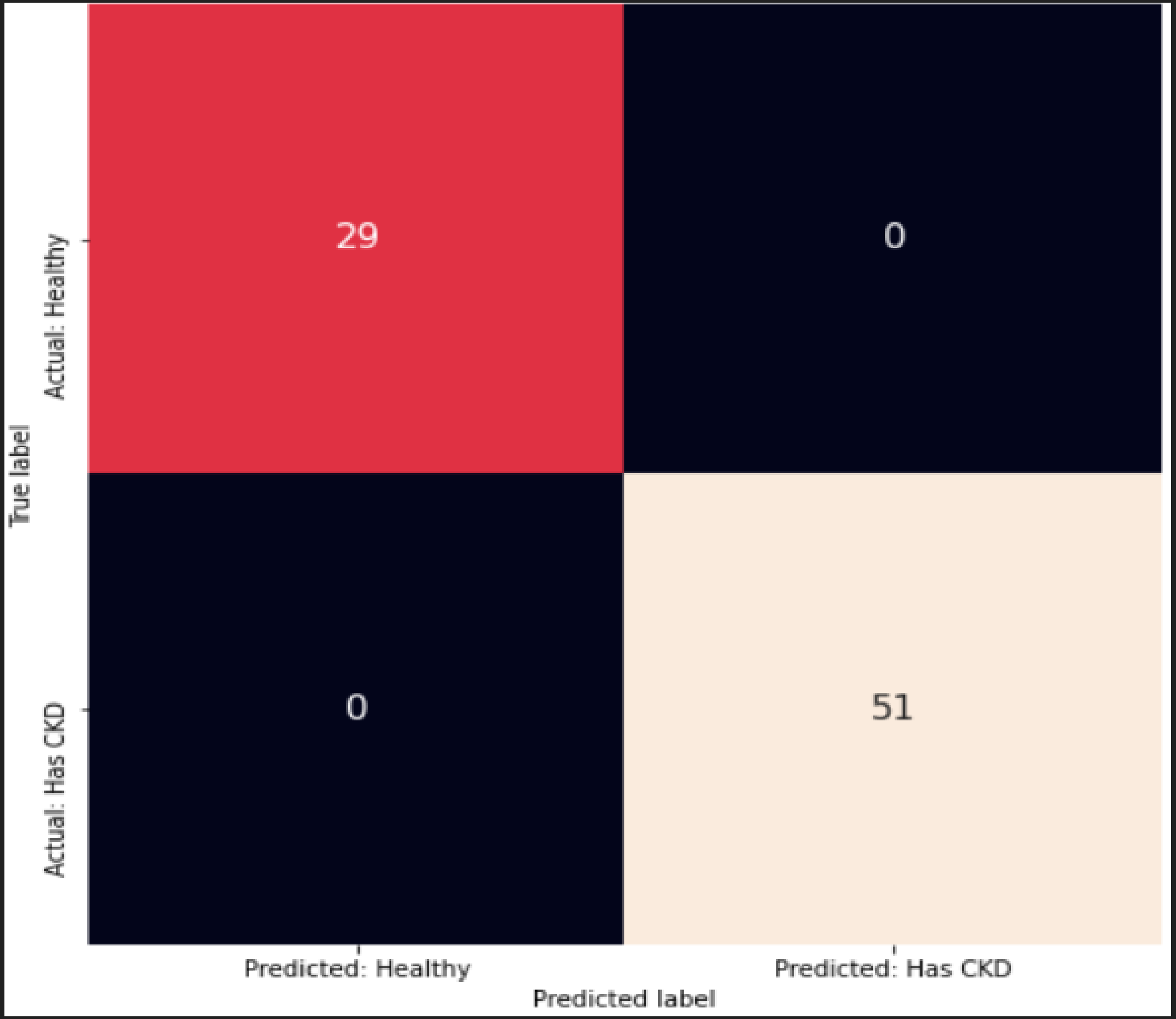


Model Building (After Scaling)

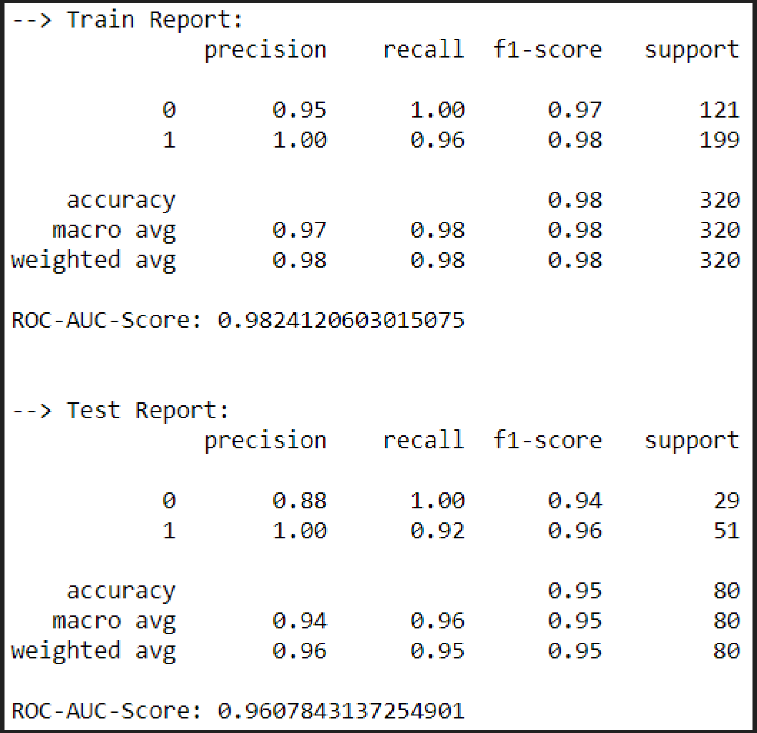
Logistic Regression



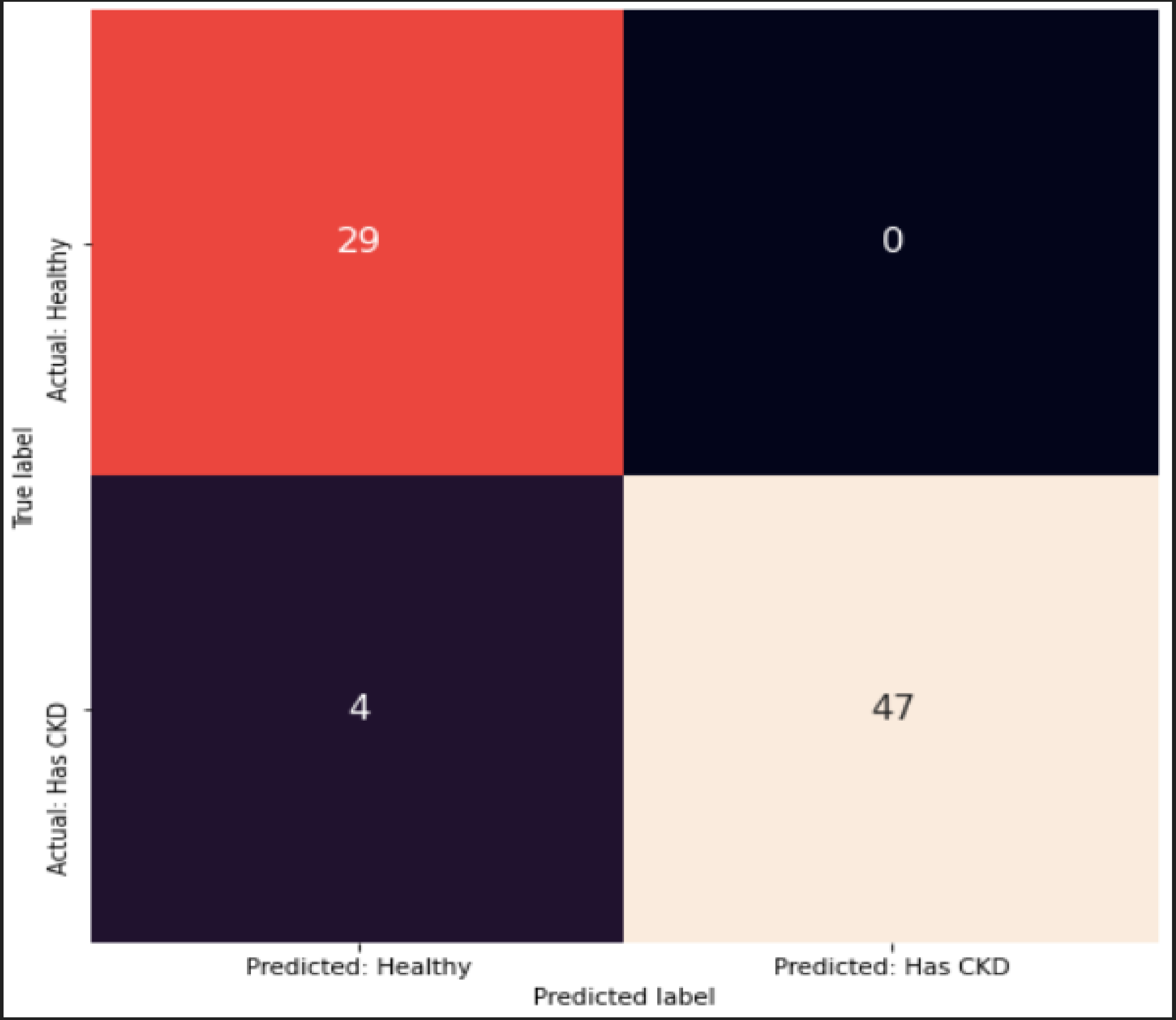
```
--> Train Report:
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       121
           1       0.99      1.00      1.00       199

    accuracy                           1.00       320
   macro avg       1.00      1.00      1.00       320
weighted avg       1.00      1.00      1.00       320

ROC-AUC-Score: 0.9958677685950413


--> Test Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        29
           1       1.00      1.00      1.00        51

    accuracy                           1.00        80
   macro avg       1.00      1.00      1.00        80
weighted avg       1.00      1.00      1.00        80

ROC-AUC-Score: 1.0
```



K-Nearest Neighbors



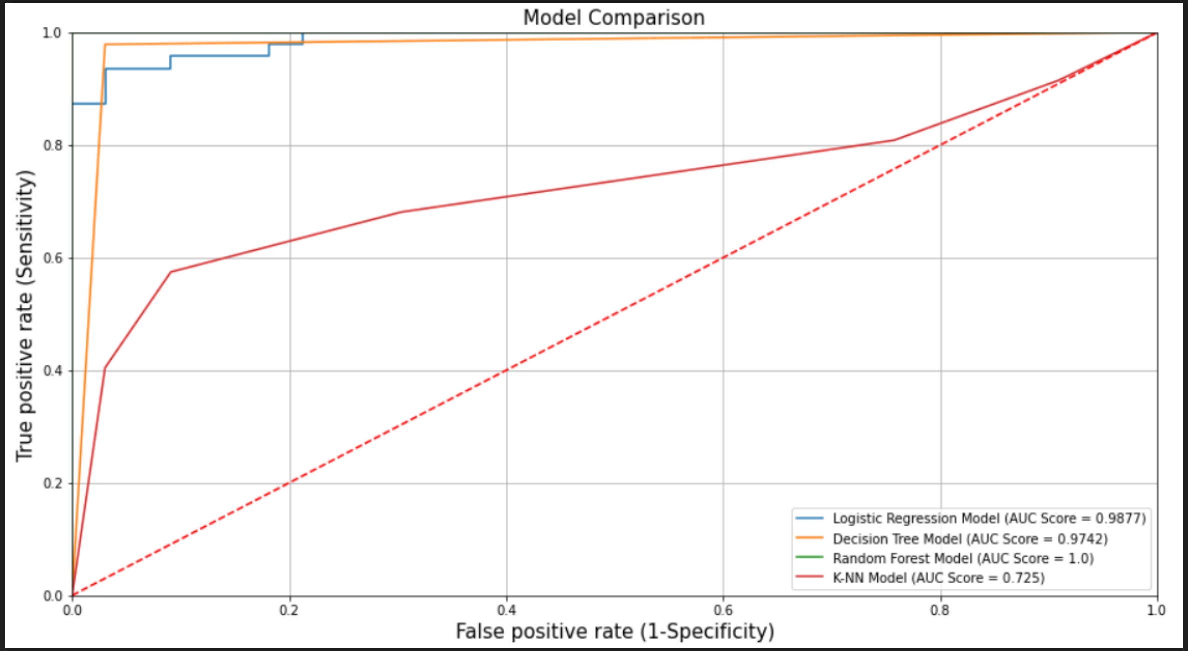
```
--> Train Report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97       121
           1       1.00      0.96      0.98       199

    accuracy                           0.98       320
   macro avg       0.97      0.98      0.98       320
weighted avg       0.98      0.98      0.98       320

ROC-AUC-Score: 0.9824120603015075


--> Test Report:
              precision    recall  f1-score   support

           0       0.88      1.00      0.94        29
           1       1.00      0.92      0.96        51

    accuracy                           0.95        80
   macro avg       0.94      0.96      0.95        80
weighted avg       0.96      0.95      0.95        80

ROC-AUC-Score: 0.9607843137254901
```



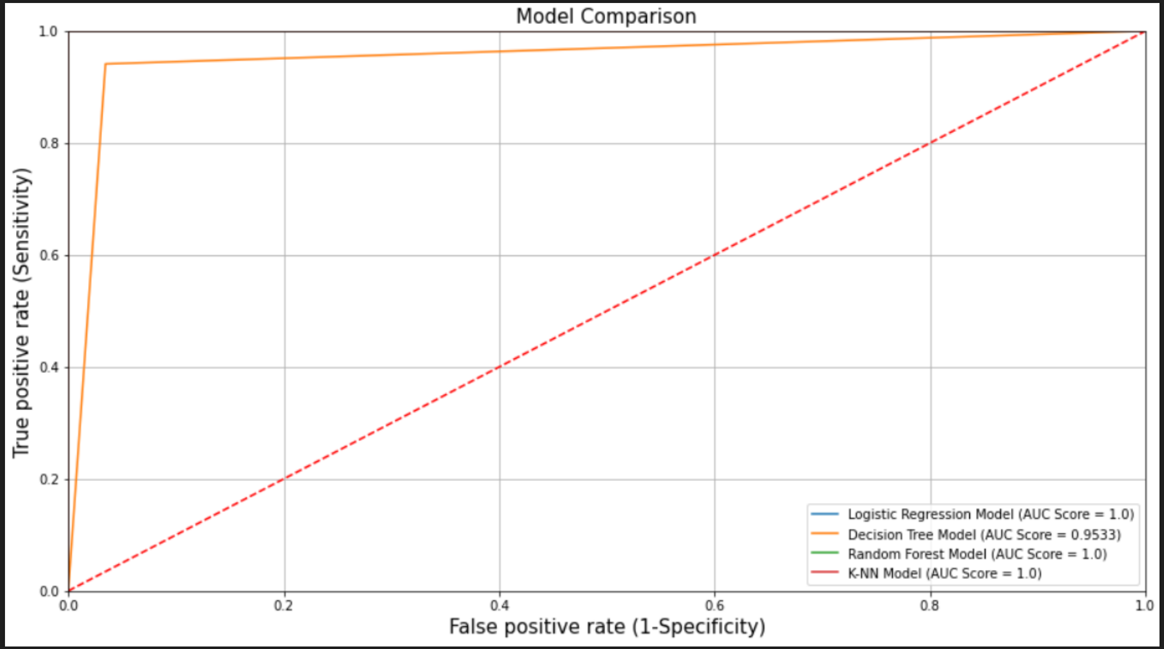
Important Point

- As this is a medical scenario, we should aim to minimize the False Positives
- Why?
  - Lets say a person has a Chronic Kidney Disease in reality but the model predicts that he is healthy, then the person will not be treated and his condition may get severe. This is False Positive.
  - On the other hand, if a person is healthy and the model predicts that he has a Chronic Kidney Disease, then this situation is acceptable/better than the previous situation. This is False Negative.

AUC-ROC Curve



Before Scaling



After Scaling

Conclusion

- From the Classification Report, Confusion Matrix and the ROC Plot we can see that, after scaling the data, the performance of Logistic Regression and KNN drastically increased.

- Best Models:
  - Random Forest (Same in both)
  - Logistic Regression (After Scaling)
  - KNN (After Scaling)
  - Decision Tree (Before Scaling)

- If any one model is to be chosen as final one, then Random Forest or Logistic Regression should be chosen.

Thank You
Stay Safe!!!