Research Article

Chronic Kidney Disease Prediction Using Machine Learning Algorithms

Rishita Agarwal

Prof. Manish Gupta

School of Engineering and Technology, Amity University, Madhya Pradesh

Assistant Professor, School of Engineering and Technology, Amity University, Madhya Pradesh

ABSTRACT

The field of biosciences have advanced to a larger extent and have generated large amounts of information from Electronic Health Records. This have given rise to the acute need of knowledge generation from this enormous amount of data. Data mining methods and machine learning play a major role in this aspect of biosciences. Chronic Kidney Disease (CKD) is a condition in which the kidneys are damaged and cannot filter blood as they always do. A family history of kidney diseases or failure, high blood pressure, type 2 diabetes may lead to CKD. This is a lasting damage to the kidney and chances of getting worser by time is high. The very common complications that results due to a kidney failure are heart diseases, anemia, bone diseases, high potassium and calcium. The worst-case situation leads to complete kidney failure and necessitates kidney transplant to live. An early detection of CKD can improve the quality of life to a greater extent. This calls for good prediction algorithm to predict CKD at an earlier stage. Literature shows a wide range of machine learning algorithms employed for the prediction of CKD. This paper uses data preprocessing, data transformation and various classifiers to predict CKD and also proposes best Prediction framework for CKD. The results of the framework show promising results of better prediction at an early stage of CKD.

Keywords: Chronic Kidney Disease, Decision Tree, Machine Learning, Random Forest, Support Vectors, Linear Regression, Logistic Regression.

1. Introduction

The disability of the kidneys to perform their regular blood filtering function and others is called Chronic Kidney Disease (CKD). The term “chronic” describes the slow degradation of the kidney cells over a long period of time. This disease is a major kidney failure where the kidney sans blood filtering process and there is a heavy fluid buildup in the body. This leads to alarming increase of potassium and calcium salts in the body. Existence of high levels of these salts result in various other ailments in the body. The prime job of kidneys is to filter extra water and wastes from blood. The efficient functioning of this process is important to balance the salts and minerals present in our body. The high balance of salts is necessary to control blood pressure, activate hormones, build red blood cells, etc. A high concentration of calcium leads to various bone diseases and cystic ovaries in women. CKD also may lead to sudden illness or allergy to certain medicines. This state is called as Acute Kidney Injury (AKI). An increased blood pressure may lead to heart problems and heart attacks. CKD in many cases leads to permanent dialysis or kidney transplants. A history of kidney disease in the family also leads to high probability of CKD. Literature shows that almost one out of three people diagnosed with diabetes have CKD. Literature also presents evidences of early identification and care of CKD can improve the quality of the patient’s life.

With the availability of biomedical data, the use of machine-learning techniques in healthcare for developing disease prediction models has become common. Further, methods such as deep learning and techniques like ensemble learning have greatly improved the predictive power of machine learning models. By deriving features from Electronic Health Records (EHR), accurate disease prediction models can be developed [5,6]. At the patient level, a physician can assess the onset of CKD using laboratory tests by looking at standard parameters such as the glomerular filtration rate (eGFR) and the albumincreatinine ratio [7]. On the other hand, from the public health perspective, laboratory data is typically not available on a large scale. However, two types of data can generally be extracted from the insurance companies’ databases: diagnoses and medications for each patient’s visit at the hospital. Prediction algorithms in machine learning can be intelligently used to predict the occurrence of CKD and presents a method of early medication. The detailed review on literature shows the application of various machine learning algorithms to predict CKD.

The rest of the paper is divided into four sections. Section1 consists of the introduction, Section 2 consists of the methodology used, Section 3 consists of the proposed System, Section 4 consists of the results and data analysis, and Section 5 consists of conclusion.

2. Methodology

Description of the Dataset. The dataset used for this research purpose was the Public Health Dataset and it is dating from 2015. It contains 25 attributes. It is integer-valued 0 = no disease and 1 = disease. Now the attributes which are used in this research purpose are described as follows and for what they are used or resemble:

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

3. Proposed System

This part describes the dataset and contains block diagrams, flow diagrams, evaluation matrices, and the study's procedure and methodology.

Figure 1 depicts the block diagram of the proposed system. The framework utilizes the CKD prediction dataset. After pre-processing and feature selection, the DT, KNN, and logistic regression algorithms have been used. All the components of this diagram have been discussed in the following sub sections.

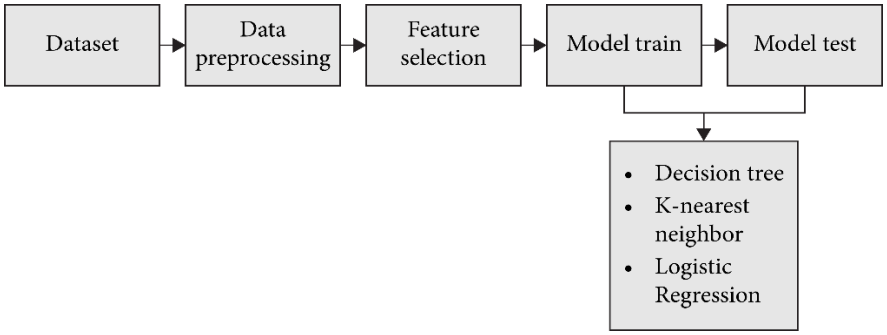

Fig 1: - Block Diagram for the proposed system

3.1 Datasets

The research was conducted using the CKD dataset. There are 400 rows and 26 columns in this dataset. The output column "class" has a value of either "1" or "0." The value "0" indicates that the patient is not a CKD patient, while the value "1" shows that the patient is a CKD patient. Before preprocessing, Figure 2 displays the total number of CKD and non-CKD entries in the output column. The overall number of CKD data is 248, whereas the total number of non-CKD data is 150.

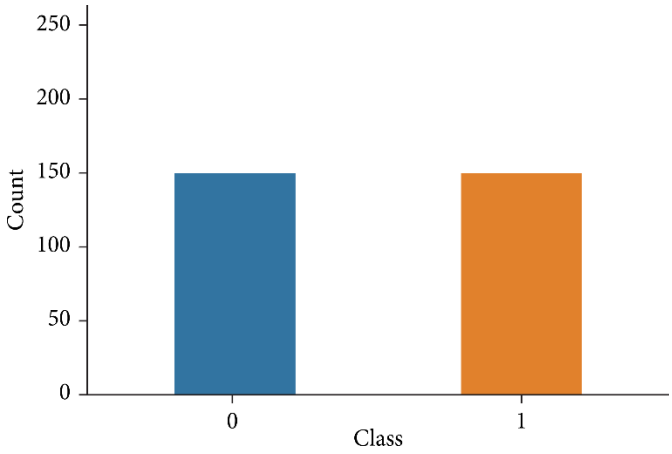

Fig 2: Total amount of CKD and Not-CKD data.

a. Data Preprocessing

Prior to model building, data preprocessing is required to remove unwanted noise and outliers from the dataset that might cause the model to diverge from the proper training set. This stage tackles anything that is impeding the model's efficiency. After collecting the necessary data, it must be cleaned and prepared for model construction. The dataset is next searched for null values. However, this dataset contains no null values. Figure 3 shows there is no missing data available in this dataset.

```
In [29]: df.isna().sum()

Out[29]: id                0
         age               0
         bp                0
         sg                0
         al                0
         su                0
         rbc               0
         pc                0
         pcc               0
         ba                0
         bgr               0
         bu                0
         sc                0
         sod               0
         pot               0
         hemo              0
         pcv               0
         wc                0
         rc                0
         htn               0
         dm                0
         cad               0
         appet             0
         pe                0
         ane               0
         classification    0
         dtype: int64
```

Fig 3: No missing data

Here, the output values "0" indicate the absence of null values. After completing data preparation and handling the unbalanced dataset, the next step is to build the model. To increase the accuracy and efficiency of this task, the data is split into training and testing segments, with an 80/20 ratio of training to testing. Following the model's splitting, it is trained using a number of classification techniques. The classification methods used in this research include the Linear Regression, Logistic Regression, Decision tree classification method, SVM, Random Forest and K-Neighbor Classifier.

b. Feature Selection

In this, the absolute values of the correlations between features and the class label show that blood pressure, albumin, sugar, blood urea, serum creatinine, potassium, white blood cell count, and hypertension all have positive links. Figures 4 show the feature correlation value.

```
In [5]: df.corr()
```

Out[5]:

	id	age	bp	sg	al	su	bgr	bu	sc	sod	pot	hemo
id	1.000000	-0.185308	-0.245744	0.642156	-0.541993	-0.283416	-0.338673	-0.307175	-0.268683	0.364251	-0.092347	0.640298
age	-0.185308	1.000000	0.159480	-0.191096	0.122091	0.220866	0.244992	0.196985	0.132531	-0.100046	0.058377	-0.192928
bp	-0.245744	0.159480	1.000000	-0.218836	0.160689	0.222576	0.160193	0.188517	0.146222	-0.116422	0.075151	-0.306540
sg	0.642156	-0.191096	-0.218836	1.000000	-0.469760	-0.296234	-0.374710	-0.314295	-0.361473	0.412190	-0.072787	0.602582
al	-0.541993	0.122091	0.160689	-0.469760	1.000000	0.269305	0.379464	0.453528	0.399198	-0.459896	0.129038	-0.634632
su	-0.283416	0.220866	0.222576	-0.296234	0.269305	1.000000	0.717827	0.168583	0.223244	-0.131776	0.219450	-0.224775
bgr	-0.338673	0.244992	0.160193	-0.374710	0.379464	0.717827	1.000000	0.143322	0.114875	-0.267848	0.066966	-0.306189
bu	-0.307175	0.196985	0.188517	-0.314295	0.453528	0.168583	0.143322	1.000000	0.586368	-0.323054	0.357049	-0.610360
sc	-0.268683	0.132531	0.146222	-0.361473	0.399198	0.223244	0.114875	0.586368	1.000000	-0.690158	0.326107	-0.401670
sod	0.364251	-0.100046	-0.116422	0.412190	-0.459896	-0.131776	-0.267848	-0.323054	-0.690158	1.000000	0.097887	0.365183
pot	-0.092347	0.058377	0.075151	-0.072787	0.129038	0.219450	0.066966	0.357049	0.326107	0.097887	1.000000	-0.133746
hemo	0.640298	-0.192928	-0.306540	0.602582	-0.634632	-0.224775	-0.306189	-0.610360	-0.401670	0.365183	-0.133746	1.000000

```
In [ ]:
```

Fig 4: Feature Correlation values

All the positively correlated features are considered for further prediction. Each albumin molecule has just five distinct sets of values. The quantity of albumin is assessed using a urine protein test. A high protein level in the urine means that the filtration units in the kidneys have been damaged by disease, fever, or intense activity. Numerous tests should be performed over many weeks to establish the diagnosis. The term serum creatinine is used interchangeably with blood creatinine and creatinine. Creatinine is the byproduct of muscle breakdown of the chemical creatine. The kidneys eliminate creatinine from the body. This test is done to find out how much creatinine is in your blood. Creatine is an element of the metabolic cycle that produces the energy needed for muscle contraction. The body produces both creatine and creatinine at the same rate. Creatinine levels in the blood can rise due to a high-protein diet, congestive heart failure, diabetic issues, and dehydration, among other factors. Creatinine levels in women should be between 0.6 and 1.1 mg/dL, while those in males should be between 0.7 and 1.3 mg/dL. Additionally, hypertension, or high blood pressure, develops whenever blood pressure against the walls of blood vessels rises. Hypertension can lead to heart attacks, strokes, and chronic kidney disease if it is not treated or managed properly. Nonetheless, CKD may result in hypertension.

c. Algorithms

The following machine learning algorithms have been used to predict chronic kidney disease: -

a) Linear Regression
b) Logistic Regression
c) Support Vector Machine (SVM)
d) Decision Tree
e) Random Forest
f) K-Neighbor Classifier

3.1 (a) Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Fig 5 shows the simple Linear Regression in Machine learning.

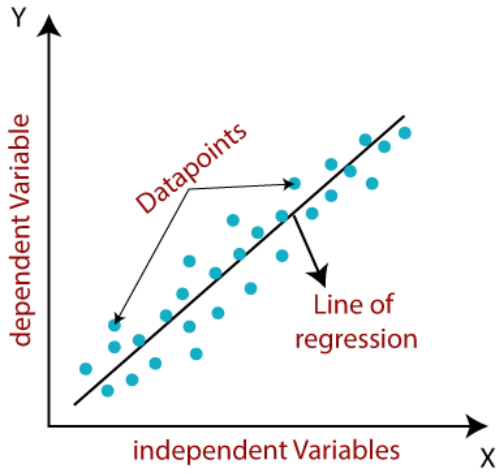

Fig 5: - Linear Regression in Machine Learning

3.1 (b) Logistic Regression

Binary outcomes are modelled using the statistical method of logistic regression, which is well known in the field. Different learning methods are used to execute logistic regression in statistical research. A variant of the neural network method was used to create the LR algorithm. This method resembles neural networks in

many ways, but it is simpler to set up and use. Figure 6 shows the block diagram of Logistic Regression.

Utilizing logistic regression, the output of a categorical dependent variable is predicted. So, the output must be discrete or categorical. It may be yes or no, 0 or 1, true or false, etc., but probability values between 0 and 1 are given. Logistic regression and linear regression are used in very similar ways. Classification problems are addressed with logistic regression, and regression problems are addressed using linear regression. Instead of a regression line, we use an "S" shaped logistic function that predicts two maximum values (0 or 1). The logistic function's curve indicates the probability of anything, such as whether cells are malignant or not, or if an animal is fat or not. Since it can classify new data using both discrete and continuous datasets, logistic regression is a common ML technique.

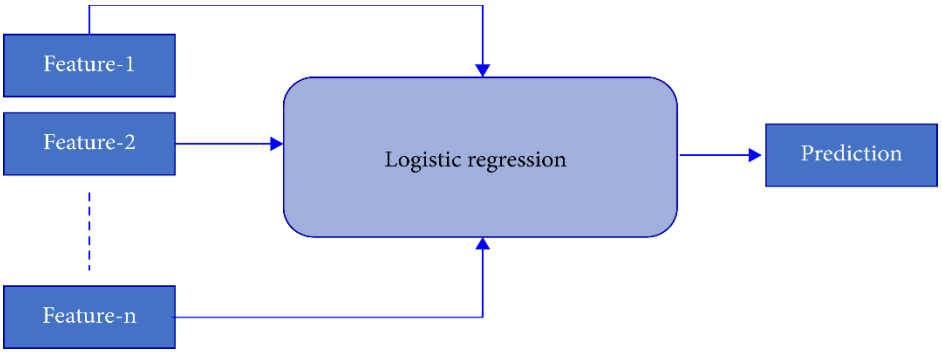

Fig 6: - Block Diagram of logistic Regression

3.1 (c) Support Vector Machine (SVM)

A linear model for classification and regression is Support Vector Machine (SVM) that can be used to solve both linear and non-linear problems. The algorithm classifies data using a hyperplane. In this algorithm, each data item will be plotted as a point in n-dimensional space (where n is the number of features) with the value of each feature being the value of a particular coordinate. Classification will be performed by finding the right hyper-plane which can differentiate the two classes efficiently. Fig 7 shows SVM Algorithm.

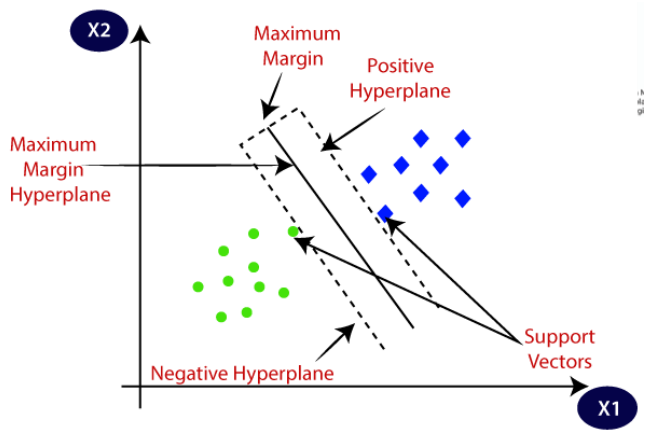

Fig 7: - Support Vector Machine Algorithm

3.1 (d) Decision Tree Classifier

The DT method is a classification and regression technique that can be used to predict both discrete and continuous characteristics. Based on the links between input columns in a dataset, the algorithm predicts discrete characteristics. It predicts the states of a column that you identify as predictable using the values of those columns, known as states. The method specifically finds the input columns that are associated with the predicted column. The DT classifier's block diagram is shown in Figure 8.

The decision tree is easy to comprehend since it replicates the phases that a person goes through while making a real-life decision. It may be quite useful in dealing with decision-making issues. It is a good idea to consider all potential solutions to an issue. Cleaning data is not as important as it is with other methods.

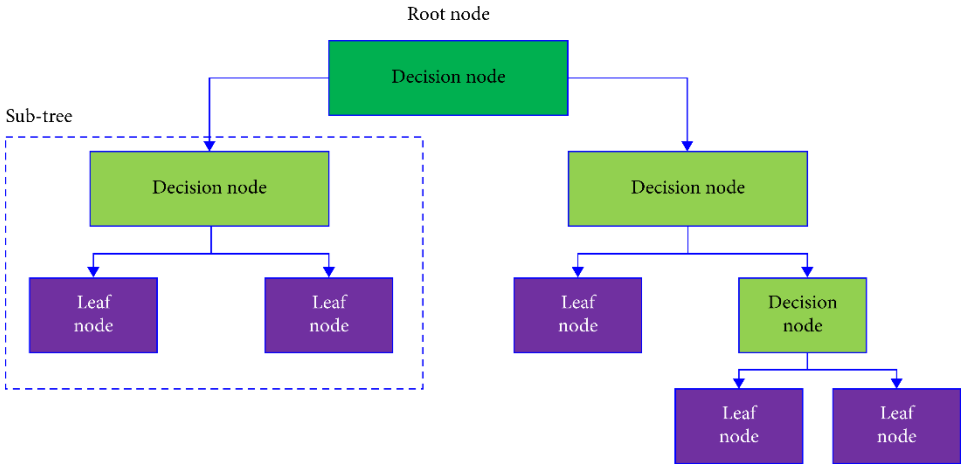

Fig 8: - Block Diagram of Decision Tree Classifier

3.1 (e) Random Forest Regressor

Random forest algorithm constructs multiple decision trees to act as an ensemble of classification and regression process. A number of decision trees are constructed using a random subset of the training data sets. A large collection of decision trees provides higher accuracy of results. The runtime of the algorithm is comparatively fast and also accommodates missing data. Random forest randomizes the algorithm and not the training data set. The decision class is the mode of classes generated by decision trees. Fig 9 shows Random Forest.

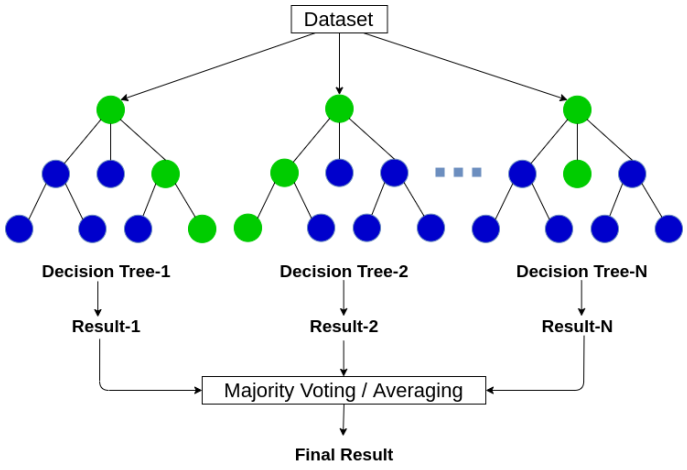

Fig 9: - Random Forest Regressor

3.1 (f) K-Neighbor Classifier (KNN)

Figure 10 depicts the whole KNN model's flowchart. One of the simplest ML algorithms is KNN, which uses the supervised learning approach. A new case is assigned to a category based on how closely it resembles prior categories. This is known as the KNN technique. With the KNN method, you can store all the data you have and then classify new data based on how similar it is to the old. This suggests that the KNN technique can rapidly classify new data into well-defined categories. Though it is often utilized for classification problems, the KNN method may be used for regression as well. There are no data assumptions made by the KNN technique, which is nonparametric and also called a "lazy learner algorithm," since it does not instantly learn from the training set but rather keeps and categorizes the data for later. If it receives new data, the KNN classifies it into a category that is quite close to the new data that was stored during training.

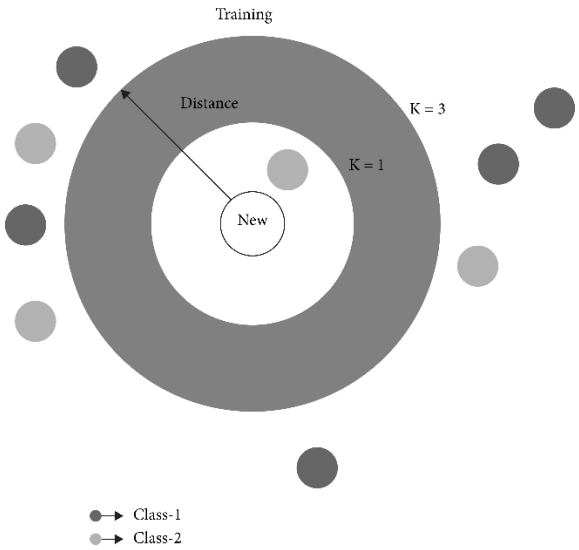

Fig 10: - KNN Classifier Working Procedure

3.1 (g) Confusion Matrix

Figure 11 shows the confusion matrix. The confusion matrix rates machine learning classification models' performance. All models were evaluated using the confusion matrix. The confusion matrix illustrates how often our models guess correctly and incorrectly. Poorly predicted values received false positives and negatives, whereas properly predicted values received genuine positives and negatives. The model's accuracy, precision-recall trade-off, and AUC were assessed after grouping all predicted values in the matrix.

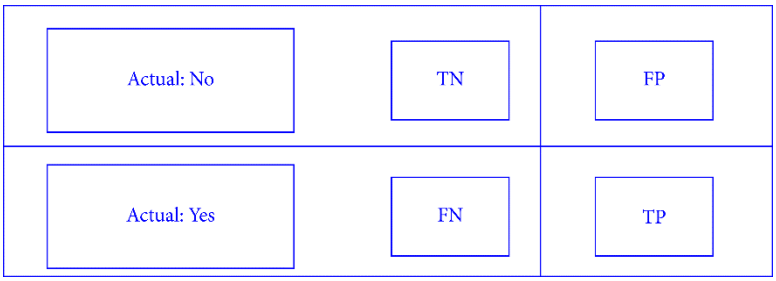

Fig 11: - Confusion Matrix Block Diagram

4. Result and Data Analysis

4.1 Linear Regression

Figure 12 shows the Linear Regression accuracy. In this case, the accuracy is 73% percent.

```
In [55]: regression_model.score(x_train,y_train) # 95% accuracy in training data using linear regression

Out[55]: 0.8182960816008278

In [56]: regression_model.score(x_test,y_test) # 75% accuracy in testing the data

Out[56]: 0.7300448634354514
```

Fig 12: - Accuracy of Linear Regression

4.2 Logistic Regression

Figure 13 shows the Logistic Regression model score. In this case, the accuracy is 73% percent.

In [77]:
```
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
#fit the model om train
model = LogisticRegression(solver = "liblinear")
model.fit(x_train,y_train)
```

```
C:\Users\HP\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversion
es, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[77]:
```
LogisticRegression(solver='liblinear')
```

In [78]:
```
y_predict = model.predict(x_test)
y_predict
```

Out[78]:
```
array(['1', '1', '1', '1', '0', '1', '1', '1', '0', '1', '1', '1', '1',
       '0', '1', '1', '1', '0', '0', '1', '0', '0', '1', '0', '1', '0',
       '1', '1', '0', '1', '1', '0', '1', '1', '1', '1', '0', '1', '1',
       '0', '1', '0', '1', '1', '1', '0', '1', '0', '0', '1', '1', '1',
       '1', '0', '1', '1', '1', '0', '0', '1', '1', '0', '0', '1', '1',
       '1', '1', '0', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1',
       '0', '1', '1', '0', '1', '0', '1', '0', '0', '1', '0', '0', '1',
       '1', '0', '0', '1', '1', '1', '1', '1', '0', '0', '0', '1', '1',
       '1', '1', '1', '1', '1', '1', '0', '1', '0', '1', '0', '1', '1',
       '0', '0', '1'], dtype=object)
```

In [79]:
```
model_score = model.score(x_test,y_test)
print(model_score)
```

```
1.0
```

Fig 13: - Model Score of LR

4.3 Support Vector Machine (SVM)

Fig 14 shows the accuracy of the model SVM. In this case, the accuracy is 73% percent.

```
In [80]: from sklearn import svm
         model = svm.SVC(C = 3)

In [81]: model.fit(x_train,y_train)

         C:\Users\HP\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A
         es, ), for example using ravel().
           return f(*args, **kwargs)
Out[81]: SVC(C=3)

In [82]: y_pred = model.predict(x_test)

In [83]: y_pred

Out[83]: array(['1', '1', '1', '0', '0', '1', '1', '0', '0', '0', '1', '0', '1',
                '1', '1', '1', '1', '0', '0', '0', '0', '0', '0', '0', '1', '0',
                '1', '1', '0', '1', '1', '0', '1', '0', '0', '1', '0', '1', '1',
                '0', '0', '1', '1', '1', '0', '0', '1', '1', '0', '0', '1', '0',
                '1', '1', '1', '1', '1', '1', '0', '1', '1', '1', '0', '1', '1',
                '1', '1', '1', '0', '1', '1', '1', '1', '1', '0', '1', '1', '1',
                '0', '1', '1', '0', '0', '0', '1', '1', '0', '1', '0', '0', '1',
                '1', '0', '0', '1', '1', '1', '1', '1', '0', '0', '1', '1', '1',
                '1', '1', '0', '1', '1', '1', '1', '1', '0', '1', '1', '0', '1',
                '1', '1', '1'], dtype=object)

In [84]: model.score(x_test,y_test)

Out[84]: 0.7333333333333333
```

Fig 14: - Accuracy of SVM

4.4 Decision Tree Classifier

Fig 15 shows the classification report of the model using Decision Tree Classifier Algorithm. In this case, the accuracy is 100 percent.

Fig 16 shows the Confusion Matrix of DT.

```
In [71]: from sklearn.tree import DecisionTreeClassifier
         regressor = DecisionTreeClassifier(random_state = 1) # initialization

In [72]: regressor.fit(x_train,y_train) # fitting training data

Out[72]: DecisionTreeClassifier(random_state=1)

In [73]: regressor.score(x_test,y_test)

Out[73]: 1.0

In [74]: y_predi = regressor.predict(x_test)

In [75]: from sklearn.metrics import classification_report
         print(classification_report(y_test, y_predi, labels=[1, 2, 3]))
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	78
2	0.00	0.00	0.00	0
3	0.00	0.00	0.00	0
micro avg	1.00	1.00	1.00	78
macro avg	0.33	0.33	0.33	78
weighted avg	1.00	1.00	1.00	78

Fig 15: - Classification Report of DT

```
In [76]: from sklearn.metrics import confusion_matrix
         print("confusion matrix")
         cm = confusion_matrix(y_test,y_predi)
         df_cm = pd.DataFrame(cm, index=(i for i in["notCKD","CKD"]),columns = (i for i in ["Predicted notCKD","Predicted CKD"]))
         sns.heatmap(df_cm, annot=True)
         plt.show()
```

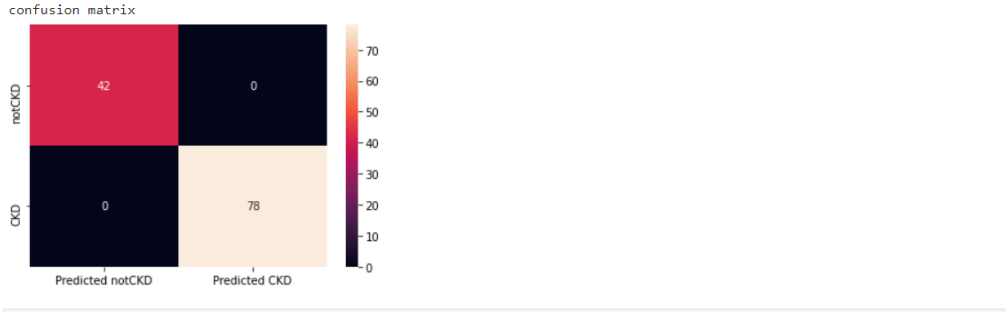

Fig 16: - Confusion Matrix of DT

4.5 Random Forest Regressor

Accuracy of the model using Random Forest Regressor is 99 percent. This is depicted in fig 17.

```
In [59]: from sklearn.ensemble import RandomForestRegressor
         regression_model1 = RandomForestRegressor(n_estimators = 100, random_state = 1) # Initialisation
         regression_model1.fit(x_train,y_train) # fitting this Regression to train data

         <ipython-input-59-e7d3cfba0be7>:3: DataConversionWarning: A column-vector y was passed when a 1d array was expect
           regression_model1.fit(x_train,y_train) # fitting this Regression to train data
Out[59]: RandomForestRegressor(random_state=1)

In [60]: regression_model1.score(x_train,y_train) # 98% accuracy in training data using Random Forest Regression

Out[60]: 0.9976500861326443

In [61]: regression_model1.score(x_test,y_test) # 86% accuracy in testing data using Random Forest Regression

Out[61]: 0.9992820512820513
```

Fig 17: - Accuracy of Random Forest Regressor

4.6 K-Neighbor Classifier (KNN)

The accuracy of the model using KNN is 92 percent. The classification report of KNN is shown in fig 18.

```
In [66]: from sklearn.neighbors import KNeighborsRegressor
         KNN = KNeighborsRegressor(n_neighbors=3) # Initialization of KNR
         KNN.fit(x_train, y_train) # fitting training data

Out[66]: KNeighborsRegressor(n_neighbors=3)

In [ ]:

In [67]: KNN = KNeighborsClassifier(n_neighbors = 1, weights = "distance")
         KNN.fit(x_train,y_train)
         KNN.score(x_test,y_test)

         C:\Users\HP\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:179: DataCon
         o (n_samples,), for example using ravel().
           return self._fit(X, y)
Out[67]: 0.925

In [68]: y_pre = KNN.predict(x_test)

In [69]: from sklearn.metrics import classification_report
         print(classification_report(y_test, y_pre, labels=[1, 2, 3]))

                       precision    recall  f1-score   support

                    1       0.96      0.92      0.94        78
                    2       0.00      0.00      0.00         0
                    3       0.00      0.00      0.00         0

            micro avg       0.96      0.92      0.94        78
            macro avg       0.32      0.31      0.31        78
         weighted avg       0.96      0.92      0.94        78
```

Fig 18: - Accuracy and Classification report of KNN

Confusion Matrix of KNN is depicted in figure 19.

```
In [70]:
from sklearn.metrics import confusion_matrix
print("confusion matrix")
cm = confusion_matrix(y_test,y_pre)
df_cm = pd.DataFrame(cm, index=(i for i in["notCKD","CKD"]),columns = (i for i in ["Predicted notCKD","Predicted CKD"]))
sns.heatmap(df_cm, annot=True)
plt.show()
```

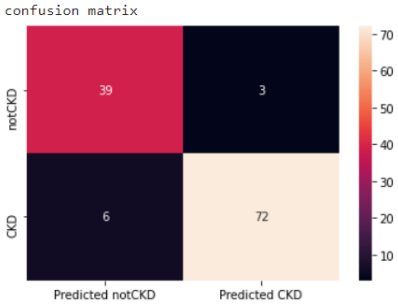

Fig 19: - Confusion Matrix of KNN

4.7 Model Conclusion

Out[85]:

	Regression	Accuracy
0	Linear Regression	0.7346691161495181
1	Logistic Regression	0.7333333333333333
2	SVM	0.7333333333333333
3	Random Forest	0.9992820512820513
4	Decision Tree	1.0
5	KNN	0.925

Fig 20: - Accuracy of different models

The chart clearly indicates that Decision Tree is the best among the many models in the framework. Using Decision Tree, this paper achieved 100 percent accuracy. The Random Forest Regressor also achieved good accuracy i.e., 99 percent. But the same paper is achieving the poor quality by using SVM i.e., 73 percent. Fig.20 and Fig.21 shows the accuracy of different models respectively.

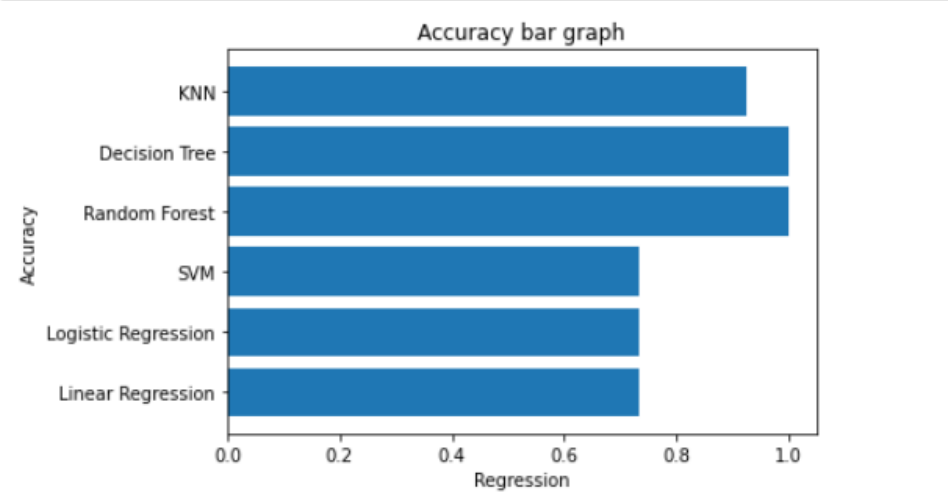

Fig 21: - Bar-graph showing the accuracy of different models

5. Conclusion

According to the findings of the study, the Random Forest Regressor approach and Decision Tree can be used to predict chronic kidney disease more accurately. According to the study, their precision was 99 percent, and their accuracy was 100 percent. Compared to prior research, the accuracy percent of the models used in this investigation is considerably higher, indicating that the models used in this study are more reliable than those used in previous studies. When cross validation measurements are used in the prediction of chronic kidney disease, the Random Forest Regressor method outperforms the other processes. Future research may build on this work by developing a web application that incorporates these algorithms and using a bigger dataset than the one utilized in this study. This will aid in the achievement of improved outcomes as well as the accuracy and efficiency with which healthcare practitioners can anticipate kidney issues. This will enhance the dependability of the framework as well as the framework's presentation. The hope is that it would encourage people to seek early treatment for chronic renal disease and to make improvements in their lives.

References

1. A. S. Levey, R. Atkins, J. Coresh et al., "Chronic kidney disease as a global public health problem: approaches and initiatives–a position statement from kidney disease improving global outcomes," *Kidney International*, vol. 72, no. 3, pp. 247–259, 2007.View at: Publisher Site | Google Scholar

2. V. Jha, G. Garcia-Garcia, K. Iseki et al., "Chronic kidney disease: global dimension and perspectives," *The Lancet*, vol. 382, no. 9888, pp. 260–272, 2013.View at: Publisher Site | Google Scholar

3. N. R. Hill, S. T. Fatoba, J. L. Oke et al., "Global prevalence of chronic kidney disease – a systematic review and meta-analysis," *PLoS One*, vol. 11, no. 7, article e0158765, 2016.View at: Publisher Site | Google Scholar

4. H. Nasri, "World kidney day 2014; chronic kidney disease and aging: a global health alert," *Iranian Journal of Public Health*, vol. 43, no. 1, pp. 126-127, 2014.View at: Google Scholar

5. G. Abraham, S. Varughese, T. Thandavan et al., "Chronic kidney disease hotspots in developing countries in South Asia," *Clinical Kidney Journal*, vol. 9, no. 1, pp. 135–141, 2016.View at: Publisher Site | Google Scholar

6. K. T. Mills, T. Xu, W. Zhang et al., "A systematic analysis of worldwide population-based data on the global burden of chronic kidney disease in 2010," *Kidney International*, vol. 88, no. 5, pp. 950–957, 2015.View at: Publisher Site | Google Scholar

7. B. Ene-Iordache, N. Perico, B. Bikbov et al., "Chronic kidney disease and cardiovascular risk in six regions of the world (ISN-KDDC): a cross-sectional study," *The Lancet Global Health*, vol. 4, no. 5, pp. e307–e319, 2016.View at: Publisher Site | Google Scholar

8. S. Anand, M. A. Khanam, J. Saquib et al., "High prevalence of chronic kidney disease in a community survey of urban Bangladeshis: a cross-sectional study," *Glob Health*, vol. 10, no. 1, p. 9, 2014.View at: Publisher Site | Google Scholar

9. L. Ali, K. Fatema, Z. Abedin et al., "Screening for chronic kidney diseases among an adult population," *Saudi Journal of Kidney Diseases and Transplantation*, vol. 24, no. 3, p. 534, 2013.View at: Publisher Site | Google Scholar.