A

Mini Project Report On

"CHRONIC KIDNEY DISEASE PREDICTION"

SUBMITTED TO:

Institute of Engineering and Technology,
Lucknow – 226021

Submitted By:
ROHIT TIWARI [2100520109002]
SAMARTH GAUTAM [2000520100050]
SHIVAM PANDEY [2000520100061]

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Institute of Engineering and Technology, Lucknow

Institute of Engineering and Technology, Lucknow

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFRICATE

This is to certify that this project report "Chronic Kidney Disease Prediction" entitled submitted by ***ROHIT TIWARI, SAMARTH GAUTAM AND SHIVAM PANDEY*** is accepted in partial fulfillment of Bachelors of Engineering in Computer Science and Engineering.

Head of Department
Dr. Y. N. Singh

DECLARATION

We hereby declare that the Thesis Titled **"CHRONIC KIDNEY DISEASE PREDICTION"** is submitted to the Department of Computer Science and Engineering of Institute of Engineering and Technology in Partial fulfillment of B tech in CSE. This is my original work and not submitted elsewhere for the award of any other degree or any other publication.

DATE:

Dr. Y. N. Singh

Head of Department

ROHIT TIWARI

SAMARTH GAUTAM

SHIVAM PANDEY

Acknowledgements

We would like to thank several individuals for their support, motivation, and inspiration white writing this thesis. Only through their constant financial, intellectual and emotional support could this work have been completed.

First and foremost, I would like to take this opportunity to thank our HOD **Dr. Y. N. Singh** for this guidance and advice on this project. At the same time, I also won't forget my friends because they quite good with sharing some of their information to complete this Minor Project successfully. Last but not least, I am very grateful to our college, lecture and friends where they gave us enough time to complete this project and at the same time, I would like to thank my friends and classmates who help me a lot to complete this project.

Thank You.

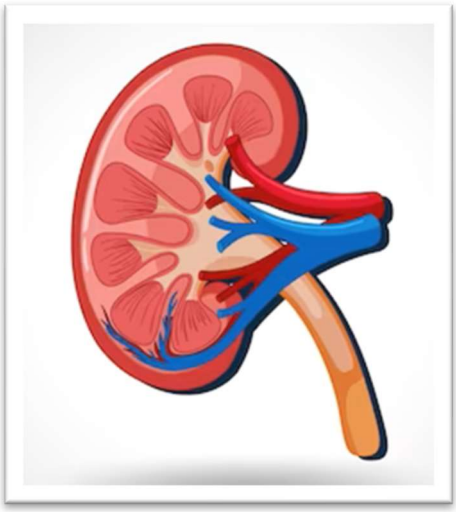
ABSTRACT

This project presents a Kidney Disease Prediction system using Machine Learning technology that can be Accessed from anywhere by an app/website. It is designed to predict the possibility of a Chronic Disease at any period of time. The model is trained with a The GPS continuously takes input data from the satellite and updates the latitude and longitude values on things peak server. This basically means that if a person has to track a vehicle, he needs to power up the tracking device and place in that vehicle, than he can easily track the vehicle just by logging on the app/website. These systems continuously watch a moving vehicle and update the location coordinates with time.

TABLE OF CONTENTS

INTRODUCTION

Chronic kidney disease (CKD) means your kidneys are damaged and can't filter blood the way they should. The disease is called "chronic" because the damage to your kidneys happens slowly over a long period of time. This damage can cause wastes to build up in your body. CKD can also cause other health problems.



Chronic kidney disease includes conditions that damage your kidneys and decrease their ability to keep you healthy by filtering wastes from your blood. If kidney disease worsens, wastes can build to high levels in your blood and make you feel sick. You may develop complications like:

- high blood pressure
- anemia (low blood count)
- weak bones
- poor nutritional health
- nerve damage

Anyone can get chronic kidney disease at any age. However, some people are more likely than others to develop kidney disease. You may have an increased risk for kidney disease if you:

- have diabetes
- have high blood pressure
- have a family history of kidney failure
- are older
- belong to a population group with a high rate of diabetes or high blood pressure, such as African Americans, Hispanic Americans, Asian, Pacific Islanders, and American Indians

OBJECTIVE

We will be going through the chronic kidney disease 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.

Problem Statement :
To develop an algorithm which can provide prediction and help in reducing Kidney related diseases using Machine Learning Algorithms.

INPUT:

- Age, Blood Pressure, Albumin, Sugar, Red Blood Cell, Pus Cell, Pus Cell Clumps, Bacteria, Blood Glucose Random, Blood Urea, Serum
- Creatinine, Potassium, White Blood Cell Count, Hypertension, Diabetes Mellitus, Coronary Artery Disease, Pedal Edema, Anemia

The data are blood tests and other measures from patients with and without CKD. There are 400 rows, one per patient; these are patients seen over a period of about two months at some point before July 2015, in a hospital in Tamil Nadu, India; maybe Apollo Reach Karaikudi.

Of the 400 rows, 250 correspond to patients with CKD and the remaining 150 rows correspond to patients without CKD. This information is in the “Class” column of the dataset.

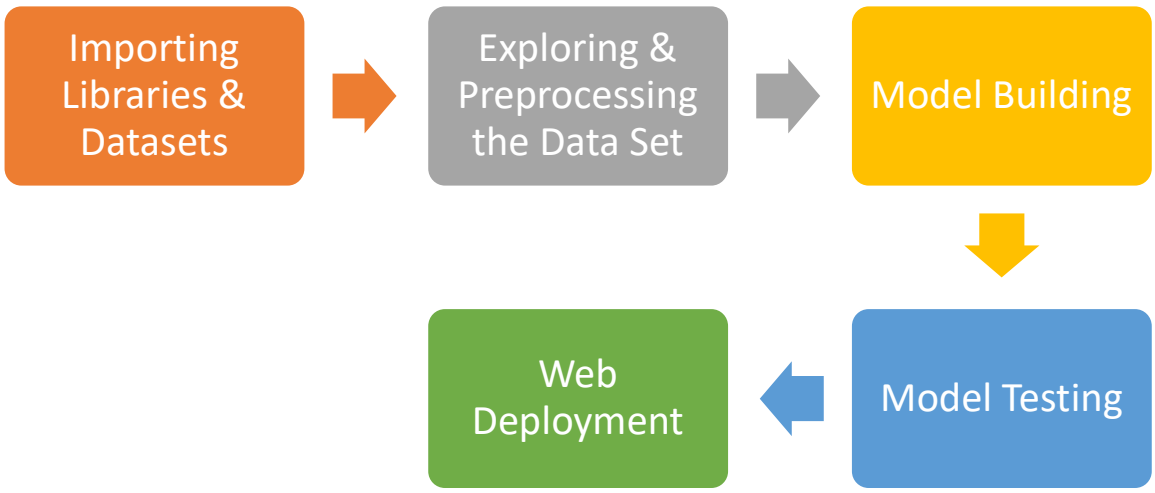
Quick View of Data Set

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd

- Age: age in years
- Blood Pressure: : BP in mm/Hg
- Specific Gravity: one of (1.005,1.010,1.015,1.020,1.025);
- Albumin: one of (0,1,2,3,4,5) (MB: in urine)
- Sugar: one of (0,1,2,3,4,5) (MB: in urine)
- Red Blood Cells: one of (“normal”, “abnormal”)
- Pus Cell: one of (“normal”, “abnormal”)
- Pus Cell clumps: one of (“present”, “notpresent”)
- Bacteria: one of (“present”, “notpresent”)
- Blood Glucose Random: in mgs/dl
- Blood Urea: in mgs/dl
- Serum Creatinine: in mgs/dl
- Sodium: in mEq/L
- Potassium: in mEq/L
- Hemoglobin: in gms
- Packed Cell Volume:
- White Blood Cell Count: in cells/cumm
- Red Blood Cell Count: in millions/cmm
- Hypertension: one of (“yes”, “no”)
- Diabetes Mellitus: one of (“yes”, “no”)
- Coronary Artery Disease: one of (“yes”, “no”)
- Appetite: one of (“good”, “poor”)
- Pedal Edema: one of (“yes”, “no”)
- Anemia: one of (“yes”, “no”)
- Class : one of (“ckd”, “notckd”) in ckd_full.csv) or (1,0) in ckd_clean.csv, where 1 corresponds to “ckd”).

Development

Project Stages



Importing Libraries and Data Sets

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt

        import warnings
        warnings.filterwarnings('ignore')
```

```
In [2]: data = pd.read_csv('kidney_disease.csv')
        data.head()
```

Out[2]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd

5 rows × 26 columns

Exploring & Pre-Processing

Data Exploring

- Create a pandas Data Frame for creating a copy of dataset on which we will carry out further processing.
- head() and tail() to see fist and last five rows of the created data frame.
- describe() to get even further insights on the created data frame.
- df.info to know data type various features present in the dataset.
- Creating Heatmap using seaborn to find correlation between features and labels

Data Pre-Processing

- Imputation of null values.
- Handling typo errors in categorical and numerical columns.
- Converting categorical columns to numerical columns.
- Dropping features having negative correlation with labels.
- Train test split

```
data.isnull().sum()
```

[13]

```
age               9
bp               12
sg               47
al               46
su               49
rbc             152
pc               65
pcc               4
ba                4
bgr              44
bu               19
sc               17
sod              87
pot              88
hemo             52
pcv              70
wc              105
rc              130
htn               2
dm                2
cad               2
appet             1
pe                1
ane               1
classification    0
dtype: int64
```

```
# These columns has blanks which we have to remove
# pcv : ''
# wc  : ''
# rc  : ''
# rc  : '\t?'
# pcv  : '\t?'
# pcv  : '\t43'
# wc  : '\t?'
# wc  : '\t6200'
# wc  : '\t8400'


# data['rc'].unique()
```

Model Building

1. Classification

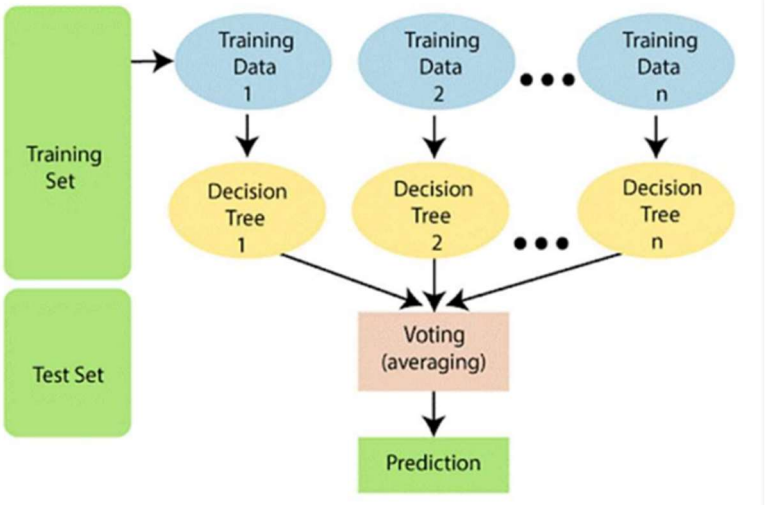
Data mining techniques have been used to define new and understandable patterns to construct classification templates [26]. Supervised and unsupervised learning techniques require the construction of models based on prior analysis and are used in medical and clinical diagnostics for classification and regression [27]. Four popular machine learning algorithms used are SVM, KNN, decision tree, and random forest, which give the best diagnostic results. Machine learning techniques work to build predictive/classification models through two stages: the training phase, in which a model is constructed from a set of training data with the expected outputs, and the validation stage, which estimates the quality of the trained models from the validation dataset without the expected output. All algorithms are supervised algorithms that are used to solve classification and regression problems.

Results:

The random forest algorithm classified all positive and negative samples correctly, as positive samples were correctly classified 250 samples (TP), and all negative samples (TN) were classified for 150 samples correctly. While the SVM, KNN, and Decision Tree algorithms rated the positive (TP) samples by 94.74%, 97.37%, and 98.68%, respectively, that is, with an error (TN) 5.26%, 2.63%, and 1.32%, respectively. Table 6 shows the results obtained from the four classifiers. The random forest algorithm outperformed the rest of the classifiers, reaching an accuracy, precision, recall, and F1-score of 100% for all measures. It was followed by the decision tree algorithm, which reached the accuracy, precision, recall, and F1-score with a score of 99.17%, 100%, 98.68%, and 99.34%, respectively. Then, the KNN algorithm came up with accuracy, precision, recall, and F1-score of 98.33%, 100% 97.37%, and 98.67%, respectively. Finally, the SVM accuracy, precision, recall, and F1-score algorithm scored 96.67%, 92%, 94.74%, and 97.30%, respectively.

Random Forest Algorithm

- It is a Supervised based Ensemble Learning.
- Ensemble Learning uses two methods
    - Bagging: Creating a different training subset from sample training data with replacement is called Bagging. The final output is based on majority voting.
    - Boosting: Combing weak learners into strong learners by creating sequential models such that the final model has the highest accuracy is called Boosting. Example: ADA BOOST, XG BOOST.
- Steps
    - Step 1: Select random samples from a given data or training set.
    - Step 2: This algorithm will construct a decision tree for every training data.
    - Step 3: Voting will take place by averaging(mean of resulting data of each tree) the decision tree.
    - Step 4: Finally, select the most voted prediction result as the final prediction result.



Model Implementation

In this section we will import Random Forest Regression model from Sklearn. Use features identifies from heatmap and label tocreate training and testing set. Finally we will train our model using training set.

1. Use train_test_split() from Sklearn to create train and test sets.

```
In [29]: from sklearn.model_selection import train_test_split

In [30]: X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 42)
```

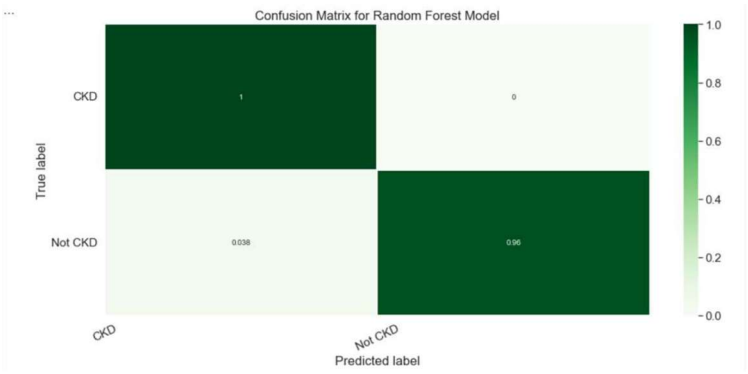
2. Algorithm on Training Data

```
In [31]: from sklearn.ensemble import RandomForestClassifier

In [34]: model = RandomForestClassifier(n_estimators = 20)
         model.fit(X_train, y_train)

Out[34]: RandomForestClassifier(n_estimators=20)
```

Model Testing

In this section we will test our prediction with testing data and calculate Confusion matrix and accuracy score to measure model accuracy.

```
In [35]: from sklearn.metrics import confusion_matrix, accuracy_score

In [36]: confusion_matrix(y_test, model.predict(X_test))

Out[36]: array([[28,  0],
                [ 3, 49]], dtype=int64)

In [37]: print(f"Accuracy is {round(accuracy_score(y_test, model.predict(X_test))*100, 2)}%")

         Accuracy is 96.25%
```



Deploying on Web

We will be using deploy our model on web with the help of Flask and Pickle.

Step 1: Saving the model using pickle library.

```
In [38]: import pickle
         pickle.dump(model, open('../models/kidney.pkl', 'wb'))
```

Step 2: Building a Flask Based App

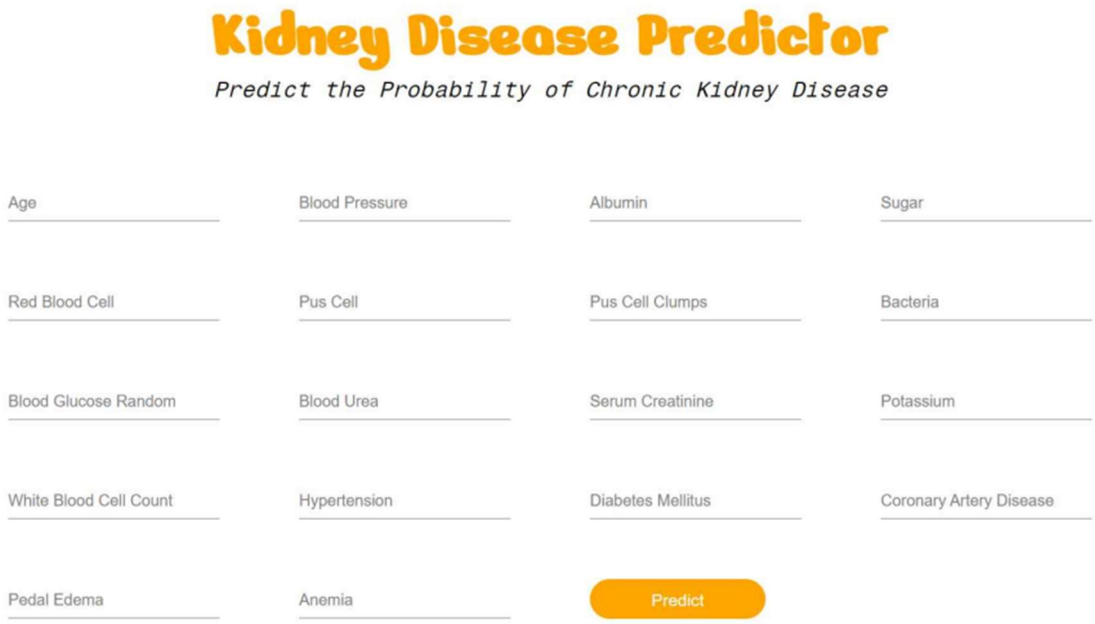
```
app = Flask(__name__)

def predict(values, dic):···

@app.route("/", methods=['GET', 'POST'])
@app.route("/home", methods=['GET', 'POST'])
def kidneyPage():
    return render_template('kidney.html')
```

Step 3: Receiving the Form Data with POST request at Back-End

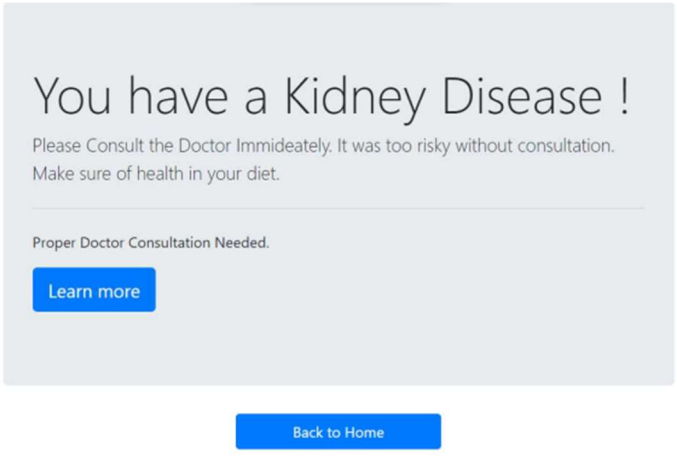
```
@app.route("/predict", methods = ['POST', 'GET'])
def predictPage():
    try:
        if request.method == 'POST':
            to_predict_dict = request.form.to_dict()
            to_predict_list = list(map(float, list(to_predict_dict.values())))
            pred = predict(to_predict_list, to_predict_dict)
    except:
        message = "Please enter valid Data"
        return render_template("home.html", message = message)

    return render_template('predict.html', pred = pred)
```

Web Interface:

: /home



: /predict



Future Scope

The increasing rate of CKD has placed a large negative impact on individuals' lives. Advanced ML technology has made the early detection of CKD easier and more accurate. Doctors and medical care professionals have used ML algorithms in the effective diagnosis of CKD. However, there is very little research on the detection of secondary infections of prolonged CKD such as albuminuria and toxin production through the ML algorithm. These secondary infections also place a negative impact, especially on diabetics and patients with high blood pressure. Therefore, effective research can be done to determine the role of ML algorithms in detecting CKD-associated diseases. Further research on effective treatment prediction and nutritional chart prediction of CKD patients through ML algorithm needs to be done in the future. Advanced technologies such as CNN, ML, random forest, and different classifiers can be used for these aspects to increase the recovery rate in CKD. By following this way, researchers and medical care professionals can enhance their service quality in accurate CKD diagnosis and treatment. Effective detection of CKD through ML algorithm is rapid and cost-effective, and due to this reason, the method can gain large popularity in the future.

Conclusion

The ML algorithm can detect the CKD stage by effectively observing the changes in blood sugar, potassium, creatinine, and pus secretion. This research has shown the accuracy of the ML algorithm in detecting these changes in a patient with CKD. The statistical significance level, the positive and negative correlation of the independent variables, indicates that the ML algorithm detects the CKD in patients more accurately and precisely and helps determine the CKD stage. There are some constraints as per the study, yet there is a virtually insufficient investigation on using the ML algorithm to detect secondary infections of persistent CKD, such as albuminuria and toxin generation.

Thank You