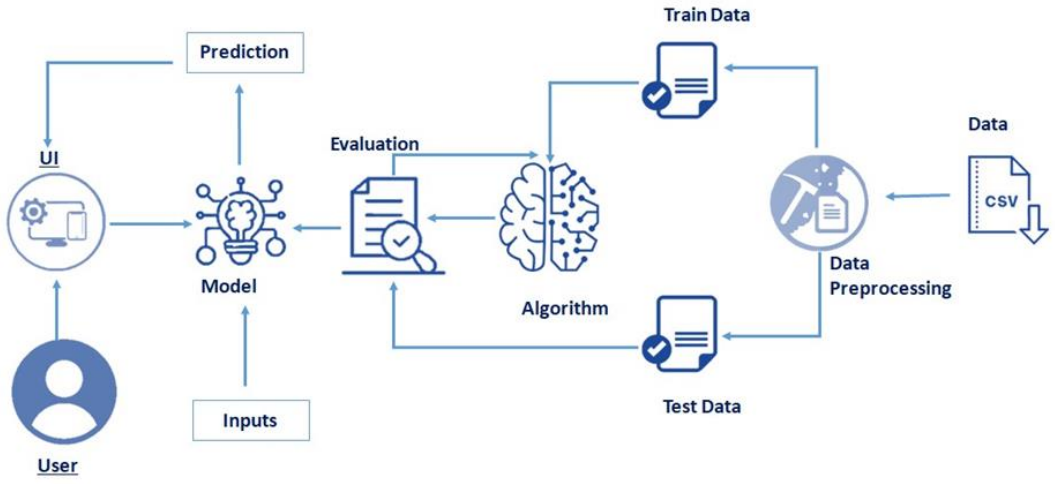
EARLY PREDICTION OF CHRONIC KIDNEY DISEASE

1 INTRODUCTION

1.1 Overview

Chronic Kidney Disease (CKD) is a major medical problem and can be cured if treated in the early stages. Usually, people are not aware that medical tests,we take for different purposes could contain valuable information concerning kidney diseases. Consequently, attributes of various medical tests are investigated to distinguish which attributes may contain helpful information about the disease. The information says that it helps us to measure the severity of the problem and we make use of such information to build a machine learning model that predicts Chronic Kidney Disease

Technical Architecture:



1.2 Purpose

- You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
- You will be able to know how to pre-process/clean the data using different data preprocessing techniques.
- You will be able to analyze or get insights from data through visualization.
- Applying different algorithms according to the dataset
- You will be able to know how to find the accuracy of the model.
- You will be able to know how to build a web application using the Flask framework.

2 LITERATURE SURVEY

2.1 Existing Problem

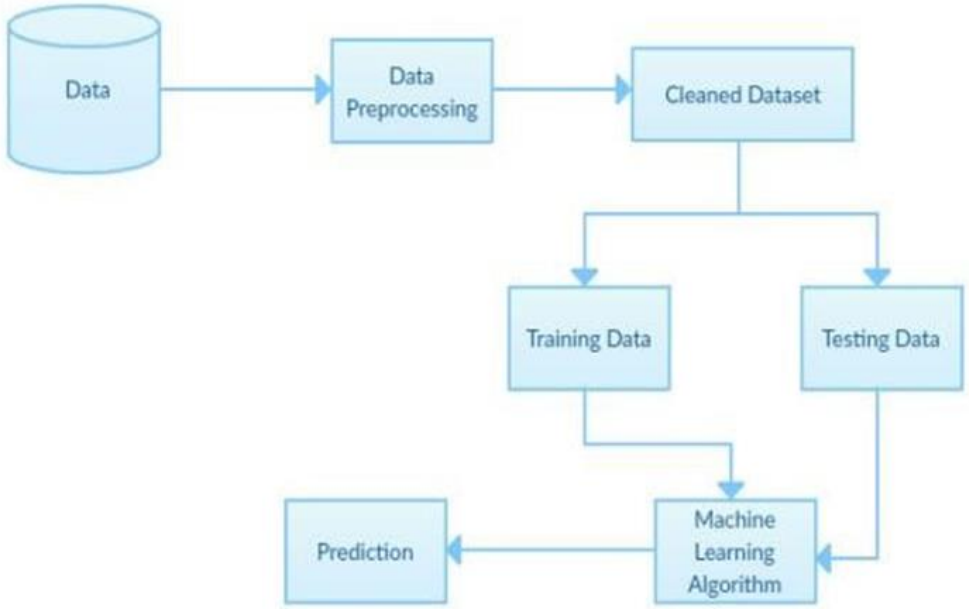
- Existing models didn't give accurate results.
- The time taken to predict the result is more.

2.2 Proposed Solution

- By using Logistic Regression techniques and by importing some packages like numpy, pandas, flask, scikit etc.
- By splitting the data into training data and testing data. • We are going to solve the existing system problems.

3 THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware/Software designing

To complete this project, you must require following software's concepts and packages
O Anaconda navigator:

- Refer to the link below to download anaconda navigator
- Link : https://www.youtube.com/watch?v=5mDYijMfSzs

O Python packages: Open anaconda prompt as administrator.

- Type "pip install numpy" and click enter.

- Type “pip install pandas” and click enter.
- Type “pip install matplotlib” and click enter.
- Type “pip install scikit-learn” and click enter.
- Type “pip install Flask” and click enter.
- Type “pip install missingno” and click enter.

The above steps allow you to install the packages in the anaconda environment

O Launch Jupyter

- Search for Anaconda Navigator and open Launch Jupyter notebook.

- Then you will be able to see that the jupyter notebook runs on local host:8888.
- To Create a new file Go to New Python3.The file in jupyter notebook is saved with .ipynb extension.

4 EXPERIMENTAL INVESTIGATIONS

Create a Project folder which contains files as shown below

- A python file called app.py for server side scripting.
- We need the model which is saved and the saved model in this content is CKD.pkl
- Templates folder which contains home.HTML file, index.HTML file, result.HTML file.
- Static folder which contains css folder which contains style.css , styles.css .

Milestone 1: Data Collection:

ML depends heavily on data, without data, it is impossible for an “AI” to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

Activity1: Download The dataset

You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository etc.
The dataset used for this project was obtained from Kaggle .

Milestone 2: Data Preprocessing

Data Pre-processing includes the following main tasks

- Import the Libraries.
- Importing the dataset.
- Checking for Null Values.
- Data Visualization.
- Label Encoding.

- Splitting Data into Train and Test.
    - Feature Scaling

Activity 1: Import Necessary Libraries

- Sklearn: Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms.
- NumPy: NumPy is a Python package that stands for 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object
- Pandas: pandas is a fast, powerful, flexible, and easy to use open source data analysis and manipulation tool,built on top of the Python programming language.
- Matplotlib: It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits

Activity 2: Importing the Dataset

- You might have your data in .csv files, .excel files
- Let's load a .csv data file into pandas using read_csv() function.We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program).
- If your dataset is in some other location ,Then
  Data=pd.read_csv(r"File_location")

Note:r stands for "raw" and will cause backslashes in the string to be interpreted as actual backslashes rather than special characters.

- If the dataset in same directory of your program, you can directly read it, without giving raw as r.

Activity 3: Analyse the data

- head() method is used to return top n (5 by default) rows of a DataFrame or series.
- info() gives information about the data
- <class 'pandas.core.frame.DataFrame'> • RangeIndex: 400 entries, 0 to 399 • Data columns (total 25 columns):
- # Column Non-Null Count Dtype
- --- ------ -------------- -----
- 0 age 391 non-null float64
- 1 blood_pressure 388 non-null float64
- 2 specific_gravity 353 non-null float64
- 3 albumin 354 non-null float64
- 4 sugar 351 non-null float64
- 5 red_blood_cells 248 non-null object
- 6 pus_cell 335 non-null object
- 7 pus_cell_clumps 396 non-null object
- 8 bacteria 396 non-null object

- 9 blood glucose random 356 non-null float64
- 10 blood_urea 381 non-null float64
- 11 serum_creatinine 383 non-null float64
- 12 sodium 313 non-null float64
- 13 potassium 312 non-null float64
- 14 hemoglobin 348 non-null float64
- 15 packed_cell_volume 330 non-null object
- 16 white_blood_cell_count 295 non-null object
- 17 red_blood_cell_count 270 non-null object
- 18 hypertension 398 non-null object
- 19 diabetesemellitus 398 non-null object
- 20 coronary_artery_disease 398 non-null object
- 21 appetite 399 non-null object
- 22 pedal_edema 399 non-null object
- 23 anemia 399 non-null object
- 24 class 400 non-null object
- dtypes: float64(11), object(14)
- memory usage: 78.2+ KB

Activity 4: Handling Missing Values

1. After loading it is important to check the complete information of data as it can indication many of the hidden infomation such as null values in a column or a row
2.Check whether any null values are there or not. if it is present then following can be done,
    a.Imputing data using Imputation method in sklearn
    b.Filling NaN values with mean, median and mode using fillna() method.

Activity 5: Label Encoding

In machine learning, we usually deal with datasets which contains multiple labels in one or more than one columns. These labels can be in the form of words or numbers. To make the data understandable or in human readable form, the training data is often labelled in words.
Label Encoding refers to converting the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

Data after label encoding
    All the data is converted into numerical values.

Activity 6: Splitting the data into Train and Test

- When you are working on a model and you want to train it, you obviously have a dataset. But after training, we have to test the model on some test dataset. For this, you will a dataset

which is different from the training set you used earlier. But it might not always be possible to have so much data during the development phase. In such cases, the solution is to split the dataset into two sets, one for training and the other for testing.

- But the question is, how do you split the data? You can't possibly manually split the dataset into two sets. And you also have to make sure you split the data in a random manner. To help us with this task, the Scikit library provides a tool, called the Model Selection library. There is a class in the library which is,'train_test_split.' Using this we can easily split the dataset into the training and the testing datasets in various proportions.
- The train-test split is a technique for evaluating the performance of a machine learning algorithm.
- Train Dataset: Used to fit the machine learning model.
- Test Dataset: Used to evaluate the fit machine learning model.
- In general you can allocate 80% of the dataset to training set and the remaining 20% to test set.We will create 4 sets— X_train (training part of the matrix of features), X_test (test part of the matrix of features), Y_train (training part of the dependent variables associated with the X train sets, and therefore also the same indices), Y_test (test part of the dependent variables associated with the X test sets, and therefore also the same indices.
- There are a few other parameters that we need to understand before we use the class:
- test_size — this parameter decides the size of the data that has to be split as the test dataset. This is given as a fraction. For example, if you pass 0.5 as the value, the dataset will be split 50% as the test dataset
- train_size — you have to specify this parameter only if you're not specifying the test_size. This is the same as test_size, but instead you tell the class what percent of the dataset you want to split as the training set.
- random_state — here you pass an integer, which will act as the seed for the random number generator during the split. Or, you can also pass an instance of the Random_state class, which will become the number generator. If you don't pass anything, the Random_state instance used by np.random will be used instead.
- Now split our dataset into train set and test using train_test_split class from scikit learn library.

Activity 7: Feature Scaling

There is huge disparity between the x values so let us use feature scaling.
Feature scaling is a method used to normalize the range of independent variables or features of data.

Milestone 3: Model Buiding

Model building includes the following main tasks

- Import the model building Libraries
- Initializing the model
- Training and testing the model
- Evaluation of Model
- Save the Model (using pickle)

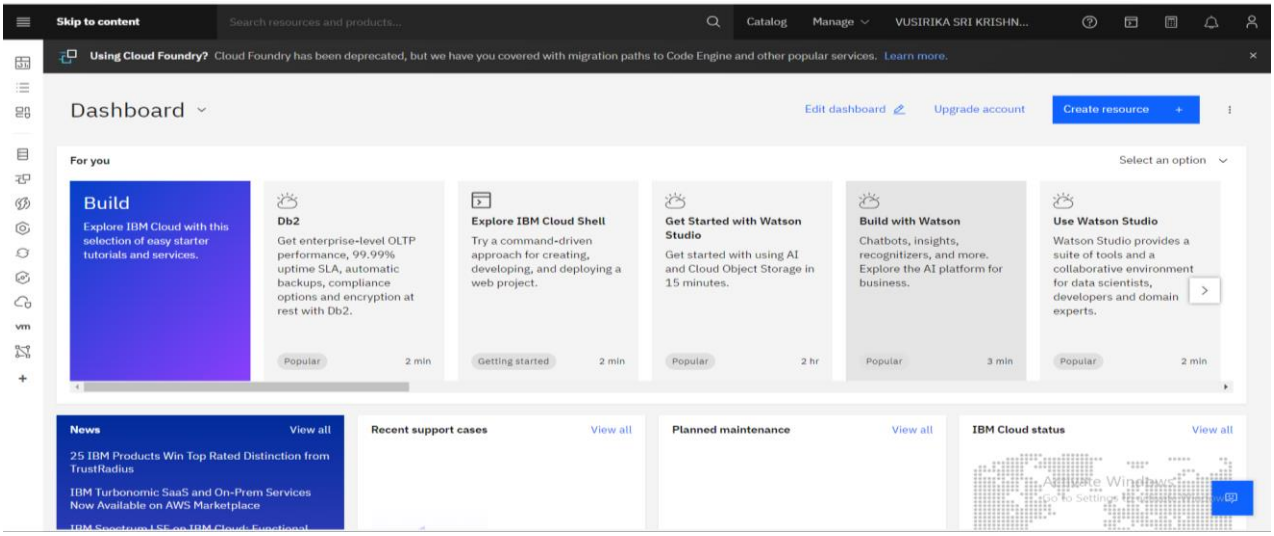
Conclusion:

As we perform 3 modes of Algorithms Logistic Regression, KNN, Navie Bayes. The Logistic Regression has high predicting accuracy of 97%. So we choose LGR among those ML Algorithms and saved the model in pickle file.

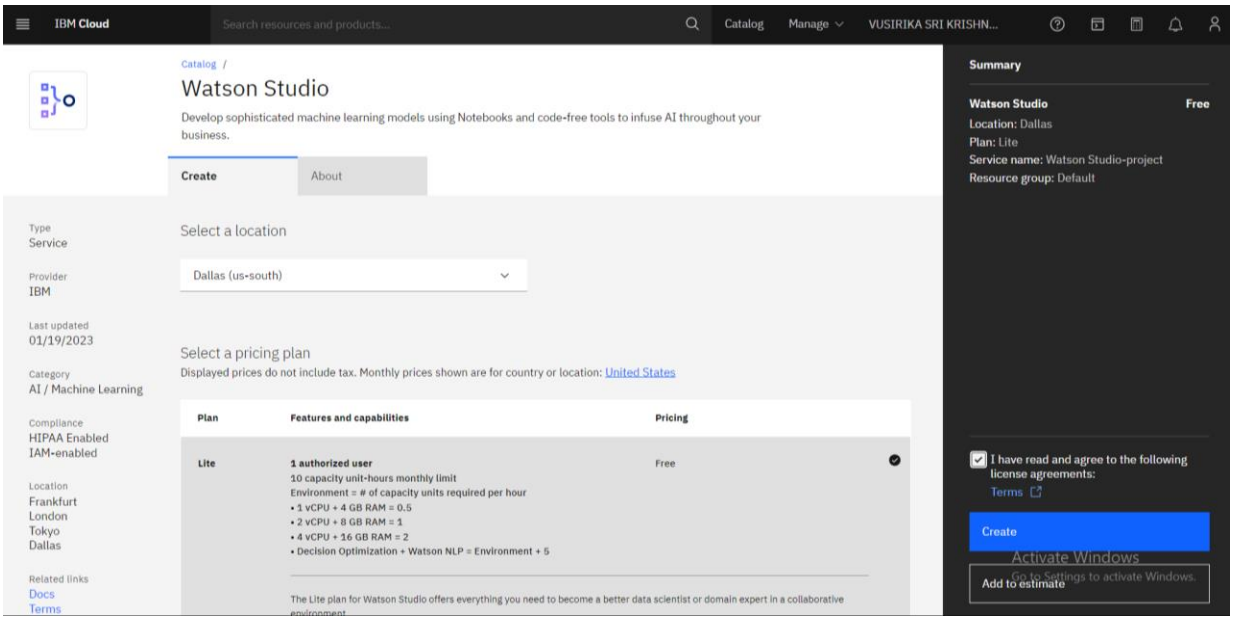
Using IBM watson studio

The other process to train model is to work with IBM watson studio. We can complete the task by performing the following steps

1. Create an IBM cloud account
2. Login to account you will see dashboard page

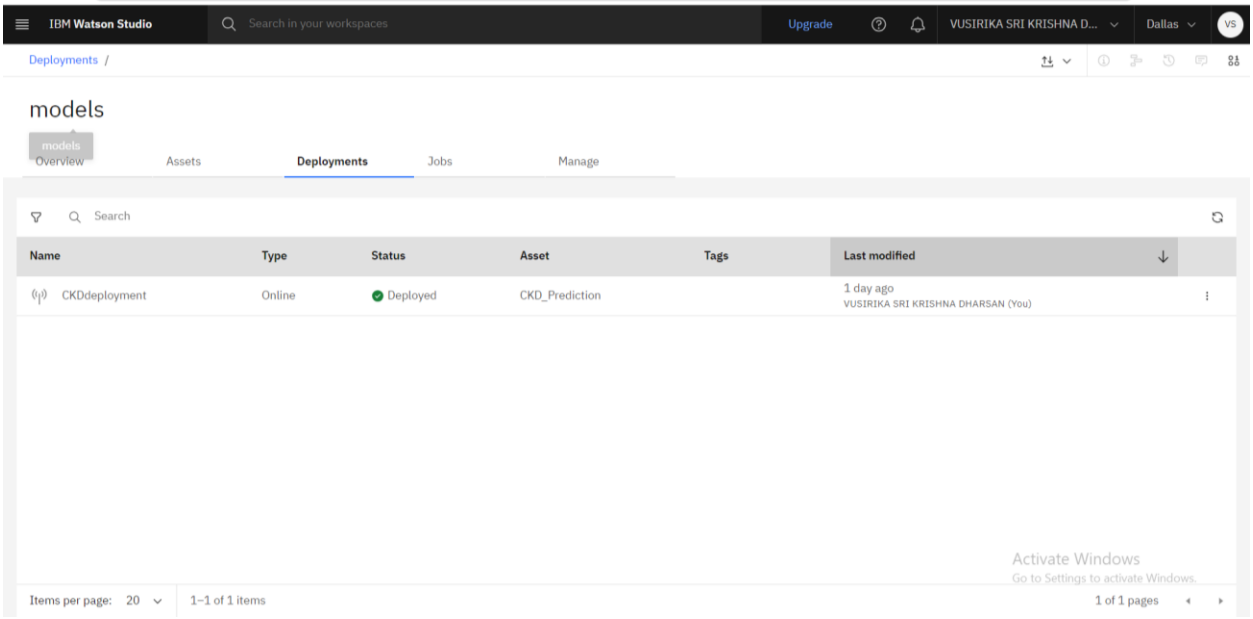


3. In Dashboard > Catalog > Watson studio create a Studio.
4. Similarly under Catalog > Machine Learning create a new project and Cloud Storage.

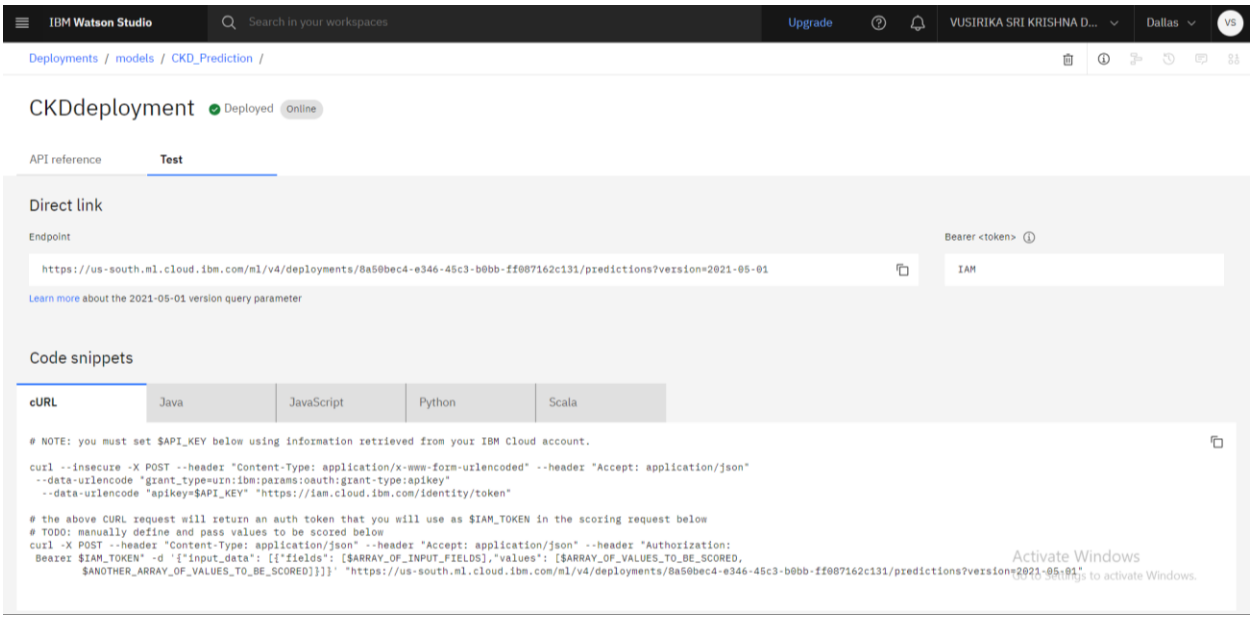


5. Add your model building file to project
6. Create a deployment space

7. Add your model to online deployment space.

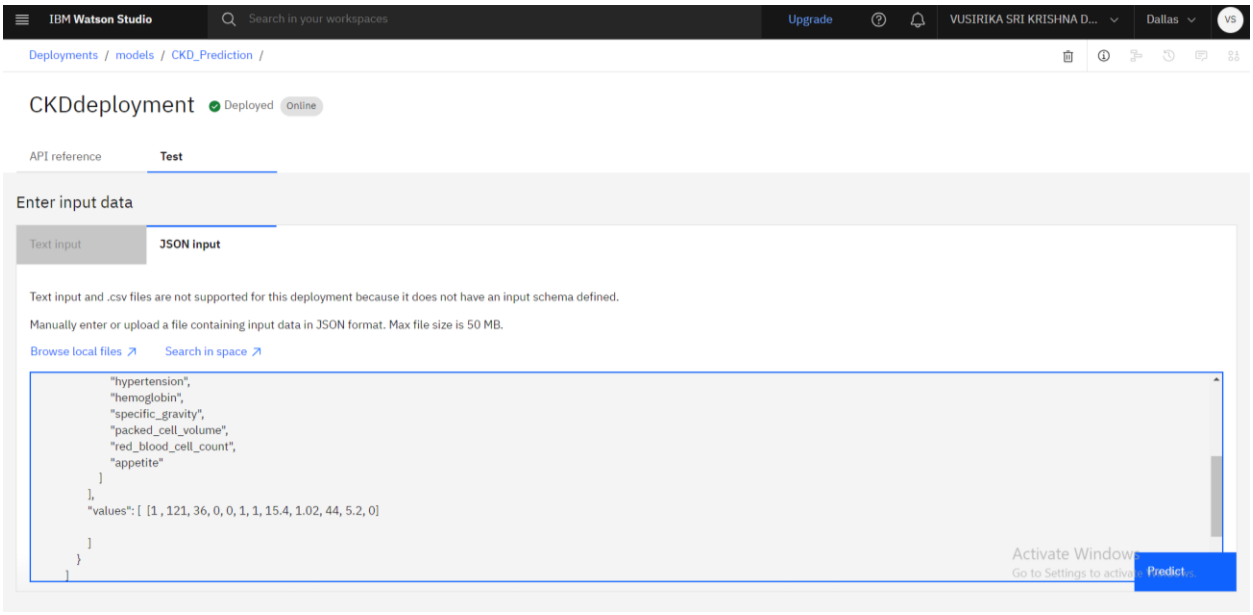


8. Once the model is deployed it will show you a ENDPOINTS link by which we can access the training data

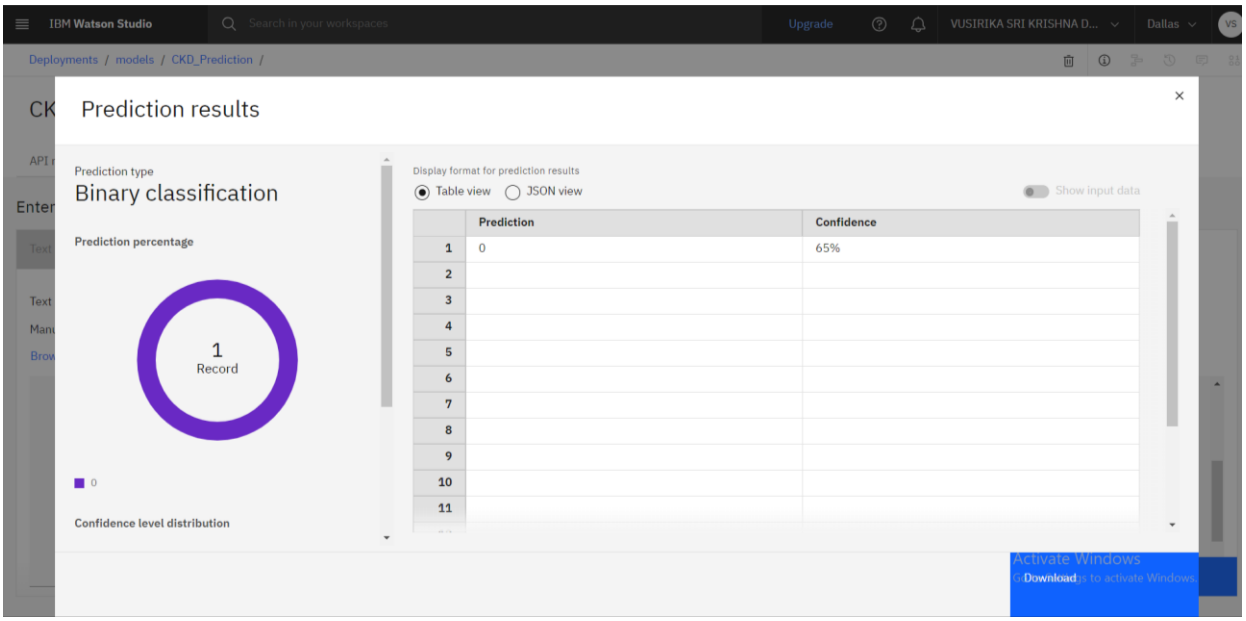


9. Create an unique API key under IAM section and note your KEY.

10. Testing the model in IBM.



11. Prediction



Activity 1: Training and Testing the Model

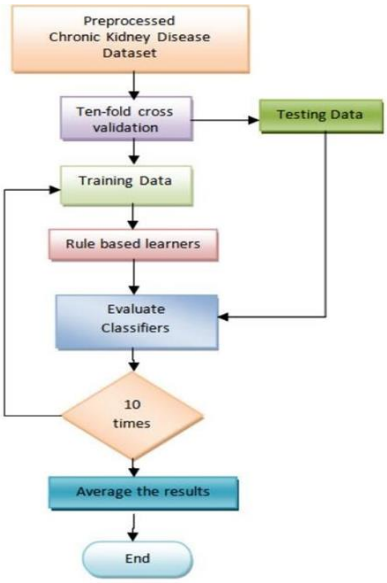
We are using the algorithm from Scikit learn library to build the model as shown below,

5 FLOW CHART

Following are the steps which are to be completed to complete this project

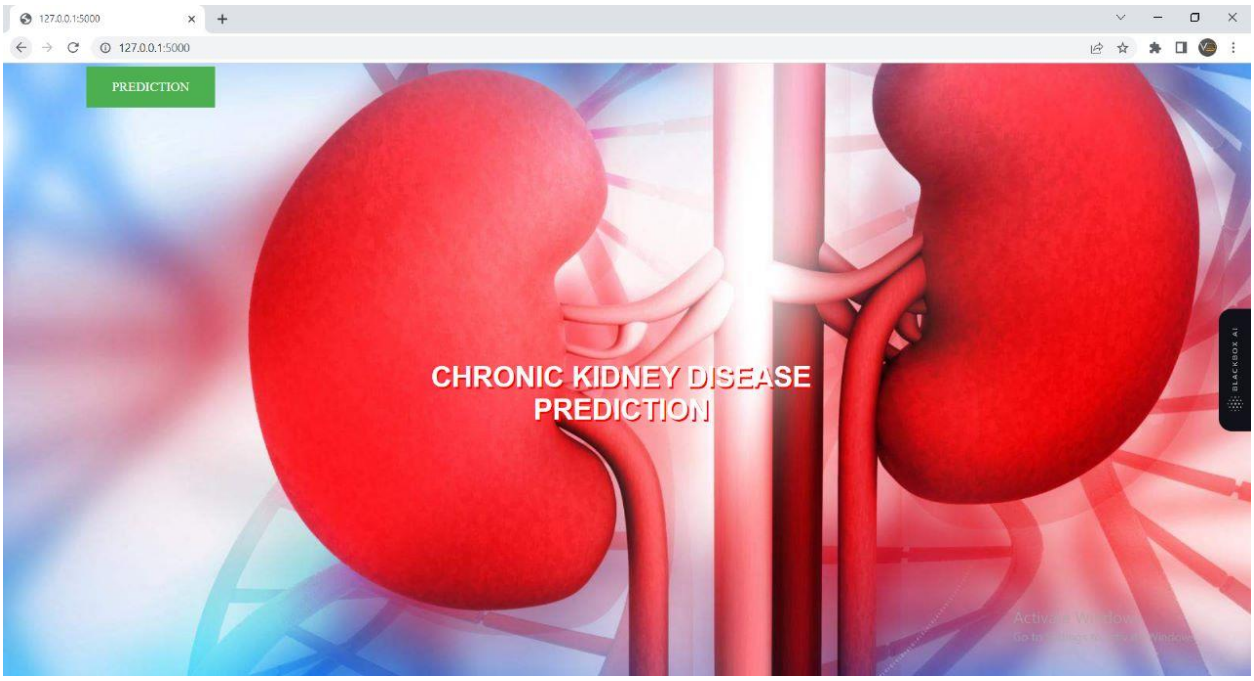
1. Download the dataset

2. Preprocess or clean the data
3. Analyze the pre-processed data
4. Train the machine with preprocessed data with an Appropriate Machine learning algorithm to build a model
5. save the model and its dependencies
6. Build a Web application using flask that integrates with Model built.

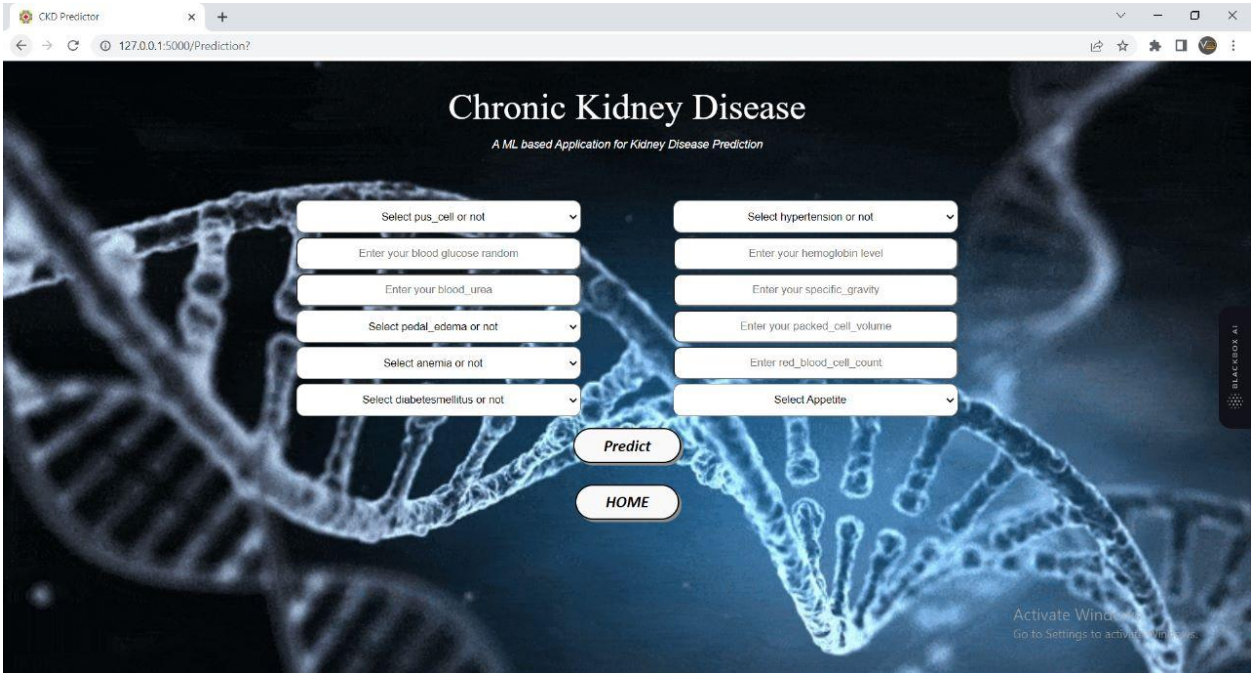


6 RESULT

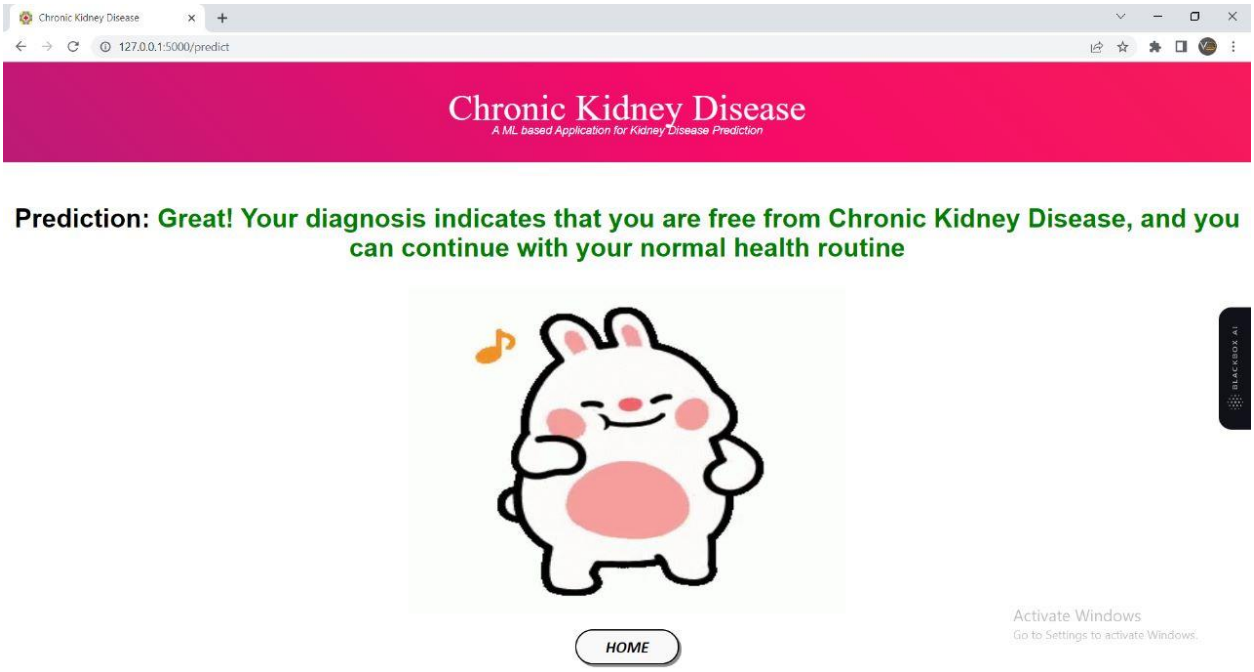
home.html

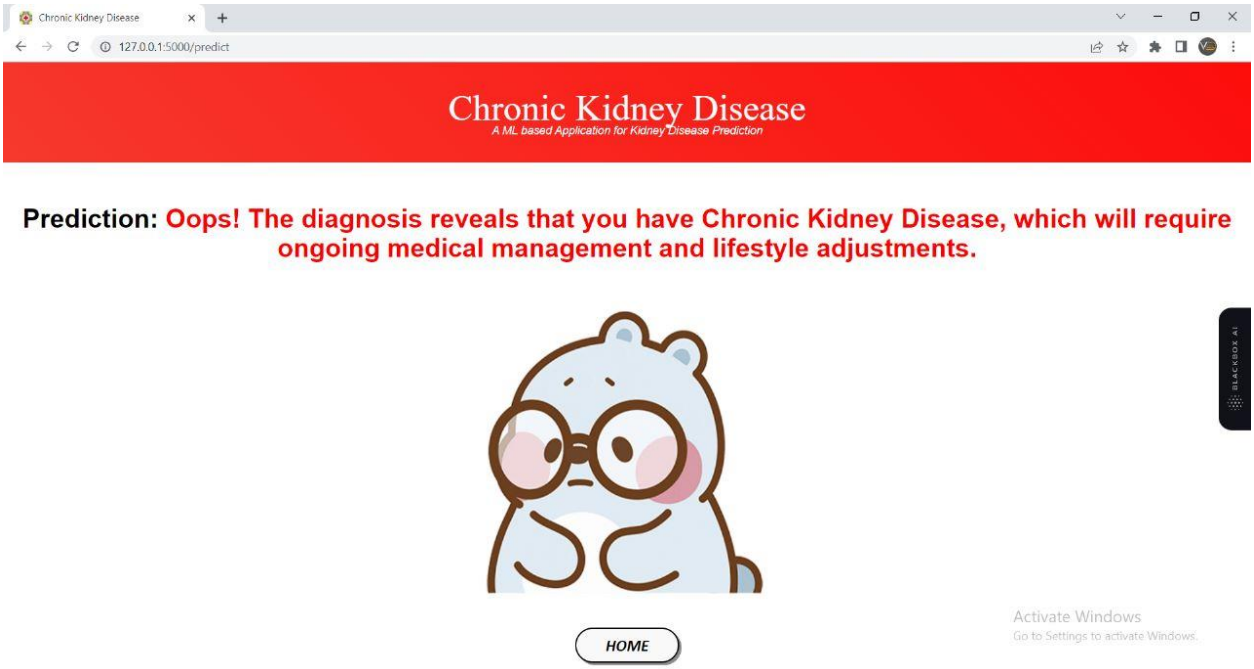


Indexnew.html



Result.html





7 ADVANTAGES

- Easy to predict the result without the help of Doctor.
- Based on the test values we can get the result that whether we have chronic kidney disease or not.

DISADVATAGES

- We cannot get the accurate results/prediction.
- Sometimes results may be wrong.

8 APPLICATIONS

- This model is used to predict CKD status based on clinical data.
- Early prediction of kidney disease status is possible with this model.
- Really a useful model for predicting kidney diseases in present situation.

9 CONCLUSION

- With the help of this model users can easily predict their disease quickly.
- The results will display whether the user have chronic kidney disease or not by taking some test values like blood urea, pus cell etc.

10 FUTURE SCOPE

- Present world is running towards Machine Learning so,

- The future scope for this model is very high because by following some simple procedure only we are getting the disease prediction results.
- This model makes the user work easy by predicting the results quickly.

11 BIBILOGRAPHY

- Flask Basics : https://www.youtube.com/watch?v=lj4I_CvBnt0
- Installation : https://youtu.be/5mDYijMfSzs
- Packages Installation : https://youtu.be/akj3_wTploU
- Supervised and Unsupervised Learning : https://youtu.be/kE5QZ8G_78c
- Regression classification and clustering : https://youtu.be/6za9_mh3uTE
- Logistic Regression : https://youtu.be/yIYKR4sgzI8