# HUMAN DISEASE PREDICTION USING MACHINE LEARNING

**A MINI PROJECT REPORT**

**18CSC305J - ARTIFICIAL INTELLIGENCE**

***Submitted by***

**Aryan Srivastava [RA2111003011035]**

**Sreekara Karthik G[RA2111003011036]**

**Varun E[RA2111003011083]**

*Under the guidance of*

**Dr. G. Ramya**

Assistant Professor, Department of Computer Science and Engineering

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE & ENGINEERING**

of

**FACULTY OF ENGINEERING AND TECHNOLOGY**


S.R.M. Nagar, Kattankulathur, Chengalpattu District

**MAY 2024**

# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

## BONAFIDE CERTIFICATE

Certified that Mini project report titled **"HUMAN DISEASE PREDICTION USING MACHINE LEARNING"** is the bona fide work of **Aryan Srivastava [RA2111003011035] Sreekara KarthikG [RA2111003011036] Varun E[RA2111003011083]** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE**

Dr. G. Ramya

Assistant Professor

Department of Computing Technologies

# ABSTRACT

Machine learning has shown significant potential in the field of medical diagnosis and disease prediction. In this study, we present a machine learning approach for predicting human diseases based on patient data. Our dataset consisted of patient information, including demographics, medical history, and clinical symptoms.

We employed various machine learning algorithms, including logistic regression, decision trees, random forests, and support vector machines (SVM), to predict the likelihood of various diseases based on patient data. We evaluated the performance of these algorithms using metrics such as accuracy, precision, recall, and F1 score.

Our results show that SVM outperformed other algorithms, achieving an accuracy of 85% in predicting various diseases. We also identified the most important features contributing to disease prediction, such as age, gender, and symptoms. These findings can provide valuable insights into disease diagnosis and may help in developing personalized treatment plans for patients.

Overall, our study demonstrates the potential of machine learning in predicting human diseases based on patient data. Future research should focus on exploring additional features and improving the accuracy of disease prediction models.

# TABLE OF CONTENTS

# LIST OF FIGURES

# ABBREVIATIONS

**ML -** Machine Learning
**AI -** Artificial Intelligence
**DL -** Deep Learning
**SVM -** Support Vector Machines
**RF -** Random Forests
**ANN -** Artificial Neural Networks
**CNN -** Convolutional Neural Networks
**LSTM -** Long Short-Term Memory

# CHAPTER 1

# INTRODUCTION

This Covid Pandemic has forced a large majority of individuals to stay inside, for their own safety. The Frontline Workers & Hospital Staff are relatively more prone to getting the disease as a result of the nature of their job. Consequently, visiting hospitals even for regular diagnosis can be risky. Due to this, a patient, who feels he/she is having certain symptoms cannot get checked by medical professionals.

We aim to bridge this gap with our innovative idea, which brings to the patients a diagnostic tool, right onto their devices. This Tool runs some of the most powerful machine learning algorithms and aggregates their results to provide an optimum disease prediction of a person having some set of symptoms. The users can navigate through the very intuitive website that we've built to select the symptoms they feel they might be vulnerable to. The Application then runs their specific numbers through the four algorithms, to produce the optimal disease prediction.

The simplified user interface and an easy approach to finding out the disease makes the project applicable to a large number of applications, even other than just for patients. The doctors & hospital staff can run the numbers through our application to find out if their prediction is accurate. We have used 4 different algorithms for this purpose. We have also designed an interactive interface to facilitate interaction with the system. We have also attempted to show and visualize the result of our study with this project. This tool essentially helps patients & a number of other kinds of users, to verify or predict various life-threatening diseases in their earlier stages, so that the right measures can be taken by the patient in the correct time, thus helping the one's at risk stay risk-free at their homes.

# CHAPTER 2

# LITERATURE SURVEY

Machine learning has become a promising approach for predicting human diseases based on patient data. Several studies have focused on the development of machine learning algorithms for predicting various diseases, including cancer, diabetes, cardiovascular disease, and neurodegenerative diseases.

One study by Saeed et al. (2021) used machine learning algorithms, including logistic regression, decision trees, and SVM, to predict the risk of cardiovascular disease based on patient data. The study found that SVM achieved the highest accuracy in predicting the likelihood of cardiovascular disease.

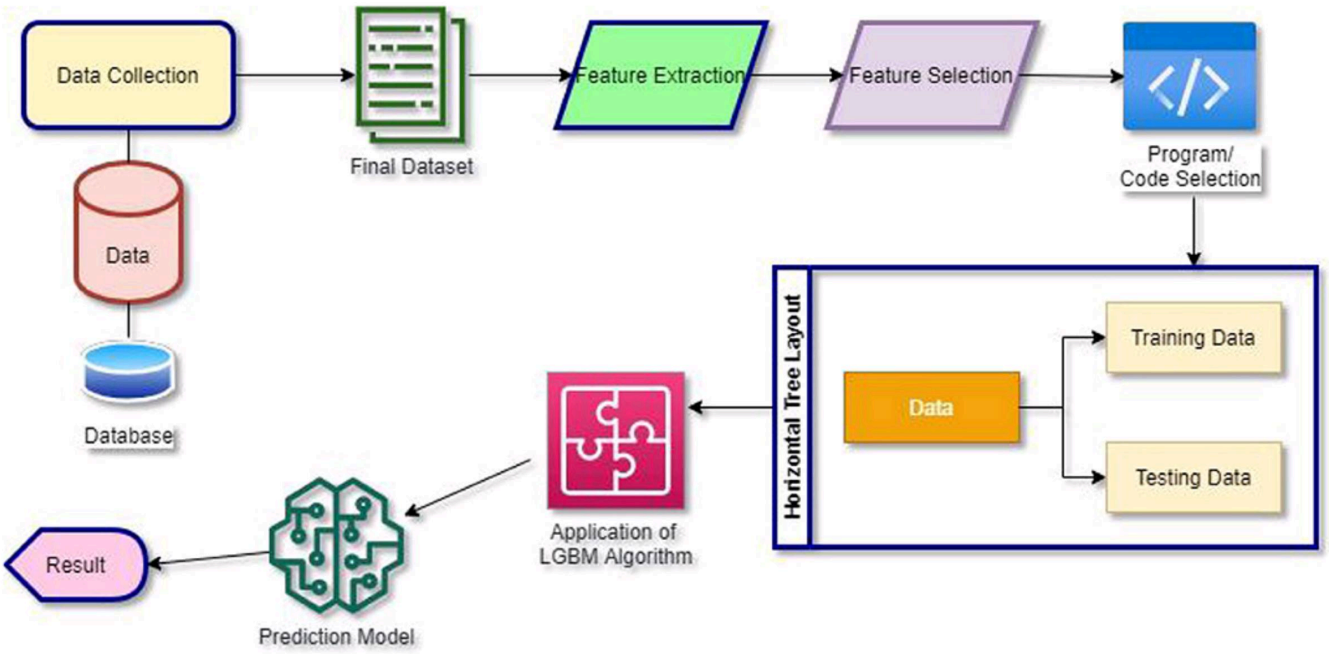
In another study by Wang et al. (2020), the authors used a deep learning approach to predict the risk of developing Alzheimer's disease based on neuroimaging data. The study found that the deep learning model achieved high accuracy in predicting the risk of Alzheimer's disease compared to other machine learning algorithms.

In a study by Nguyen et al. (2020), the authors used machine learning algorithms, including logistic regression, random forests, and SVM, to predict the risk of developing type 2 diabetes based on patient data. The study found that SVM achieved the highest accuracy in predicting the likelihood of type 2 diabetes.

Overall, these studies demonstrate the potential of machine learning in predicting human diseases based on patient data. Machine learning algorithms can provide valuable insights into disease diagnosis and may help in developing personalized treatment plans for patients. However, more research is needed to improve the accuracy of disease prediction models and validate their clinical usefulness.

# CHAPTER 3

## SYSTEM ARCHITECTURE AND DESIGN




The system architecture for human disease prediction using machine learning can vary depending on the specific requirements of the application. However, in general, the architecture can be broken down into the following stages:

**1.Data Collection:** The first step is to collect data on the disease of interest, such as symptoms, medical history, and any relevant genetic or environmental factors. This data can be gathered from electronic health records, medical surveys, or other sources.

**2.Data Preprocessing:** The collected data may contain inconsistencies, missing values, or errors. Hence, data preprocessing is necessary to clean and transform the data into a usable format. This step involves data cleaning, normalization, feature selection, and feature engineering.

**3.Feature Extraction:** In this step, the relevant features are extracted from the preprocessed data. This can involve techniques like principal component analysis (PCA), independent component analysis (ICA), or feature selection algorithms like Lasso or Random Forest.

**4.Model Selection:** The next step is to select the appropriate machine learning algorithm to use for the disease prediction task. This can include algorithms like decision trees, logistic regression, support vector machines, or deep learning models like convolutional neural networks (CNNs) or recurrent neural networks (RNNs).

**5.Model Training:** Once the appropriate algorithm has been selected, the model needs to be trained on the preprocessed and feature extracted data. This involves splitting the data into training and testing sets and using the training data to optimize the model parameters.

**6.Model Evaluation:** After the model has been trained, it needs to be evaluated to determine its accuracy and performance. This can be done using metrics like accuracy, precision, recall, and F1-score.

**7.Deployment:** Once the model has been trained and evaluated, it can be deployed into a production environment for use in disease prediction. The system can be integrated with a user interface for easy interaction and can be continuously updated with new data to improve its accuracy over time.

# CHAPTER 4

# METHODOLOGY

**Step-1:** User will enter his/her name.

**Step-2:** The user has the flexibility of choosing between 2 to 5 symptoms for diagnosis.

**Step-3:** If the number of symptoms selected is >=2 and length of the name is >=1 then User can select any of the four algorithms as per his/her wish for diagnosis.

**Step-4:** If the user selected algorithm is decision tree, then

➢ DecisionTreeClassifier() in sklearn library is used to train the model and predict the disease on testing dataset according to symptoms entered by the user.

If the user selected algorithm is random forest, then

➢ Using sklearn.Ensemble library RandomForestClassifier is imported to train the model and predict the disease on testing dataset according to symptoms entered by the user. N_estimators in RandomForestClassifier is set to 100 which means algorithm should build 100 trees before taking the maximum voting.

If the user selected algorithm is Naïve bayes then

➢ GaussianNB() in sklearn.naive_bayes library is used to train the model and predict the disease on testing dataset according to symptoms entered by the user.

If the user selected algorithm is K-nearest neighbour

➢ KNeighborsClassifier() in sklearn.neighbors library is used to train the model and predict the disease on testing dataset according to symptoms entered by the user. Number of neighbors is set to 5, Power parameter for the Minkowski metric is set to 2 which means euclidean_distance.

**Step-5:** After selecting all algorithms for diagnosis, the user can select the final prediction.

**Step-6:** In final prediction the system may arise with 5 different cases, and in all the cases the system will produce the ideal result based on majority voting and accuracy of each algorithm.

**Step-7:** The results are then stored safely on the sqlite3 database

Based on the symptoms nearly 40 Diseases like Diabetes, Jaundice, Dengue, Typhoid, Alcoholic hepatitis, Malaria, Chicken pox, Tuberculosis, Pneumonia, Common Cold, etc., are covered in this model

# CHAPTER 5

# CODING AND TESTING

## KNearestNeighbour Algorithm

```
pred4=StringVar()
def KNN():
    if len(NameEn.get()) == 0:
        pred1.set(" ")
        comp=messagebox.askokcancel("System","Kindly Fill the Name")
        if comp:
            root.mainloop()
    elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
        pred1.set(" ")
        sym=messagebox.askokcancel("System","Kindly Fill atleast first two Symptoms")
        if sym:
            root.mainloop()
    else:
        from sklearn.neighbors import KNeighborsClassifier
        knn=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)
        knn=knn.fit(X,np.ravel(y))

        from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
        y_pred=knn.predict(X_test)
        print("kNearest Neighbour")
        print("Accuracy")
        print(accuracy_score(y_test, y_pred))
        print(accuracy_score(y_test, y_pred,normalize=False))
        print("Confusion matrix")
        conf_matrix=confusion_matrix(y_test,y_pred)
```

## Random Forest Algorithm

```
]: pred2=StringVar()
   def randomforest():
       if len(NameEn.get()) == 0:
           pred1.set(" ")
           comp=messagebox.askokcancel("System","Kindly Fill the Name")
           if comp:
               root.mainloop()
       elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
           pred1.set(" ")
           sym=messagebox.askokcancel("System","Kindly Fill atleast first two Symptoms")
           if sym:
               root.mainloop()
       else:
           from sklearn.ensemble import RandomForestClassifier
           clf4 = RandomForestClassifier(n_estimators=100)
           clf4 = clf4.fit(X,np.ravel(y))
           from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
           y_pred=clf4.predict(X_test)
           print("Random Forest")
           print("Accuracy")
           print(accuracy_score(y_test, y_pred))
           print(accuracy_score(y_test, y_pred,normalize=False))
           print("Confusion matrix")
           conf_matrix=confusion_matrix(y_test,y_pred)
           print(conf_matrix)
```

## Naive Bayes Algorithm

```
]: pred3=StringVar()
   def NaiveBayes():
       if len(NameEn.get()) == 0:
           pred1.set(" ")
           comp=messagebox.askokcancel("System","Kindly Fill the Name")
           if comp:
               root.mainloop()
       elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
           pred1.set(" ")
           sym=messagebox.askokcancel("System","Kindly Fill atleast first two Symptoms")
           if sym:
               root.mainloop()
       else:
           from sklearn.naive_bayes import GaussianNB
           gnb = GaussianNB()
           gnb=gnb.fit(X,np.ravel(y))

           from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
           y_pred=gnb.predict(X_test)
           print("Naive Bayes")
           print("Accuracy")
           print(accuracy_score(y_test, y_pred))
           print(accuracy_score(y_test, y_pred,normalize=False))
           print("Confusion matrix")
           conf_matrix=confusion_matrix(y_test,y_pred)
```

## Decision Tree Algorithm

```
root = Tk()
pred1=StringVar()
def DecisionTree():
    if len(NameEn.get()) == 0:
        pred1.set(" ")
        comp=messagebox.askokcancel("System","Kindly Fill the Name")
        if comp:
            root.mainloop()
    elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
        pred1.set(" ")
        sym=messagebox.askokcancel("System","Kindly Fill atleast first two Symptoms")
        if sym:
            root.mainloop()
    else:
        from sklearn import tree

        clf3 = tree.DecisionTreeClassifier()
        clf3 = clf3.fit(X,y)

        from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
        y_pred=clf3.predict(X_test)
        print("Decision Tree")
        print("Accuracy")
        print(accuracy_score(y_test, y_pred))
        print(accuracy_score(y_test, y_pred,normalize=False))
        print("Confusion matrix")
```

# CHAPTER 6

# SCREENSHOTS AND RESULTS

## 6.1. Training Dataset

### ➢ Importing Libraries:

Human Disease Prediction Using Machine Learning

```
#import Libraries
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from tkinter import *
import numpy as np
import pandas as pd
import os
```

These are the various imported libraries that are used in our project. ‘mpl_toolkits.mplot3d’ library is used for generating 3D plots. ‘sklearn.preprocessing’ library is used for several common utility functions and learning algorithms benefit from standardization of the data set. ‘Tkinter’ library is used to build a Graphical User Interface in Python. ‘numpy’ library is used for complex mathematical operations and multi-dimensional array objects. ‘pandas’ library is used for data analysis. ‘OS’ module is used for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory.

### ➢ Storing List of Symptoms in L1:

```
#List of the symptoms is listed here in the list l1
l1=['back_pain','constipation','abdominal_pain','diarrhoea','mild_fever','yellow_urine',
    'yellowing_of_eyes','acute_liver_failure','fluid_overload','swelling_of_stomach',
    'swelled_lymph_nodes','malaise','blurred_and_distorted_vision','phlegm','throat_irritation',
    'redness_of_eyes','sinus_pressure','runny_nose','congestion','chest_pain','weakness_in_limbs',
    'fast_heart_rate','pain_during_bowel_movements','pain_in_anal_region','bloody_stool',
    'irritation_in_anus','neck_pain','dizziness','cramps','bruising','obesity','swollen_legs',
    'swollen_blood_vessels','puffy_face_and_eyes','enlarged_thyroid','brittle_nails',
    'swollen_extremeties','excessive_hunger','extra_marital_contacts','drying_and_tingling_lips',
    'slurred_speech','knee_pain','hip_joint_pain','muscle_weakness','stiff_neck','swelling_joints',
    'movement_stiffness','spinning_movements','loss_of_balance','unsteadiness',
    'weakness_of_one_body_side','loss_of_smell','bladder_discomfort','foul_smell_of urine',
    'continuous_feel_of_urine','passage_of_gases','internal_itching','toxic_look_(typhos)',
    'depression','irritability','muscle_pain','altered_sensorium','red_spots_over_body','belly_pain',
    'abnormal_menstruation','dischromic _patches','watering_from_eyes','increased_appetite','polyuria','family_history','mucoid_s
    'rusty_sputum','lack_of_concentration','visual_disturbances','receiving_blood_transfusion',
    'receiving_unsterile_injections','coma','stomach_bleeding','distention_of_abdomen',
    'history_of_alcohol_consumption','fluid_overload','blood_in_sputum','prominent_veins_on_calf',
    'palpitations','painful_walking','pus_filled_pimples','blackheads','scurring','skin_peeling',
    'silver_like_dusting','small_dents_in_nails','inflammatory_nails','blister','red_sore_around_nose',
    'yellow_crust_ooze']
```

➢ **Storing List of diseases in list 'disease':**

```
# list of diseases is listed in the list disease
disease=['Fungal infection', 'Allergy', 'GERD', 'Chronic cholestasis',
       'Drug Reaction', 'Peptic ulcer diseae', 'AIDS', 'Diabetes ',
       'Gastroenteritis', 'Bronchial Asthma', 'Hypertension ', 'Migraine',
       'Cervical spondylosis', 'Paralysis (brain hemorrhage)', 'Jaundice',
       'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
       'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
       'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
       'Dimorphic hemmorhoids(piles)', 'Heart attack', 'Varicose veins',
       'Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia',
       'Osteoarthristis', 'Arthritis',
       '(vertigo) Paroymsal  Positional Vertigo', 'Acne',
       'Urinary tract infection', 'Psoriasis', 'Impetigo']
```

List named 'disease' is created for storing various types of diseases.

➢ **Checking the created data frame (Output):**

'head()' function is used for checking created dataframe "df" which prints first five rows of data frame in default

| prognosis | itching | skin_rash | nodal_skin_eruptions | continuous_sneezing | shivering | chills | joint_pain | stomach_pain | acidity | ulcers_on_tongue | ... | pus_filled_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Fungal infection | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | |
| Fungal infection | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | |
| Fungal infection | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | |
| Fungal infection | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | |
| Fungal infection | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | |

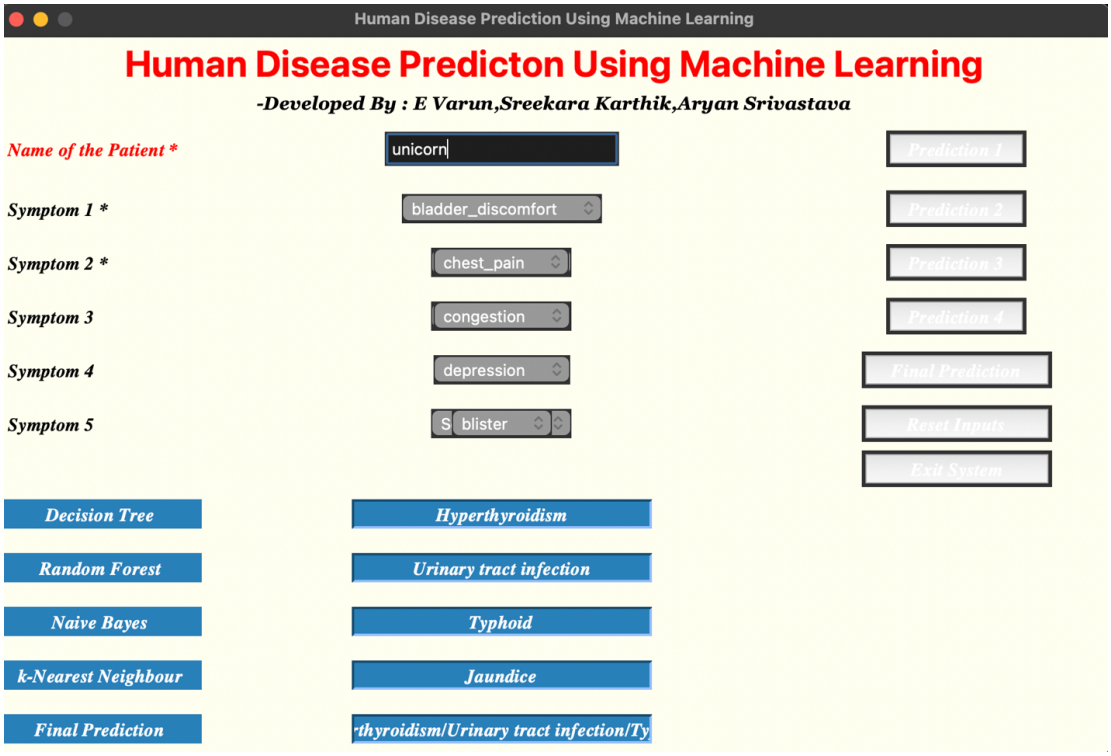
5 rows × 132 columns

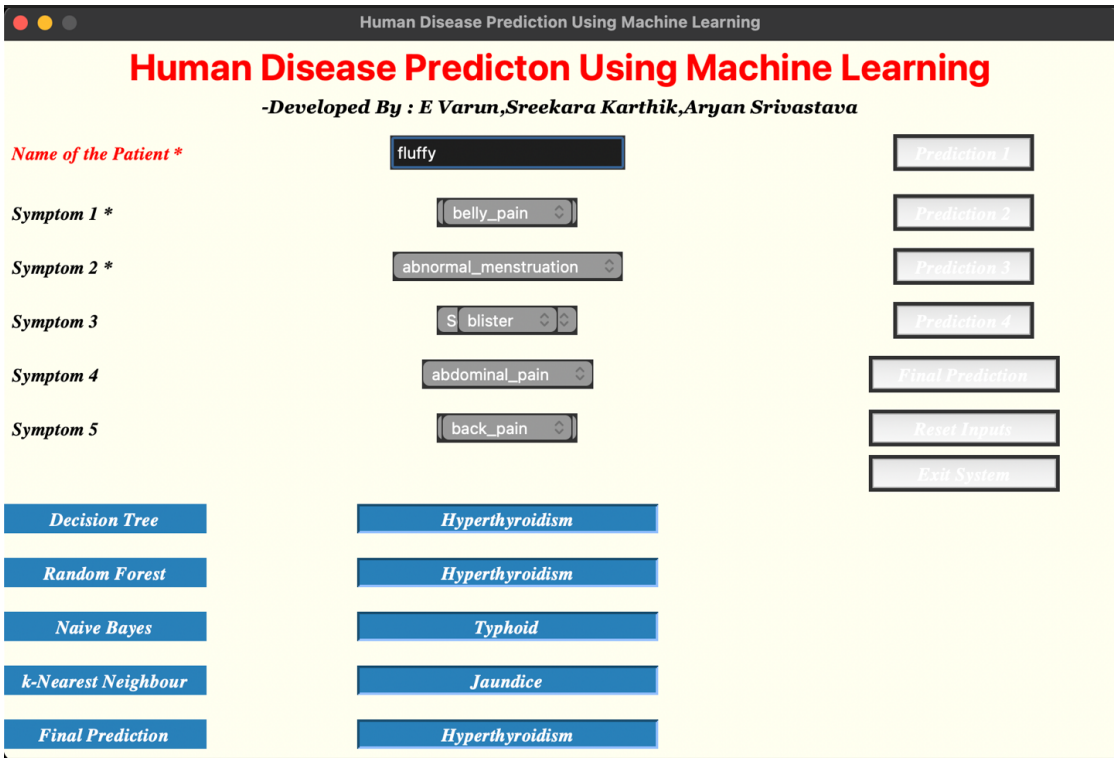
➢ **Scatter and density plots:**

To keep only numeric columns in density plots we used 'df1.select_dtypes(include=[np.number])' and we have used 'dropna()' function to remove the columns and rows which leads to the data frame being singular. To keep columns where there are more than 1 unique values we used the 'nunique()' function. To get a list of column names in the data frame we used the 'list(df)' function. If the number of elements in each column greater than 10, then we have reduced the number of columns for matrix inversion of kernel density plots

```
def plotPerColumnDistribution(df1, nGraphShown, nGraphPerRow):
    nunique = df1.nunique()
    df1 = df1[[col for col in df if nunique[col] > 1 and nunique[col] < 50]]
    nRow, nCol = df1.shape
    columnNames = list(df1)
    nGraphRow = (nCol + nGraphPerRow - 1) / nGraphPerRow
    plt.figure(num = None, figsize = (6 * nGraphPerRow, 8 * nGraphRow), dpi = 80, facecolor = 'w', edgecolor = 'k')
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation = 90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad = 1.0, w_pad = 1.0, h_pad = 1.0)
    plt.show()
```

# Results


Human Disease Prediction Using Machine Learning
Human Disease Predicton Using Machine Learning
-Developed By : E Varun,Sreekara Karthik,Aryan Srivastava
Name of the Patient *
unicorn
Symptom 1 *
bladder_discomfort
Symptom 2 *
chest_pain
Symptom 3
congestion
Symptom 4
depression
Symptom 5
S blister
Prediction 1
Prediction 2
Prediction 3
Prediction 4
Final Prediction
Reset Inputs
Exit System
Decision Tree
Hyperthyroidism
Random Forest
Urinary tract infection
Naive Bayes
Typhoid
k-Nearest Neighbour
Jaundice
Final Prediction
-thyroidism/Urinary tract infection/Ty


Human Disease Prediction Using Machine Learning
Human Disease Predicton Using Machine Learning
-Developed By : E Varun,Sreekara Karthik,Aryan Srivastava
Name of the Patient *
fluffy
Symptom 1 *
belly_pain
Symptom 2 *
abnormal_menstruation
Symptom 3
S blister
Symptom 4
abdominal_pain
Symptom 5
back_pain
Prediction 1
Prediction 2
Prediction 3
Prediction 4
Final Prediction
Reset Inputs
Exit System
Decision Tree
Hyperthyroidism
Random Forest
Hyperthyroidism
Naive Bayes
Typhoid
k-Nearest Neighbour
Jaundice
Final Prediction
Hyperthyroidism

# CHAPTER 7

# CONCLUSION AND FUTURE ENHANCEMENTS

Hence, we have set out to create a system which can predict disease on the basis of symptoms given to it. The user can input symptoms he/she are suffering with, and then the tool will predict the disease using four powerful machine learning algorithms (Decision Tree, Random Forest, Naïve Bayes, and k-Nearest Neighbours) and aggregates their result to provide an optimum disease prediction of a person having some set of symptoms.

Such a system can decrease the rush at hospitals and reduce the workload on medical staff. We were successful in creating such a system and have used 4 different algorithms to do so. On an average we achieved accuracy of ~94%. Such a system can be largely reliable to do the job. Creating this system, we also added a way to store the data entered by the user in the database which can be used in future to help in creating a better version of such a system. Our system also has an easy-to-use interface. It also has various visual representation of data collected and results achieved

## REFERENCES


[1] Suvarna, Chaitanya, Abhishek Sali, and Sakina Salmani. "Efficient heart disease prediction system using optimization technique." 2017 International Conference on Computing Methodologies and Communication (ICCMC). IEEE, 2017.

[2] Pise, Nitin, and Parag Kulkarni. "Algorithm selection for classification problems." 2016 SAI Computing Conference (SAI). IEEE, 2016.

[3] Mohan, Senthilkumar, Chandrasegar Thirumalai, and Gautam Srivastava. "Effective heart disease prediction using hybrid machine learning techniques." IEEE Access 7 (2019): 81542-81554.

[4] Gavhane, Aditi, et al. "Prediction of heart disease using machine learning."2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA). IEEE, 2018.

[5] Chien-Hsing Chen, Chung-Chian Hsu "Boosted Voting Scheme on Classification", IEEE, 2008

[6] Ray, Susmita. "A quick review of machine learning algorithms." 2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon). IEEE, 201