

# DISEASE PREDICTION BY SYMPTOMS USING AIML

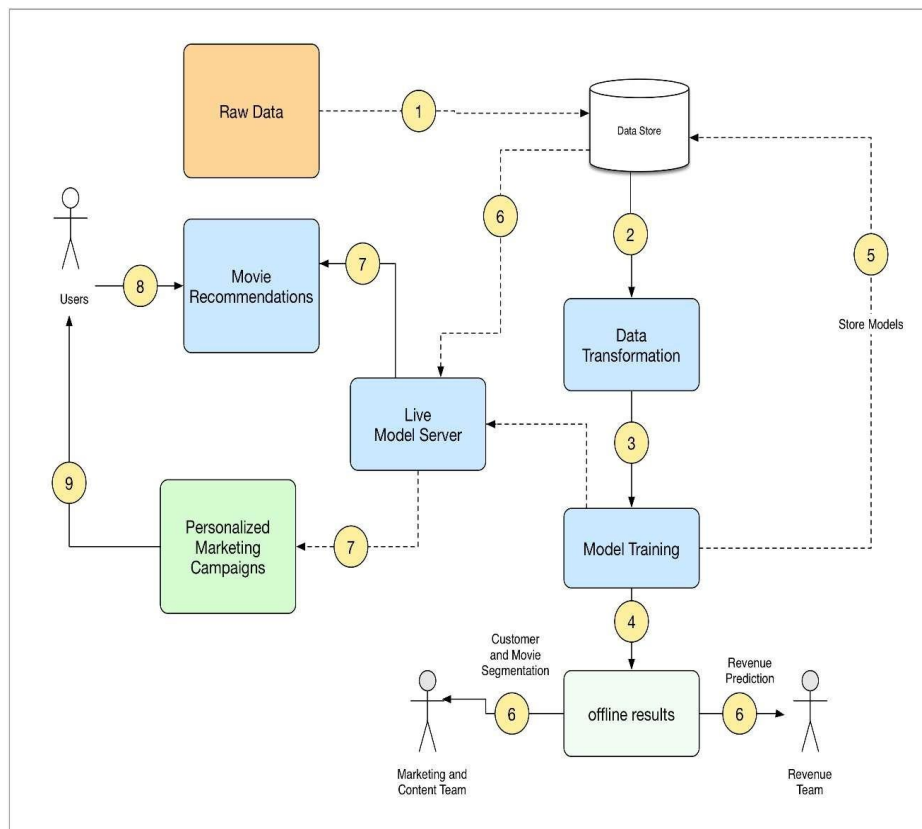
Done by M.Kalpana Devi

## Learning Algorithms :

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

- ✚ Supervised Learning
- ✚ Semi-Supervised Learning
- ✚ Unsupervised Learning
- ✚ Reinforcement Learning
- ✚ Features Learning

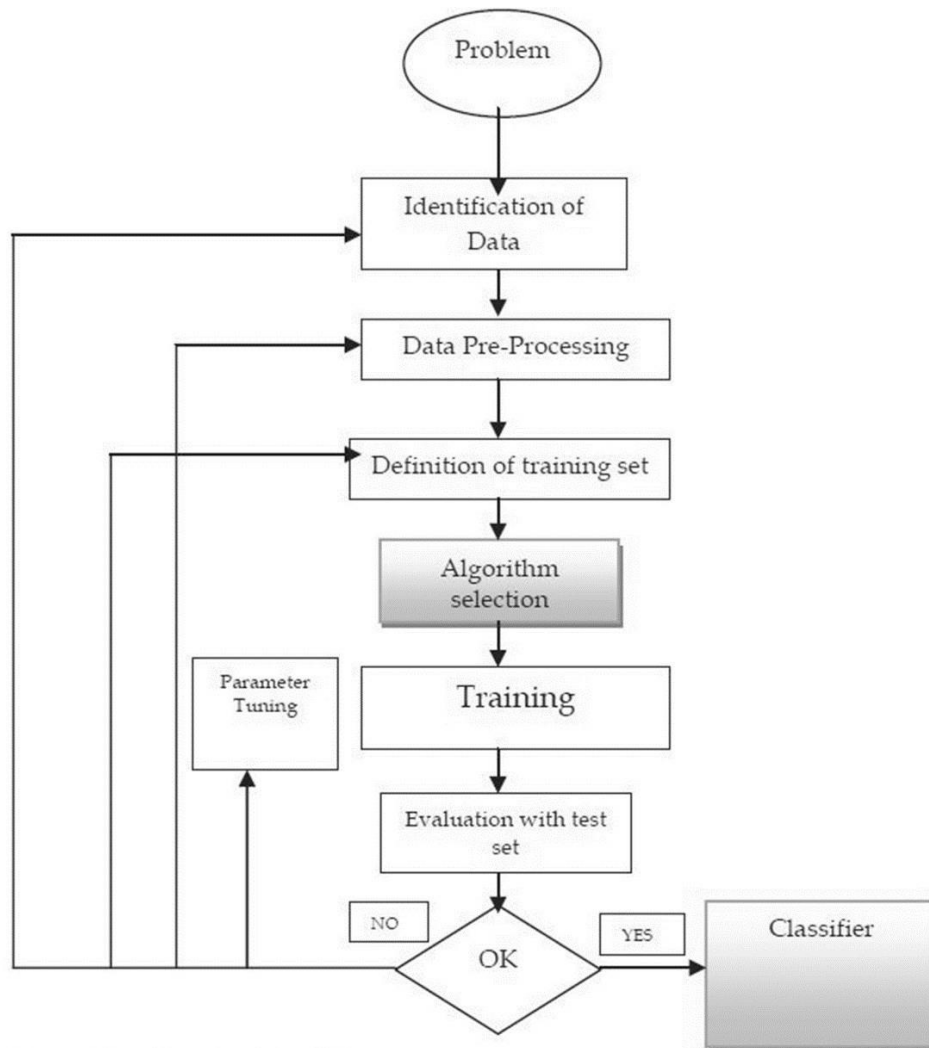
## Machine Learning Architecture



## Machine Learning Algorithms :

### 1) Supervised Learning

Supervised learning is a machine learning technique for learning a function from training data. The training data consist of pairs of input objects (typically vectors), and desired outputs. The output of the function can be a continuous value (called regression), or can predict a class label of the input object (called classification).



## **2) Unsupervised Learning -**

Unsupervised learning is a type of machine learning where manual labels of inputs are not used. It is distinguished from supervised learning approaches which learn how to perform a task, such as classification or regression, using a set of human prepared examples.

## **3) Semi-supervised Learning -**

Semi-supervised learning combines both labeled and unlabeled examples to generate an appropriate function or classifier.

## **4) Reinforcement Learning -**

Reinforcement Learning is where the algorithm learns a policy of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm.

## **Dataset Preparation and Pre-processing :**

Data is the foundation for any machine learning project. The second stage of project implementation is complex and involves data collection, selection, preprocessing, and transformation. Each of these phases can be split into several steps.

### *1) Data Collection*

It's time for a data analyst to pick up the baton and lead the way to machine learning implementation. The job of a data analyst is to find ways and sources of collecting relevant and comprehensive data, interpreting it, and analyzing results with the help of statistical techniques.

## 2) Data Visualization

A large amount of information represented in graphic form is easier to understand and analyze. Some companies specify that a data analyst must know how to create slides, diagrams, charts, and templates.

### 3) *Data Cleaning*

This set of procedures allows for removing noise and fixing inconsistencies in data. A data scientist can fill in missing data using imputation techniques, e.g. substituting missing values with mean attributes.

File Home Insert Page Layout Formulas Data Review View Tell me what you want to do.

Clipboard Font Alignment Merge & Center Number Styles Cells Editing

Calibri 11 Wrap Text General Conditional Formatting Normal Bad Good Neutral Calculation Check Cell Insert Delete Format Autosum Fill Sort & Find & Filter Select

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
	Itching	skin_rash	nodal_skin_continuous	shivering	chills	joint_pain	stomach_acidity	ulcers_on_mucosa	vomiting	burning	spots	fatigue	weight_loss	anxiety	cold	hand_mood	swelling	weight_loss	restlessness	lethargy	patches_on_skin		
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	
5	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
6	1	1	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	
7	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	
10	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	
17	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
18	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	
19	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	
20	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
21	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
22	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
25	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	
26	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
27	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	
28	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Testing

## Dataset Splitting

A dataset used for machine learning should be partitioned into three subsets - training, test, and validation sets.

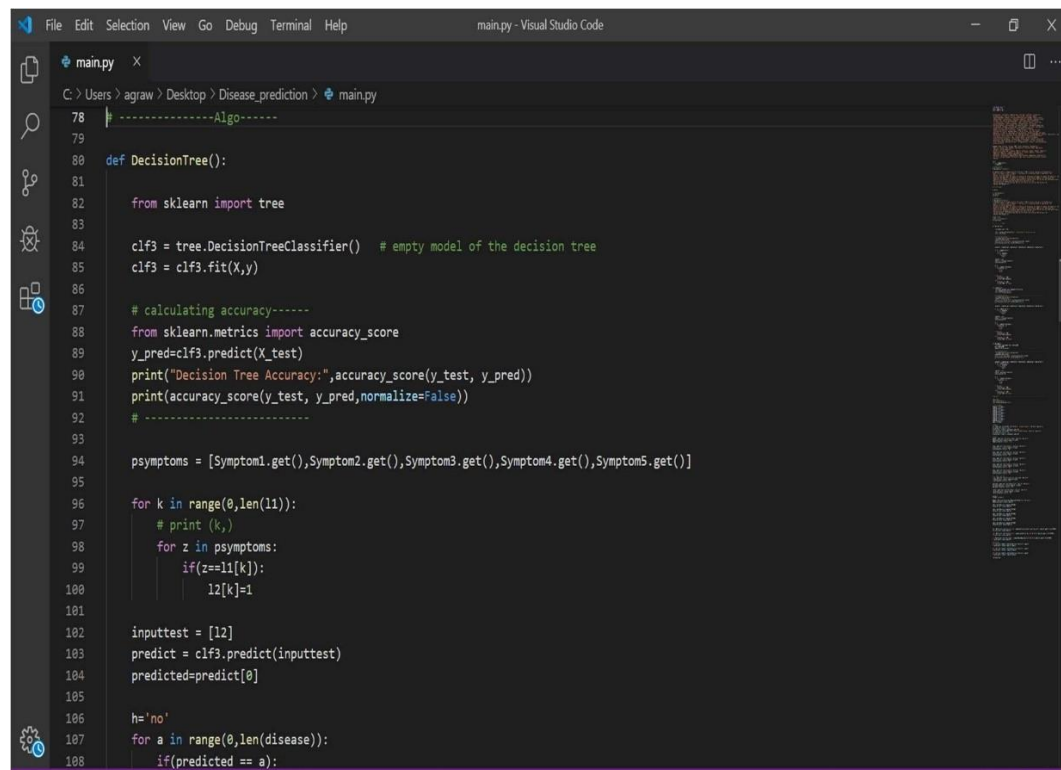
1) *Training Set:*

A data scientist uses a training set to train a model and define its optimal parameters - parameters it has to learn from data.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	itching	skin_rash	nodal_skin_continuous	shivering	chills	joint_pain	stomach_acidity			ulcers_or_muscle_wounds	worming	burning	nummular_spotting	fatigue	weight_gain	anxiety	cold_hand	mood_swing	weight_loss	restlessness	lethargy	patches	
2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
12	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
14	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
16	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
18	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
19	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
22	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	
23	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	
25	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
26	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	
27	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	
28	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	
29	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	
30	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	
Training																							

### a) *Decision Tree Algorithm*

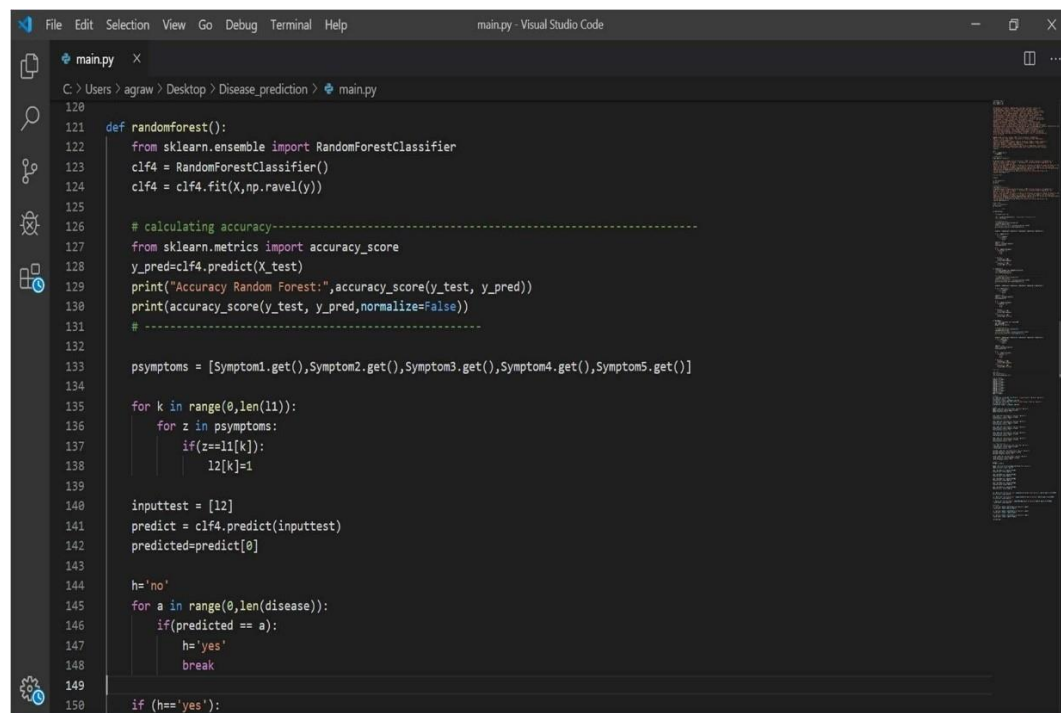
- Decision tree algorithm falls under the category of supervised learning. They can be used to solve both regression and classification problems.
- Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree.



```
78 -----Algo-----
79
80 def DecisionTree():
81
82     from sklearn import tree
83
84     clf3 = tree.DecisionTreeClassifier() # empty model of the decision tree
85     clf3 = clf3.fit(X,y)
86
87     # calculating accuracy-----
88     from sklearn.metrics import accuracy_score
89     y_pred=clf3.predict(X_test)
90     print("Decision Tree Accuracy:",accuracy_score(y_test, y_pred))
91     print(accuracy_score(y_test, y_pred,normalize=False))
92     # -----
93
94     psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]
95
96     for k in range(0,len(11)):
97         # print (k,)
98         for z in psymptoms:
99             if(z==l1[k]):
100                 l2[k]=1
101
102     inputtest = [12]
103     predict = clf3.predict(inputtest)
104     predicted=predict[0]
105
106     h='no'
107     for a in range(0,len(disease)):
108         if(predicted == a):
```

### b) Random Forest Algorithm

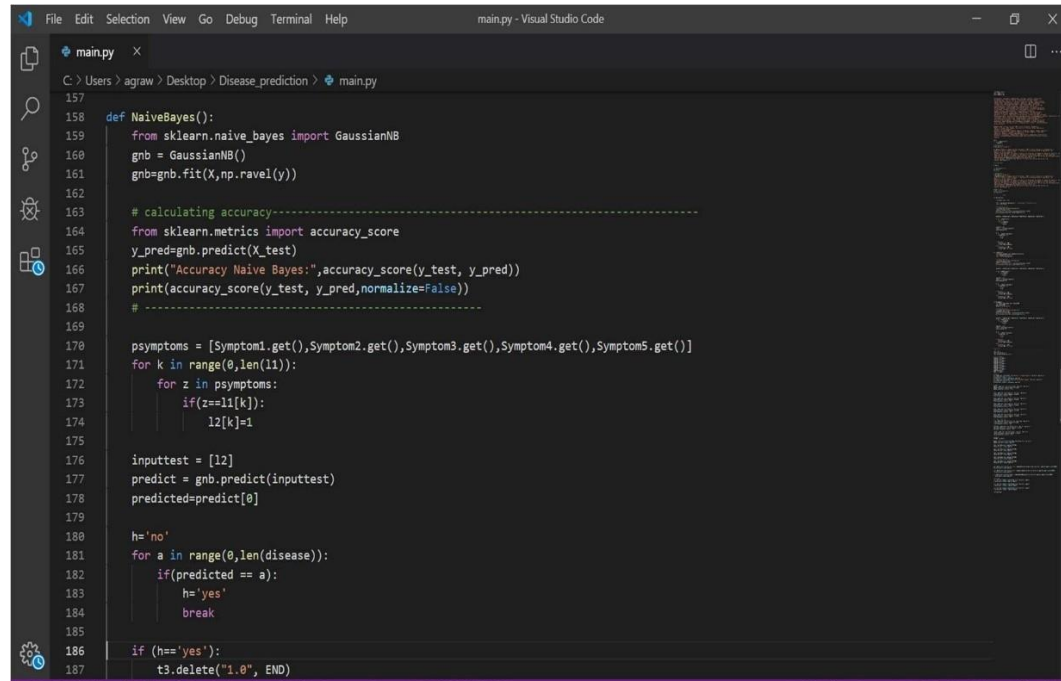
- A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap Aggregation, commonly known as **bagging**.
- The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.



```
120
121 def randomforest():
122     from sklearn.ensemble import RandomForestClassifier
123     clf4 = RandomForestClassifier()
124     clf4 = clf4.fit(X,np.ravel(y))
125
126     # calculating accuracy-----
127     from sklearn.metrics import accuracy_score
128     y_pred=clf4.predict(X_test)
129     print("Accuracy Random Forest:",accuracy_score(y_test, y_pred))
130     print(accuracy_score(y_test, y_pred,normalize=False))
131     # -----
132
133     symptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]
134
135     for k in range(0,len(l1)):
136         for z in psymptoms:
137             if(z==l1[k]):
138                 l2[k]=1
139
140     inputtest = [l2]
141     predict = clf4.predict(inputtest)
142     predicted=predict[0]
143
144     h='no'
145     for a in range(0,len(disease)):
146         if(predicted == a):
147             h='yes'
148             break
149
150     if (h=='yes'):
```

### c) Naïve Bayes Algorithm

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.



```
157
158 def NaiveBayes():
159     from sklearn.naive_bayes import GaussianNB
160     gnb = GaussianNB()
161     gnb=gnb.fit(X,np.ravel(y))
162
163     # calculating accuracy-----
164     from sklearn.metrics import accuracy_score
165     y_pred=gnb.predict(X_test)
166     print("Accuracy Naive Bayes:",accuracy_score(y_test, y_pred))
167     print(accuracy_score(y_test, y_pred,normalize=False))
168     # -----
169
170     psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]
171     for k in range(0,len(l1)):
172         for z in psymptoms:
173             if(z==l1[k]):
174                 l2[k]=1
175
176     inputtest = [l2]
177     predict = gnb.predict(inputtest)
178     predicted=predict[0]
179
180     h='no'
181     for a in range(0,len(disease)):
182         if(predicted == a):
183             h='yes'
184             break
185
186     if (h=='yes'):
187         t3.delete("1.0", END)
```



## 2) Module Evaluation and Testing

The goal of this step is to develop the simplest model able to formulate a target value fast and well enough. A data scientist can achieve this goal through model tuning. That's the optimization of model parameters to achieve an algorithm's best performance.

```
File Edit Selection View Go Debug Terminal Help main.py - Visual Studio Code

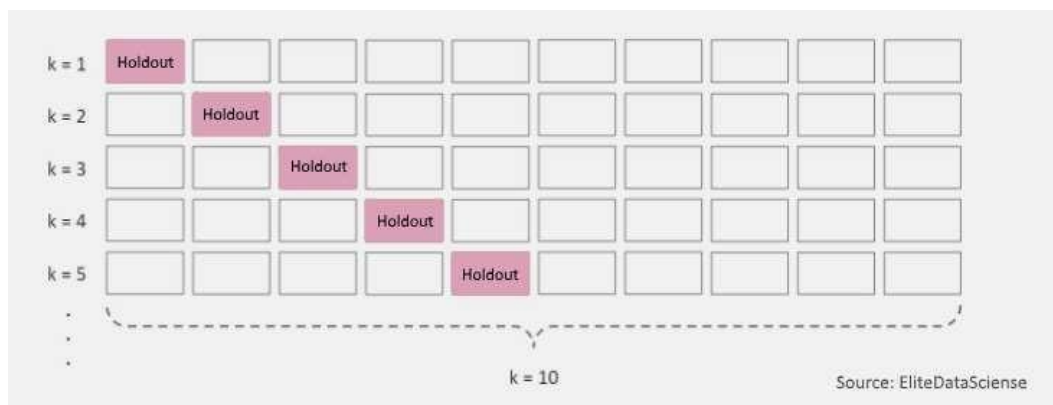
main.py x
C:\Users\agraw\Desktop> Disease_prediction> main.py > ...
13 |swollen_extremeties','excessive_hunger','extra_marital_contacts','drying_and_tingling_lips',
14 |'slurred_speech','knee_pain','hip_joint_pain','muscle_weakness','stiff_neck','swelling_joints',
15 |'movement_stiffness','spinning_movements','loss_of_balance','unsteadiness',
16 |'weakness_of_one_body_side','loss_of_smell','bladder_discomfort','foul_smell_of_urine',
17 |'continuous_feel_of_urine','passage_of_gases','internal_itching','toxic_look_typhos',
18 |'depression','irritability','muscle_pain','altered_sensorium','red_spots_over_body','belly_pain',
19 |'abnormal_menstruation','dischromic_patches','watering_from_eyes','increased_appetite','polyuria','family_history','mucoid_sputum',
20 |'rusty_sputum','lack_of_concentration','visual_disturbances','receiving_blood_transfusion',
21 |'receiving_unsterile_injections','coma','stomach_bleeding','distention_of_abdomen',
22 |'history_of_alcohol_consumption','fluid_overload','blood_in_sputum','prominent_veins_on_calf',
23 |'palpitations','painful_walking','pus_filled_pimples','blackheads','scurrying','skin_peeling',
24 |'silver_like_dusting','small_dents_in_nails','inflammatory_nails','blister','red_sore_around_nose',
25 |'yellow_crust_ooze']
26
27 |disease=['Fungal infection','Allergy','GERD','Chronic cholestasis','Drug Reaction',
28 |'Peptic ulcer disease','AIDS','Diabetes','Gastroenteritis','Bronchial Asthma','Hypertension',
29 |'Migraine','Cervical spondylosis',
30 |'Paralysis (brain hemorrhage)','Jaundice','Malaria','Chicken pox','Dengue','Typhoid','hepatitis A',
31 |'Hepatitis B','Hepatitis C','Hepatitis D','Hepatitis E','Alcoholic hepatitis','Tuberculosis',
32 |'Common Cold','Pneumonia','Dimorphic hemorrhoids(piles)',
33 |'Heartattack','Varicoseveins','Hypothyroidism','Hyperthyroidism','Hypoglycemia','Osteoarthritis',
34 |'Arthritis','(vertigo) Paroymsal Positional Vertigo','Acne','Urinary tract infection','Psoriasis',
35 |'Impetigo']
36
37 |l2=[]
38 |for x in range(0,len(l1)):
39 |    l2.append(0)
40
41 |# TESTING DATA df -----
42 |df=pd.read_csv("Training.csv")
43
```

```
File Edit Selection View Go Debug Terminal Help main.py - Visual Studio Code

main.py x
C:\Users\agraw\Desktop> Disease_prediction> main.py > ...
32 |'Common Cold','Pneumonia','Dimorphic hemorrhoids(piles)',
33 |'Heartattack','Varicoseveins','Hypothyroidism','Hyperthyroidism','Hypoglycemia','Osteoarthritis',
34 |'Arthritis','(vertigo) Paroymsal Positional Vertigo','Acne','Urinary tract infection','Psoriasis',
35 |'Impetigo']
36
37 |l2=[]
38 |for x in range(0,len(l1)):
39 |    l2.append(0)
40
41 |# TESTING DATA df -----
42 |df=pd.read_csv("Training.csv")
43
44 |df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,
45 |'Peptic ulcer disease':5,'AIDS':6,'Diabetes':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension':10,
46 |'Migraine':11,'Cervical spondylosis':12,
47 |'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
48 |'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,
49 |'Common Cold':26,'Pneumonia':27,'Dimorphic hemorrhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,
50 |'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthritis':34,'Arthritis':35,
51 |'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,
52 |'Impetigo':40}},inplace=True)
53
54 |# print(df.head())
55
56 |X= df[l1]
57
58 |y = df[['prognosis']]
59 |np.ravel(y)
60 |# print(y)
61
62 |# TRAINING DATA tr -----
```

### *Cross-validation:*

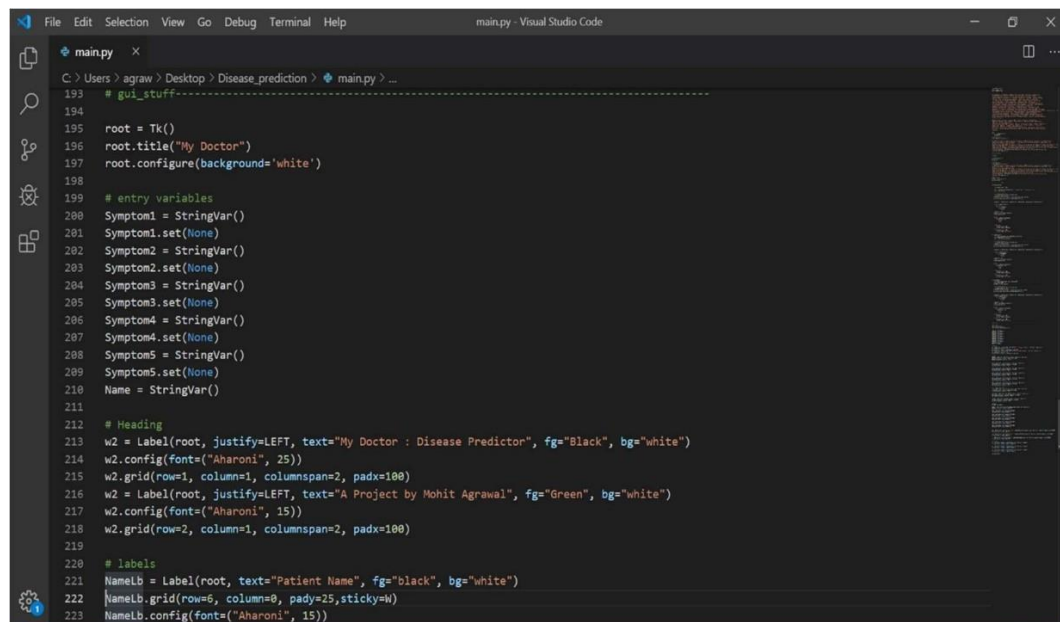
Cross-validation is the most commonly used tuning method. It entails splitting a training dataset into ten equal parts (folds). A given model is trained on only nine folds and then tested on the tenth one (the one previously left out). Training continues until every fold is left aside and used for testing. As a result of model performance measure, a specialist calculates a cross-validated score for each set of hyper parameters. A data scientist trains models with different sets of hyper parameters to define which model has the highest prediction accuracy. The cross-validated score indicates average model performance across ten hold-out folds.



# 1. Model Deployment

The model deployment stage covers putting a model into production use.

Once a *data scientist* has chosen a reliable model and specified its performance requirements, he or she delegates its deployment to a *data engineer* or *database administrator*. The distribution of roles depends on your organization's structure and the amount of data you store.



```
File Edit Selection View Go Debug Terminal Help main.py - Visual Studio Code
main.py x
C:\Users\agraw> Desktop > Disease_prediction > main.py > ...
193 # gui_stuff-----
194
195 root = Tk()
196 root.title("My Doctor")
197 root.configure(background='white')
198
199 # entry variables
200 Symptom1 = StringVar()
201 Symptom1.set(None)
202 Symptom2 = StringVar()
203 Symptom2.set(None)
204 Symptom3 = StringVar()
205 Symptom3.set(None)
206 Symptom4 = StringVar()
207 Symptom4.set(None)
208 Symptom5 = StringVar()
209 Symptom5.set(None)
210 Name = StringVar()
211
212 # Heading
213 w2 = Label(root, justify=LEFT, text="My Doctor : Disease Predictor", fg="black", bg="white")
214 w2.config(font=("Aharoni", 25))
215 w2.grid(row=1, column=1, columnspan=2, padx=100)
216 w2 = Label(root, justify=LEFT, text="A Project by Mohit Agrawal", fg="Green", bg="white")
217 w2.config(font=("Aharoni", 15))
218 w2.grid(row=2, column=1, columnspan=2, padx=100)
219
220 # labels
221 NameLb = Label(root, text="Patient Name", fg="black", bg="white")
222 NameLb.grid(row=6, column=0, pady=25, sticky=W)
223 NameLb.config(font=("Aharoni", 15))
```

## TKINTER in Python

### Create GUI Window

```
# gui_stuff.....  
  
root = Tk()  
root.title("My Doctor")
```

### Heading in Window

```
# Heading  
  
w2 = Label(root, justify=LEFT, text="My Doctor : Disease Predictor", fg="Black", bg="white")  
  
w2.config(font=("Aharoni", 25))  
w2.grid(row=1, column=1, columnspan=2, padx=100)  
  
w2 = Label(root, justify=LEFT, text="A Project by Mohit Agrawal", fg="Green", bg="white")
```

### Create Levels for Symptoms

```
# labels

NameLb = Label(root, text="Patient Name", fg="black", bg="white")
NameLb.grid(row=6, column=0, pady=25, sticky=W)
NameLb.config(font=("Aharoni", 15))


S1Lb = Label(root, text="Symptom 1", fg="black", bg="white")
S1Lb.grid(row=7, column=0, pady=20, sticky=W)
S1Lb.config(font=("Aharoni", 15))
```

```

S2Lb = Label(root, text="Symptom 2", fg="black", bg="white")
S2Lb.grid(row=8, column=0, pady=20, sticky=W)
S2Lb.config(font=("Aharoni", 15))

S3Lb = Label(root, text="Symptom 3", fg="black", bg="white")
S3Lb.grid(row=9, column=0, pady=20, sticky=W)
S3Lb.config(font=("Aharoni", 15))

S4Lb = Label(root, text="Symptom 4", fg="black", bg="white")
S4Lb.grid(row=10, column=0, pady=20, sticky=W)
S4Lb.config(font=("Aharoni", 15))

S5Lb = Label(root, text="Symptom 5", fg="black", bg="white")

```

## List View

```

# entries

OPTIONS = sorted(l1)

NameEn = Entry(root, textvariable=Name, width=50, bg="black", fg="white")
NameEn.grid(row=6, column=1, padx=10)

S1En = OptionMenu(root, Symptom1, *OPTIONS)
S1En.grid(row=7, column=1, padx=10)

S2En = OptionMenu(root, Symptom2, *OPTIONS)
S2En.grid(row=8, column=1, padx=10)

S3En = OptionMenu(root, Symptom3, *OPTIONS)
S3En.grid(row=9, column=1, padx=10)

S4En = OptionMenu(root, Symptom4, *OPTIONS)
S4En.grid(row=10, column=1, padx=10)

```

## Button

```
dst = Button(root, text="Decision Tree", command=DecisionTree,bg="orange",
fg="white", padx=10, pady=5,relief=RIDGE)

dst.grid(row=8, column=2,padx=10)

rnf = Button(root, text="Random Forest", command=randomforest,bg="red",fg=
"white",padx=10, pady=5,relief=RIDGE)

rnf.grid(row=9, column=2,padx=10)

lr = Button(root, text="Naive Bayes", command=NaiveBayes,bg="blue",fg="whi
```

## Text Fields

```
#textfileds

t1 = Text(root, height=1, width=40,bg="black",fg="white", pady=5)
t1.grid(row=15, column=1, padx=10, pady=5)

t2 = Text(root, height=1, width=40,bg="black",fg="white", pady=5)
t2.grid(row=17, column=1 , padx=10, pady=5)

t3 = Text(root, height=1, width=40,bg="black",fg="white", pady=5)
```

## Disease Predictor Prototype

- This Machine Learning project is used to predict the disease based on the symptoms given by the user. So, the output is accurate.
- The patient can fill up to 5 symptoms and based on these symptoms Machine Learning will predict disease.
- It predicts disease by using **three different machine learning algorithms**.
- It uses **tkinter** for GUI and **Numpy, Pandas** for data mining.

### My Doctor : Disease Predictor A Project by Kalpana Devi

Patient Name	<input type="text"/>	
Symptom 1	<input type="text" value="None"/>	
Symptom 2	<input type="text" value="None"/>	<input type="button" value="Decision Tree"/>
Symptom 3	<input type="text" value="None"/>	<input type="button" value="Random Forest"/>
Symptom 4	<input type="text" value="None"/>	<input type="button" value="Naive Bayes"/>
Symptom 5	<input type="text" value="None"/>	
Decision Tree	<input type="text"/>	
Random Forest	<input type="text"/>	
Naive Bayes	<input type="text"/>	



### Testing Set:

A test set is needed for an evaluation of the trained model and its capability for generalization. The latter means a model's ability to identify patterns in new unseen data after having been trained over a training data. It's crucial to use different subsets for training and testing to avoid model overfitting, which is the incapacity for generalization we mentioned above.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	itching	skin_rash	nodal_skin	continuous	shivering	chills	joint_pain	stomach	acidity	ulcers_on	muscle	w/vomiting	burning	m/spotting	t/fatigue	weight_ga	anxiety	cold_hand	mood_swi	weight_lo	restlessne	lethargy	patches
2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0
5	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1
10	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
16	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0
17	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
18	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
19	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
20	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
22	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0
28	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
29	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### Validation Set:

The purpose of a validation set is to tweak a model's hyper parameters — higher-level structural settings that can't be directly learned from data. These settings can express, for instance, how complex a model is and how fast it finds patterns in data.

