DISEASE PREDICTION BY SYMPTOMS USING AIML

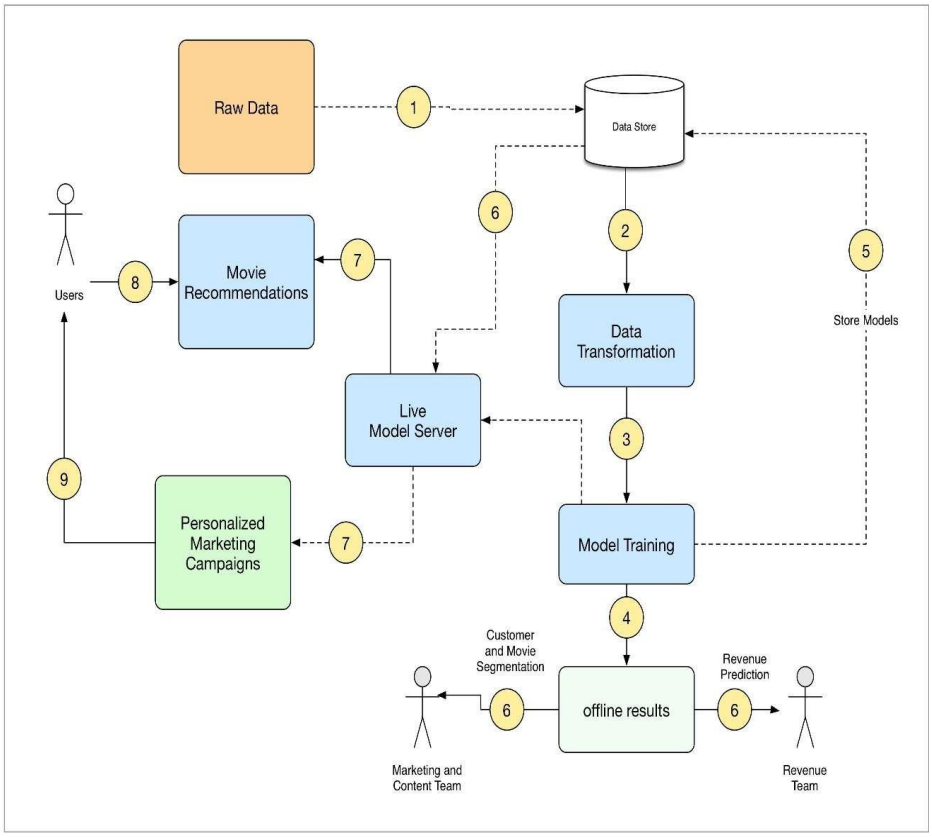
Done by M.Kalpana Devi

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

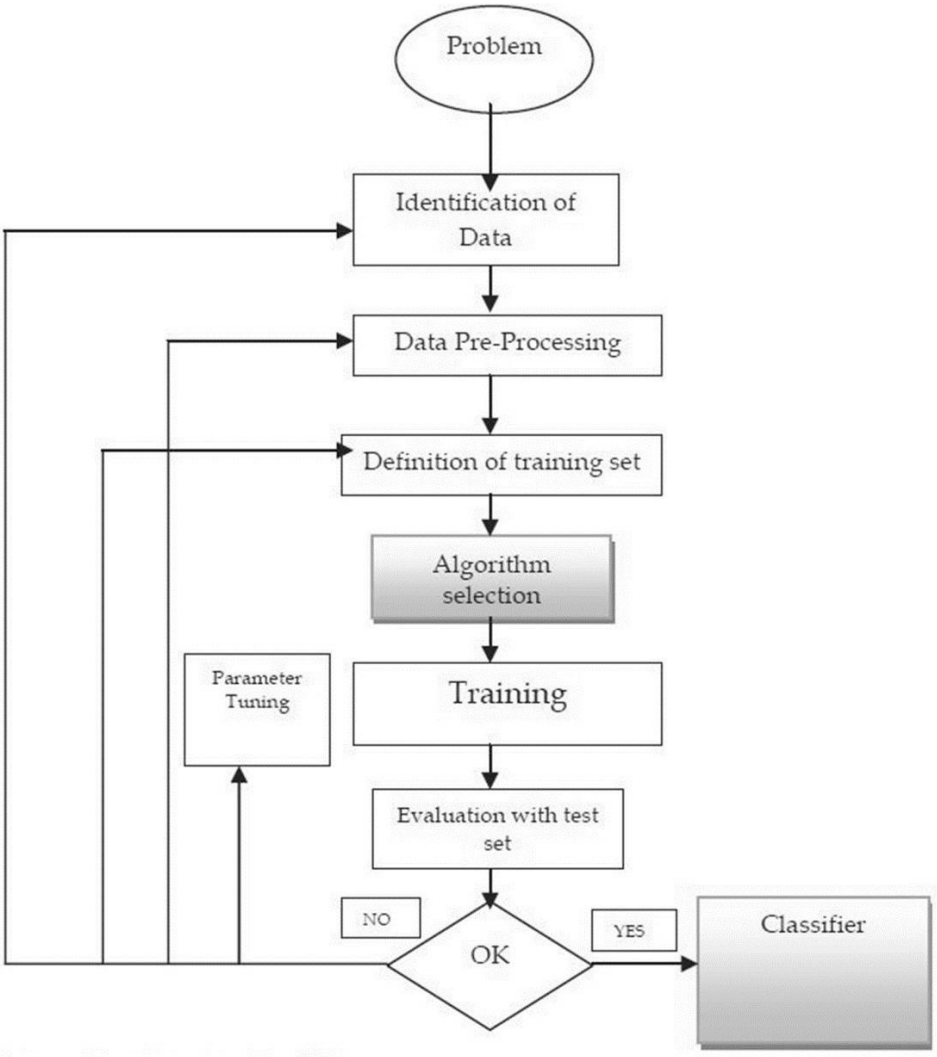
Machine Learning Architecture



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

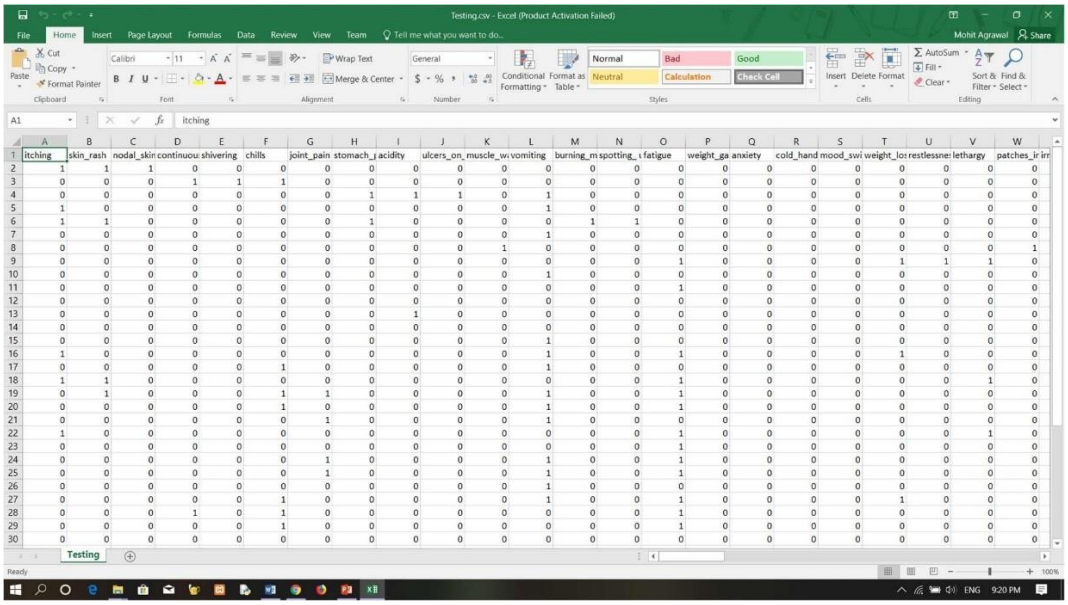
It's time for a data analyst to pick up the baton and lead the way to machine learning implementation. The job of a data analyst is to find ways and sources of collecting relevant and comprehensive data, interpreting it, and analyzing results with the help of statistical techniques.

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.

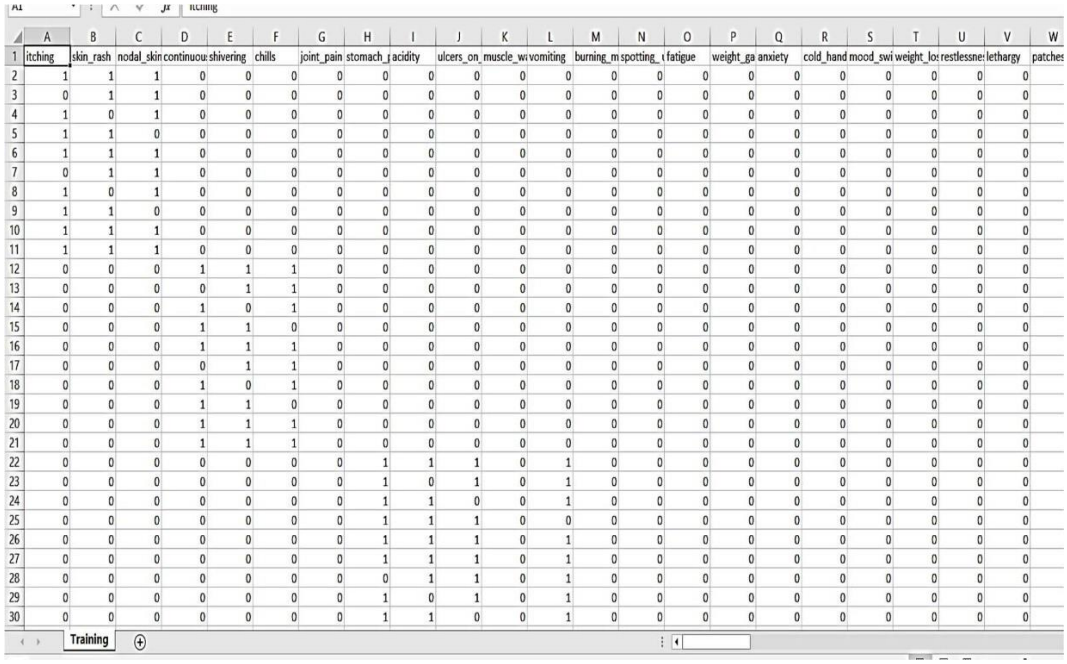


Dataset Splitting

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.

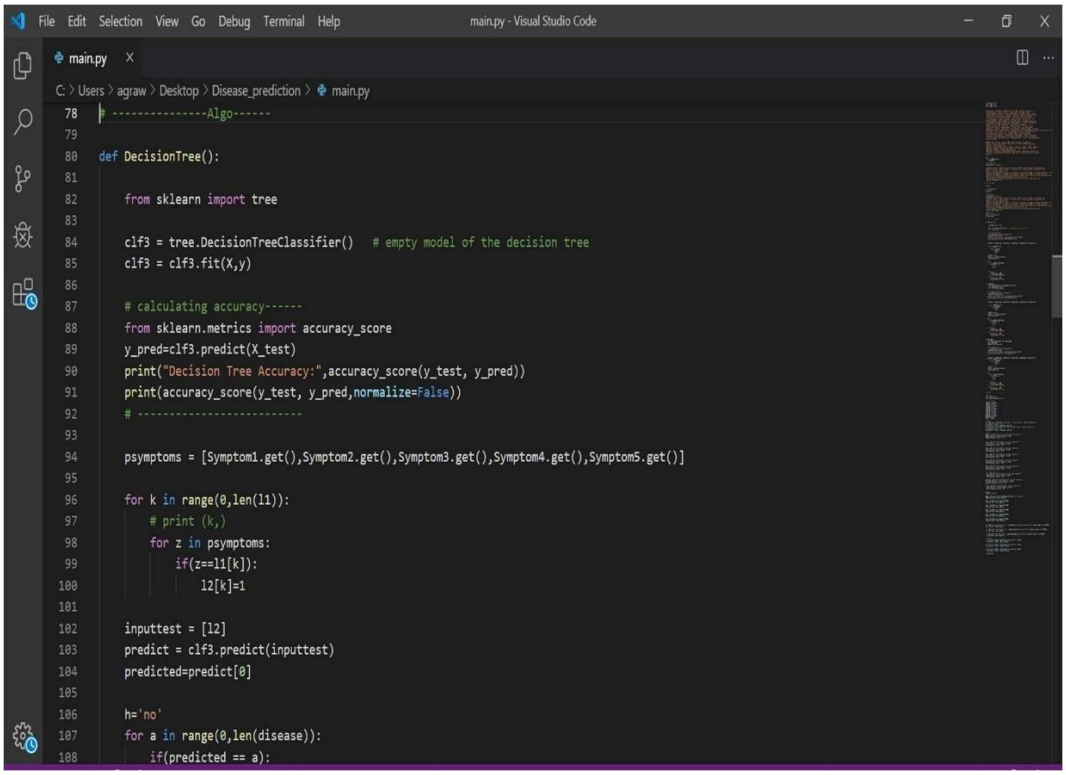


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	itching	skin_rash	nodal_skin	continuous	shivering	chills	joint_pain	stomach_p	acidity	ulcers_on_	muscle_wa	vomiting	burning_m	spotting_u	fatigue	weight_ga	anxiety	cold_hand	mood_swi	weight_lo	restlessne	lethargy	patches
2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
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	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	
23	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	
25	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	
26	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	
27	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	
28	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	
29	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	
30	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	

Training

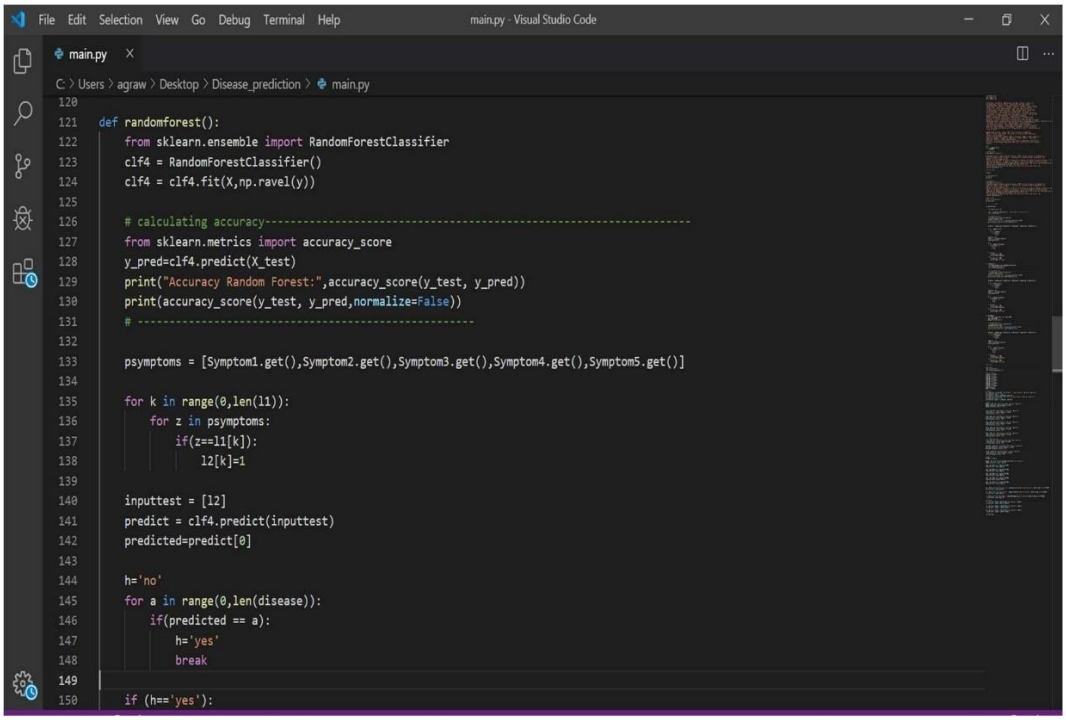
a) Decision Tree Algorithm

- 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.



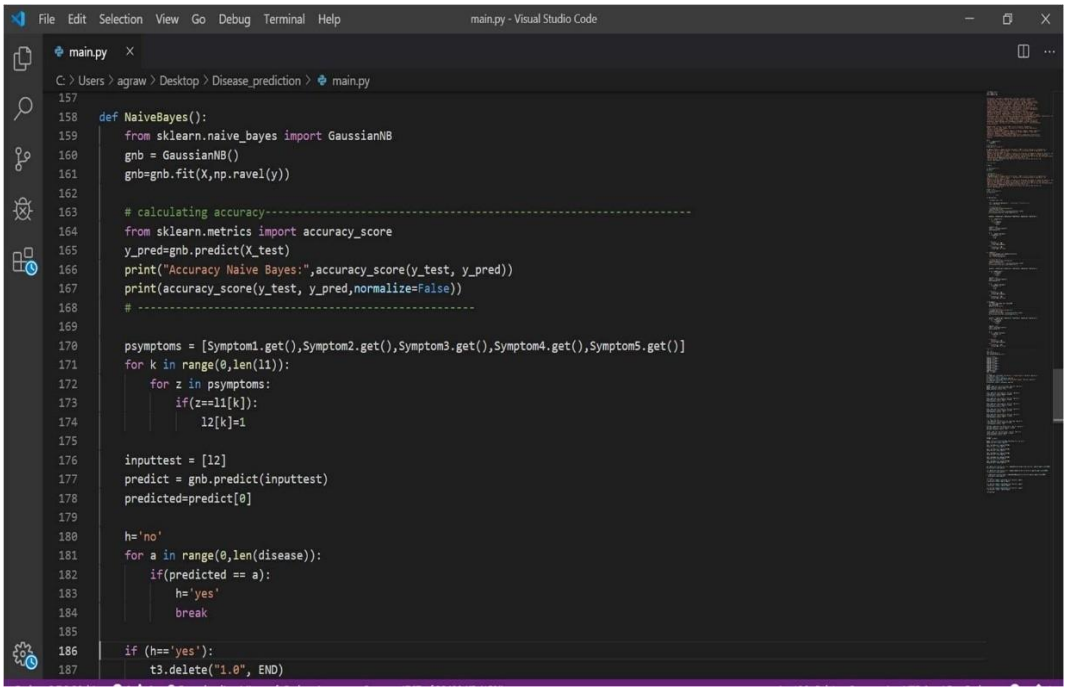
b) Random Forest Algorithm

- 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.



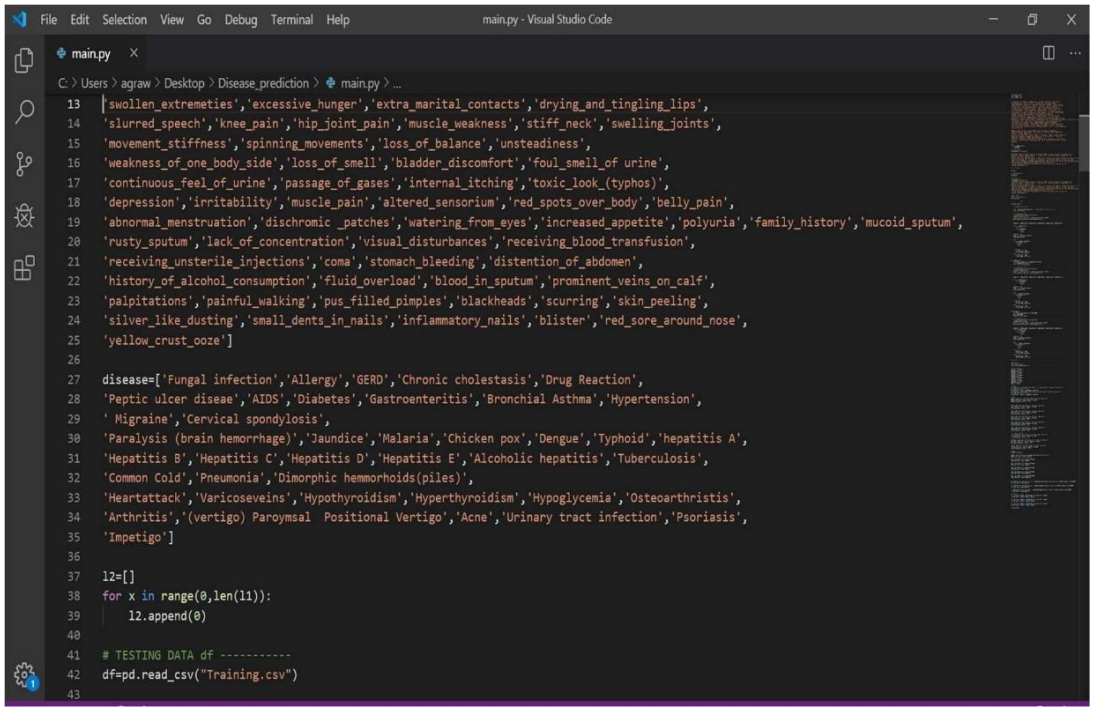
c) Naïve Bayer Algorithm

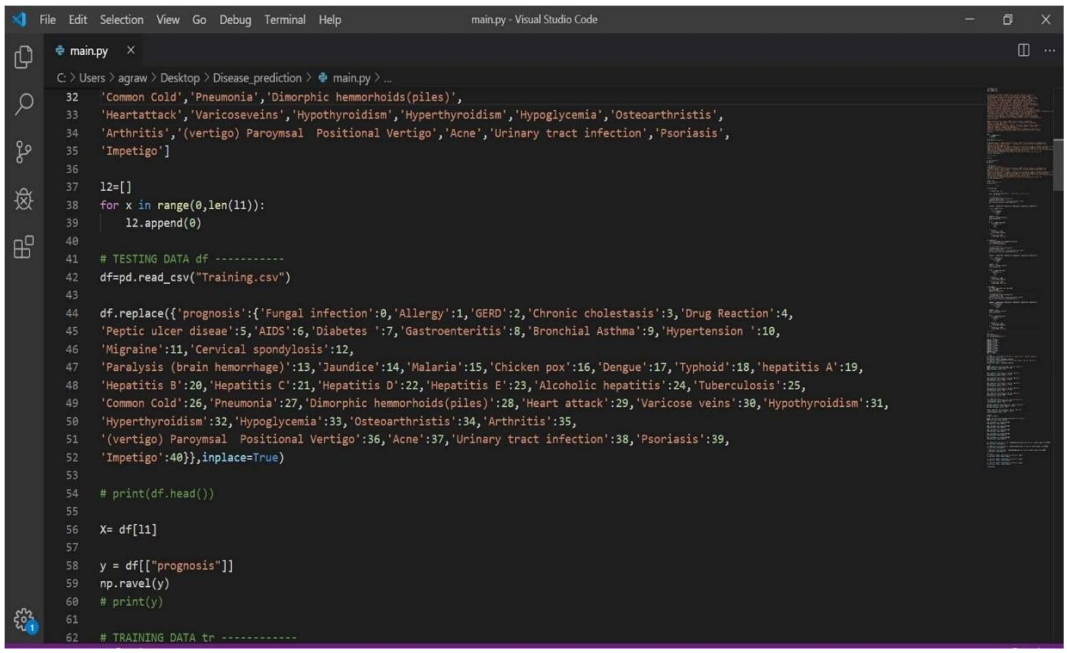
Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.



2) Module Evaluation and Testing

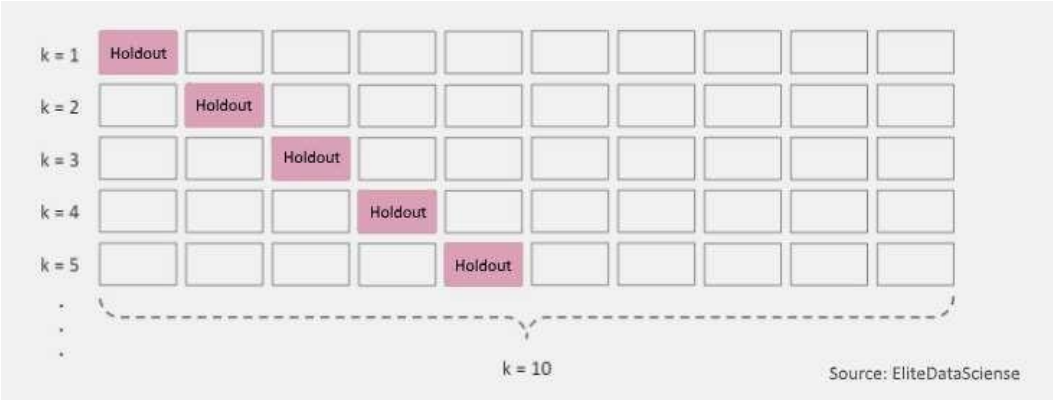
The goal of this step is to develop the simplest model able to formulate a target value fast and well enough. A data scientist can achieve this goal through model tuning. That's the optimization of model parameters to achieve an algorithm's best performance.





Cross-validation:

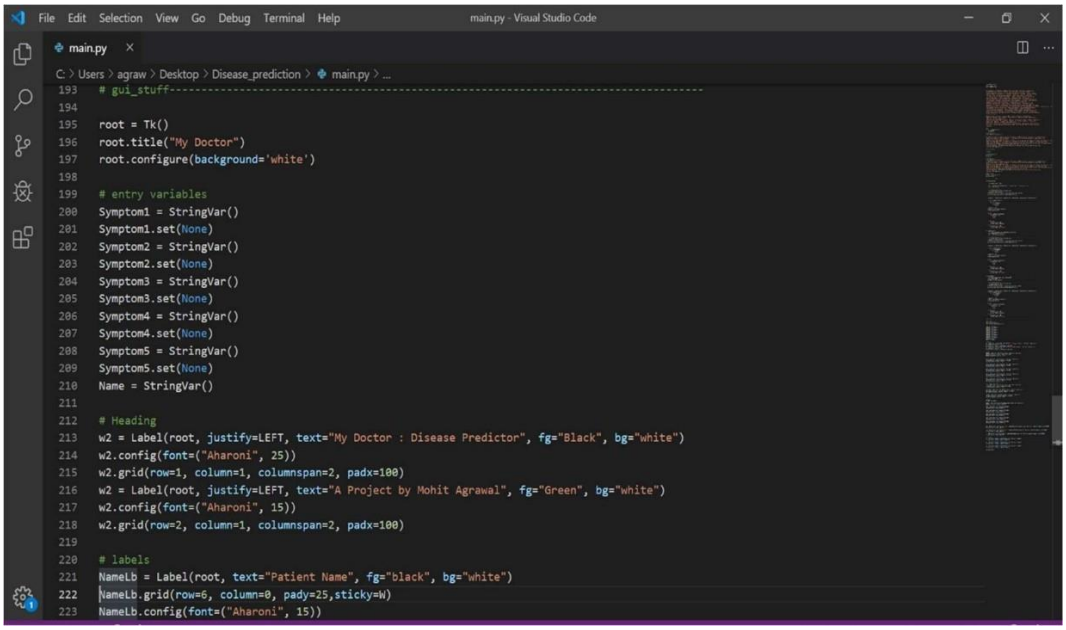
Cross-validation is the most commonly used tuning method. It entails splitting a training dataset into ten equal parts (folds). A given model is trained on only nine folds and then tested on the tenth one (the one previously left out). Training continues until every fold is left aside and used for testing. 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.



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

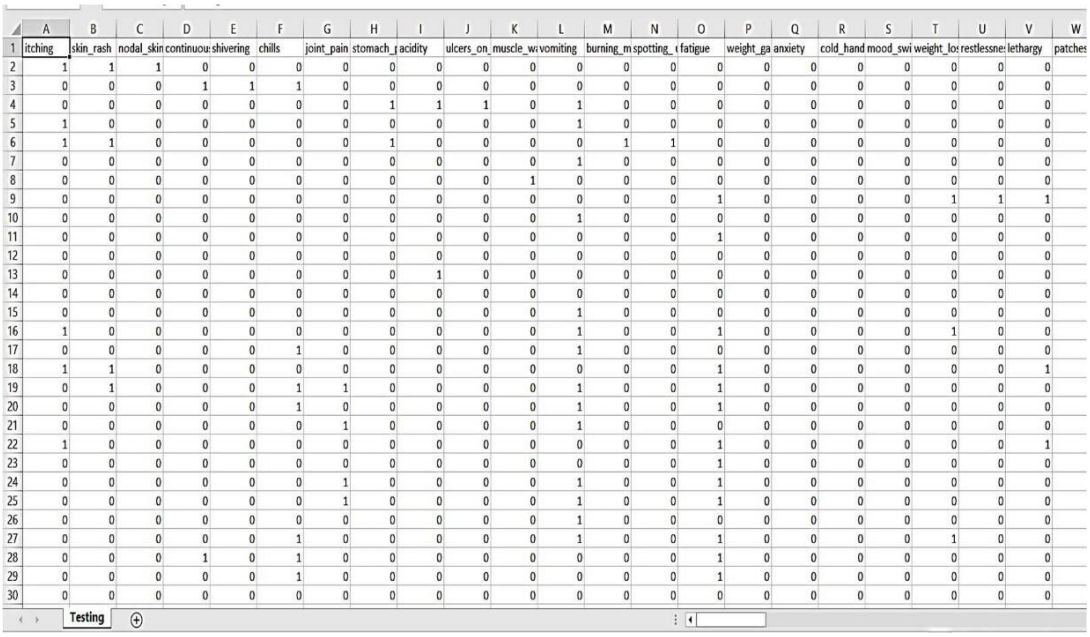
```
#textfileds

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.

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_p	acidity	ulcers_on_	muscle_wa	vomiting	burning_m	spotting_u	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	
3	0	0	0	1	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	0	1	0	0	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	0	1	1	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	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	1	0	0	0	0	1	1	1	
10	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	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	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	
15	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	1	0	0	1	0	0	0	0	1	0	0	
17	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	1	0	0	0	0	0	0	1	
19	0	1	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	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	
21	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	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	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	
25	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	1	0	0	0	0	0	0	0	0	0	0	
27	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	1	0	0	0	0	0	0	0	
29	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	

Testing

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.