MACHINE LEARNING PROJECT DSCI-6003-01

Disease Prediction from Symptoms

CHETANA

WHAT AND WHY?

- What is Disease Prediction from Symptoms?

→ It is a machine learning application used to predict the diseases from symptoms.

- Why do we need this Application?

→We can predict the disease in early stages.

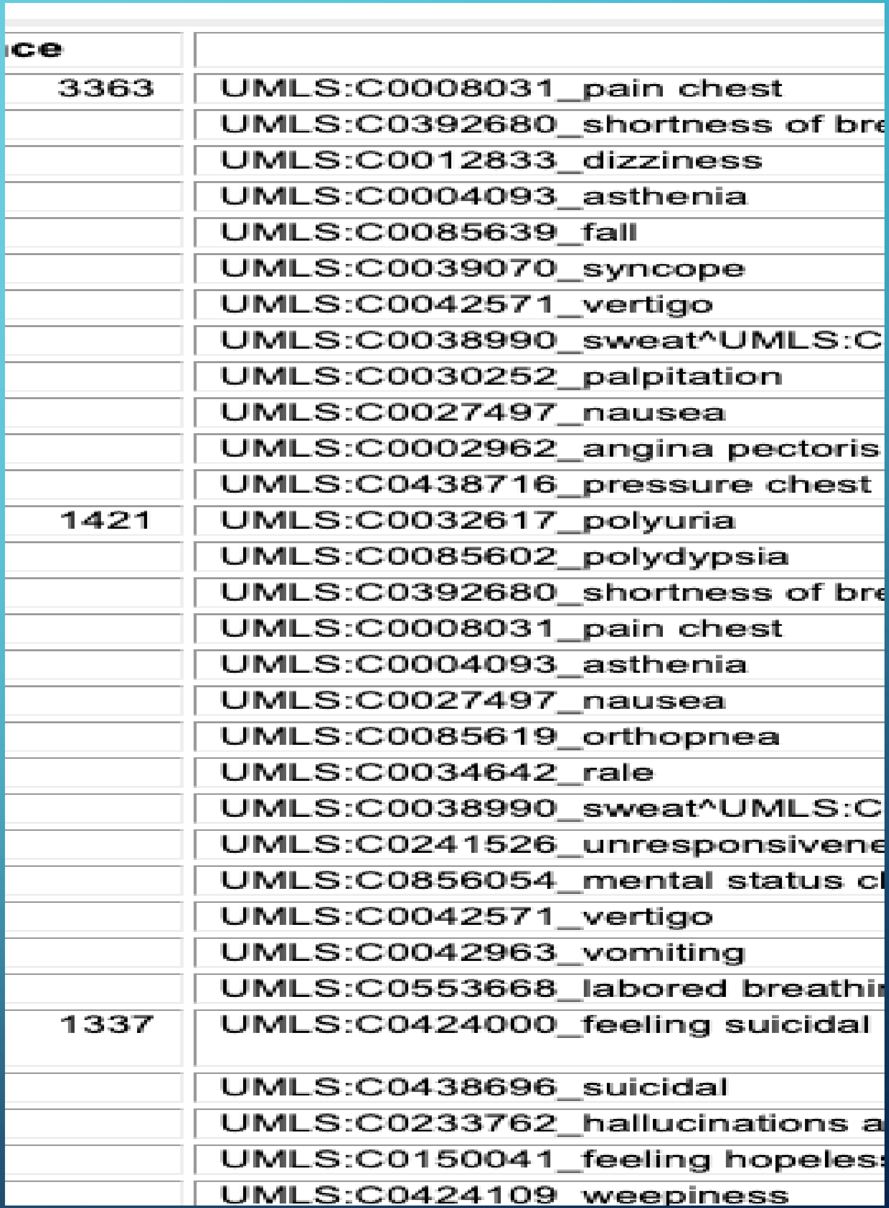
→We can utilize the resources efficiently i.e.; we can prioritize patients who need urgent medical attention.

→It helps doctors to find accurate diagnosis, which can lead to a better treatment

→It helps the patients more engaged in their own health, By providing patients with access to information about their symptoms, conditions, and potential health risks, an application can help them to make better decisions about their health and well-being.

DATASET SELECTION

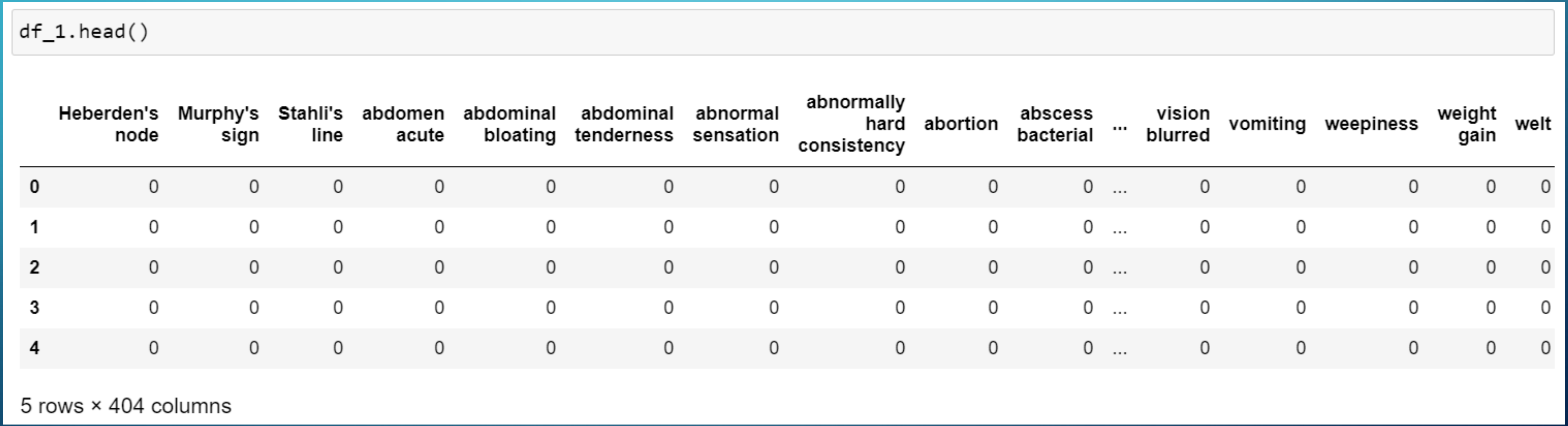
- The dataset is a public dataset acquired from Kaggle. It is a raw dataset which consists of Disease, Count of disease occurrence and Symptoms.
- https://www.kaggle.com/datasets/itachi9604/disease-symptom-description-dataset



ce	
3363	UMLS:C0008031_pain chest
	UMLS:C0392680_shortness of bre
	UMLS:C0012833_dizziness
	UMLS:C0004093_asthenia
	UMLS:C0085639_fall
	UMLS:C0039070_syncope
	UMLS:C0042571_vertigo
	UMLS:C0038990_sweat^UMLS:C
	UMLS:C0030252_palpitation
	UMLS:C0027497_nausea
	UMLS:C0002962_angina pectoris
	UMLS:C0438716_pressure chest
1421	UMLS:C0032617_polyuria
	UMLS:C0085602_polydypsia
	UMLS:C0392680_shortness of bre
	UMLS:C0008031_pain chest
	UMLS:C0004093_asthenia
	UMLS:C0027497_nausea
	UMLS:C0085619_orthopnea
	UMLS:C0034642_rale
	UMLS:C0038990_sweat^UMLS:C
	UMLS:C0241526_unresponsivene
	UMLS:C0856054_mental status c
	UMLS:C0042571_vertigo
	UMLS:C0042963_vomiting
	UMLS:C0553668_labored breathi
1337	UMLS:C0424000_feeling suicidal
	UMLS:C0438696_suicidal
	UMLS:C0233762_hallucinations a
	UMLS:C0150041_feeling hopeles
	UMLS:C0424109_weepiness

DATA PREPROCESSING

- Final data after cleaning the dataset.



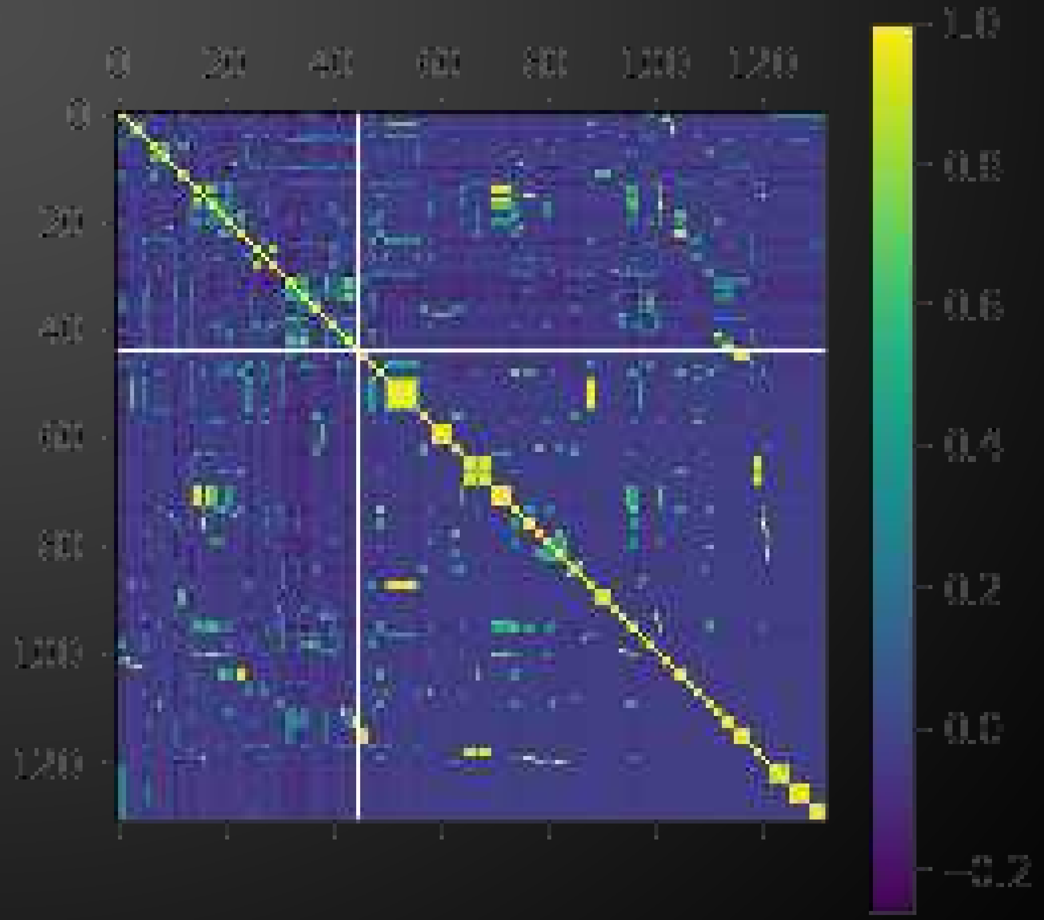
```
df_1.head()
```

	Heberden's node	Murphy's sign	Stahli's line	abdomen acute	abdominal bloating	abdominal tenderness	abnormal sensation	abnormally hard consistency	abortion	abscess bacterial	...	vision blurred	vomiting	weepiness	weight gain	welt
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0

5 rows × 404 columns

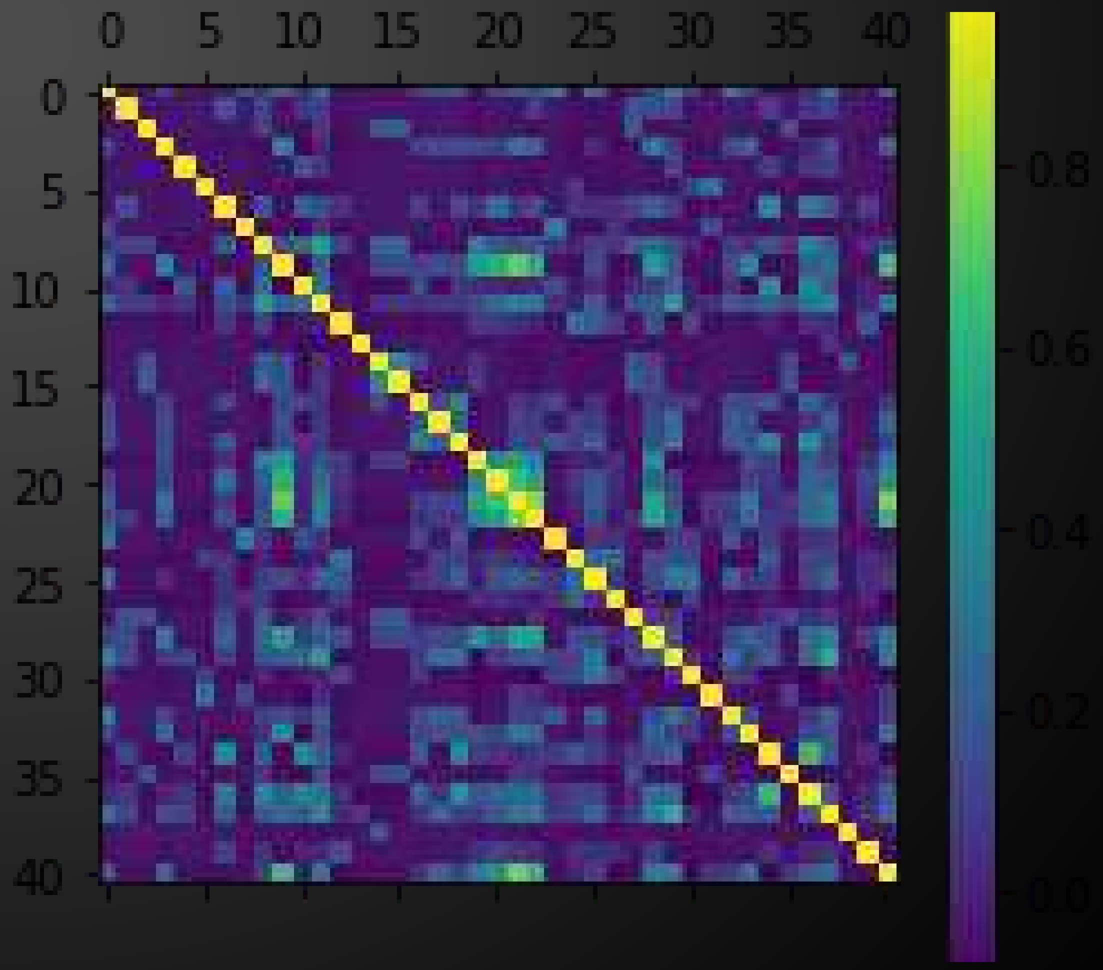
DATA VISUALIZATIONS

- Heatmap showing correlation between the symptoms.



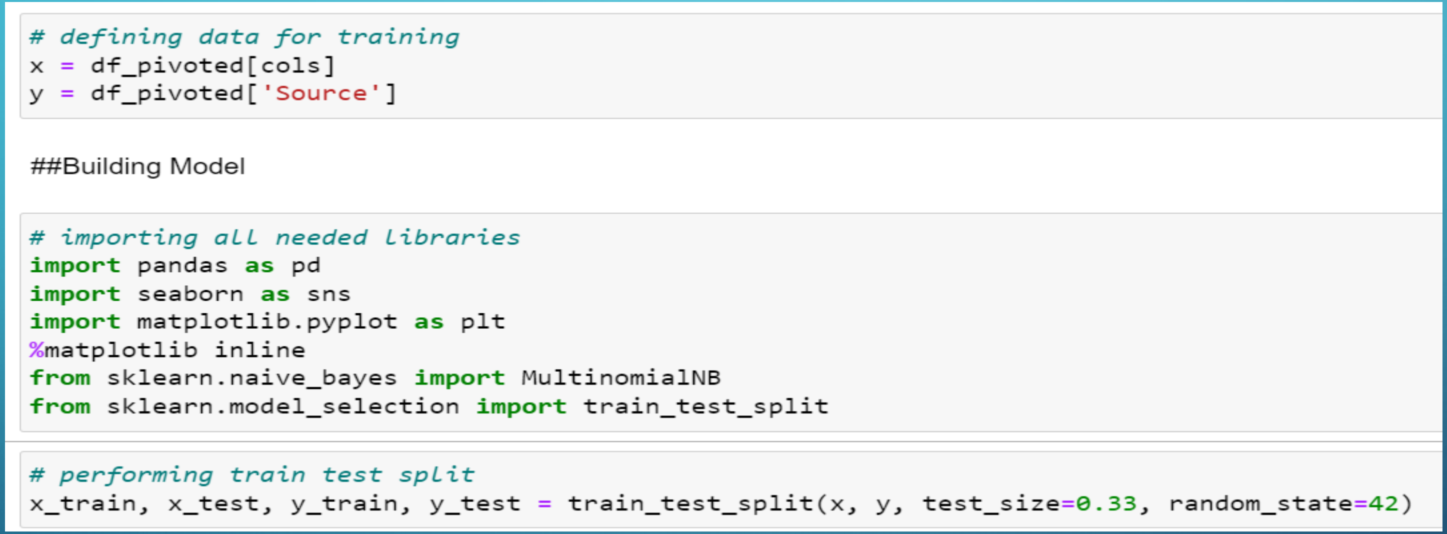
DATA VISUALIZATIONS

- Heatmap showing the correlation between the diseases.



PERFORMING TRAIN TEST SPLIT

- I had divided data for x and y to perform train and test the dataset.



```
# defining data for training
x = df_pivoted[cols]
y = df_pivoted['Source']
```

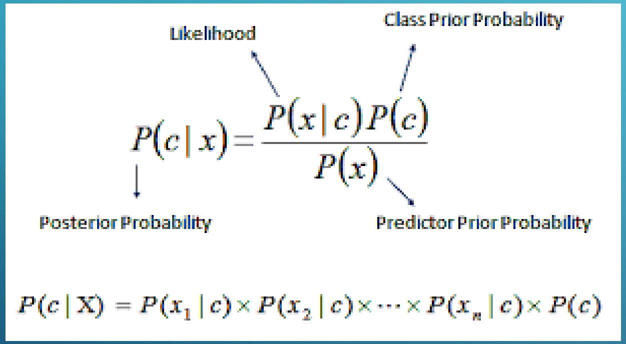
##Building Model

```
# importing all needed libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
```

```
# performing train test split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=42)
```

ALGORITHMS USED

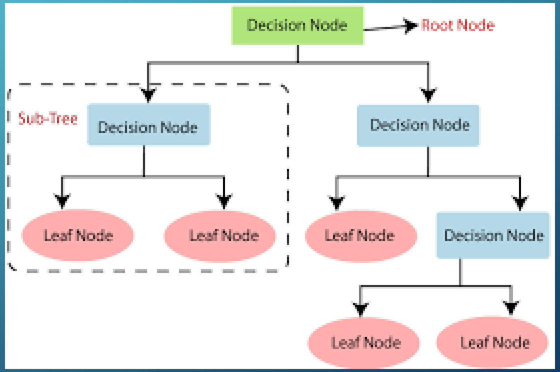
- Multinomial Bayes Classifier.



$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

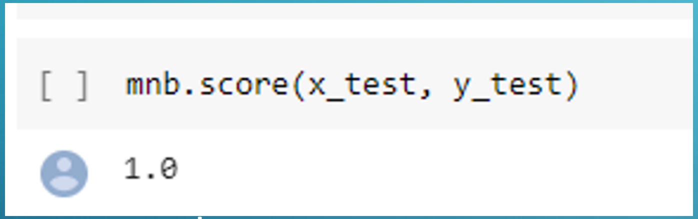
- Decision Tree Classifier.



IMPLEMENTATION

IMPLEMENTATION ON MULTINOMIAL BAYES CLASSIFIER:

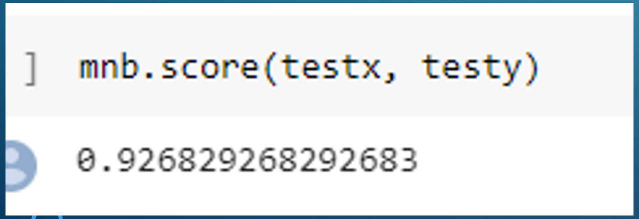
- Training results



```
[ ]  mnb.score(x_test, y_test)

     1.0
```

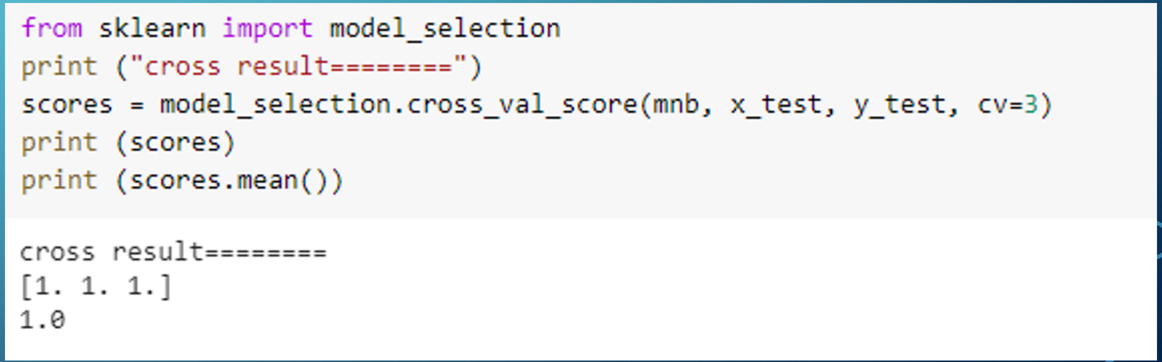
- Test results



```
]  mnb.score(testx, testy)

   0.926829268292683
```

- Validation results



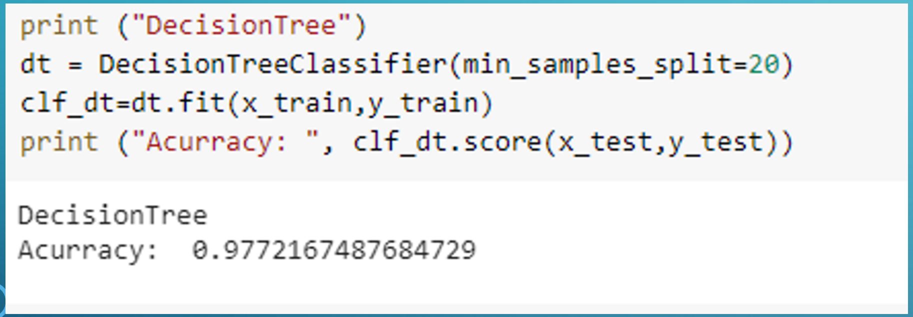
```
from sklearn import model_selection
print ("cross result========")
scores = model_selection.cross_val_score(mnb, x_test, y_test, cv=3)
print (scores)
print (scores.mean())

cross result========
[1. 1. 1.]
1.0
```

IMPLEMENTATION

IMPLEMENTATION ON DECISION TREE CLASSIFIER:

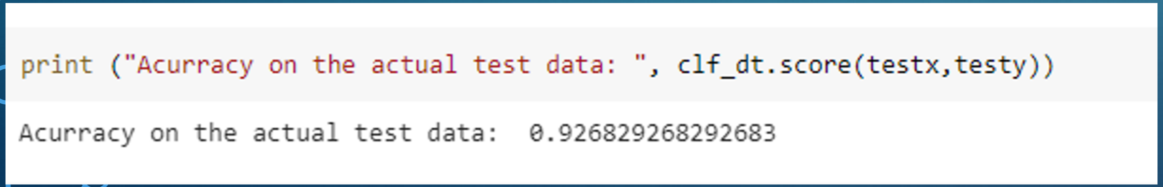
- Training results



```
print ("DecisionTree")
dt = DecisionTreeClassifier(min_samples_split=20)
clf_dt=dt.fit(x_train,y_train)
print ("Acurracy: ", clf_dt.score(x_test,y_test))
```

```
DecisionTree
Acurracy:  0.9772167487684729
```

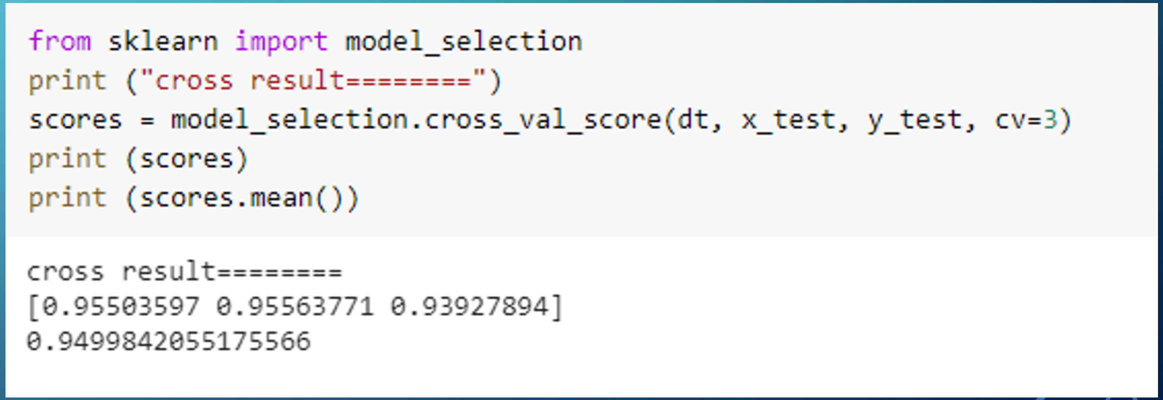
- Testing results



```
print ("Acurracy on the actual test data: ", clf_dt.score(testx,testy))
```

```
Acurracy on the actual test data:  0.926829268292683
```

- Validation results



```
from sklearn import model_selection
print ("cross result========")
scores = model_selection.cross_val_score(dt, x_test, y_test, cv=3)
print (scores)
print (scores.mean())
```

```
cross result========
[0.95503597 0.95563771 0.93927894]
0.9499842055175566
```

EVALUATION & RESULTS

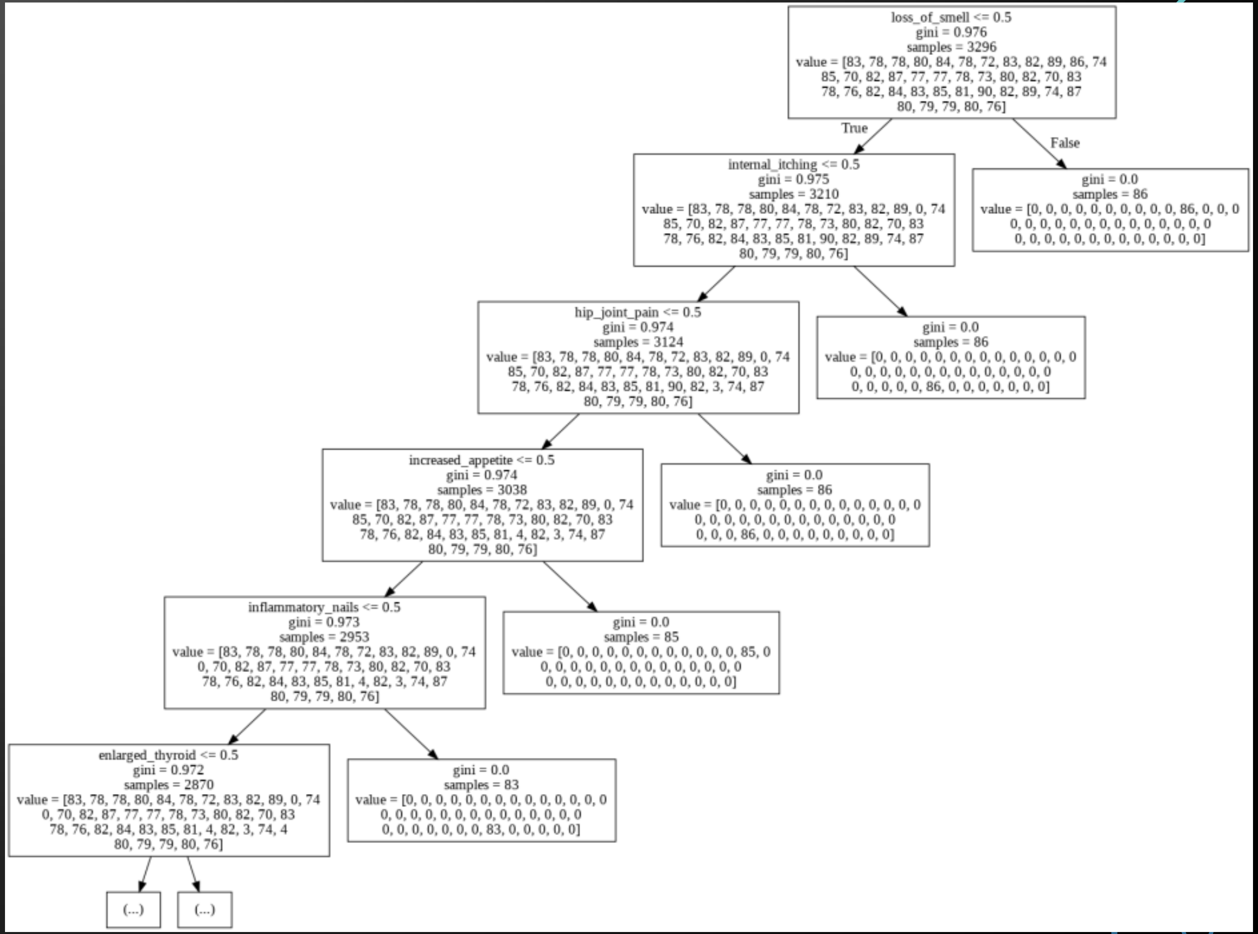
- On our training and testing sets, we ran the two methods, and the results are displayed above. We now concentrate on determining how significant the model considers each symptom (input) to be. This study makes use of a component called importance, which measures how much the input of the variable contributed to the projected outcome. This fits the following description.
- Every node in the decision tree has an associated importance metric. It evaluates the impurity weighted by weighted inputs reduction. The parent's contribution is then subtracted from the children's contribution. As a result, we arrive at the formula below. This approach is employed in widespread implementations like sklearn.

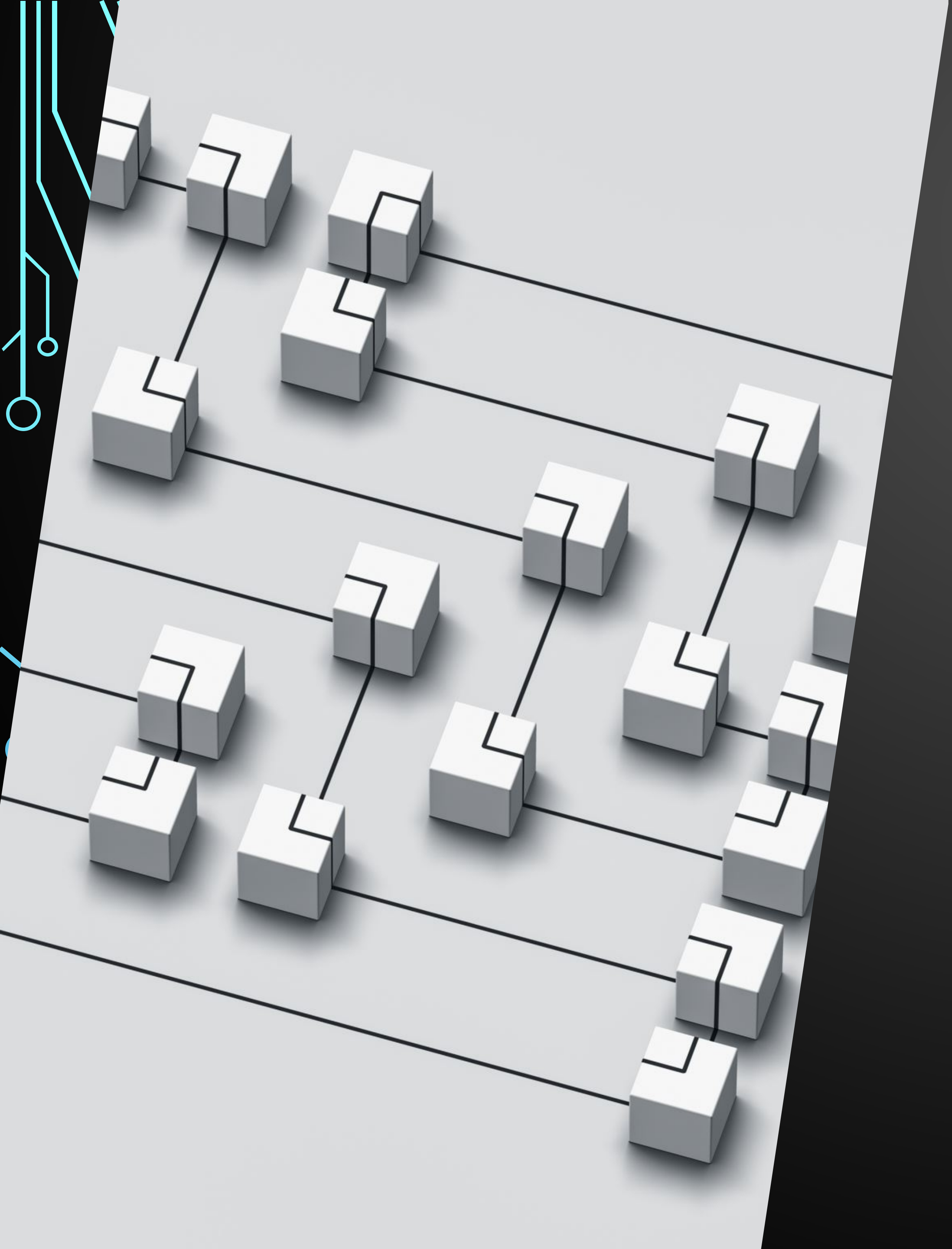
$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)}$$

EVALUATION AND RESULTS :

- We thus obtain the following results with decision tree.
- These are the top10 symptoms with their importance in brackets:

1. feature 88 - loss_of_smell (0.026973)
2. feature 93 - internal_itching (0.026919)
3. feature 79 - hip_joint_pain (0.026895)
4. feature 104 - increased_appetite (0.026657)
5. feature 48 - malaise (0.026156)
6. feature 128 - inflammatory_nails (0.025972)
7. feature 71 - enlarged_thyroid (0.025737)
8. feature 118 - blood_in_sputum (0.025126)
9. feature 131 - yellow_crust_ooze (0.025049)
10. feature 2 - nodal_skin_eruptions (0.024985)


loss_of_smell <= 0.5 gini = 0.976 samples = 3296 value = [83, 78, 78, 80, 84, 78, 72, 83, 82, 89, 86, 74 85, 70, 82, 87, 77, 77, 78, 73, 80, 82, 70, 83 78, 76, 82, 84, 83, 85, 81, 90, 82, 89, 74, 87 80, 79, 79, 80, 76]
True
False
internal_itching <= 0.5 gini = 0.975 samples = 3210 value = [83, 78, 78, 80, 84, 78, 72, 83, 82, 89, 0, 74 85, 70, 82, 87, 77, 77, 78, 73, 80, 82, 70, 83 78, 76, 82, 84, 83, 85, 81, 90, 82, 89, 74, 87 80, 79, 79, 80, 76]
gini = 0.0 samples = 86 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 86, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
hip_joint_pain <= 0.5 gini = 0.974 samples = 3124 value = [83, 78, 78, 80, 84, 78, 72, 83, 82, 89, 0, 74 85, 70, 82, 87, 77, 77, 78, 73, 80, 82, 70, 83 78, 76, 82, 84, 83, 85, 81, 90, 82, 3, 74, 87 80, 79, 79, 80, 76]
gini = 0.0 samples = 86 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 86, 0, 0, 0, 0, 0, 0, 0]
increased_appetite <= 0.5 gini = 0.974 samples = 3038 value = [83, 78, 78, 80, 84, 78, 72, 83, 82, 89, 0, 74 85, 70, 82, 87, 77, 77, 78, 73, 80, 82, 70, 83 78, 76, 82, 84, 83, 85, 81, 4, 82, 3, 74, 87 80, 79, 79, 80, 76]
gini = 0.0 samples = 86 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 86, 0, 0, 0, 0, 0, 0, 0, 0, 0]
inflammatory_nails <= 0.5 gini = 0.973 samples = 2953 value = [83, 78, 78, 80, 84, 78, 72, 83, 82, 89, 0, 74 0, 70, 82, 87, 77, 77, 78, 73, 80, 82, 70, 83 78, 76, 82, 84, 83, 85, 81, 4, 82, 3, 74, 87 80, 79, 79, 80, 76]
gini = 0.0 samples = 85 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 85, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
enlarged_thyroid <= 0.5 gini = 0.972 samples = 2870 value = [83, 78, 78, 80, 84, 78, 72, 83, 82, 89, 0, 74 0, 70, 82, 87, 77, 77, 78, 73, 80, 82, 70, 83 78, 76, 82, 84, 83, 85, 81, 4, 82, 3, 74, 4 80, 79, 79, 80, 76]
gini = 0.0 samples = 83 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 83, 0, 0, 0, 0, 0]
(...)
(...)



CONCLUSION

- I developed a machine learning model utilizing techniques like Naive Bayes and Decision Tree that can accurately predict the kind of disease given the various types of symptoms experienced by the user after data collection and pre-processing or cleaning. The decision tree approach is superior because, in our dataset, several symptoms taken together can effectively describe the condition, unlike naive Bayes, which believes that all attributes are independent of one another and provides the probability.

- ***Thankyou***