ព្រះរាជាណាចក្រកម្ពុជា

ជាតិ សាសនា ព្រះមហាក្សត្រ

PROJECT REPORT: *Introduction to Data Science*

Disease Prediction using Machine Learning

GROUP: I3-AMS-B

Name of Students	**ID of Students**	**Score**
1. LY Chhaythean	e20220963	....
2. LUN Rathana	e20220368	....
3. MIN Sivee	e20220283	....
4. LY Chhungheang	e20220980	....

Lecturer: **Dr. PHAUK Sokkhey (Course)**
Mr. PEN Chentra (TP)

Academic year 2024-2025

CONTENTS

I. Introduction

1. Project Purpose

The objective of this project is to use modern technology such as machine learning to build algorithms capable of analyzing patient data, medical history, and other records to generate the probability of sickness. Because health professionals will be able to provide more accurate predictions of illness, they will be better able to make relevant treatment decisions and intervene quickly. This project attempts to develop solutions for better predictive healthcare by using these insights from data.

2. Problem Statement

Due to lack of materials, time, and skills, healthcare systems across the world have a lot of difficulty with disease detection and diagnosis at an early stage, which is one of the biggest problems in the world today. Existing approaches in diagnostics can be inefficient, slow, and depend greatly on the operator’s skills. This project resolves these issues through the design of a machine learning system that is able to sift through data and find correlations and, therefore, make predictions regarding diseases.

3. Objective

These are the objective that we aim to archive are listed in down below:

- Develop a machine learning model that can predict specific diseases based on particular input features, for example, symptoms, demographic data, and case history.
- Enhancement of accuracy in disease prediction and reliability using state of-the-art algorithms and feature engineering techniques.
- Create a user-friendly interface system which will help health practitioners enter the data and gets the prediction easily.
- Test and validate the performance of the model on real-world datasets for its robustness and applicability.

4. Scope

This project focuses on the development and implementation of a machine learning model for disease prediction. The key features include:

- Data preprocessing and feature selection to improve model performance.
- Implementation of supervised learning algorithms to train the model.

- Evaluation of the model’s performance using metrics like accuracy, precision, recall, and F1 score.

Overview of the Report

This report is structured as follows:

- Literature Review: A review of existing research and technologies in disease prediction using machine learning.
- Methodology: Detailed explanation of the dataset, preprocessing steps, model selection, and training process.
- Results and Discussion: Presentation and analysis of the model’s performance and findings.
- Conclusion and Future Work: Summary of the project outcomes and suggestions for future enhancements.

By the end of this report, the reader will have a comprehensive understanding of how machine learning can be applied to predict diseases effectively and the potential impact of such systems on healthcare.

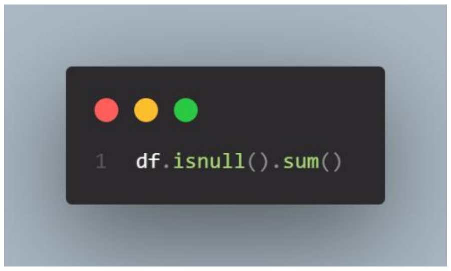
II. Methodology

To carry out this project, we followed several steps:

- Data Collection: We gathered data from various sources, including hospitals or online databases. This data included patient records and medical history.
- Data Preprocessing: Before using the data, we cleaned it. This involved fixing missing values, removing duplicates, and converting the data into a format suitable for analysis. We used libraries like Pandas, NumPy, and Scikit-learn for this process.
- Feature Selection: We identified which features (or variables) were most important for predicting diseases. This step helps in improving the model's performance.
- Model Selection: We tested different machine learning algorithms, such as logistic regression, decision trees, and neural networks. Each model has its strengths, and we aimed to find the best one for our data.
- Training and Testing: We split the data into two parts: one for training the model and the other for testing its accuracy. This helps ensure that the model can make accurate predictions on new, unseen data.

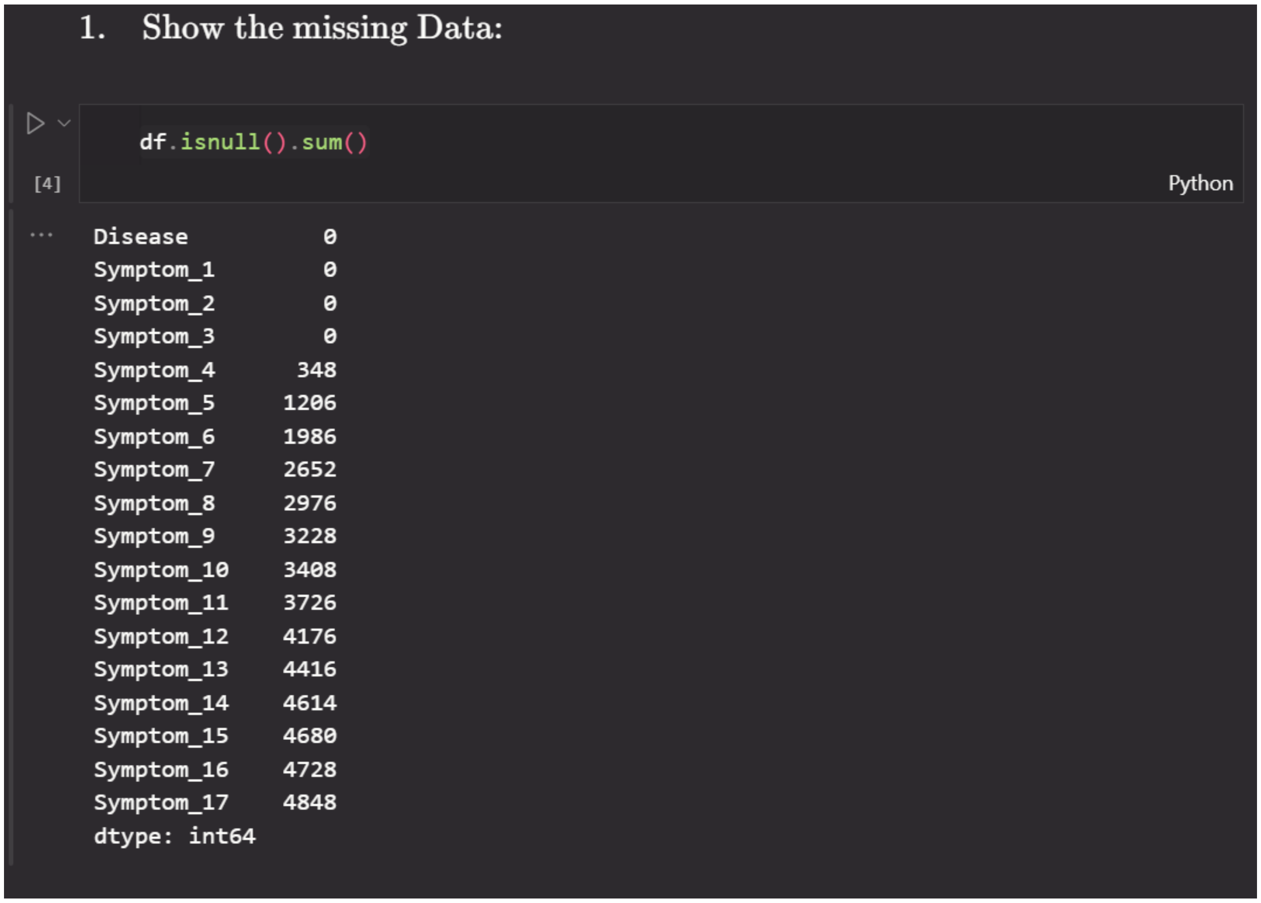
III. Implementation

Data cleaning was a crucial step in our project. We focused on:



```
df.isnull().sum()
```

After load the data, one must show the missing data we use this command code:



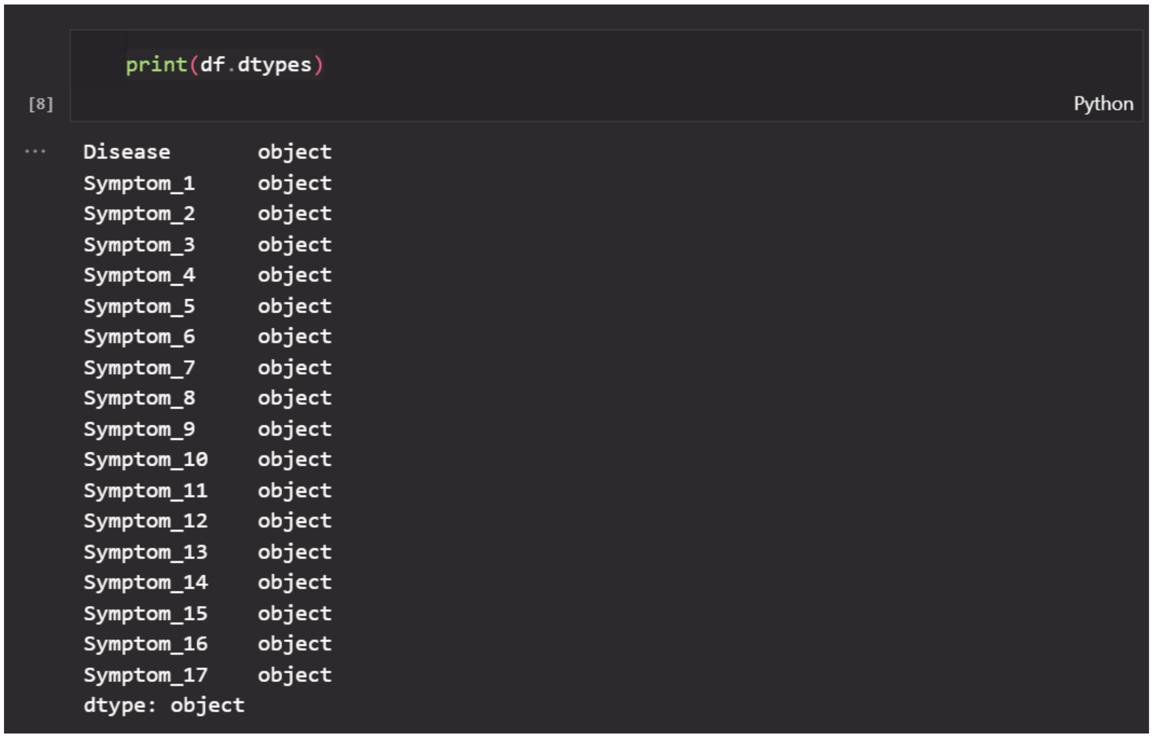
1. Show the missing Data:

```
df.isnull().sum()
```

[4] Python

```
Disease        0
Symptom_1      0
Symptom_2      0
Symptom_3      0
Symptom_4    348
Symptom_5   1206
Symptom_6   1986
Symptom_7   2652
Symptom_8   2976
Symptom_9   3228
Symptom_10  3408
Symptom_11  3726
Symptom_12  4176
Symptom_13  4416
Symptom_14  4614
Symptom_15  4680
Symptom_16  4728
Symptom_17  4848
dtype: int64
```

Now we see the missing data in our CSV file, so one must need to know what type of our data is, we use this command:

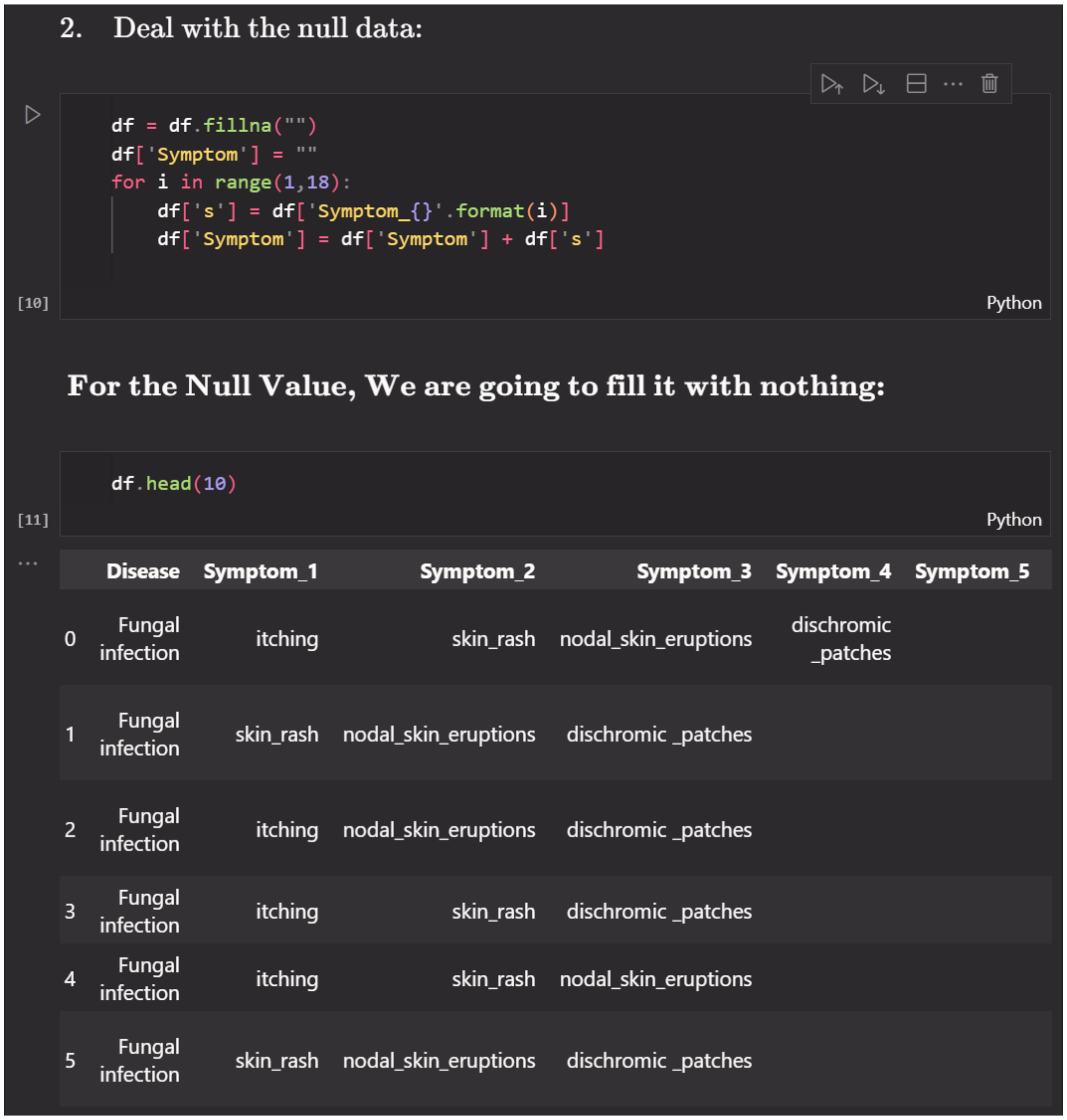


```
print(df.dtypes)
```

[8] Python

```
Disease       object
Symptom_1     object
Symptom_2     object
Symptom_3     object
Symptom_4     object
Symptom_5     object
Symptom_6     object
Symptom_7     object
Symptom_8     object
Symptom_9     object
Symptom_10    object
Symptom_11    object
Symptom_12    object
Symptom_13    object
Symptom_14    object
Symptom_15    object
Symptom_16    object
Symptom_17    object
dtype: object
```

Since, our data is an object type one said it is text type so for the null data we are going to fill it with nothing:



2. Deal with the null data:

```python
df = df.fillna("")
df['Symptom'] = ""
for i in range(1,18):
    df['s'] = df['Symptom_{}'.format(i)]
    df['Symptom'] = df['Symptom'] + df['s']
```

[10] Python

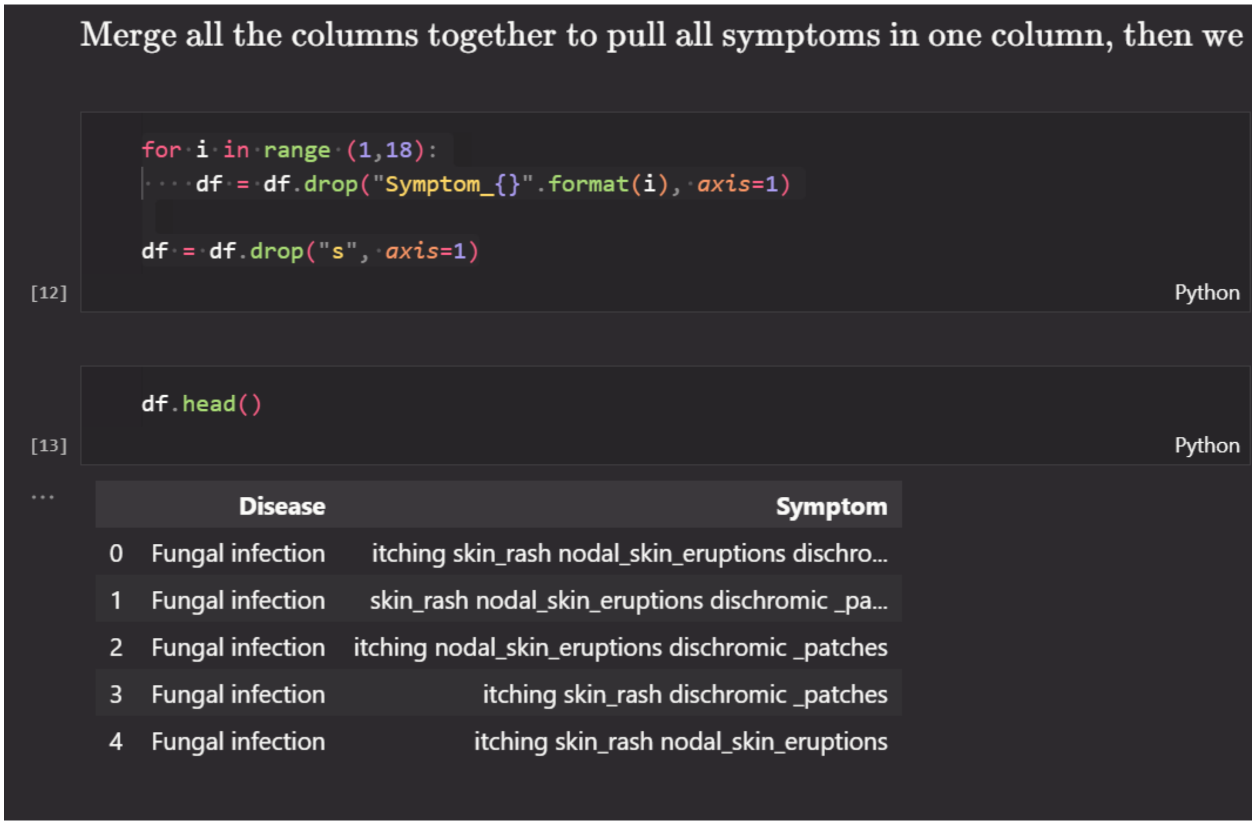
For the Null Value, We are going to fill it with nothing:

```python
df.head(10)
```

[11] Python

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5
0	Fungal infection	itching	skin_rash	nodal_skin_eruptions	dischromic _patches	
1	Fungal infection	skin_rash	nodal_skin_eruptions	dischromic _patches		
2	Fungal infection	itching	nodal_skin_eruptions	dischromic _patches		
3	Fungal infection	itching	skin_rash	dischromic _patches		
4	Fungal infection	itching	skin_rash	nodal_skin_eruptions		
5	Fungal infection	skin_rash	nodal_skin_eruptions	dischromic _patches		

After completing the deal with the null data, one need to merge the data all together for the further NLP techniques:


Merge all the columns together to pull all symptoms in one column, then we

```
for i in range (1,18):
    df = df.drop("Symptom_{}".format(i), axis=1)

df = df.drop("s", axis=1)
```

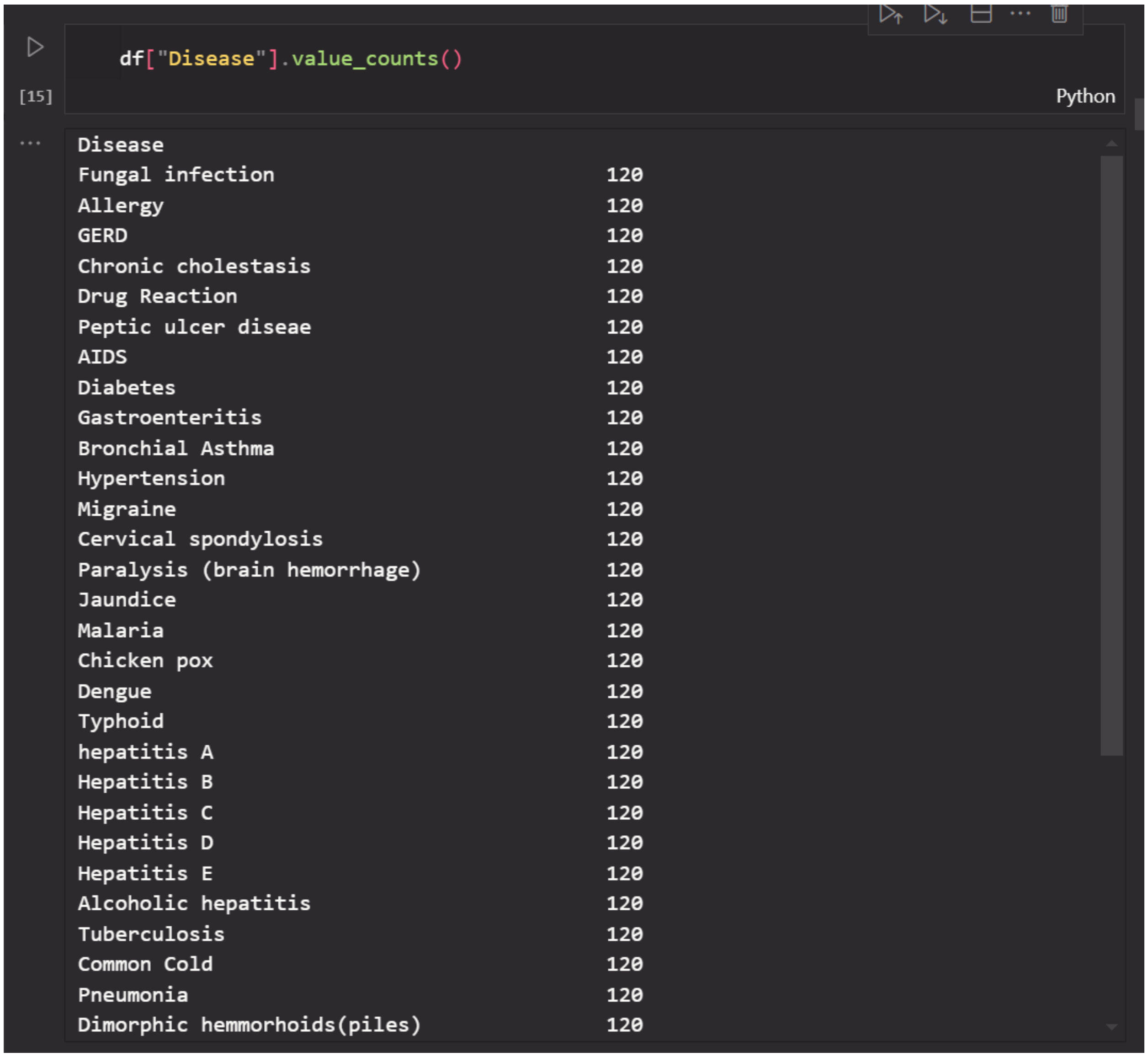
[12] Python

```
df.head()
```

[13] Python

	Disease	Symptom
0	Fungal infection	itching skin_rash nodal_skin_eruptions dischro...
1	Fungal infection	skin_rash nodal_skin_eruptions dischromic _pa...
2	Fungal infection	itching nodal_skin_eruptions dischromic _patches
3	Fungal infection	itching skin_rash dischromic _patches
4	Fungal infection	itching skin_rash nodal_skin_eruptions

After Merge the data, we might want to see if the data is balanced or not then we use this Python’s implementation:



```
df["Disease"].value_counts()
```

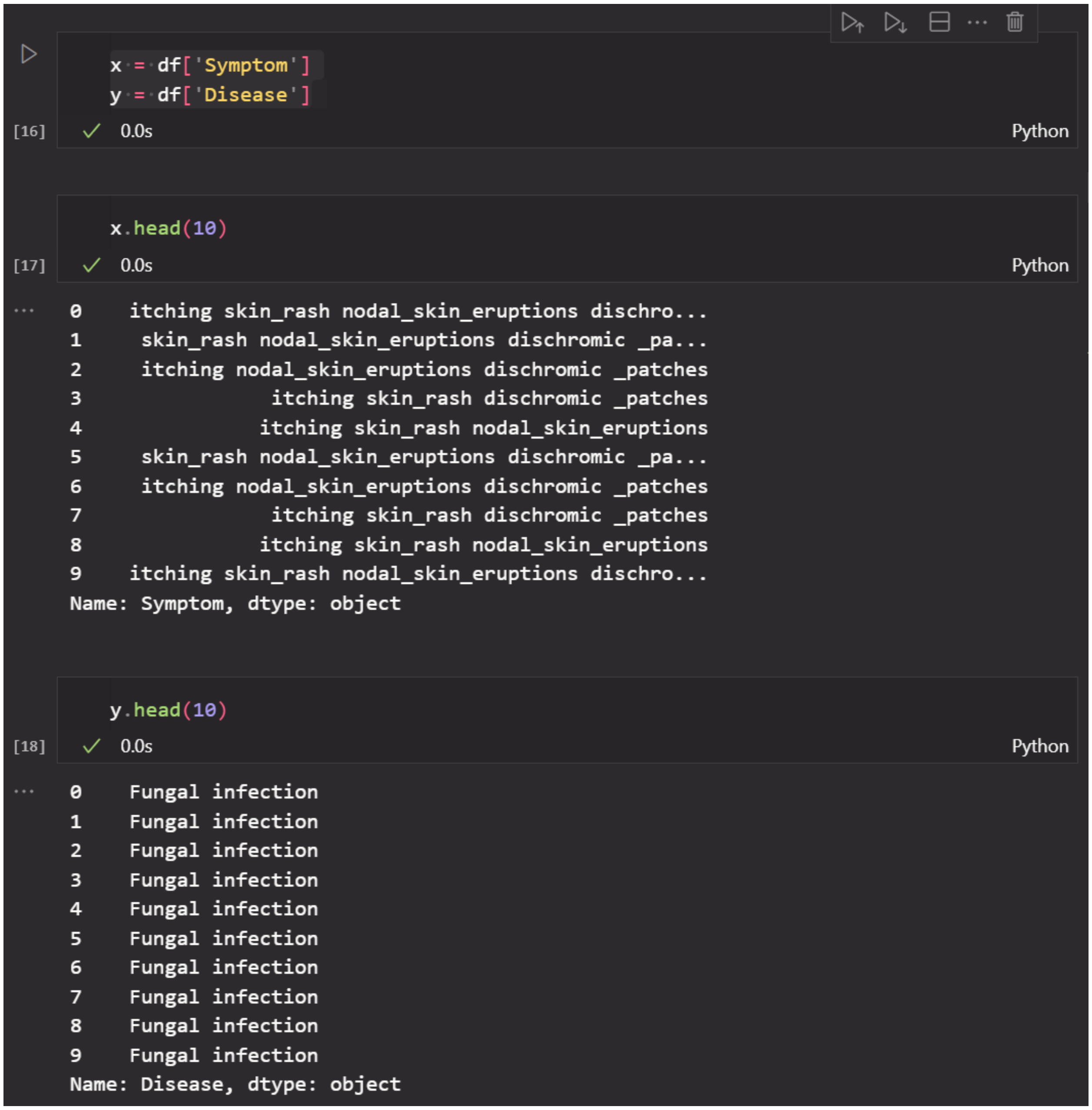
[15]

Python

```
Disease
Fungal infection                           120
Allergy                                    120
GERD                                       120
Chronic cholestasis                        120
Drug Reaction                              120
Peptic ulcer diseae                        120
AIDS                                       120
Diabetes                                   120
Gastroenteritis                            120
Bronchial Asthma                           120
Hypertension                               120
Migraine                                   120
Cervical spondylosis                       120
Paralysis (brain hemorrhage)               120
Jaundice                                   120
Malaria                                    120
Chicken pox                                120
Dengue                                     120
Typhoid                                    120
hepatitis A                                120
Hepatitis B                                120
Hepatitis C                                120
Hepatitis D                                120
Hepatitis E                                120
Alcoholic hepatitis                        120
Tuberculosis                               120
Common Cold                                120
Pneumonia                                  120
Dimorphic hemmorhoids(piles)               120
```

All goods our data is completely balanced but we want to see some graph for further visualization:

We assigned our data into 2D arrays where we denoted the disease is y and the symptom is x.



```
x = df['Symptom']
y = df['Disease']
```
[16] ✓ 0.0s Python

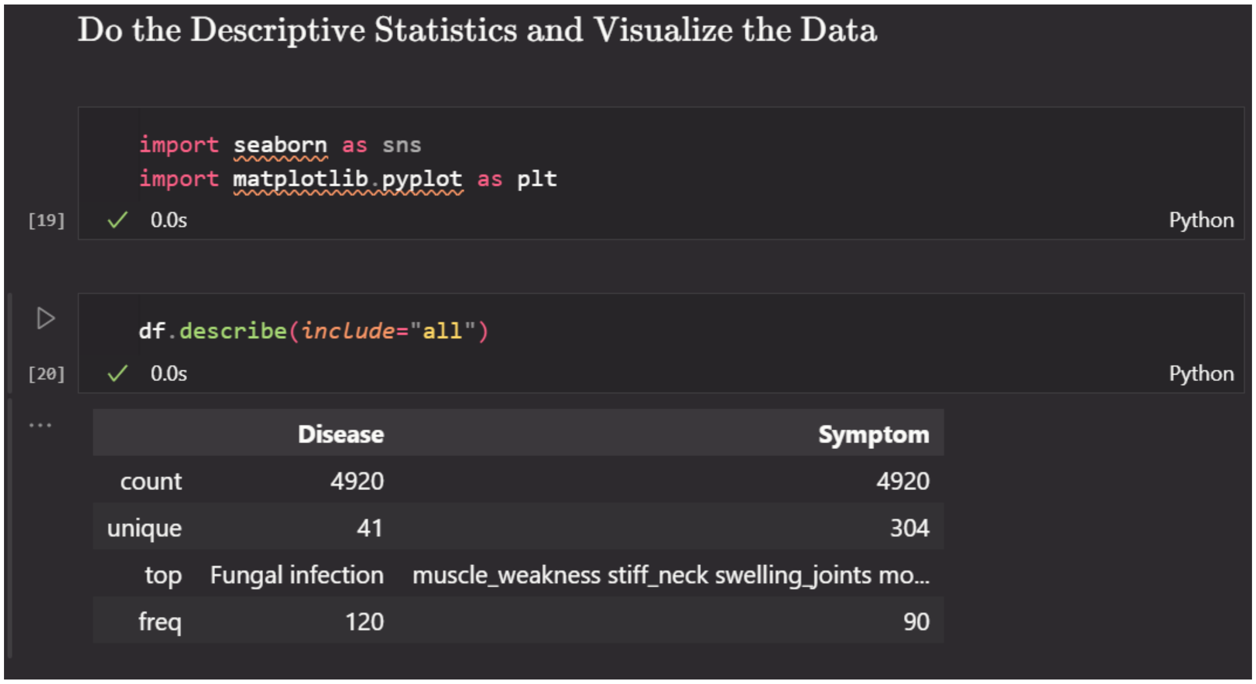
```
x.head(10)
```
[17] ✓ 0.0s Python

```
0    itching skin_rash nodal_skin_eruptions dischro...
1     skin_rash nodal_skin_eruptions dischromic _pa...
2     itching nodal_skin_eruptions dischromic _patches
3                itching skin_rash dischromic _patches
4               itching skin_rash nodal_skin_eruptions
5     skin_rash nodal_skin_eruptions dischromic _pa...
6     itching nodal_skin_eruptions dischromic _patches
7                itching skin_rash dischromic _patches
8               itching skin_rash nodal_skin_eruptions
9    itching skin_rash nodal_skin_eruptions dischro...
Name: Symptom, dtype: object
```

```
y.head(10)
```
[18] ✓ 0.0s Python

```
0    Fungal infection
1    Fungal infection
2    Fungal infection
3    Fungal infection
4    Fungal infection
5    Fungal infection
6    Fungal infection
7    Fungal infection
8    Fungal infection
9    Fungal infection
Name: Disease, dtype: object
```

Now we are going to do descriptive statistics for further understanding:



Do the Descriptive Statistics and Visualize the Data

```
import seaborn as sns
import matplotlib.pyplot as plt
```

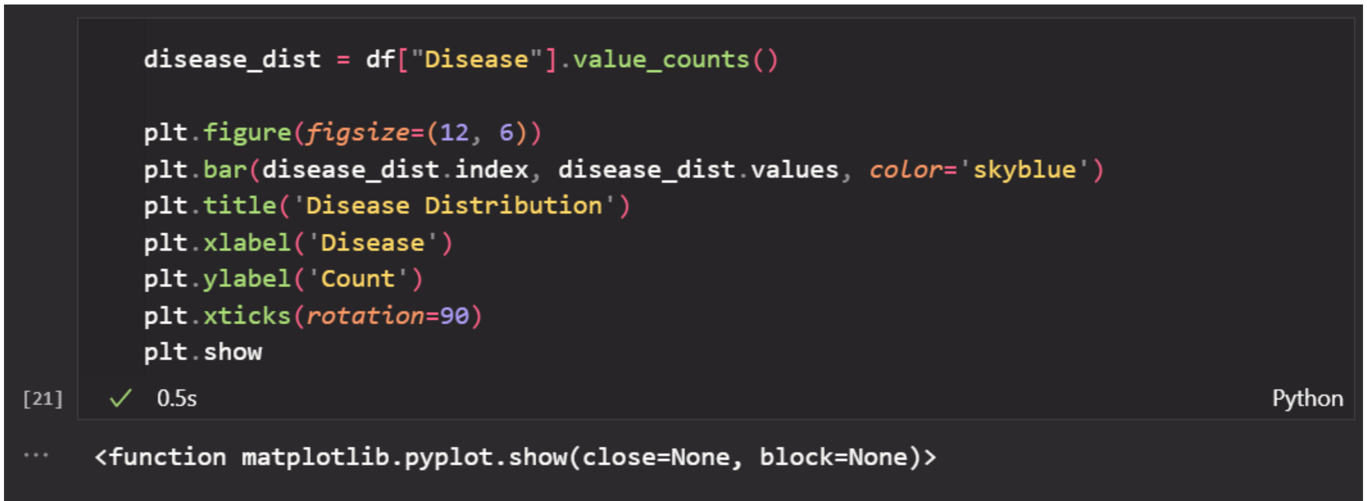
```
[19] ✓ 0.0s                                                    Python
```

```
df.describe(include="all")
```

```
[20] ✓ 0.0s                                                    Python
```

	Disease	Symptom
count	4920	4920
unique	41	304
top	Fungal infection	muscle_weakness stiff_neck swelling_joints mo...
freq	120	90

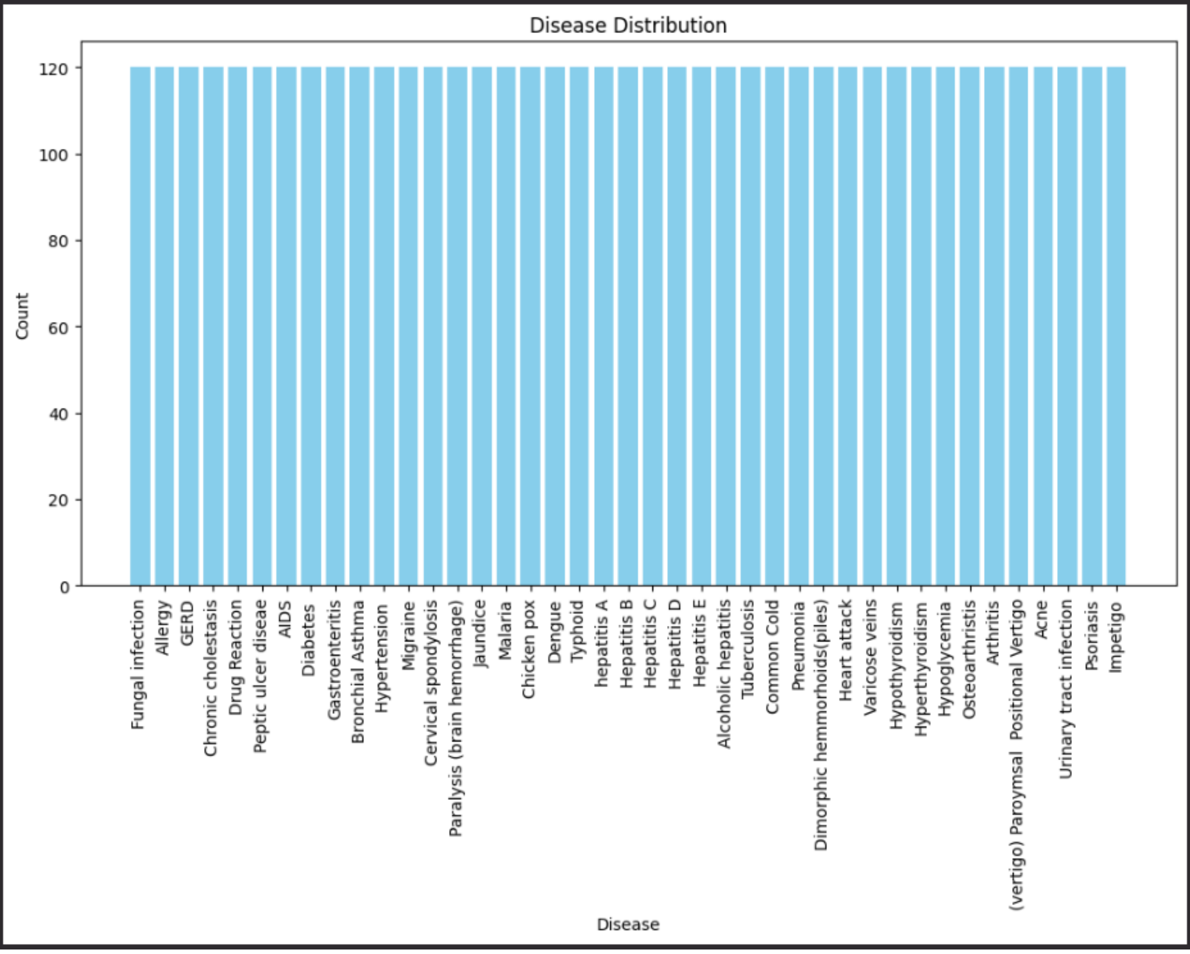
After doing the descriptive statistics we now can construct the plot:



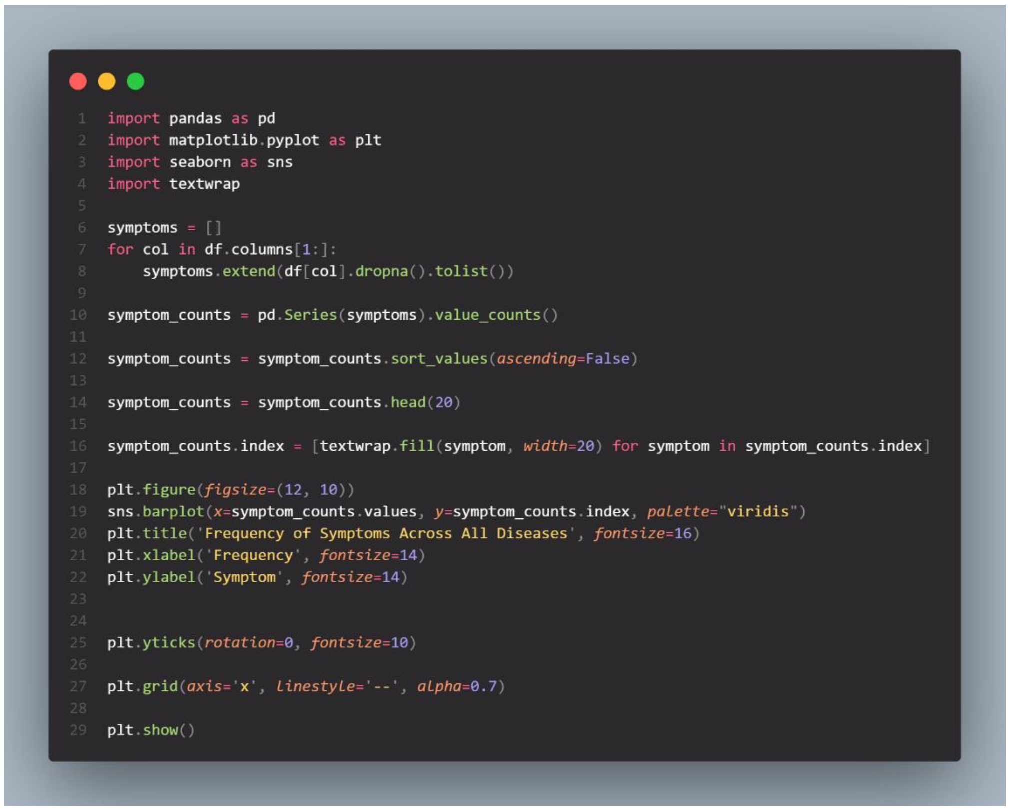
```
disease_dist = df["Disease"].value_counts()

plt.figure(figsize=(12, 6))
plt.bar(disease_dist.index, disease_dist.values, color='skyblue')
plt.title('Disease Distribution')
plt.xlabel('Disease')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show
```

[21] ✓ 0.5s Python

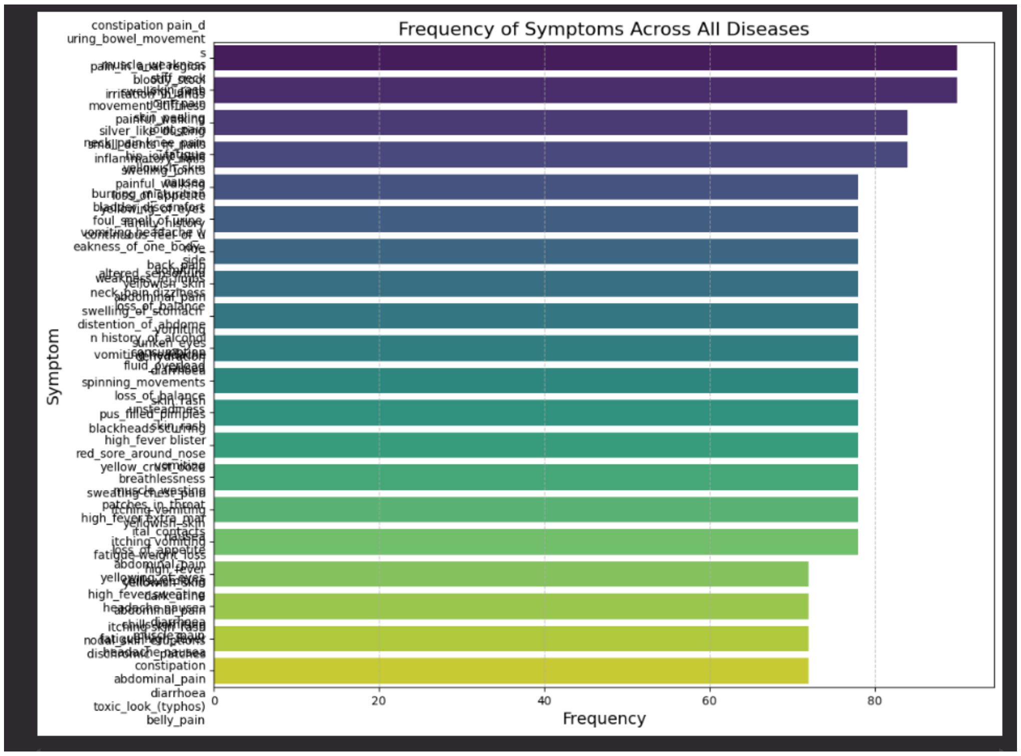
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Now plot the symptoms that give the impact to our process:

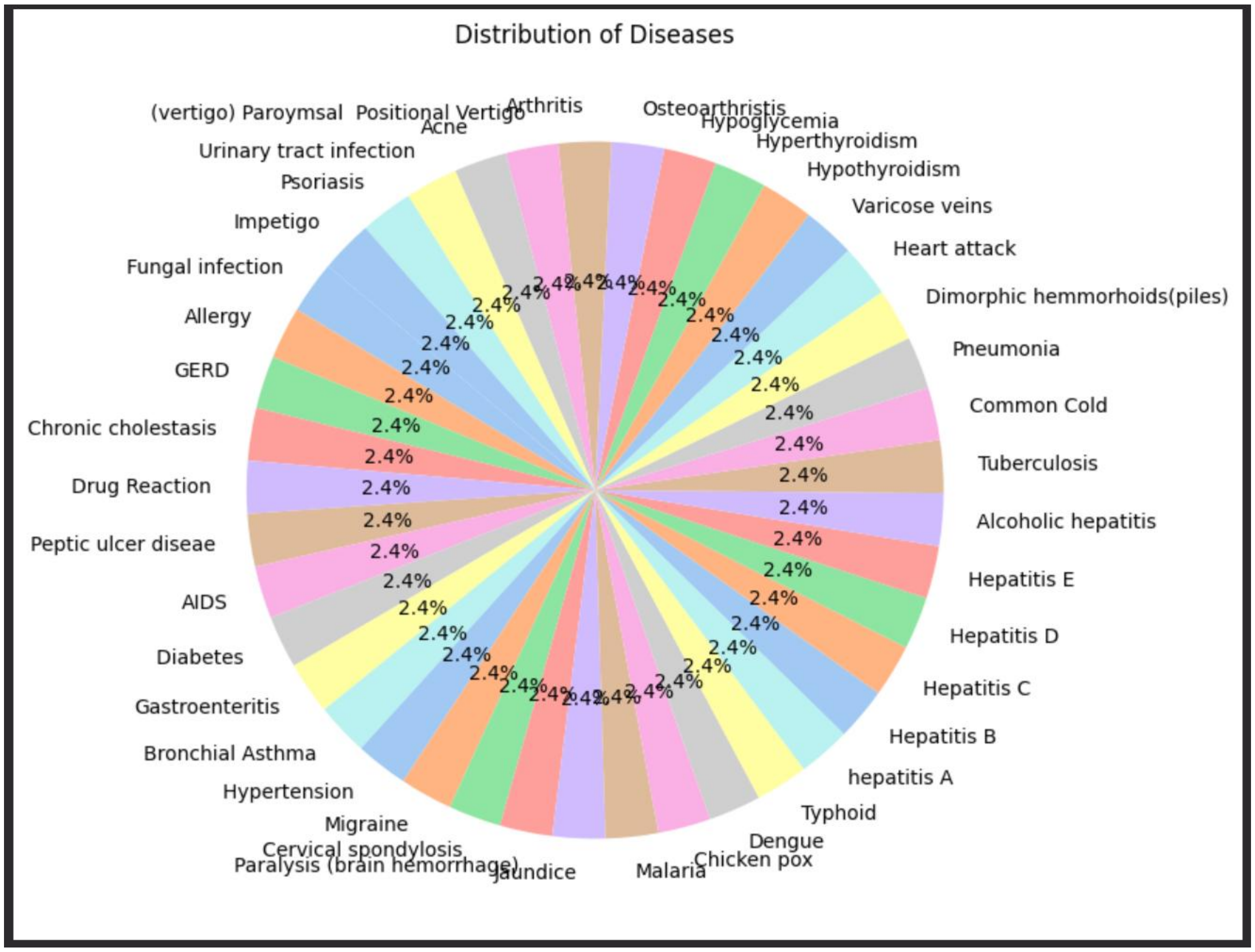


```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import textwrap

symptoms = []
for col in df.columns[1:]:
    symptoms.extend(df[col].dropna().tolist())

symptom_counts = pd.Series(symptoms).value_counts()

symptom_counts = symptom_counts.sort_values(ascending=False)

symptom_counts = symptom_counts.head(20)

symptom_counts.index = [textwrap.fill(symptom, width=20) for symptom in symptom_counts.index]

plt.figure(figsize=(12, 10))
sns.barplot(x=symptom_counts.values, y=symptom_counts.index, palette="viridis")
plt.title('Frequency of Symptoms Across All Diseases', fontsize=16)
plt.xlabel('Frequency', fontsize=14)
plt.ylabel('Symptom', fontsize=14)


plt.yticks(rotation=0, fontsize=10)

plt.grid(axis='x', linestyle='--', alpha=0.7)

plt.show()
```



Now we are going to see how many percent of the disease contributed in our data simply Pie Chart

```
disease_counts = df['Disease'].value_counts()

plt.figure(figsize=(8, 8))
plt.pie(disease_counts, labels=disease_counts.index, autopct='%1.1f%%',
startangle=140, colors=sns.color_palette("pastel"))
plt.title('Distribution of Diseases')
plt.show()
```



Feature Selection and Model Selection

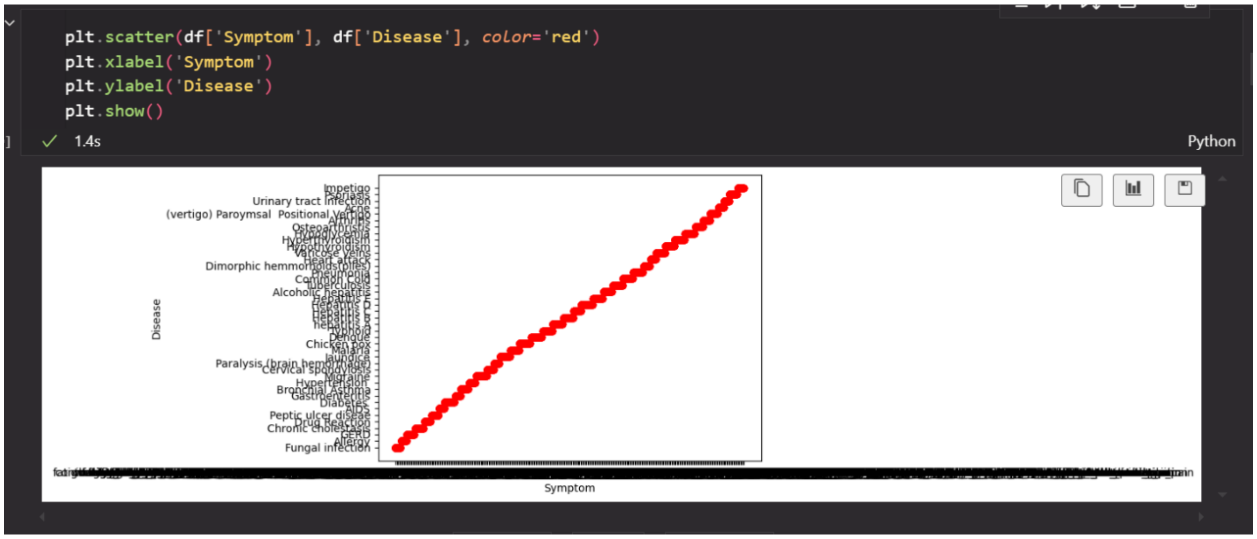
In here, we are testing about three different models in other to see with one is more accurate and well performed:

a) Linear SVC

b) GradientBoostingClassification

➢ Linear SVC

By seeing the data in scatter plot we once see that there is data point so up as linear and on the other hand, Linear SVC works with the large sample numbers.



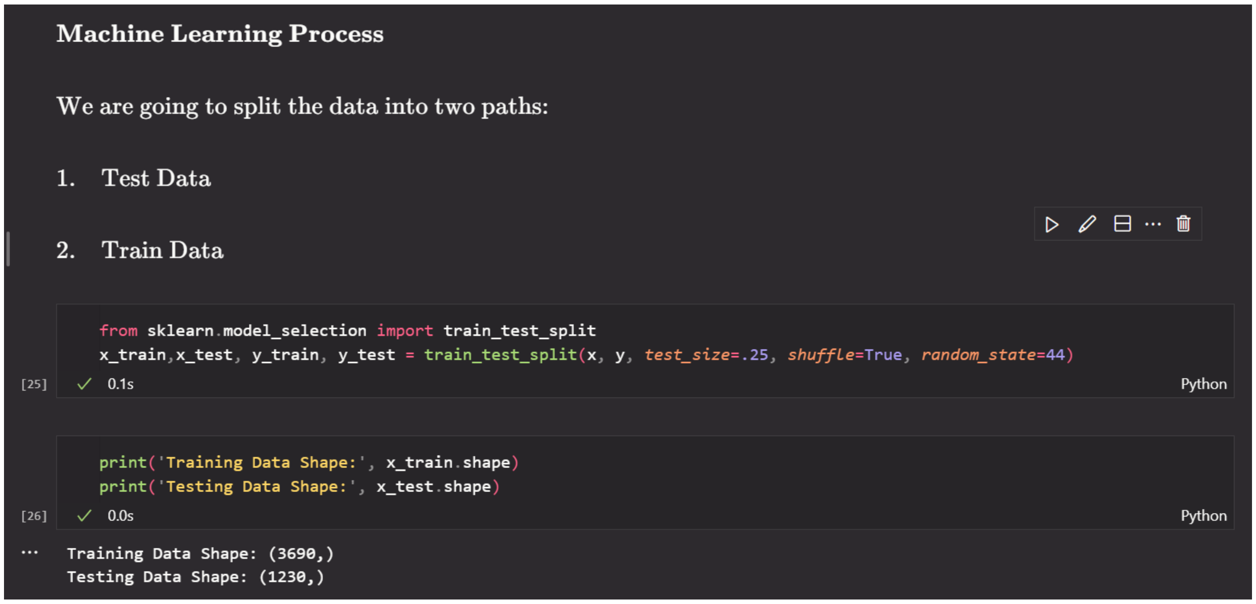
```
plt.scatter(df['Symptom'], df['Disease'], color='red')
plt.xlabel('Symptom')
plt.ylabel('Disease')
plt.show()
```

✓ 1.4s Python

Model Processing

In this case, we are going to split the data into Test and Train dataset,

We split 25% of data for the test size and 75% for the train size so we use this implement in order to do so:



Machine Learning Process

We are going to split the data into two paths:

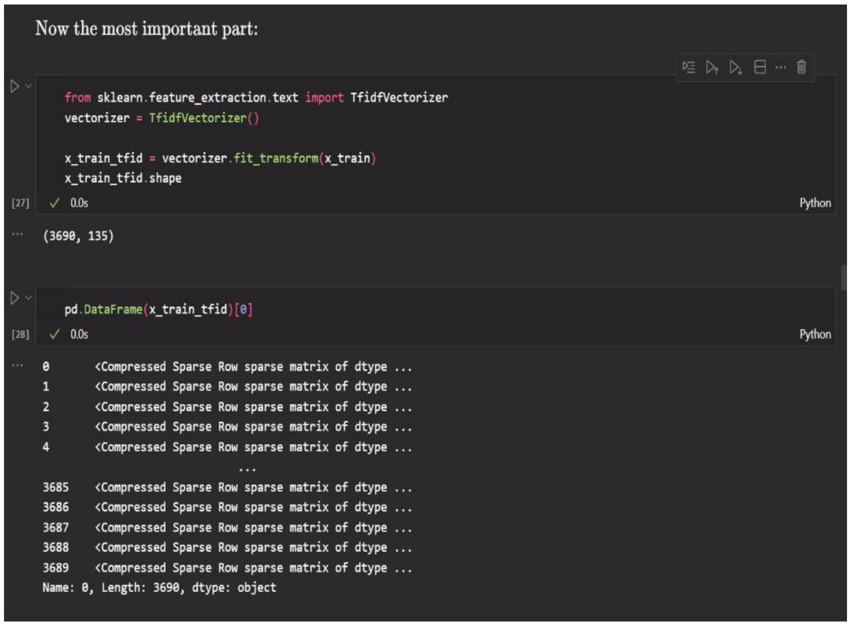
1. Test Data
2. Train Data

```
from sklearn.model_selection import train_test_split
x_train,x_test, y_train, y_test = train_test_split(x, y, test_size=.25, shuffle=True, random_state=44)
```
[25] ✓ 0.1s Python

```
print('Training Data Shape:', x_train.shape)
print('Testing Data Shape:', x_test.shape)
```
[26] ✓ 0.0s Python

```
Training Data Shape: (3690,)
Testing Data Shape: (1230,)
```

Since, our Dataset is the text type file, we have to encode our dataset into numeric in order to work with *Linear SVC.*



Now the most important part:

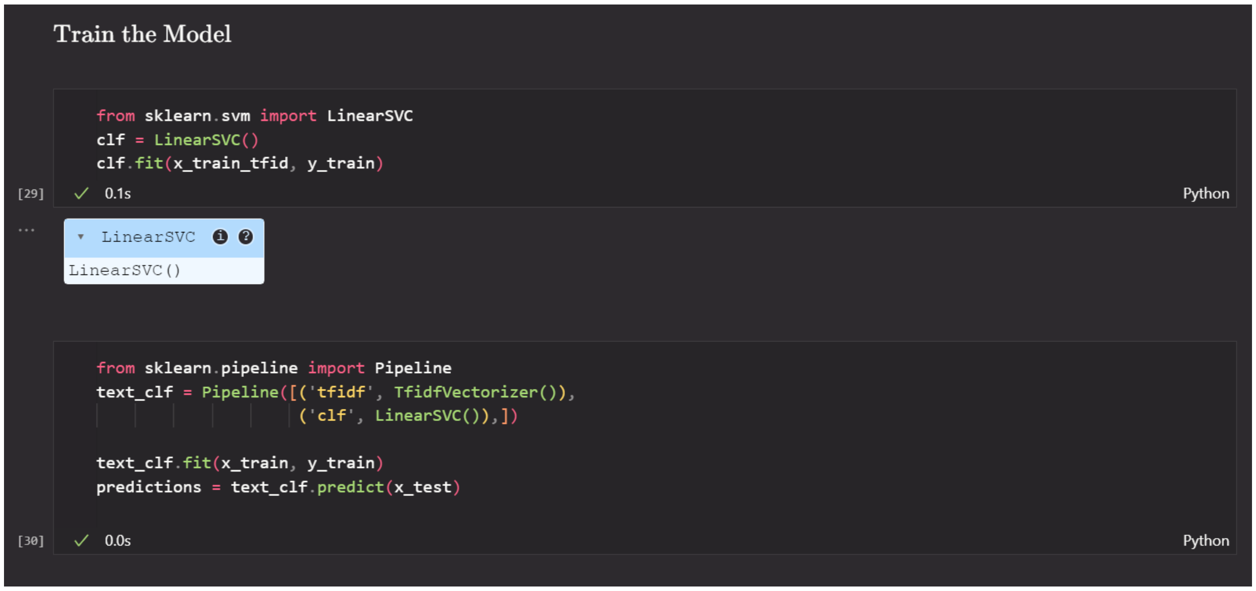
```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer()

x_train_tfid = vectorizer.fit_transform(x_train)
x_train_tfid.shape
```
[27] ✓ 0.0s Python

```
(3690, 135)
```

```
pd.DataFrame(x_train_tfid)[0]
```
[28] ✓ 0.0s Python

```
0       <Compressed Sparse Row sparse matrix of dtype ...
1       <Compressed Sparse Row sparse matrix of dtype ...
2       <Compressed Sparse Row sparse matrix of dtype ...
3       <Compressed Sparse Row sparse matrix of dtype ...
4       <Compressed Sparse Row sparse matrix of dtype ...
                              ...
3685    <Compressed Sparse Row sparse matrix of dtype ...
3686    <Compressed Sparse Row sparse matrix of dtype ...
3687    <Compressed Sparse Row sparse matrix of dtype ...
3688    <Compressed Sparse Row sparse matrix of dtype ...
3689    <Compressed Sparse Row sparse matrix of dtype ...
Name: 0, Length: 3690, dtype: object
```

Train the model:



Train the Model

```
from sklearn.svm import LinearSVC
clf = LinearSVC()
clf.fit(x_train_tfid, y_train)
```

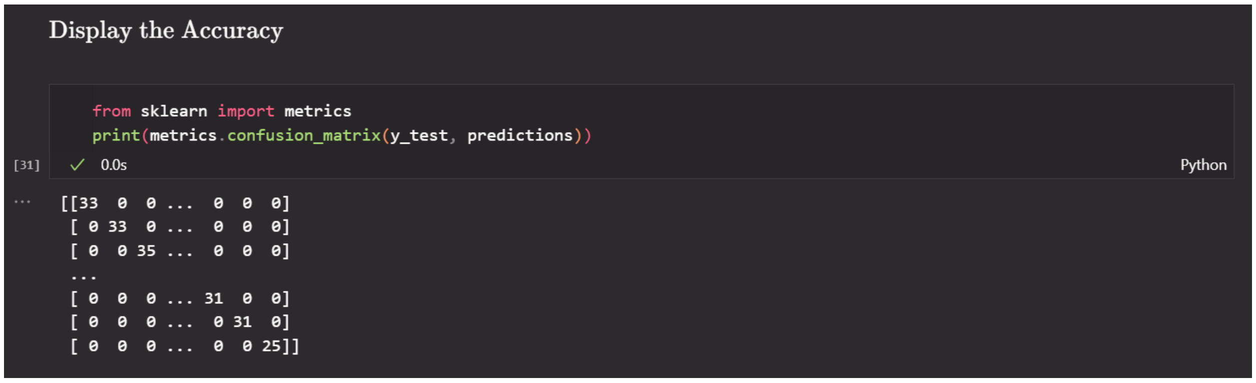
[29] ✓ 0.1s Python

```
LinearSVC()
```

```
from sklearn.pipeline import Pipeline
text_clf = Pipeline([('tfidf', TfidfVectorizer()),
                     ('clf', LinearSVC()),])

text_clf.fit(x_train, y_train)
predictions = text_clf.predict(x_test)
```

[30] ✓ 0.0s Python

Display the Accuracy:

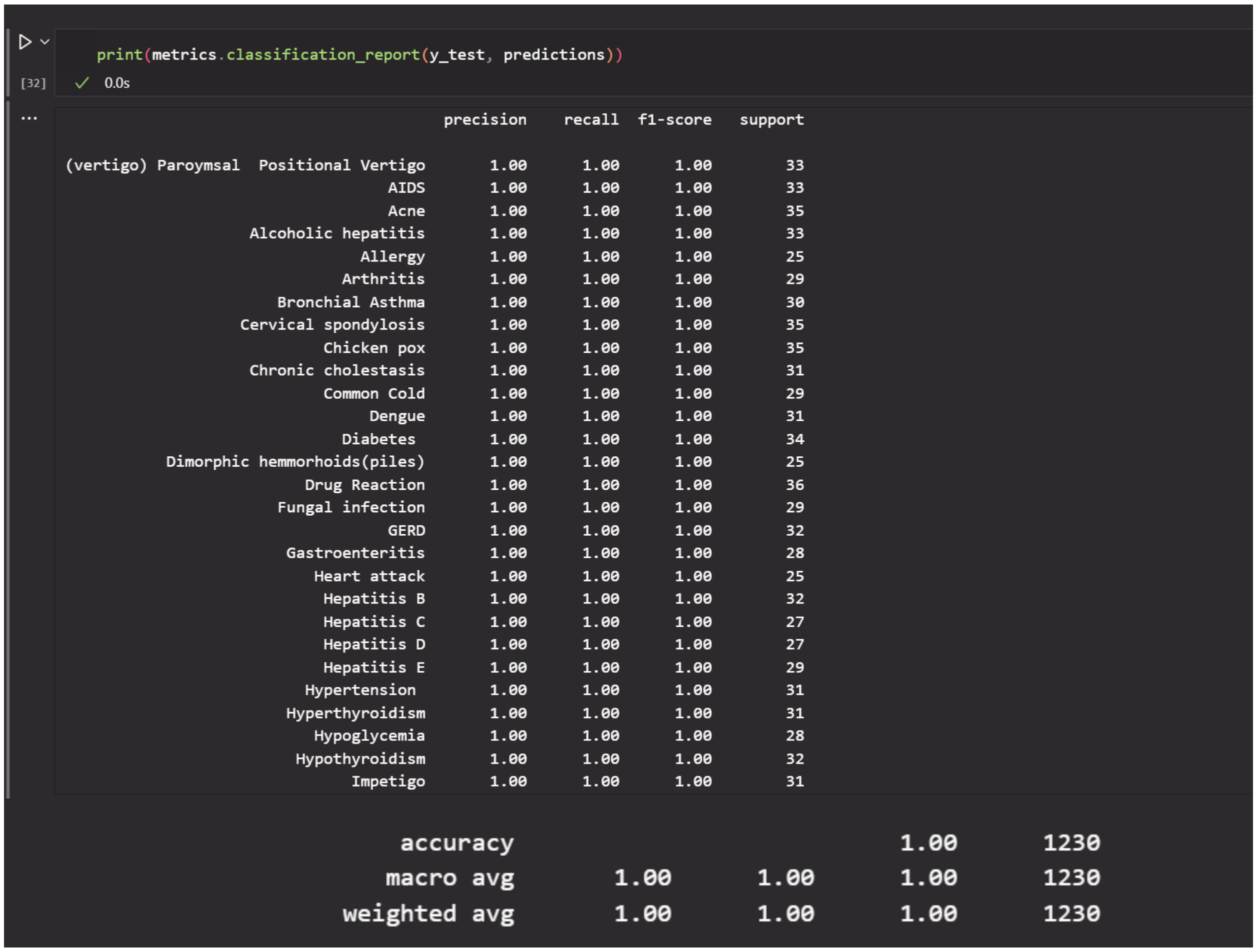


Display the Accuracy

```
from sklearn import metrics
print(metrics.confusion_matrix(y_test, predictions))
```

[31] ✓ 0.0s Python

```
[[33  0  0 ...  0  0  0]
 [ 0 33  0 ...  0  0  0]
 [ 0  0 35 ...  0  0  0]
 ...
 [ 0  0  0 ... 31  0  0]
 [ 0  0  0 ...  0 31  0]
 [ 0  0  0 ...  0  0 25]]
```

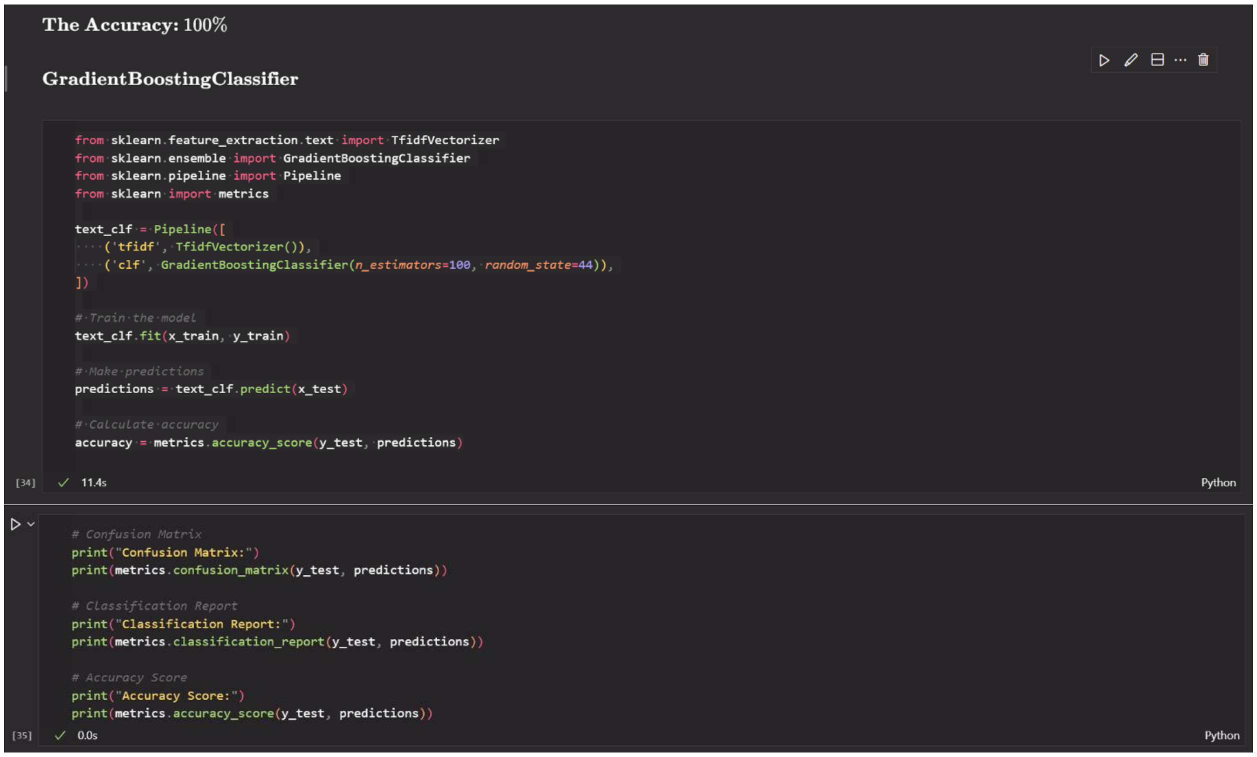


```
print(metrics.classification_report(y_test, predictions))
```

[32] ✓ 0.0s

```
                                          precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00        33
                                   AIDS       1.00      1.00      1.00        33
                                   Acne       1.00      1.00      1.00        35
                    Alcoholic hepatitis       1.00      1.00      1.00        33
                                Allergy       1.00      1.00      1.00        25
                              Arthritis       1.00      1.00      1.00        29
                       Bronchial Asthma       1.00      1.00      1.00        30
                   Cervical spondylosis       1.00      1.00      1.00        35
                            Chicken pox       1.00      1.00      1.00        35
                    Chronic cholestasis       1.00      1.00      1.00        31
                            Common Cold       1.00      1.00      1.00        29
                                 Dengue       1.00      1.00      1.00        31
                              Diabetes        1.00      1.00      1.00        34
        Dimorphic hemmorhoids(piles)          1.00      1.00      1.00        25
                           Drug Reaction      1.00      1.00      1.00        36
                        Fungal infection      1.00      1.00      1.00        29
                                   GERD       1.00      1.00      1.00        32
                       Gastroenteritis        1.00      1.00      1.00        28
                            Heart attack      1.00      1.00      1.00        25
                            Hepatitis B       1.00      1.00      1.00        32
                            Hepatitis C       1.00      1.00      1.00        27
                            Hepatitis D       1.00      1.00      1.00        27
                            Hepatitis E       1.00      1.00      1.00        29
                          Hypertension        1.00      1.00      1.00        31
                        Hyperthyroidism       1.00      1.00      1.00        31
                           Hypoglycemia       1.00      1.00      1.00        28
                         Hypothyroidism       1.00      1.00      1.00        32
                               Impetigo       1.00      1.00      1.00        31

                               accuracy                           1.00      1230
                              macro avg       1.00      1.00      1.00      1230
                           weighted avg       1.00      1.00      1.00      1230
```

➢ GradientBoostingClassification



The Accuracy: 100%

GradientBoostingClassifier

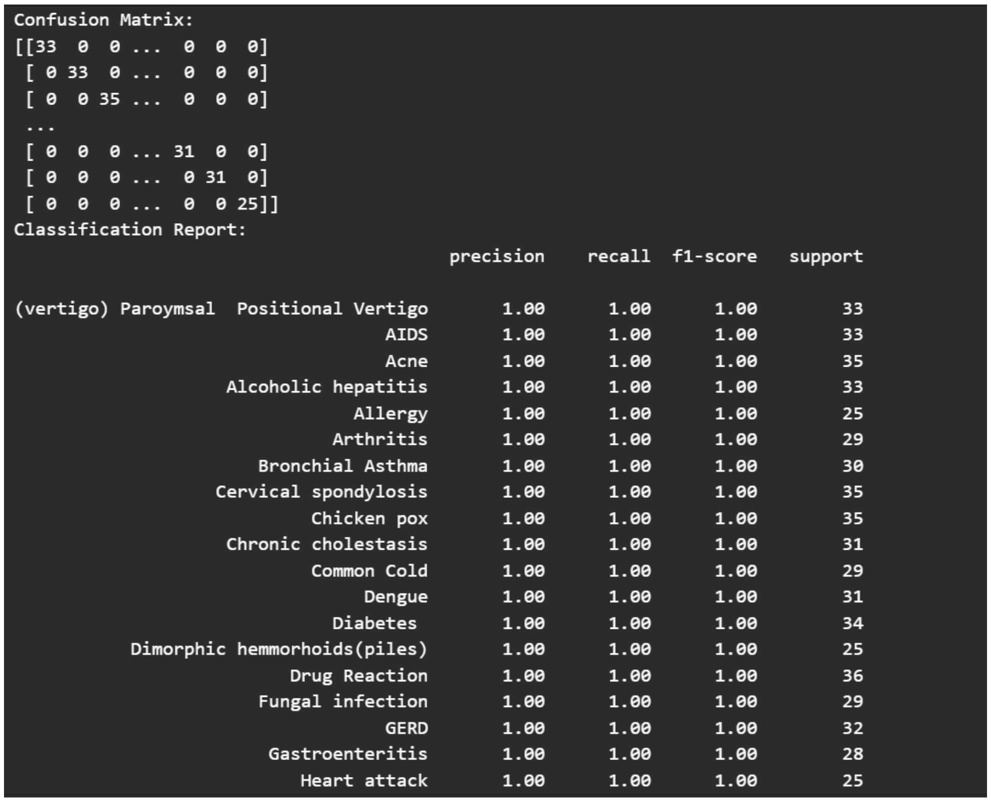
```python
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.pipeline import Pipeline
from sklearn import metrics

text_clf = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('clf', GradientBoostingClassifier(n_estimators=100, random_state=44)),
])

# Train the model
text_clf.fit(x_train, y_train)

# Make predictions
predictions = text_clf.predict(x_test)

# Calculate accuracy
accuracy = metrics.accuracy_score(y_test, predictions)
```

[34] ✓ 11.4s Python

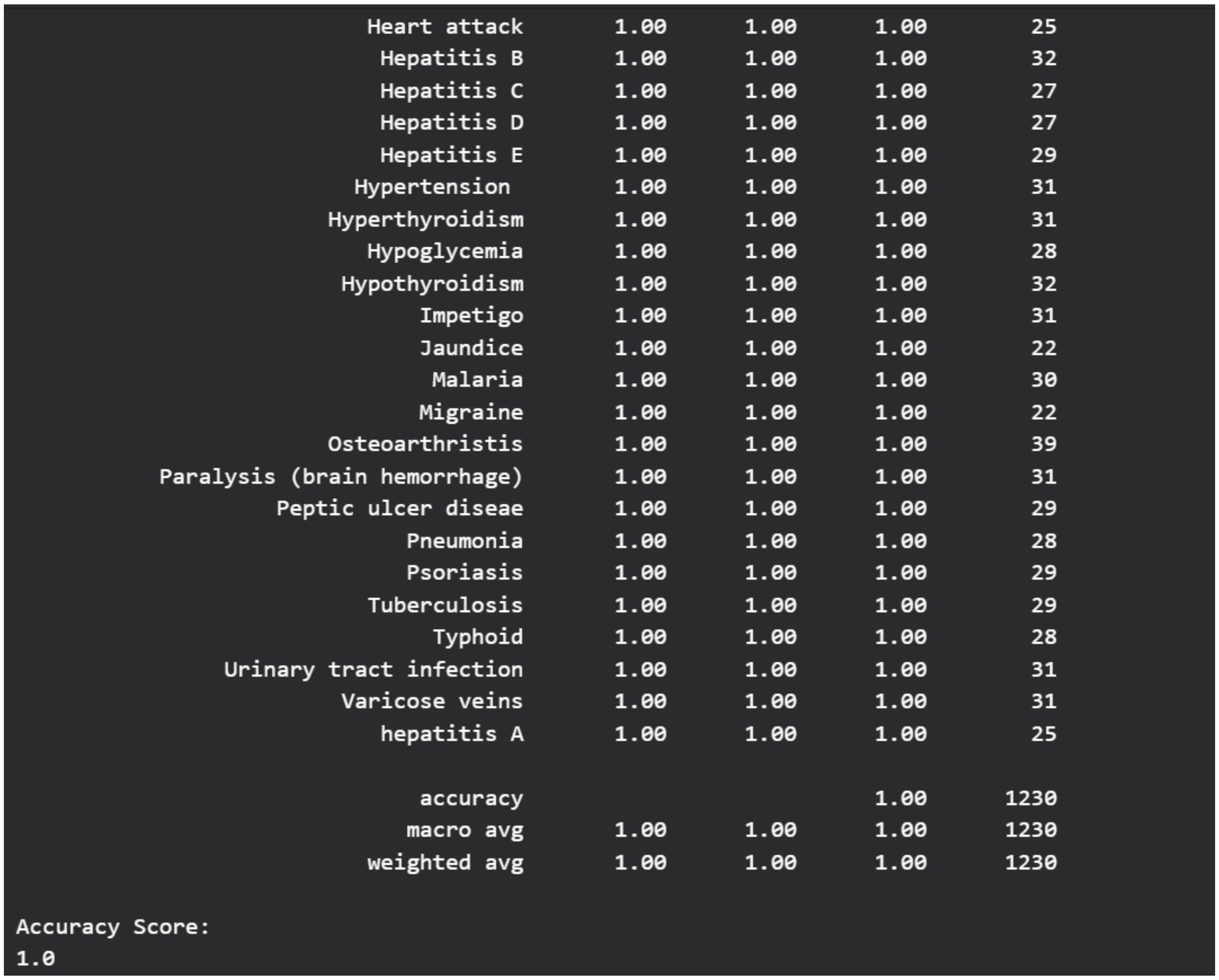
```python
# Confusion Matrix
print("Confusion Matrix:")
print(metrics.confusion_matrix(y_test, predictions))

# Classification Report
print("Classification Report:")
print(metrics.classification_report(y_test, predictions))

# Accuracy Score
print("Accuracy Score:")
print(metrics.accuracy_score(y_test, predictions))
```

[35] ✓ 0.0s Python

The Result:



```
Confusion Matrix:
[[33  0  0 ...  0  0  0]
 [ 0 33  0 ...  0  0  0]
 [ 0  0 35 ...  0  0  0]
 ...
 [ 0  0  0 ... 31  0  0]
 [ 0  0  0 ...  0 31  0]
 [ 0  0  0 ...  0  0 25]]
Classification Report:
                                          precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00        33
                                   AIDS       1.00      1.00      1.00        33
                                   Acne       1.00      1.00      1.00        35
                    Alcoholic hepatitis       1.00      1.00      1.00        33
                                Allergy       1.00      1.00      1.00        25
                              Arthritis       1.00      1.00      1.00        29
                       Bronchial Asthma       1.00      1.00      1.00        30
                   Cervical spondylosis       1.00      1.00      1.00        35
                            Chicken pox       1.00      1.00      1.00        35
                    Chronic cholestasis       1.00      1.00      1.00        31
                            Common Cold       1.00      1.00      1.00        29
                                 Dengue       1.00      1.00      1.00        31
                              Diabetes        1.00      1.00      1.00        34
        Dimorphic hemmorhoids(piles)       1.00      1.00      1.00        25
                          Drug Reaction       1.00      1.00      1.00        36
                       Fungal infection       1.00      1.00      1.00        29
                                   GERD       1.00      1.00      1.00        32
                        Gastroenteritis       1.00      1.00      1.00        28
                           Heart attack       1.00      1.00      1.00        25
```

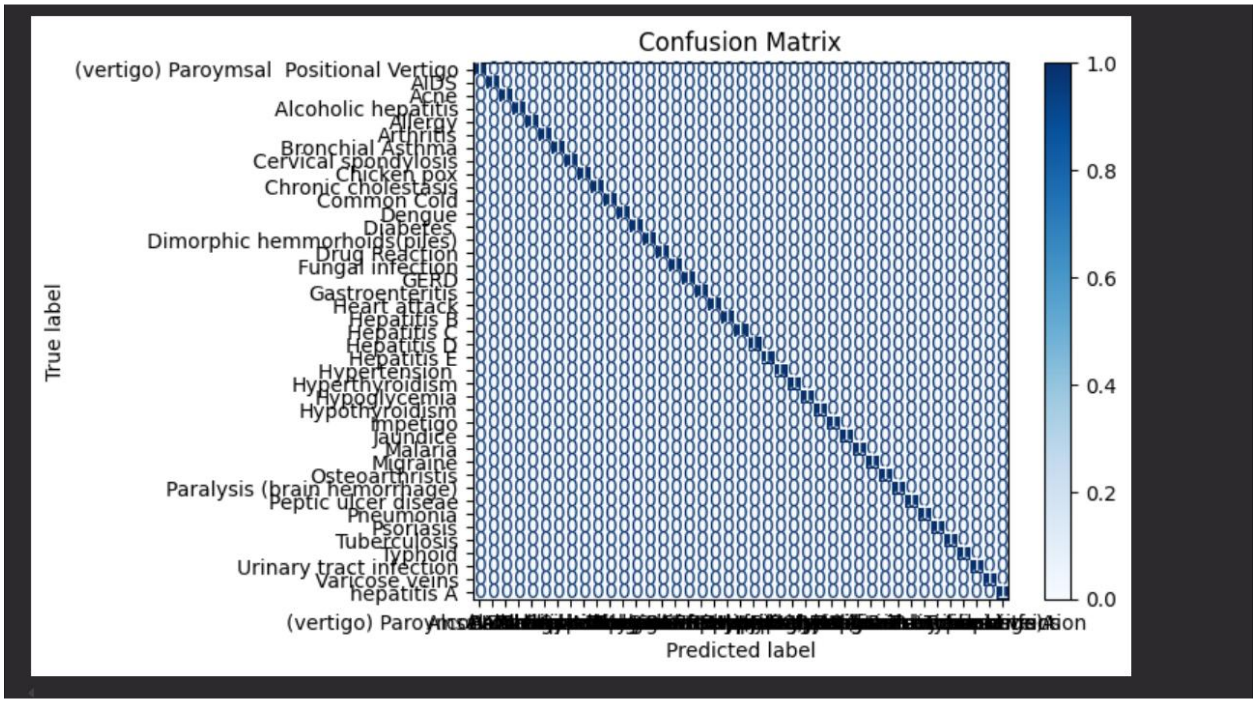


```
                  Heart attack       1.00      1.00      1.00        25
                   Hepatitis B       1.00      1.00      1.00        32
                   Hepatitis C       1.00      1.00      1.00        27
                   Hepatitis D       1.00      1.00      1.00        27
                   Hepatitis E       1.00      1.00      1.00        29
                 Hypertension        1.00      1.00      1.00        31
               Hyperthyroidism       1.00      1.00      1.00        31
                  Hypoglycemia       1.00      1.00      1.00        28
                Hypothyroidism       1.00      1.00      1.00        32
                      Impetigo       1.00      1.00      1.00        31
                      Jaundice       1.00      1.00      1.00        22
                       Malaria       1.00      1.00      1.00        30
                      Migraine       1.00      1.00      1.00        22
               Osteoarthristis       1.00      1.00      1.00        39
  Paralysis (brain hemorrhage)       1.00      1.00      1.00        31
           Peptic ulcer diseae       1.00      1.00      1.00        29
                     Pneumonia       1.00      1.00      1.00        28
                     Psoriasis       1.00      1.00      1.00        29
                  Tuberculosis       1.00      1.00      1.00        29
                       Typhoid       1.00      1.00      1.00        28
       Urinary tract infection       1.00      1.00      1.00        31
                Varicose veins       1.00      1.00      1.00        31
                   hepatitis A       1.00      1.00      1.00        25

                      accuracy                           1.00      1230
                     macro avg       1.00      1.00      1.00      1230
                  weighted avg       1.00      1.00      1.00      1230

Accuracy Score:
1.0
```

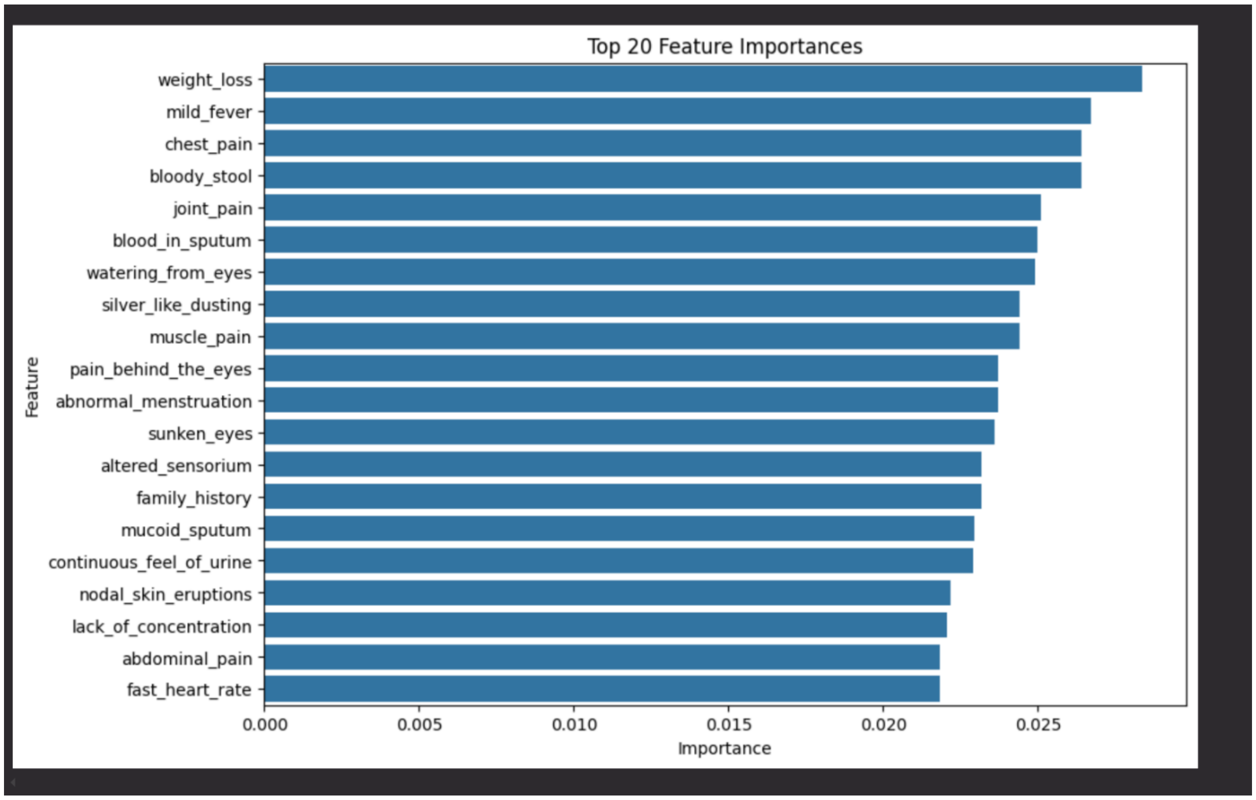
Conclusion of the models: After testing all these two models we get the accuracy of our data is up to 1.0 or be said 100% due to the data cleaning and its clean dataset.

Prediction Result be seen in Confusion Matrix

It is absolutely accuracy and efficiency.



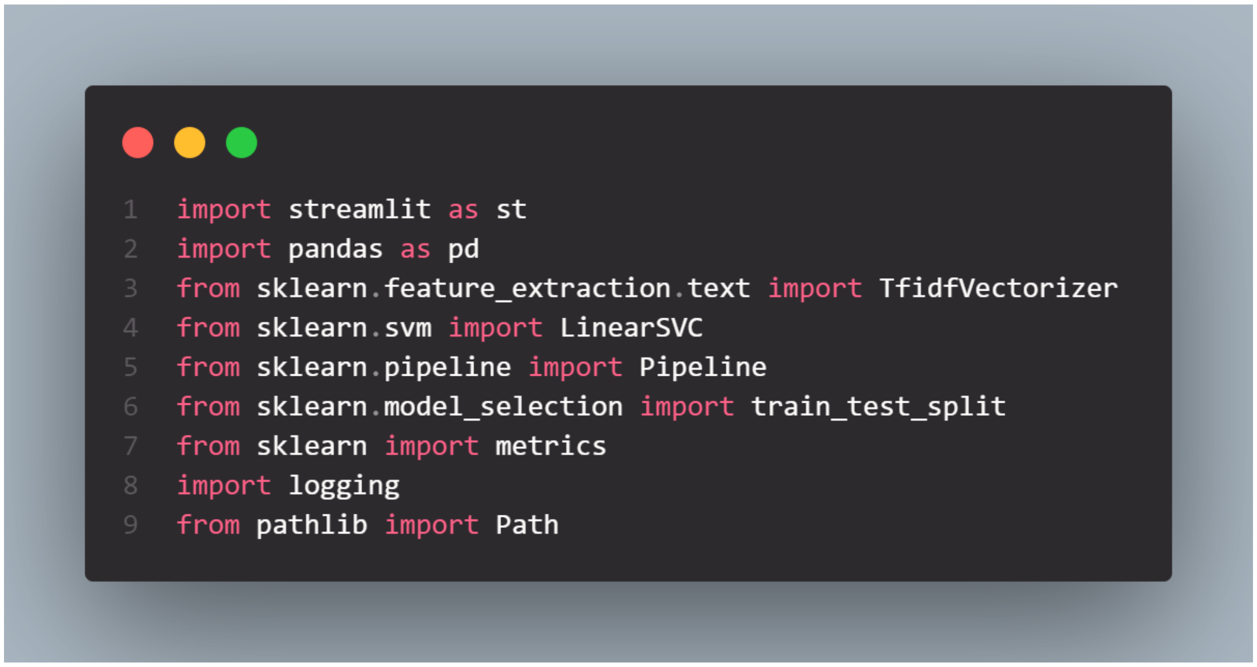
Now, we know the data is balanced but now we want to see what impact the most of our data, so we are going to display Top 10 or 20 symptoms:



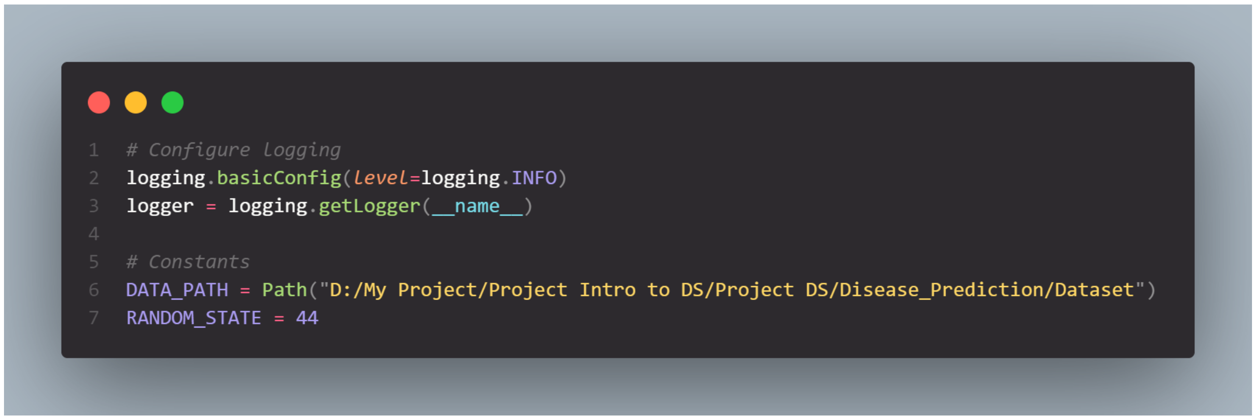
Deployment

Once, we complete all the process, we are going to deploy the model with *Streamlit* to run it as website and easy interface for user.

Important Libraries:

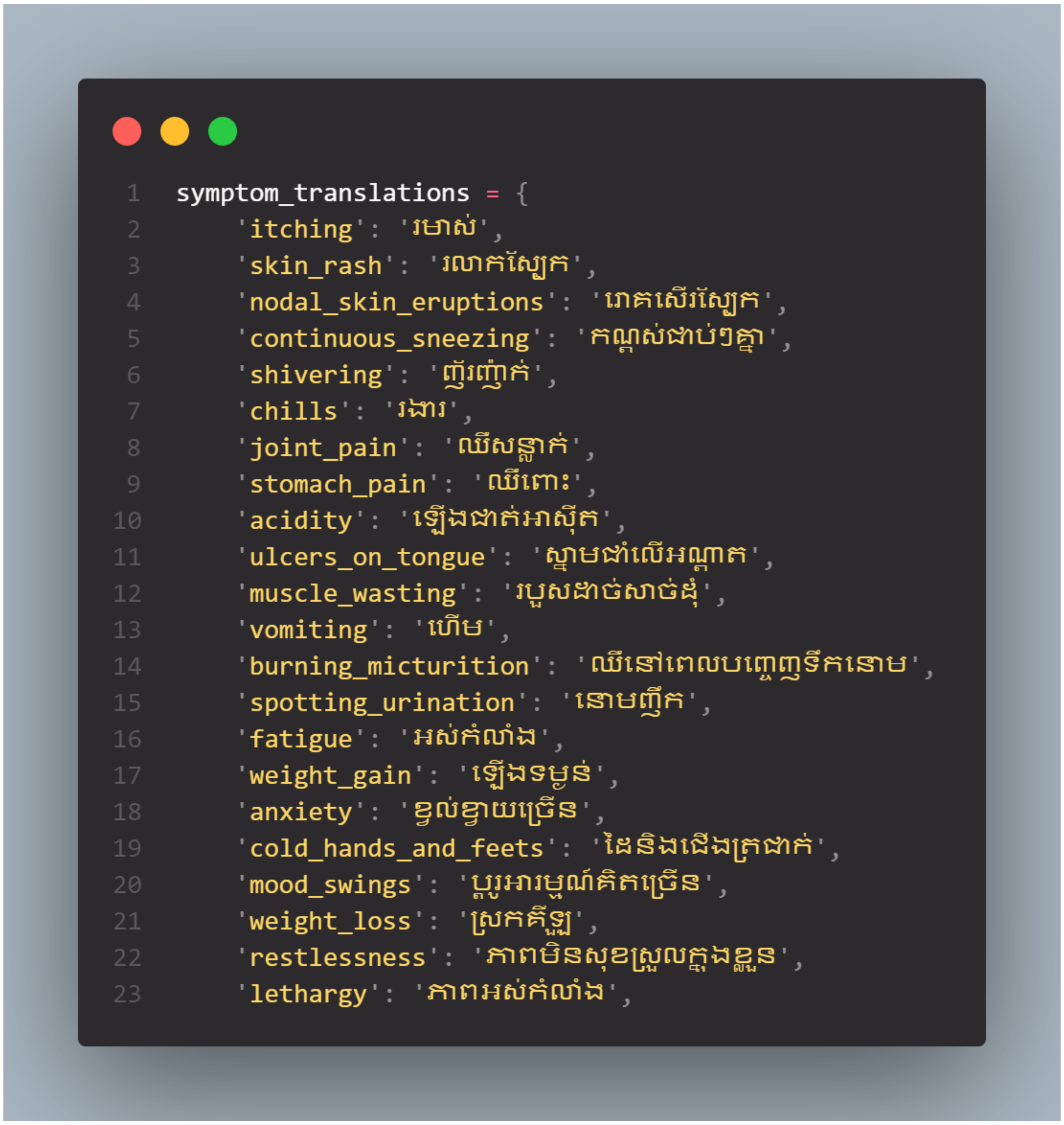


```python
import streamlit as st
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import LinearSVC
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn import metrics
import logging
from pathlib import Path
```



```python
# Configure Logging
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

# Constants
DATA_PATH = Path("D:/My Project/Project Intro to DS/Project DS/Disease_Prediction/Dataset")
RANDOM_STATE = 44
```

In my application file, we are doing to translation of symptoms easy for users to enter their symptoms. Furthermore, we also add the precautions and diseases description for the users in our website.



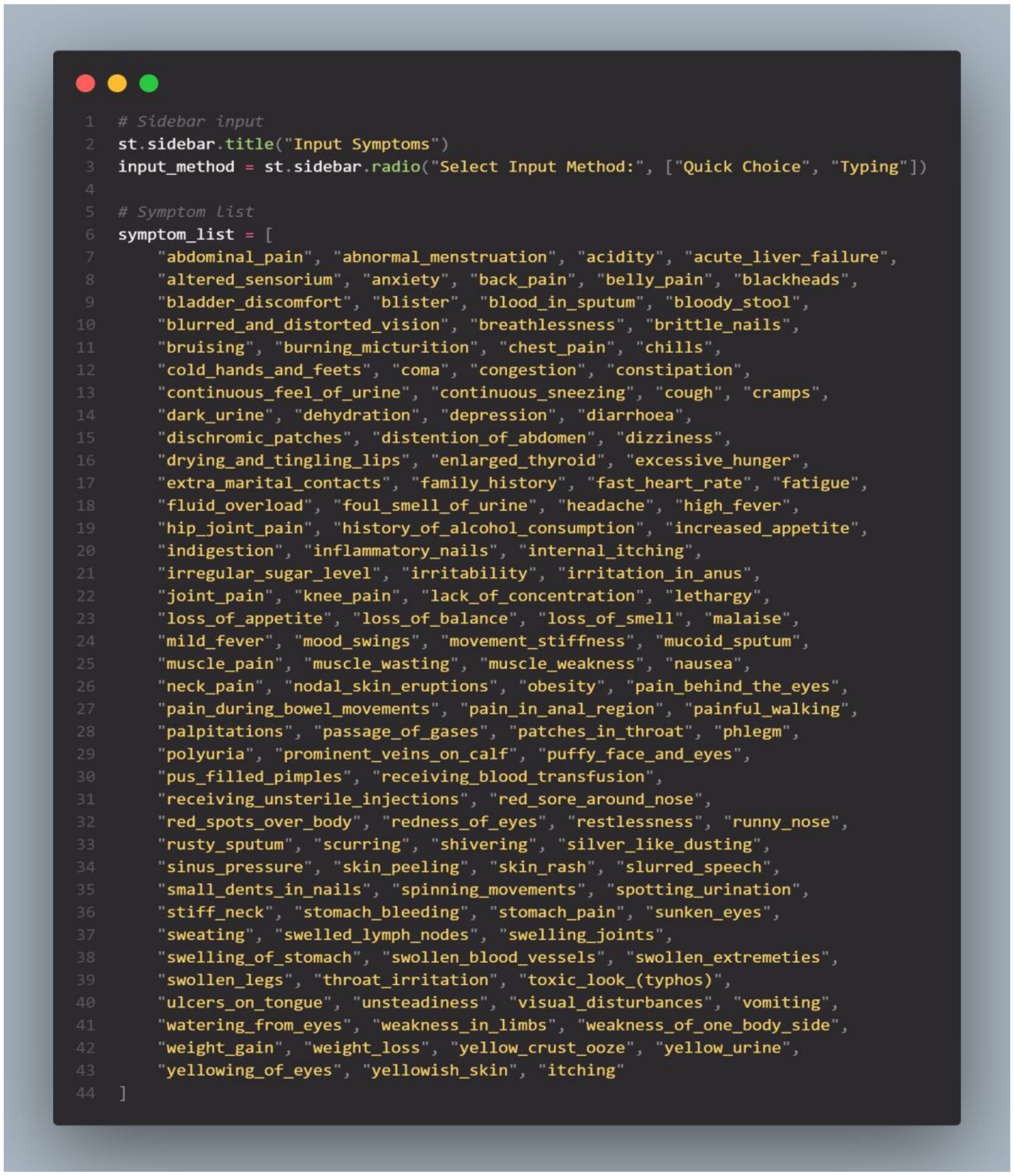
```python
symptom_translations = {
    'itching': 'រមាស់',
    'skin_rash': 'រលាកស្បែក',
    'nodal_skin_eruptions': 'រោគសើរស្បែក',
    'continuous_sneezing': 'កណ្តាស់ជាប់ៗគ្នា',
    'shivering': 'ញ័រញាក់',
    'chills': 'រងារ',
    'joint_pain': 'ឈឺសន្លាក់',
    'stomach_pain': 'ឈឺពោះ',
    'acidity': 'ឡើងជាត់អាស៊ីត',
    'ulcers_on_tongue': 'ស្នាមដំាលើអណ្តាត',
    'muscle_wasting': 'របួសដាច់សាច់ដុំ',
    'vomiting': 'ហើម',
    'burning_micturition': 'ឈឺនៅពេលបញ្ចេញទឹកនោម',
    'spotting_urination': 'នោមញឹក',
    'fatigue': 'អស់កំលាំង',
    'weight_gain': 'ឡើងទម្ងន់',
    'anxiety': 'ខ្វល់ខ្វាយច្រើន',
    'cold_hands_and_feets': 'ដៃនិងជើងត្រជាក់',
    'mood_swings': 'ប្តូរអារម្មណ៍គិតច្រើន',
    'weight_loss': 'ស្រកគីឡូ',
    'restlessness': 'ភាពមិនសុខស្រួលក្នុងខ្លួន',
    'lethargy': 'ភាពអស់កំលាំង',
```

In our streamlit website, we include two options there

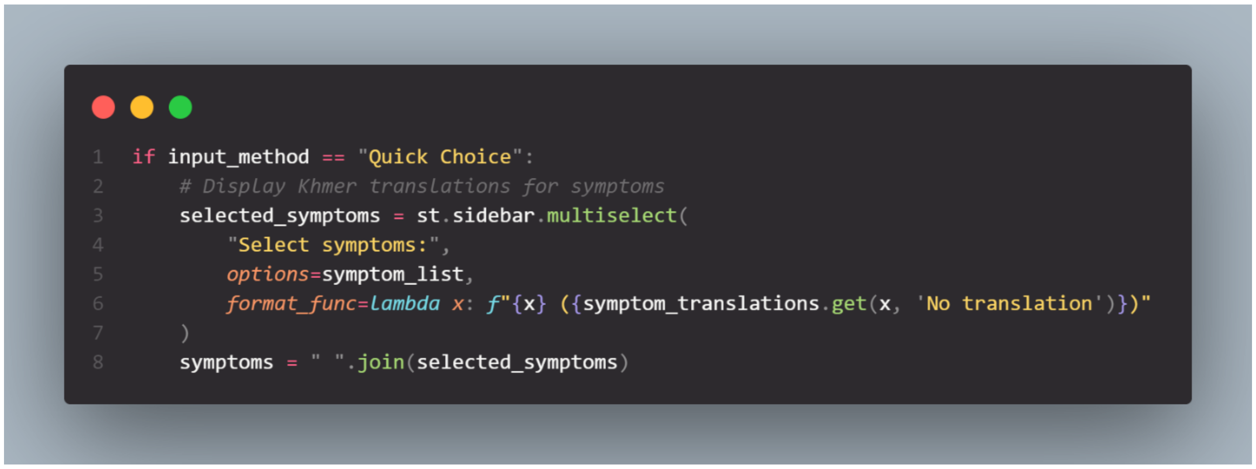
1) Quick Choose

2) Typing

This is the implement for Quick Choose

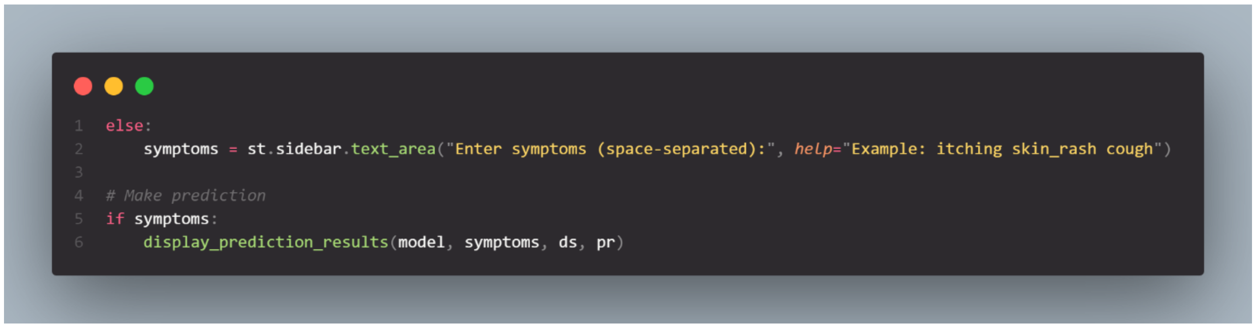


```python
# Sidebar input
st.sidebar.title("Input Symptoms")
input_method = st.sidebar.radio("Select Input Method:", ["Quick Choice", "Typing"])

# Symptom List
symptom_list = [
    "abdominal_pain", "abnormal_menstruation", "acidity", "acute_liver_failure",
    "altered_sensorium", "anxiety", "back_pain", "belly_pain", "blackheads",
    "bladder_discomfort", "blister", "blood_in_sputum", "bloody_stool",
    "blurred_and_distorted_vision", "breathlessness", "brittle_nails",
    "bruising", "burning_micturition", "chest_pain", "chills",
    "cold_hands_and_feets", "coma", "congestion", "constipation",
    "continuous_feel_of_urine", "continuous_sneezing", "cough", "cramps",
    "dark_urine", "dehydration", "depression", "diarrhoea",
    "dischromic_patches", "distention_of_abdomen", "dizziness",
    "drying_and_tingling_lips", "enlarged_thyroid", "excessive_hunger",
    "extra_marital_contacts", "family_history", "fast_heart_rate", "fatigue",
    "fluid_overload", "foul_smell_of_urine", "headache", "high_fever",
    "hip_joint_pain", "history_of_alcohol_consumption", "increased_appetite",
    "indigestion", "inflammatory_nails", "internal_itching",
    "irregular_sugar_level", "irritability", "irritation_in_anus",
    "joint_pain", "knee_pain", "lack_of_concentration", "lethargy",
    "loss_of_appetite", "loss_of_balance", "loss_of_smell", "malaise",
    "mild_fever", "mood_swings", "movement_stiffness", "mucoid_sputum",
    "muscle_pain", "muscle_wasting", "muscle_weakness", "nausea",
    "neck_pain", "nodal_skin_eruptions", "obesity", "pain_behind_the_eyes",
    "pain_during_bowel_movements", "pain_in_anal_region", "painful_walking",
    "palpitations", "passage_of_gases", "patches_in_throat", "phlegm",
    "polyuria", "prominent_veins_on_calf", "puffy_face_and_eyes",
    "pus_filled_pimples", "receiving_blood_transfusion",
    "receiving_unsterile_injections", "red_sore_around_nose",
    "red_spots_over_body", "redness_of_eyes", "restlessness", "runny_nose",
    "rusty_sputum", "scurring", "shivering", "silver_like_dusting",
    "sinus_pressure", "skin_peeling", "skin_rash", "slurred_speech",
    "small_dents_in_nails", "spinning_movements", "spotting_urination",
    "stiff_neck", "stomach_bleeding", "stomach_pain", "sunken_eyes",
    "sweating", "swelled_lymph_nodes", "swelling_joints",
    "swelling_of_stomach", "swollen_blood_vessels", "swollen_extremeties",
    "swollen_legs", "throat_irritation", "toxic_look_(typhos)",
    "ulcers_on_tongue", "unsteadiness", "visual_disturbances", "vomiting",
    "watering_from_eyes", "weakness_in_limbs", "weakness_of_one_body_side",
    "weight_gain", "weight_loss", "yellow_crust_ooze", "yellow_urine",
    "yellowing_of_eyes", "yellowish_skin", "itching"
]
```



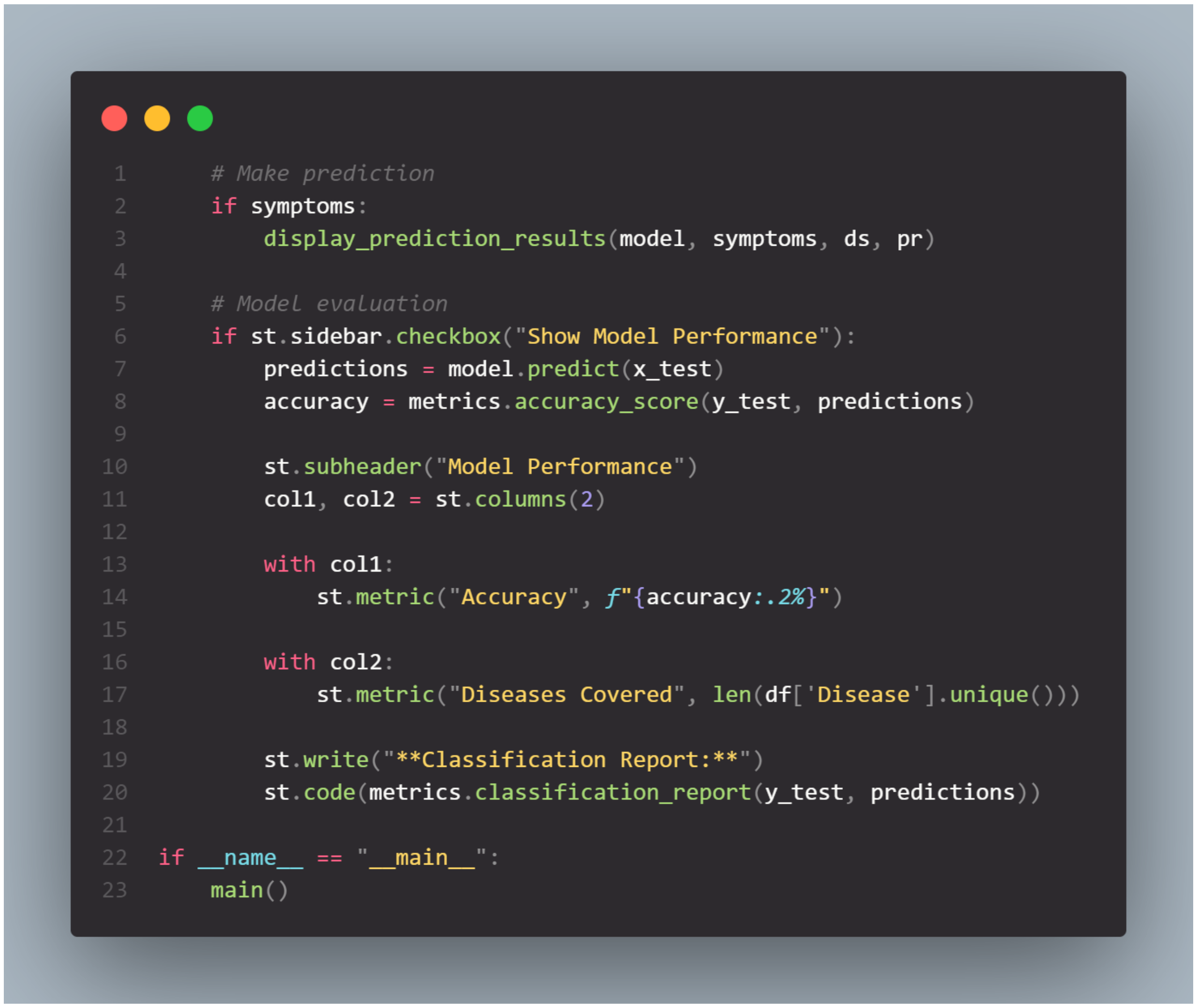
```python
if input_method == "Quick Choice":
    # Display Khmer translations for symptoms
    selected_symptoms = st.sidebar.multiselect(
        "Select symptoms:",
        options=symptom_list,
        format_func=lambda x: f"{x} ({symptom_translations.get(x, 'No translation')})"
    )
    symptoms = " ".join(selected_symptoms)
```

For Typing Option:



```python
else:
    symptoms = st.sidebar.text_area("Enter symptoms (space-separated):", help="Example: itching skin_rash cough")

# Make prediction
if symptoms:
    display_prediction_results(model, symptoms, ds, pr)
```

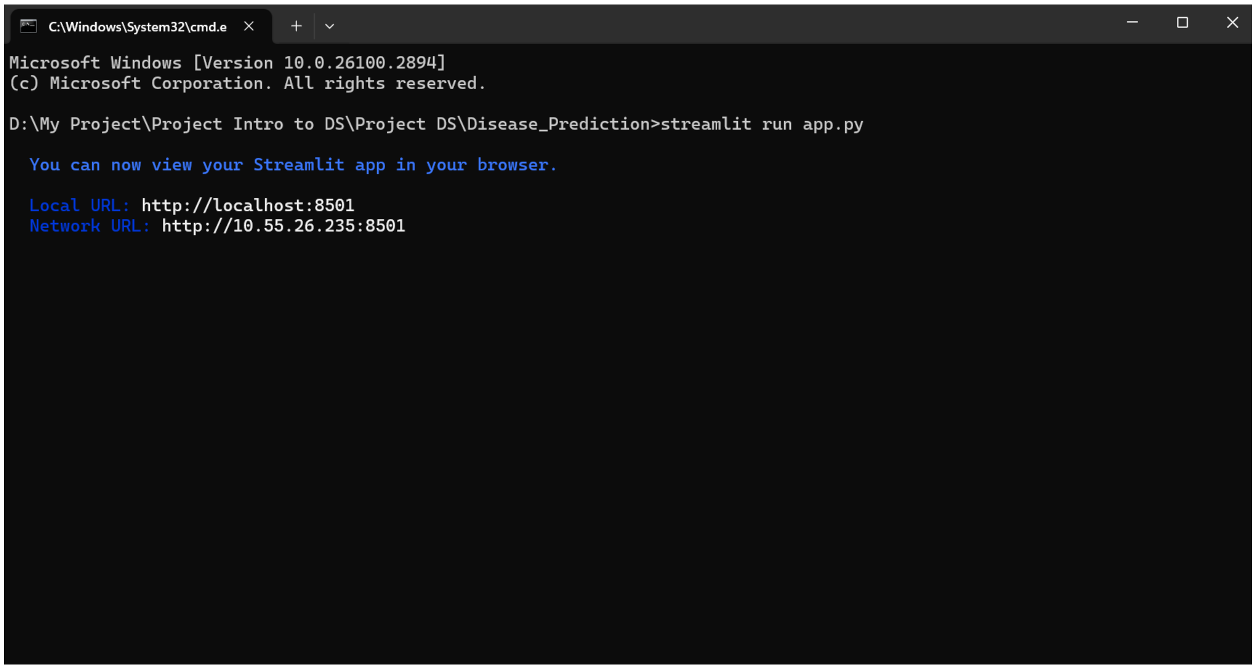
Make Predictions and Model evaluation



```python
    # Make prediction
    if symptoms:
        display_prediction_results(model, symptoms, ds, pr)

    # Model evaluation
    if st.sidebar.checkbox("Show Model Performance"):
        predictions = model.predict(x_test)
        accuracy = metrics.accuracy_score(y_test, predictions)

        st.subheader("Model Performance")
        col1, col2 = st.columns(2)

        with col1:
            st.metric("Accuracy", f"{accuracy:.2%}")

        with col2:
            st.metric("Diseases Covered", len(df['Disease'].unique()))

        st.write("**Classification Report:**")
        st.code(metrics.classification_report(y_test, predictions))

if __name__ == "__main__":
    main()
```

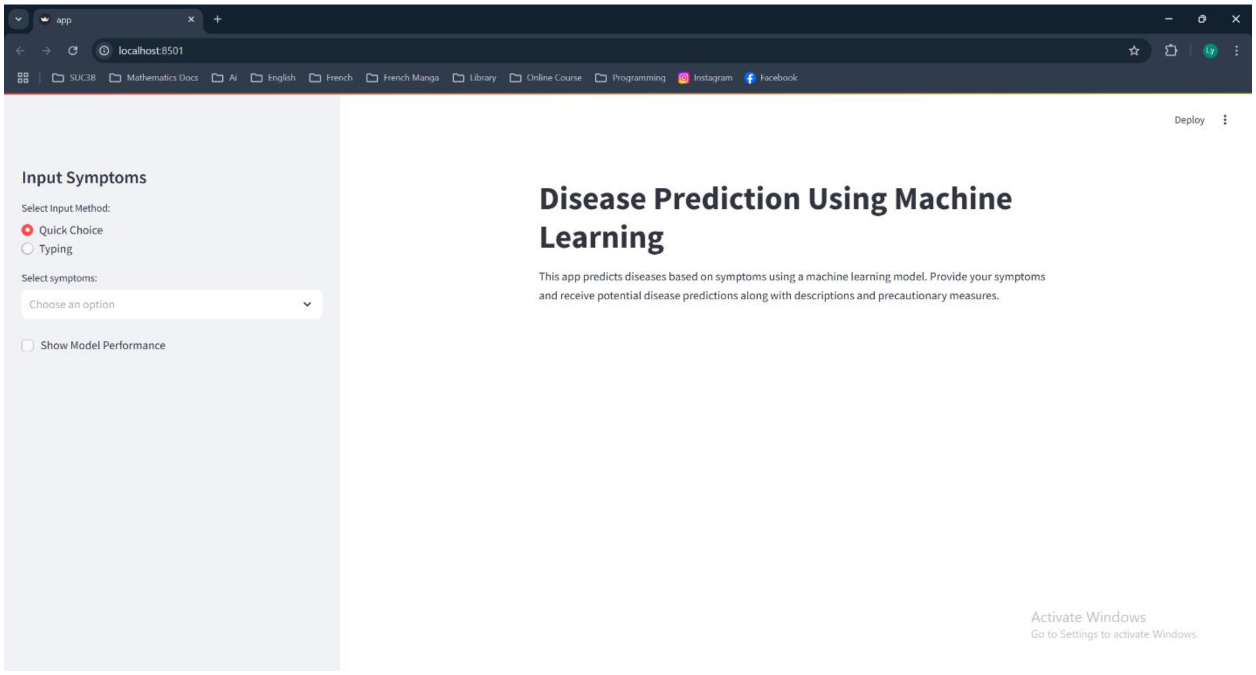
The Output

The command to run our app:



```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

D:\My Project\Project Intro to DS\Project DS\Disease_Prediction>streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://10.55.26.235:8501
```

Web Interface



Input Symptoms

Select Input Method:

[✓ Quick Choice]
Typing

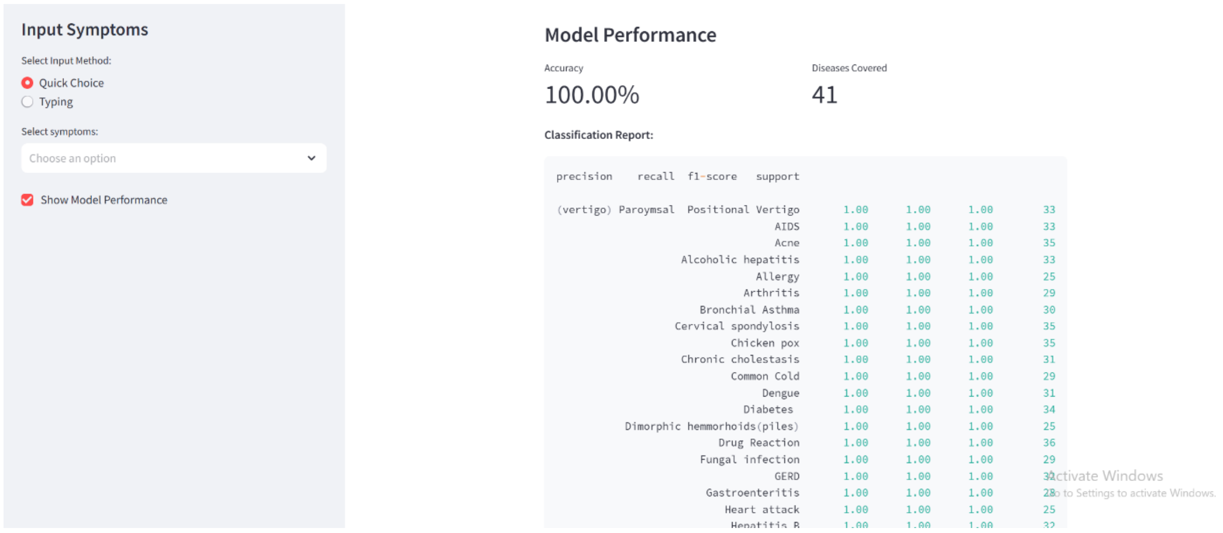
Select symptoms:

Choose an option

Show Model Performance

Disease Prediction Using Machine Learning

This app predicts diseases based on symptoms using a machine learning model. Provide your symptoms and receive potential disease predictions along with descriptions and precautionary measures.



Input Symptoms

Select Input Method:

[✓ Quick Choice]
Typing

Select symptoms:

Choose an option

[✓ Show Model Performance]

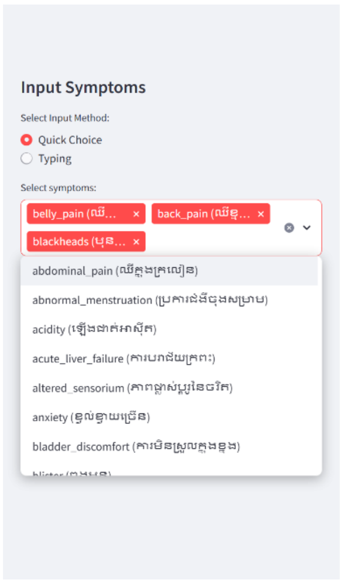
Model Performance

Accuracy
100.00%

Diseases Covered
41

Classification Report:

```
precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00        33
                                   AIDS       1.00      1.00      1.00        33
                                   Acne       1.00      1.00      1.00        35
                    Alcoholic hepatitis       1.00      1.00      1.00        33
                                Allergy       1.00      1.00      1.00        25
                              Arthritis       1.00      1.00      1.00        29
                       Bronchial Asthma       1.00      1.00      1.00        30
                   Cervical spondylosis       1.00      1.00      1.00        35
                            Chicken pox       1.00      1.00      1.00        35
                    Chronic cholestasis       1.00      1.00      1.00        31
                            Common Cold       1.00      1.00      1.00        29
                                 Dengue       1.00      1.00      1.00        31
                              Diabetes        1.00      1.00      1.00        34
        Dimorphic hemmorhoids(piles)       1.00      1.00      1.00        25
                           Drug Reaction       1.00      1.00      1.00        36
                         Fungal infection       1.00      1.00      1.00        29
                                   GERD       1.00      1.00      1.00        3[illegible]
                        Gastroenteritis       1.00      1.00      1.00        28
                           Heart attack       1.00      1.00      1.00        25
                            Hepatitis B       1.00      1.00      1.00        32
```



Input Symptoms

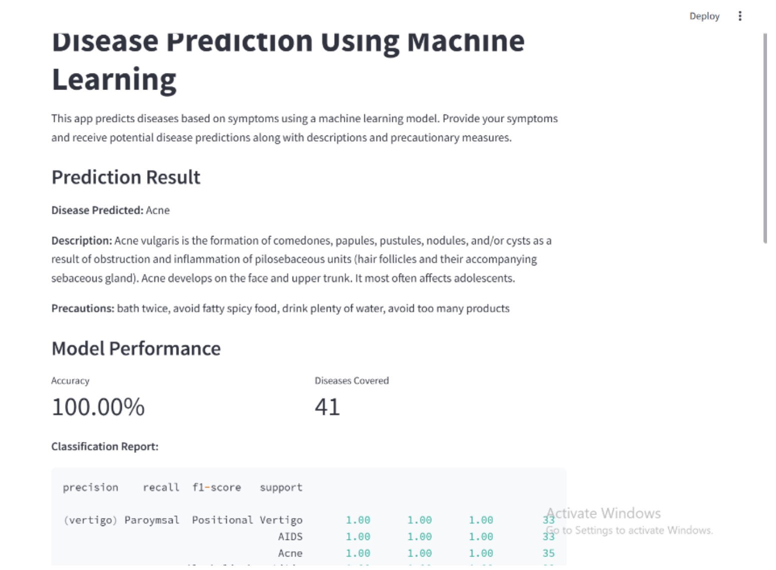
Select Input Method:

[✓ Quick Choice]
Typing

Select symptoms:

belly_pain (ឈឺ... × back_pain (ឈឺខ្ន... × blackheads (មុន... ×

- abdominal_pain (ឈឺក្នុងក្រលៀន)
- abnormal_menstruation (ប្រការជំងឺចុងសម្រាម)
- acidity (ឡើងជាត់អាស៊ីត)
- acute_liver_failure (ការបរាជ័យក្រពះ)
- altered_sensorium (ភាពផ្លាស់ប្តូរនៃចរិត)
- anxiety (ខ្វល់ខ្វាយច្រើន)
- bladder_discomfort (ការមិនស្រួលក្នុងខ្នង)



Disease Prediction Using Machine Learning

This app predicts diseases based on symptoms using a machine learning model. Provide your symptoms and receive potential disease predictions along with descriptions and precautionary measures.

Prediction Result

Disease Predicted: Acne

Description: Acne vulgaris is the formation of comedones, papules, pustules, nodules, and/or cysts as a result of obstruction and inflammation of pilosebaceous units (hair follicles and their accompanying sebaceous gland). Acne develops on the face and upper trunk. It most often affects adolescents.

Precautions: bath twice, avoid fatty spicy food, drink plenty of water, avoid too many products

Model Performance

Accuracy

100.00%

Diseases Covered

41

Classification Report:

```
precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00        33
                                   AIDS       1.00      1.00      1.00        33
                                   Acne       1.00      1.00      1.00        35
```

IV. Future Work

- **Deploy our app into the website**
- **Improve our interface**
- **Create other models to predict**
- **Add more data to predict more disease**
- **Can provide the medicine type for each disease**
- **Can discuss with our Ai ChatBot**

V. Result

After training and testing our models, we evaluated their performance using various metrics, such as accuracy, precision, and recall. The results showed that some models performed better than others. For instance, the neural network model demonstrated a higher accuracy compared to traditional models.

We also created visualizations to represent the results and understand the model's predictions better. This helped us communicate our findings effectively.

VI. Conclusion

In conclusion, this project successfully demonstrated the potential of machine learning in healthcare. By using models like *linear SVC* and *GradientBoostingClassification*, we were able to enhance disease prediction accuracy. This work can lead to better patient outcomes and more efficient healthcare systems in the future. As technology continues to advance, we believe that integrating machine learning into medical practices will be essential.

VII. References

- Dataset: https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning
- Exploration of Data Analysis: Lecture course from **Dr. PHAUK Sokkhey**
- Model of Linear SVC: https://scikit-learn.org/dev/modules/generated/sklearn.svm.LinearSVC.html
- GradientBoostingClassifcation: https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html [1], https://www.kaggle.com/code/beagle01/prediction-with-gradient-boosting-classifier
- Youtube
- https://medium.com/geekculture/gradient-boosting-classifier-f7a6834979d8
- TP Practice from Mr. PEN Chentra
- https://www.geeksforgeeks.org/ml-gradient-boosting/