Supervised Learning Final Project

February 26, 2025

Problem description

The problem of identifying diseases based on symptopms involves several challenges nowdays: - Overlapping symptoms: Many diseases share similar symptoms, this make to reach an accurate dianosis - Lack of medical services: Patients withput access to specialist may not receiba an early and accurate diagnosis - Even experienced doctors can make mistakes - Limited time and resources: In high-demand hospitals like Mexico, doctos may not have enough time to analyze all factors of a patient's condition

1.0 Exploratory Data Analysis procedure

1.1 Dataset

Dataset: Disease Symptom Prediction, Kaggle https://www.kaggle.com/datasets/itachi9604/disease-symptom-description-dataset

1.2 Machine Learning Models

- RandomForestClassifier: This is an ensemble learning method that operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification) of the individual trees.
- LogisticRegression: This is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. It is used for binary classification problems.

[12]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from IPython.display import display, HTML

data  = pd.read_csv("../data/dataset.csv")
```

With EDA, I start to explore and analysis the data. Starting using.shape to look at the dimeniosn of dataset, with proving the number of rows and columns below.

```
[13]: #Display the colums of the data set
      print("Number of Disease(Rows): " + str(data.shape[0]))
      print("Number of Symptom(Columns): " + str(data.shape[1]))
      print(data)
```

```
Number of Disease(Rows): 4920
Number of Symptom(Columns): 18
                                       Disease              Symptom_1  \
0                             Fungal infection                itching
1                             Fungal infection              skin_rash
2                             Fungal infection                itching
3                             Fungal infection                itching
4                             Fungal infection                itching
...                                        ...                    ...
4915  (vertigo) Paroymsal  Positional Vertigo               vomiting
4916                                      Acne              skin_rash
4917                   Urinary tract infection   burning_micturition
4918                                 Psoriasis              skin_rash
4919                                  Impetigo              skin_rash

                Symptom_2              Symptom_3                 Symptom_4  \
0               skin_rash   nodal_skin_eruptions       dischromic _patches
1    nodal_skin_eruptions    dischromic _patches                       NaN
2    nodal_skin_eruptions    dischromic _patches                       NaN
3               skin_rash    dischromic _patches                       NaN
4               skin_rash   nodal_skin_eruptions                       NaN
...                   ...                    ...                       ...
4915             headache                 nausea        spinning_movements
4916    pus_filled_pimples             blackheads                  scurring
4917    bladder_discomfort    foul_smell_of urine   continuous_feel_of_urine
4918            joint_pain            skin_peeling        silver_like_dusting
4919           high_fever                blister       red_sore_around_nose

                Symptom_5            Symptom_6 Symptom_7 Symptom_8  \
0                     NaN                  NaN       NaN       NaN
1                     NaN                  NaN       NaN       NaN
2                     NaN                  NaN       NaN       NaN
3                     NaN                  NaN       NaN       NaN
4                     NaN                  NaN       NaN       NaN
...                   ...                  ...       ...       ...
4915      loss_of_balance         unsteadiness       NaN       NaN
4916                  NaN                  NaN       NaN       NaN
4917                  NaN                  NaN       NaN       NaN
4918  small_dents_in_nails  inflammatory_nails       NaN       NaN
4919     yellow_crust_ooze                  NaN       NaN       NaN

     Symptom_9 Symptom_10 Symptom_11 Symptom_12 Symptom_13 Symptom_14  \
0          NaN        NaN        NaN        NaN        NaN        NaN
```

```
1           NaN         NaN         NaN         NaN         NaN         NaN
2           NaN         NaN         NaN         NaN         NaN         NaN
3           NaN         NaN         NaN         NaN         NaN         NaN
4           NaN         NaN         NaN         NaN         NaN         NaN
...         ...         ...         ...         ...         ...         ...
4915        NaN         NaN         NaN         NaN         NaN         NaN
4916        NaN         NaN         NaN         NaN         NaN         NaN
4917        NaN         NaN         NaN         NaN         NaN         NaN
4918        NaN         NaN         NaN         NaN         NaN         NaN
4919        NaN         NaN         NaN         NaN         NaN         NaN

     Symptom_15 Symptom_16 Symptom_17
0           NaN         NaN         NaN
1           NaN         NaN         NaN
2           NaN         NaN         NaN
3           NaN         NaN         NaN
4           NaN         NaN         NaN
...         ...         ...         ...
4915        NaN         NaN         NaN
4916        NaN         NaN         NaN
4917        NaN         NaN         NaN
4918        NaN         NaN         NaN
4919        NaN         NaN         NaN

[4920 rows x 18 columns]
```

1.3 Statistical Metrics

data.describe() displays the mean, std, minimum, and maximum values for the dataset which will assist in understand the range/distribution of the data while data.head() displays the first rows of a DataFrame, it hellp me with quicky inspeact the structure and contect of dataset

[14]:
```
data.describe()
```

[14]:
```
                 Disease   Symptom_1   Symptom_2 Symptom_3    Symptom_4  \
count               4920        4920        4920      4920         4572
unique                41          34          48        54           50
top     Fungal infection    vomiting    vomiting   fatigue  high_fever
freq                 120         822         870       726          378

        Symptom_5 Symptom_6       Symptom_7        Symptom_8  \
count        3714      2934            2268             1944
unique         38        32              26               21
top      headache    nausea  abdominal_pain   abdominal_pain
freq          348       390             264              276

                 Symptom_9          Symptom_10      Symptom_11 Symptom_12  \
count                 1692                1512            1194         744
```

```
unique                  22                21            18         11
top      yellowing_of_eyes  yellowing_of_eyes  irritability    malaise
freq                   228               198           120        126

          Symptom_13   Symptom_14   Symptom_15        Symptom_16    Symptom_17
count            504          306          240               192            72
unique             8            4            3                 3             1
top      muscle_pain   chest_pain   chest_pain   blood_in_sputum   muscle_pain
freq              72           96          144                72            72
```

[15]:
```
data.head()
```

[15]:
```
            Disease   Symptom_1              Symptom_2              Symptom_3  \
0  Fungal infection     itching              skin_rash   nodal_skin_eruptions
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

             Symptom_4 Symptom_5 Symptom_6 Symptom_7 Symptom_8 Symptom_9  \
0  dischromic _patches       NaN       NaN       NaN       NaN       NaN
1                  NaN       NaN       NaN       NaN       NaN       NaN
2                  NaN       NaN       NaN       NaN       NaN       NaN
3                  NaN       NaN       NaN       NaN       NaN       NaN
4                  NaN       NaN       NaN       NaN       NaN       NaN

  Symptom_10 Symptom_11 Symptom_12 Symptom_13 Symptom_14 Symptom_15  \
0        NaN        NaN        NaN        NaN        NaN        NaN
1        NaN        NaN        NaN        NaN        NaN        NaN
2        NaN        NaN        NaN        NaN        NaN        NaN
3        NaN        NaN        NaN        NaN        NaN        NaN
4        NaN        NaN        NaN        NaN        NaN        NaN

  Symptom_16 Symptom_17
0        NaN        NaN
1        NaN        NaN
2        NaN        NaN
3        NaN        NaN
4        NaN        NaN
```

1.4 Data cleaning

[16]:
```
# Identify symptom columns (excluding disease label)
symptom_cols = [col for col in data.columns if "Symptom" in col]
label_col = "Disease"  # Adjust this based on your dataset

# Encode disease labels using Label Encoding
```

```
label_encoder = LabelEncoder()
data[label_col] = label_encoder.fit_transform(data[label_col])

# =====================
# **Identify and Encode Categorical Columns**
# =====================

# Identify categorical symptom columns (if any)
categorical_cols = data[symptom_cols].select_dtypes(include=['object']).columns.
 ↪tolist()

# One-hot encode categorical symptoms if needed
onehot_encoder = OneHotEncoder(handle_unknown='ignore', sparse_output=False)
onehot_encoder.fit(data[categorical_cols])

# =====================
# **Apply One-Hot Encoding and Finalize Dataset**
# =====================

# Convert categorical values to one-hot encoding
data_encoded = pd.DataFrame(onehot_encoder.transform(data[categorical_cols]),
                          columns=onehot_encoder.
 ↪get_feature_names_out(categorical_cols))

# Combine original dataset (excluding categorical columns) with the one-hot␣
 ↪encoded data
data_final = pd.concat([data.drop(columns=categorical_cols), data_encoded],␣
 ↪axis=1)

# Separate features and labels
X = data_final.drop(columns=[label_col])
y = data_final[label_col]

# =====================
# **Correlation Matrix**
# =====================
data_encoded = data.apply(LabelEncoder().fit_transform)
correlation_matrix = data_encoded.corr()
print(correlation_matrix)

plt.figure(figsize=(11,11))
sns.heatmap(correlation_matrix,
            cmap= 'coolwarm',
            annot=True,
            fmt=".3f",
            annot_kws={"size": 6, "weight": "bold"},
            square=True,
             linewidths=.05)
plt.savefig('../results/heat_mat.png', dpi=300)
plt.show()
```

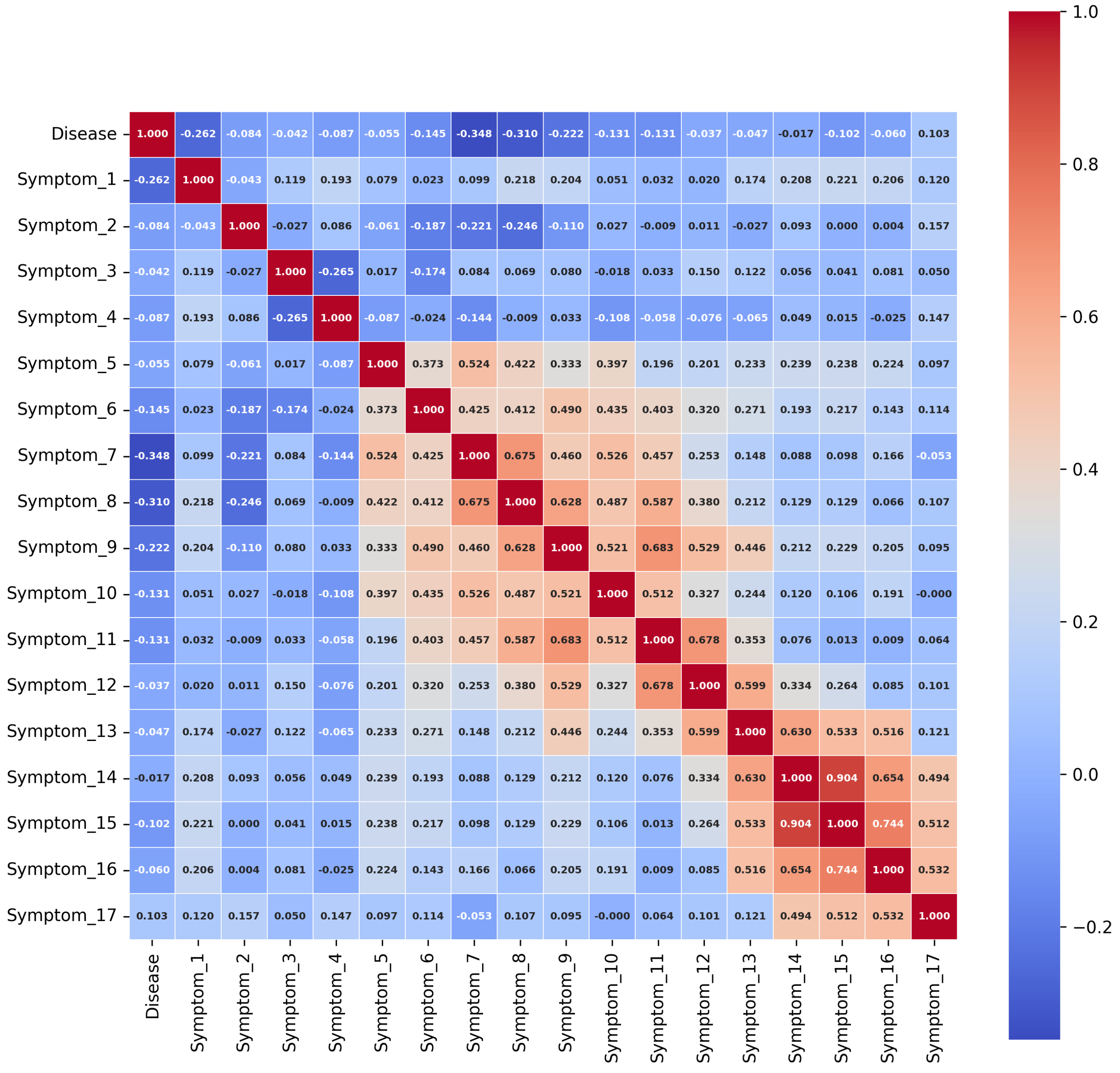
```
             Disease  Symptom_1  Symptom_2  Symptom_3  Symptom_4  Symptom_5  \
Disease     1.000000  -0.261814  -0.084411  -0.042227  -0.086792  -0.055276
Symptom_1  -0.261814   1.000000  -0.043362   0.118788   0.193146   0.078957
Symptom_2  -0.084411  -0.043362   1.000000  -0.026983   0.086465  -0.060654
Symptom_3  -0.042227   0.118788  -0.026983   1.000000  -0.264756   0.016827
Symptom_4  -0.086792   0.193146   0.086465  -0.264756   1.000000  -0.087494
Symptom_5  -0.055276   0.078957  -0.060654   0.016827  -0.087494   1.000000
Symptom_6  -0.145203   0.022503  -0.186539  -0.173736  -0.024463   0.372599
Symptom_7  -0.347849   0.098601  -0.221452   0.084273  -0.143803   0.524071
Symptom_8  -0.309583   0.217522  -0.246351   0.068856  -0.009458   0.421903
Symptom_9  -0.221644   0.203916  -0.110205   0.079941   0.032849   0.333107
Symptom_10 -0.130663   0.051263   0.026727  -0.017539  -0.108092   0.397099
Symptom_11 -0.130510   0.032281  -0.008550   0.033142  -0.057731   0.195535
Symptom_12 -0.037365   0.020023   0.010555   0.150479  -0.076284   0.201141
Symptom_13 -0.046965   0.173927  -0.026524   0.121668  -0.064585   0.233271
Symptom_14 -0.017249   0.207757   0.092681   0.056134   0.048586   0.239486
Symptom_15 -0.101953   0.220917   0.000254   0.041384   0.014555   0.237652
Symptom_16 -0.060041   0.206212   0.003978   0.081329  -0.025116   0.224450
Symptom_17  0.102996   0.120216   0.156542   0.049979   0.146808   0.097143

            Symptom_6  Symptom_7  Symptom_8  Symptom_9  Symptom_10  \
Disease     -0.145203  -0.347849  -0.309583  -0.221644   -0.130663
Symptom_1    0.022503   0.098601   0.217522   0.203916    0.051263
Symptom_2   -0.186539  -0.221452  -0.246351  -0.110205    0.026727
Symptom_3   -0.173736   0.084273   0.068856   0.079941   -0.017539
Symptom_4   -0.024463  -0.143803  -0.009458   0.032849   -0.108092
Symptom_5    0.372599   0.524071   0.421903   0.333107    0.397099
Symptom_6    1.000000   0.425125   0.412414   0.489612    0.435085
Symptom_7    0.425125   1.000000   0.675436   0.460296    0.525827
Symptom_8    0.412414   0.675436   1.000000   0.628161    0.487290
Symptom_9    0.489612   0.460296   0.628161   1.000000    0.521321
Symptom_10   0.435085   0.525827   0.487290   0.521321    1.000000
Symptom_11   0.402794   0.457050   0.586505   0.682528    0.512028
Symptom_12   0.320493   0.252574   0.379705   0.529157    0.327123
Symptom_13   0.270822   0.148477   0.211525   0.446362    0.243859
Symptom_14   0.193321   0.088069   0.129252   0.212035    0.119875
Symptom_15   0.216582   0.097767   0.128619   0.229046    0.105714
Symptom_16   0.143429   0.166294   0.065548   0.205173    0.191301
Symptom_17   0.113912  -0.053184   0.107253   0.094584   -0.000393

            Symptom_11  Symptom_12  Symptom_13  Symptom_14  Symptom_15  \
Disease      -0.130510   -0.037365   -0.046965   -0.017249   -0.101953
Symptom_1     0.032281    0.020023    0.173927    0.207757    0.220917
Symptom_2    -0.008550    0.010555   -0.026524    0.092681    0.000254
```

```
Symptom_3     0.033142   0.150479   0.121668   0.056134   0.041384
Symptom_4    -0.057731  -0.076284  -0.064585   0.048586   0.014555
Symptom_5     0.195535   0.201141   0.233271   0.239486   0.237652
Symptom_6     0.402794   0.320493   0.270822   0.193321   0.216582
Symptom_7     0.457050   0.252574   0.148477   0.088069   0.097767
Symptom_8     0.586505   0.379705   0.211525   0.129252   0.128619
Symptom_9     0.682528   0.529157   0.446362   0.212035   0.229046
Symptom_10    0.512028   0.327123   0.243859   0.119875   0.105714
Symptom_11    1.000000   0.678369   0.352873   0.076045   0.013043
Symptom_12    0.678369   1.000000   0.599215   0.333772   0.264498
Symptom_13    0.352873   0.599215   1.000000   0.629942   0.532589
Symptom_14    0.076045   0.333772   0.629942   1.000000   0.904473
Symptom_15    0.013043   0.264498   0.532589   0.904473   1.000000
Symptom_16    0.009409   0.085288   0.515840   0.654178   0.743645
Symptom_17    0.063949   0.100924   0.120903   0.494255   0.511913

             Symptom_16  Symptom_17
Disease      -0.060041   0.102996
Symptom_1     0.206212   0.120216
Symptom_2     0.003978   0.156542
Symptom_3     0.081329   0.049979
Symptom_4    -0.025116   0.146808
Symptom_5     0.224450   0.097143
Symptom_6     0.143429   0.113912
Symptom_7     0.166294  -0.053184
Symptom_8     0.065548   0.107253
Symptom_9     0.205173   0.094584
Symptom_10    0.191301  -0.000393
Symptom_11    0.009409   0.063949
Symptom_12    0.085288   0.100924
Symptom_13    0.515840   0.120903
Symptom_14    0.654178   0.494255
Symptom_15    0.743645   0.511913
Symptom_16    1.000000   0.531626
Symptom_17    0.531626   1.000000
```



	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	Symptom_11	Symptom_12	Symptom_13	Symptom_14	Symptom_15	Symptom_16	Symptom_17
Disease	1.000	-0.262	-0.084	-0.042	-0.087	-0.055	-0.145	-0.348	-0.310	-0.222	-0.131	-0.131	-0.037	-0.047	-0.017	-0.102	-0.060	0.103
Symptom_1	-0.262	1.000	-0.043	0.119	0.193	0.079	0.023	0.099	0.218	0.204	0.051	0.032	0.020	0.174	0.208	0.221	0.206	0.120
Symptom_2	-0.084	-0.043	1.000	-0.027	0.086	-0.061	-0.187	-0.221	-0.246	-0.110	0.027	-0.009	0.011	-0.027	0.093	0.000	0.004	0.157
Symptom_3	-0.042	0.119	-0.027	1.000	-0.265	0.017	-0.174	0.084	0.069	0.080	-0.018	0.033	0.150	0.122	0.056	0.041	0.081	0.050
Symptom_4	-0.087	0.193	0.086	-0.265	1.000	-0.087	-0.024	-0.144	-0.009	0.033	-0.108	-0.058	-0.076	-0.065	0.049	0.015	-0.025	0.147
Symptom_5	-0.055	0.079	-0.061	0.017	-0.087	1.000	0.373	0.524	0.422	0.333	0.397	0.196	0.201	0.233	0.239	0.238	0.224	0.097
Symptom_6	-0.145	0.023	-0.187	-0.174	-0.024	0.373	1.000	0.425	0.412	0.490	0.435	0.403	0.320	0.271	0.193	0.217	0.143	0.114
Symptom_7	-0.348	0.099	-0.221	0.084	-0.144	0.524	0.425	1.000	0.675	0.460	0.526	0.457	0.253	0.148	0.088	0.098	0.166	-0.053
Symptom_8	-0.310	0.218	-0.246	0.069	-0.009	0.422	0.412	0.675	1.000	0.628	0.487	0.587	0.380	0.212	0.129	0.129	0.066	0.107
Symptom_9	-0.222	0.204	-0.110	0.080	0.033	0.333	0.490	0.460	0.628	1.000	0.521	0.683	0.529	0.446	0.212	0.229	0.205	0.095
Symptom_10	-0.131	0.051	0.027	-0.018	-0.108	0.397	0.435	0.526	0.487	0.521	1.000	0.512	0.327	0.244	0.120	0.106	0.191	-0.000
Symptom_11	-0.131	0.032	-0.009	0.033	-0.058	0.196	0.403	0.457	0.587	0.683	0.512	1.000	0.678	0.353	0.076	0.013	0.009	0.064
Symptom_12	-0.037	0.020	0.011	0.150	-0.076	0.201	0.320	0.253	0.380	0.529	0.327	0.678	1.000	0.599	0.334	0.264	0.085	0.101
Symptom_13	-0.047	0.174	-0.027	0.122	-0.065	0.233	0.271	0.148	0.212	0.446	0.244	0.353	0.599	1.000	0.630	0.533	0.516	0.121
Symptom_14	-0.017	0.208	0.093	0.056	0.049	0.239	0.193	0.088	0.129	0.212	0.120	0.076	0.334	0.630	1.000	0.904	0.654	0.494
Symptom_15	-0.102	0.221	0.000	0.041	0.015	0.238	0.217	0.098	0.129	0.229	0.106	0.013	0.264	0.533	0.904	1.000	0.744	0.512
Symptom_16	-0.060	0.206	0.004	0.081	-0.025	0.224	0.143	0.166	0.066	0.205	0.191	0.009	0.085	0.516	0.654	0.744	1.000	0.532
Symptom_17	0.103	0.120	0.157	0.050	0.147	0.097	0.114	-0.053	0.107	0.095	-0.000	0.064	0.101	0.121	0.494	0.512	0.532	1.000

Fig 1. Correlation Matrix

2.0 Analysis

[17]:
```
# ======================
# **Train Model**
# ======================
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X, y)

lr_model = LogisticRegression(max_iter=1000, random_state=42)
lr_model.fit(X, y)

# ======================
# **Define New Symptoms for Prediction**
# ======================
input_symptoms = {
    "Symptom_1": "0",
    "Symptom_2": "0",
    "Symptom_3": "1",
    "Symptom_4": "1",
    "Symptom_5": "0",
    "Symptom_6": "0",
    "Symptom_7": "0",
    "Symptom_8": "0",
    "Symptom_9": "0",
    "Symptom_10": "0",
    "Symptom_11": "0",
    "Symptom_12": "1",
    "Symptom_13": "0",
    "Symptom_14": "0",
    "Symptom_15": "0",
    "Symptom_16": "0",
    "Symptom_17": "1",
}

# Convert to DataFrame
input_data = pd.DataFrame([input_symptoms])

# Ensure input matches dataset columns, filling missing columns with "0"
input_data = input_data.reindex(columns=symptom_cols, fill_value="0")

# Encode categorical values if needed
if categorical_cols:
    input_encoded = pd.DataFrame(
        onehot_encoder.transform(input_data[categorical_cols]),
        columns=onehot_encoder.get_feature_names_out(categorical_cols),
        index=input_data.index
    )
    input_data = pd.concat([input_data.drop(columns=categorical_cols),␣
 ↪input_encoded], axis=1)

# Ensure final input matches trained feature columns, filling missing columns␣
 ↪with "0"
input_data = input_data.reindex(columns=X.columns, fill_value=0)

# ======================
# **Predict Disease (Random Forest)**
# ======================
rf_predicted_disease_index = rf_model.predict(input_data)[0]
rf_predicted_disease = label_encoder.
 ↪inverse_transform([rf_predicted_disease_index])[0]

rf_probabilities = rf_model.predict_proba(input_data)
rf_disease_probabilities = dict(zip(label_encoder.classes_, rf_probabilities[0]))
rf_sorted_predictions = sorted(rf_disease_probabilities.items(), key=lambda x:␣
 ↪x[1], reverse=True)

# ======================
# **Predict Disease (Logistic Regression)**
# ======================
lr_predicted_disease_index = lr_model.predict(input_data)[0]
lr_predicted_disease = label_encoder.
 ↪inverse_transform([lr_predicted_disease_index])[0]

lr_probabilities = lr_model.predict_proba(input_data)
lr_disease_probabilities = dict(zip(label_encoder.classes_, lr_probabilities[0]))
lr_sorted_predictions = sorted(lr_disease_probabilities.items(), key=lambda x:␣
 ↪x[1], reverse=True)
```

3.0 Results

3.1 Predict Disease(Random Forest and Logistic Regression) Results

[18]:
```
print(f"\n[Logistic Regression] Predicted Disease: {lr_predicted_disease}")
for disease, probability in lr_sorted_predictions[:5]:
    print(f"{disease}: {probability:.2f}")

print(f"\n[Random Forest] Predicted Disease: {rf_predicted_disease}")
for disease, probability in rf_sorted_predictions[:5]:
    print(f"{disease}: {probability:.2f}")
```

```

[Logistic Regression] Predicted Disease: Common Cold
Common Cold: 0.06
```

```
Tuberculosis: 0.05
Dengue: 0.04
Hypothyroidism: 0.04
Hepatitis E: 0.04

[Random Forest] Predicted Disease: Common Cold
Common Cold: 0.40
Tuberculosis: 0.27
Dengue: 0.08
Hypothyroidism: 0.07
Hyperthyroidism: 0.04
```

Confusion matrix

[19]:
```
cm = confusion_matrix(y, rf_model.predict(X))
print("\nConfusion matrix:")
display(HTML('<div style="height: 20px; overflow: auto;">' + str(cm) + '</div>'))
```

```

Confusion matrix:

<IPython.core.display.HTML object>
```

Model Performance on Test Data

[20]:
```
print("\nModel Performance on Test Data:")
print(classification_report(y, rf_model.predict(X), target_names=label_encoder.
 ↪classes_))
```

```

Model Performance on Test Data:
                                         precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00       120
                                   AIDS       1.00      1.00      1.00       120
                                   Acne       1.00      1.00      1.00       120
                    Alcoholic hepatitis       1.00      1.00      1.00       120
                                Allergy       1.00      1.00      1.00       120
                              Arthritis       1.00      1.00      1.00       120
                       Bronchial Asthma       1.00      1.00      1.00       120
                   Cervical spondylosis       1.00      1.00      1.00       120
                            Chicken pox       1.00      1.00      1.00       120
                    Chronic cholestasis       1.00      1.00      1.00       120
                            Common Cold       1.00      1.00      1.00       120
                                 Dengue       1.00      1.00      1.00       120
                              Diabetes        1.00      1.00      1.00       120
           Dimorphic hemmorhoids(piles)       1.00      1.00      1.00       120
                          Drug Reaction       1.00      1.00      1.00       120
                       Fungal infection       1.00      1.00      1.00       120
                                   GERD       1.00      1.00      1.00       120
```

```
              Gastroenteritis       1.00      1.00      1.00       120
                 Heart attack       1.00      1.00      1.00       120
                  Hepatitis B       1.00      1.00      1.00       120
                  Hepatitis C       1.00      1.00      1.00       120
                  Hepatitis D       1.00      1.00      1.00       120
                  Hepatitis E       1.00      1.00      1.00       120
                Hypertension        1.00      1.00      1.00       120
              Hyperthyroidism       1.00      1.00      1.00       120
                 Hypoglycemia       1.00      1.00      1.00       120
               Hypothyroidism       1.00      1.00      1.00       120
                     Impetigo       1.00      1.00      1.00       120
                     Jaundice       1.00      1.00      1.00       120
                      Malaria       1.00      1.00      1.00       120
                     Migraine       1.00      1.00      1.00       120
              Osteoarthristis       1.00      1.00      1.00       120
Paralysis (brain hemorrhage)        1.00      1.00      1.00       120
          Peptic ulcer diseae       1.00      1.00      1.00       120
                    Pneumonia       1.00      1.00      1.00       120
                    Psoriasis       1.00      1.00      1.00       120
                 Tuberculosis       1.00      1.00      1.00       120
                      Typhoid       1.00      1.00      1.00       120
      Urinary tract infection       1.00      1.00      1.00       120
               Varicose veins       1.00      1.00      1.00       120
                  hepatitis A       1.00      1.00      1.00       120

                     accuracy                           1.00      4920
                    macro avg       1.00      1.00      1.00      4920
                 weighted avg       1.00      1.00      1.00      4920
```

4.0 Conclusion

Random Forest algorithm demonstrated higher condidence in its predictions compared to Logistic regression, in this case we can see the difference for the Common Cold. This suggest that Random Forest is more effective for classification.

4.1 Confusion matrix Analysis

The confusion matrix shos that there are no missclassifications, with all true in positives correctly, no false positives o false negatives. Each disease category has 120 correct redictions.

4.2 Performance

The model achieved an overall accuracy of 100%, correctly classifying all instances in the test set - Precision: The proportion of true predictions among all positive predictions - Recall: The proportion of true positive predictions among all actual positives - F1-Score: The harmonic precision and recall, providing a single metric for both

4.3 Benefits

- **Doctors**:
  - An AI-based system could assist doctors in pre-diagnosing diseases.
  - It would help reduce the margin of error in diagnosis.
- **Patients**:
  - Patients without immediate access to healthcare could receive preliminary guidance about their health.
  - This system provides an initial step for self-assessment.
- **Hospitals**:
  - Hospitals could optimize medical care by reducing unnecessary consultations.
  - It would help prioritize urgent cases, ensuring that resources are allocated efficiently.