**ML Project CS6301**

**B.E CSE VI P - BATCH**

**TEAM MEMBERS: TEAM NO: 11**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **REG. NO.** | **NAME** |
| **1** | **2020103558** | **Rishi Gauthama T** |
| **2** | **2020103085** | **Velmurugan J** |
| **3** | **2020103581** | **Tarini Thuthika** |

**Project Title:**

**Prediction of Diseases Using Different Machine Learning Approaches**

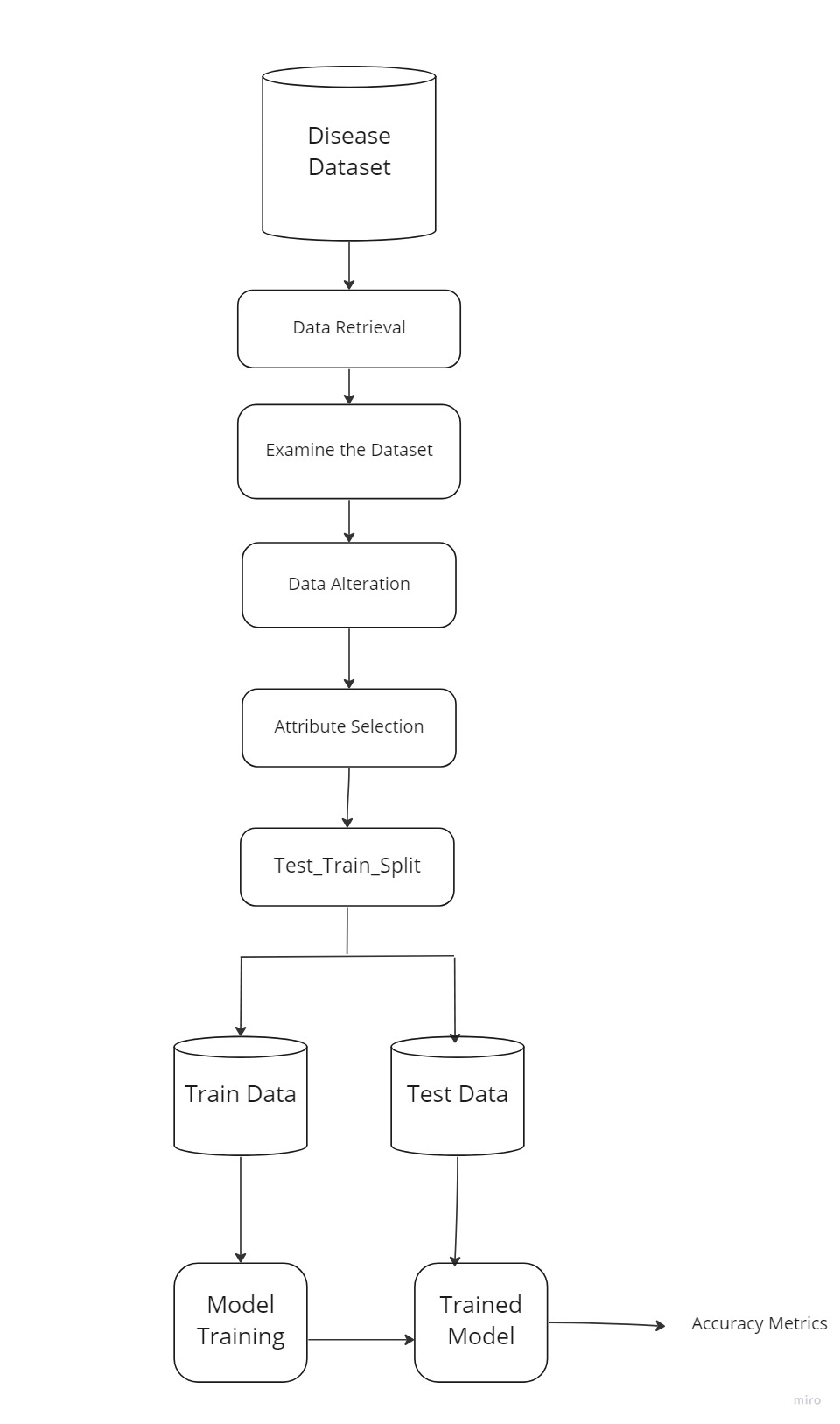
**Problem Statement**

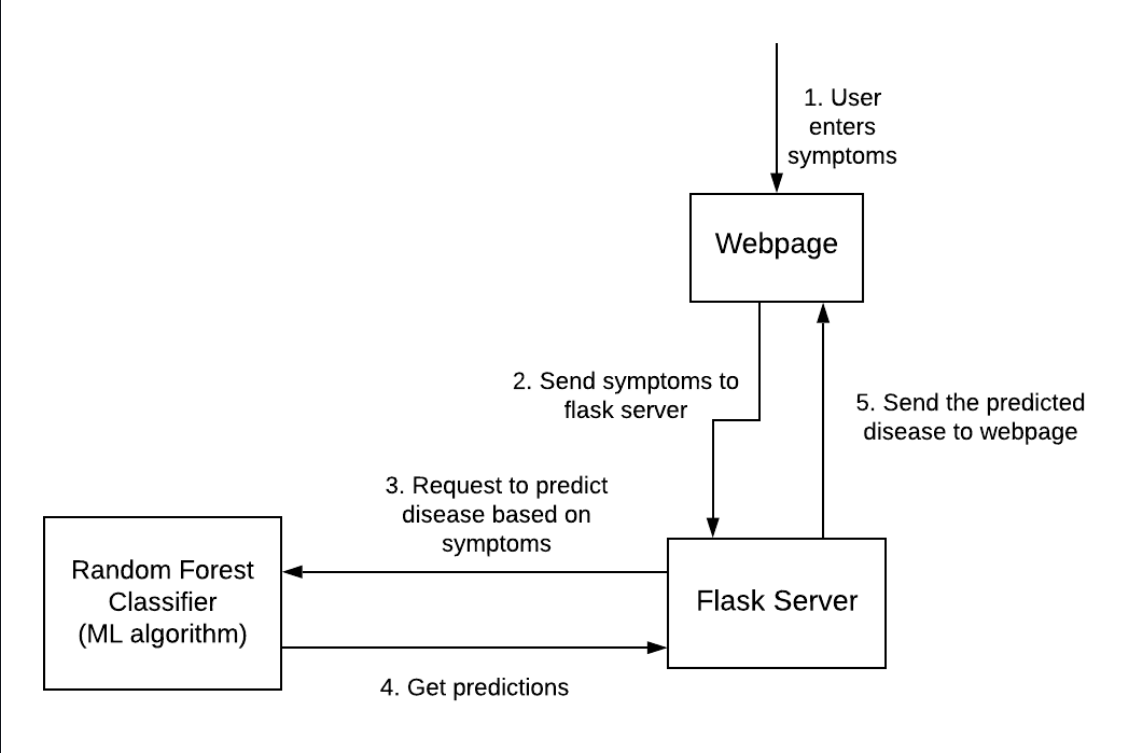
This project seeks to address the challenges of accurately diagnosing diseases by developing a machine learning model that predicts diseases based on patient symptoms. The model will be trained on a large dataset of patient symptoms and corresponding diagnoses. The project aims to improve diagnosis accuracy, reduce misdiagnosis rates, and enhance the efficiency of the healthcare system. The project's success will be evaluated based on the model's accuracy and its impact on improving healthcare outcomes.

**Objective of our project**

The medical system is overworked and becoming more and more expensive in many countries each year as the number of patients and diseases rises. In order to be treated, the bulk of the ailment requires a visit to the doctor. A simple and affordable algorithm can predict diseases if there is enough data. A key part of treatment is the ability to diagnose disease based on symptoms. Using the patient's symptoms, we sought to accurately predict a disease in our project. We are going to use four different algorithms to accomplish this to obtain better accuracy. The use of such a technology in medical care in the future is quite promising. In order to facilitate interaction with the data, we are also planning to develop an interactive interface.

**Overall Architecture**





**List of modules :**

Data Input:

This module involves collecting relevant patient data, such as symptoms, medical history, and demographics, to be used as input for the disease prediction model.

Data Preprocessing:

This module involves cleaning and transforming the collected data to prepare it for analysis. This may include tasks such as removing missing or erroneous data, normalizing or standardizing the data, and converting categorical variables to numerical ones.

Model Training and Validation:

This module involves selecting and training different machine learning algorithms, such as decision trees, random forests, logistic regression, support vector machines, or neural networks, using the prepared data. The trained models are then validated using various evaluation metrics, such as accuracy, precision, recall, and F1 score, to assess their performance and identify the best-performing algorithm.

Model Evaluation and Deployment:

This module involves deploying the final model in a real-world healthcare setting and evaluating its performance on new patient data. Ongoing monitoring and evaluation of the model can help identify opportunities for improvement and fine-tuning of the architecture.

Disease Output:

This module represents the output of the disease prediction model, which is the predicted disease based on the patient's symptoms. The output can be presented in various formats, such as a probability score or a binary prediction (e.g., disease or no disease).

**Algorithm of Each Module:**

1. Data Collection Module:

Loading dataset which we downloaded from Kaggle.

2. Data Preprocessing Module:

##### Handling Missing data[¶](#Handling-Missing-data)

In [ ]:

df.isna().sum()

df.isnull().sum()

Out[ ]:

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

In [ ]:

df = df.apply(lambda x: x.str.strip() )

df.iloc[:,1:] = df.iloc[:,1:].apply(lambda x: x.str.lower() )

df.iloc[:,1:] = df.iloc[:,1:].apply(lambda x: x.str.replace(' ', ''))

df = df.fillna(0)

df

Out[ ]:

4920 rows × 18 columns

##### Encoding the symptoms with their severity weight[¶](#Encoding--the-symptoms-with-their-sever)

In [ ]:

# Create a dictionary mapping each symptom to its severity

severity\_map = dict(zip(df1['Symptom'], df1['weight']))

# replace the symptoms with their severity

encoded\_df = df.replace(severity\_map)

encoded\_df

Out[ ]:

4920 rows × 18 columns

#### Storing the diseases and encoded symptoms in seperate dataframes[¶](" \l "Storing-the-diseases-and-encoded-sympto)

In [ ]:

cols = df.columns

(encoded\_df[cols] == 0).all()

encoded\_df['Disease'].value\_counts()

encoded\_df['Disease'].unique()

data = encoded\_df.iloc[:,1:].values

labels = encoded\_df['Disease'].values

#### [¶](#Storing-the-diseases-and-encoded-sympto)

3. Model Training and Evaluation

#### Random Forest Classifier[¶](#Random-Forest-Classifier)

In [ ]:

rfc = RandomForestClassifier()

rfc.fit(x\_train, y\_train)

result = rfc.predict(x\_test)

#### Decision Tree Classifier[¶](#Decision-Tree-Classifier)

In [ ]:

dtc = DecisionTreeClassifier()

dtc.fit(x\_train, y\_train)

result = dtc.predict(x\_test)

print( 'Accuracy% =', accuracy\_score(y\_test, result)\*100)

Accuracy% = 98.91598915989161

#### Naives Bayes Classifier[¶](#Naives-Bayes-Classifier)

In [ ]:

nbc = GaussianNB()

nbc.fit(x\_train, y\_train)

result = nbc.predict(x\_test)

print( 'Accuracy% =', accuracy\_score(y\_test, result)\*100)

Accuracy% = 88.27913279132791

#### K-Nearest Neighbors Classifier[¶](#K-Nearest-Neighbors-Classifier)

In [ ]:

knc = KNeighborsClassifier()

knc.fit(x\_train, y\_train)

result = knc.predict(x\_test)

print( 'Accuracy% =', accuracy\_score(y\_test, result)\*100)

Accuracy% = 98.3739837398374

#### Support Vector Classifier[¶](#Support-Vector-Classifier)

In [ ]:

svc = SVC(probability=True)

svc.fit(x\_train, y\_train)

result = svc.predict(x\_test)

### [¶](#Making-Predictions)

4. Deployment Module:

● Web application frameworks: Flask or Django

**Dataset : disease-symptom-description-dataset**

1. dataset.csv: This dataset file contains information about different patients, including their symptoms and the corresponding disease. Each row of the dataset corresponds to a patient, and the columns contain information such as the patient ID, symptoms, and disease.

Example row:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Patient ID | Symptom 1 | Symptom 2 | Symptom 3 | ... | Disease |
| 1 | Fever | Cough | Fatigue | ... | Pneumonia |
| 2 | Headache | Nausea | Vomiting | ... | Migraine |
| 3 | Diarrhea | Fever | Fatigue | ... | Typhoid |

1. symptom-severity.csv: This dataset file contains information about the severity of different symptoms associated with various diseases. Each row of the dataset corresponds to a symptom, and the columns contain information such as the name of the symptom, the associated disease, and the severity of the symptom on a scale of 1 to 10.

Example row:

|  |  |
| --- | --- |
| Symptom Name | Severity |
| Cough | 6 |
| Fever | 8 |
| Headache | 9 |

1. symptom-description.csv: This dataset file contains descriptions of various symptoms associated with different diseases. Each row of the dataset corresponds to a symptom, and the columns contain information such as the name of the symptom and a description of the symptom.

Example row:

|  |  |
| --- | --- |
| Symptom Name | Description |
| Cough | A sudden and repetitive reflexive action of the lungs, often accompanied by a distinctive sound. |
| Fever | A rise in body temperature above the normal range |
|  | due to an infection or illness. |
| Headache | A pain in the head, scalp or neck. |

1. symptom-precaution.csv: This dataset file contains information about the precautions that can be taken for different symptoms associated with various diseases. Each row of the dataset corresponds to a symptom, and the columns contain information such as the name of the symptom and a precaution that can be taken.

Example row:

|  |  |
| --- | --- |
| Symptom Name | Precaution |
| Cough | Avoid exposure to irritants such as smoke or pollutants. Drink plenty of fluids. |
| Fever | Rest and drink plenty of fluids. |
| Headache | Apply a cold compress or use pain relief medication as recommended by a healthcare professional. |

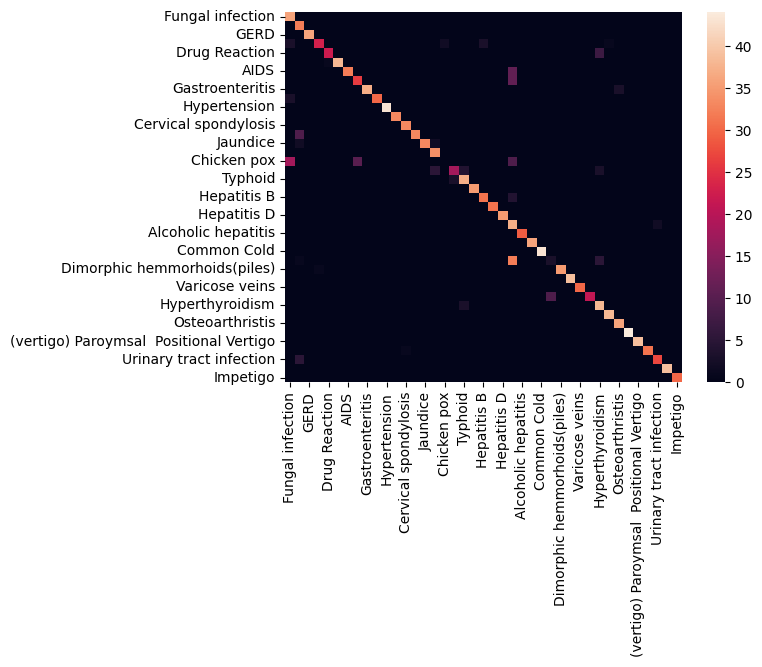
**Performance measures :**

1. Accuracy: Accuracy measures the percentage of correctly predicted diseases out of all the diseases. It is the most commonly used performance measure for classification problems.

print( 'Accuracy% =', accuracy\_score(y\_test, result)\*100)

Accuracy% = 88.27913279132791

1. Confusion matrix: A confusion matrix is a table that summarizes the predictions of the model in terms of true positive, true negative, false positive, and false negative cases. It is a useful measure when the nature of the errors needs to be analyzed.

****

**Result :**

Making Predictions[¶](#Making-Predictions)

In [ ]:

|  |
| --- |
| def predict\_disease(symptoms\_list,model,top\_k=5): |
| # Encode the symptoms based on their severity |
| encoded\_symptoms = [severity\_map[symptom] for symptom in symptoms\_list] |
|  |
| for i in range(len(encoded\_symptoms),17): |
| encoded\_symptoms.append(0) |
|  |
| # Create a numpy array from the encoded symptoms |
| symptoms\_array = np.array(encoded\_symptoms).reshape(1,-1) |
|  |
| # Use the trained model to predict the probabilities of all diseases |
| disease\_probs = model.predict\_proba(symptoms\_array)[0] |
|  |
| # Sort the predicted probabilities in descending order |
| sorted\_probs\_idx = np.argsort(disease\_probs)[::-1] |
|  |
| # Get the top-k predicted diseases and their probabilities |
| top\_k\_diseases = [(model.classes\_[idx], disease\_probs[idx]) for idx in sorted\_probs\_idx[:top\_k]] |
|  |
| return top\_k\_diseases |

In [ ]:

symptoms = [ 'itching', 'skin\_peeling']

predicted\_disease = predict\_disease(symptoms,rfc)

print('The predicted disease is {} with probability {:.2f}%\n'.format(predicted\_disease[0][0], predicted\_disease[0][1]\*100))

print(predicted\_disease)

The predicted disease is Fungal infection with probability 54.00%

**[('Fungal infection', 0.54), ('Paralysis (brain hemorrhage)', 0.18), ('Psoriasis', 0.09), ('Jaundice', 0.07), ('Chronic cholestasis', 0.06)]**

**References :**

[1]“Prediction of Diseases Using Different Machine Learning Approaches” Dr. Anish Gupta, Professor ,Department of Computer Science & Engineering Apex Institute of Technology ,Chandigarh University, published on 3rd International Conference on Intelligent Engineering and Management (ICIEM), 2022