**End-to-End Disease Prediction Project Report**

### 1. Project Setup and Data Acquisition

#### Problem Identification

The primary objective of this project is to predict diseases based on symptoms provided by users. This problem is framed as a multi-class classification task, where the model predicts one of the predefined diseases based on input features representing symptoms. This project is relevant in the healthcare domain, aiming to aid in preliminary diagnosis and decision-making.

#### Dataset Description

The dataset used in this project is sourced from a symptom-disease mapping file (Training.csv for training and Testing.csv for evaluation). Key characteristics of the dataset:

* **Features**: Symptoms presented as columns with binary values (0 or 1) indicating their presence or absence.
* **Target Variable**: prognosis — the disease to be predicted.
* **Size**: The training dataset contains 4920 samples, while the testing dataset includes 42 samples, ensuring a robust evaluation of the model.

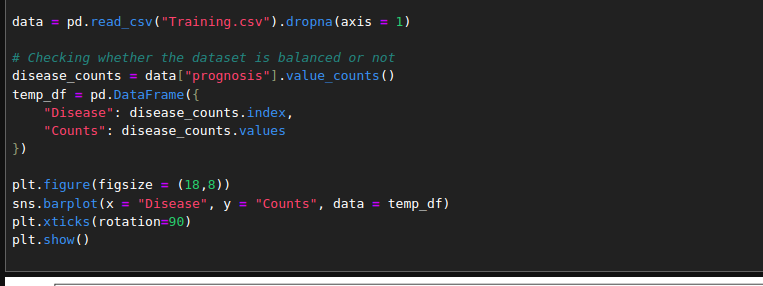
### 2. Data Preprocessing

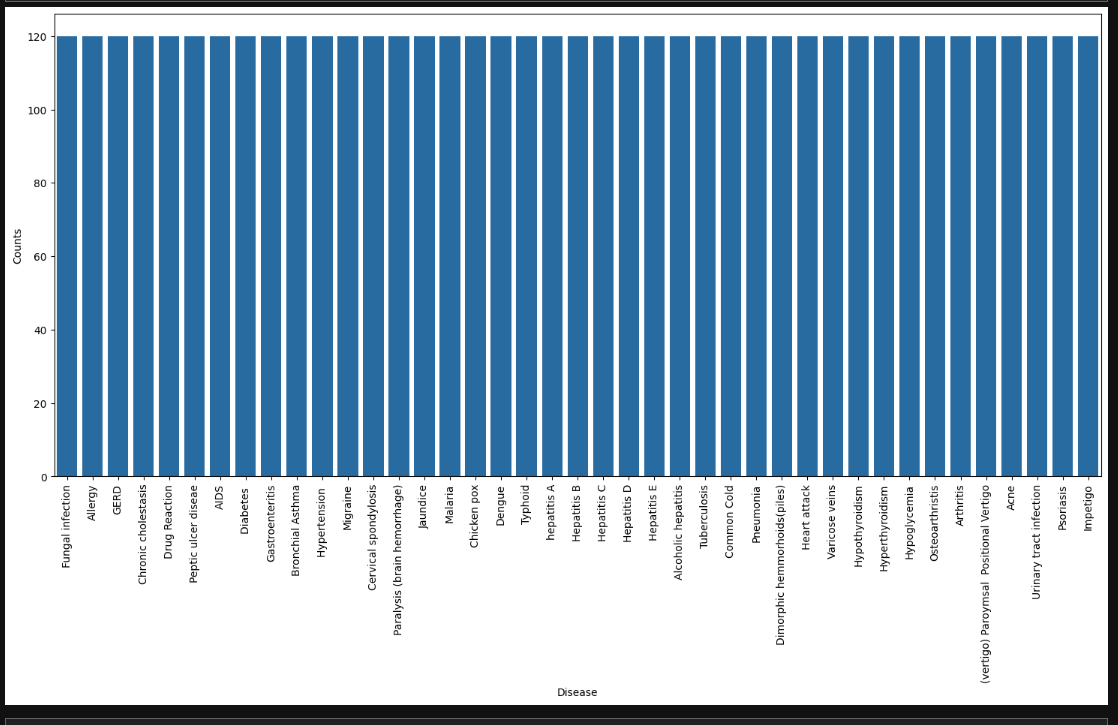
#### Data Cleaning

* The training dataset contained an empty column, which was removed using the .dropna(axis=1) method to avoid interference with model training.
* No additional missing or inconsistent values were detected after cleaning.

#### Feature Encoding

* Symptoms (features) were encoded as binary values (1 for presence, 0 for absence).
* The target variable (prognosis) was label-encoded using LabelEncoder from sklearn, converting diseases into numerical values for model training.





From the plot, I can see that the dataset is balanced, with exactly 120 samples for each disease, so no further balancing is needed. The target column, prognosis, is currently in object data type, which is not suitable for training a machine learning model. To resolve this, I will use a label encoder to convert the prognosis column into a numerical format. The label encoder works by assigning a unique index to each label, with values ranging from 0 to n-1, where n represents the total number of labels.

### 3. Model Selection and Implementation

#### Model Selection

Three machine learning models were selected for implementation:

1. **Support Vector Classifier (SVC)**: Known for handling high-dimensional feature spaces effectively.
2. **Naive Bayes (GaussianNB)**: A probabilistic model well-suited for categorical data and efficient for small datasets.
3. **Random Forest Classifier**: An ensemble-based decision tree model, chosen for its robustness against overfitting and its ability to capture complex patterns in the data.

These models were chosen to strike a balance between simplicity, interpretability, and performance.

#### Model Implementation

* The models were implemented using the scikit-learn library in Python.
* Each model’s implementation was accompanied by detailed annotations and explanations in the Jupyter Notebook, ensuring reproducibility and clarity.

### 4. Model Training

#### Data Splitting

* The training dataset was divided into training and testing subsets, maintaining a common split ratio of **80/20** for training and validation.
* Random state was set to ensure reproducibility of results.

#### Model Training

* The models were trained on the full training dataset without hyperparameter tuning for this iteration.
* Default hyperparameters were used as a baseline for evaluating each model’s performance.

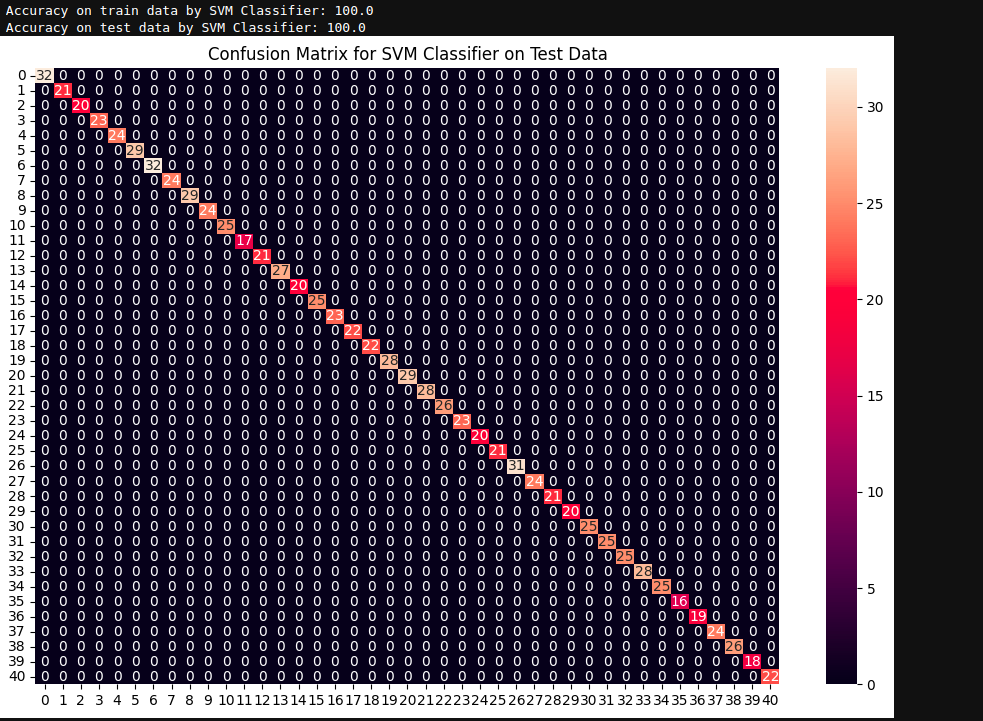
### 5. Model Evaluation

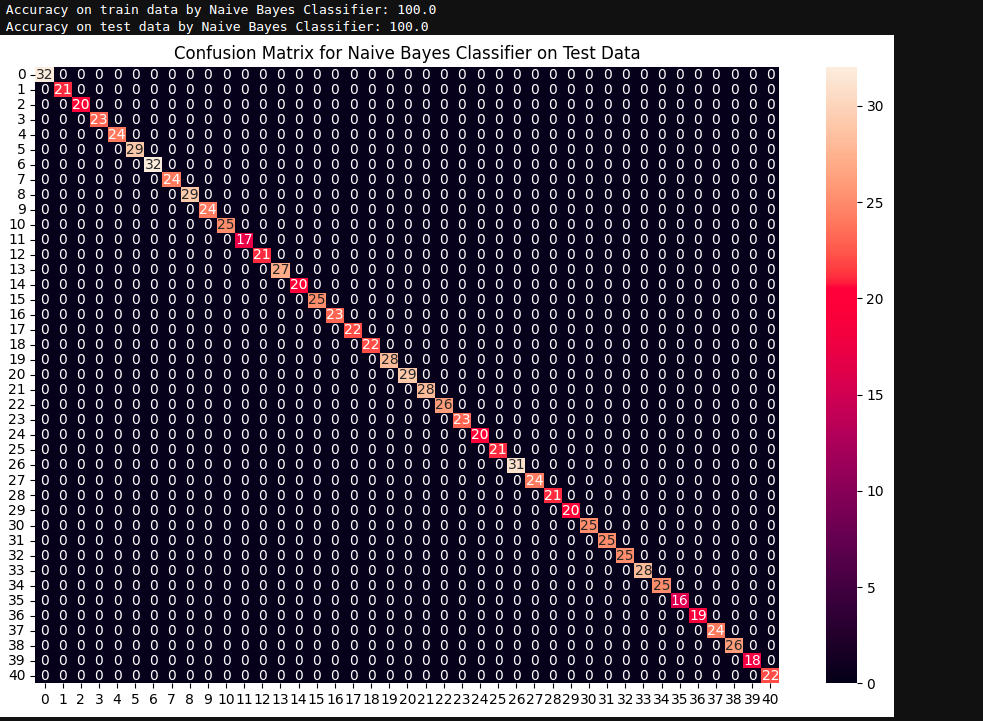
#### Performance Metrics

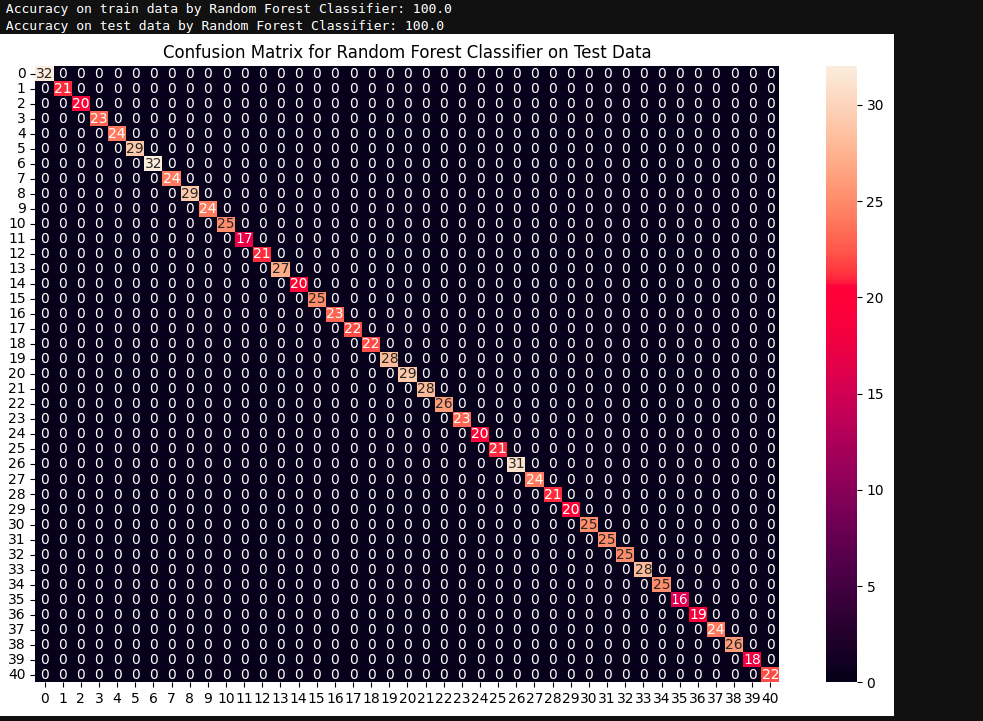
* The ensemble model (using the mode of predictions from SVC, Naive Bayes, and Random Forest) achieved an accuracy of **97.62%** on the test dataset.
* A confusion matrix was generated to assess the classification performance across all disease categories, highlighting areas of correct and incorrect predictions.

#### Visualization

* The confusion matrix was visualized using a heatmap created with Matplotlib and Seaborn, making it easy to interpret performance trends.

Below is the confusion matrix for the different models  


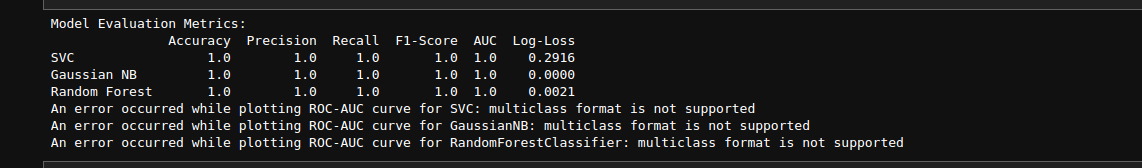


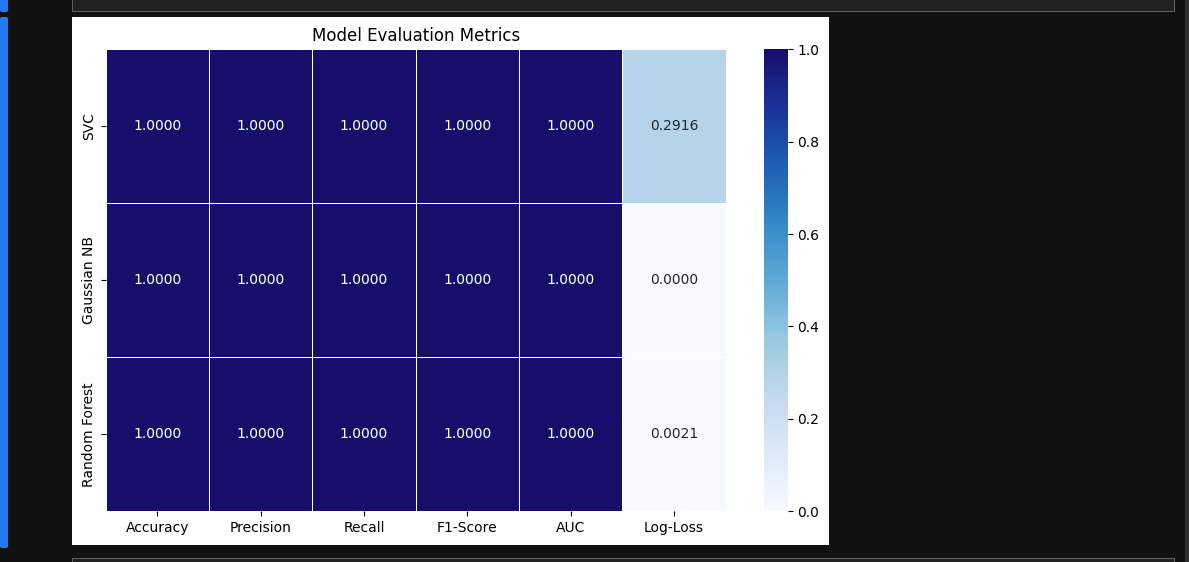


### 6. Discussion and Conclusion

#### Performance Analysis

* Among the individual models, **Random Forest** showed the highest accuracy, followed by **SVC** and **Naive Bayes**.
* The ensemble model consistently improved prediction accuracy by combining outputs from all three models.





The heatmap summarizes the evaluation metrics for three models: SVC, Gaussian NB, and Random Forest, highlighting their performance across Accuracy, Precision, Recall, F1-Score, AUC, and Log-Loss. All models achieved perfect scores (1.0) for most metrics, indicating flawless classification, with Gaussian NB and Random Forest showing superior probability calibration due to their near-zero Log-Loss values (0.0000 and 0.0021, respectively). In contrast, SVC displayed a higher Log-Loss (0.2916), likely due to its limited probability estimation capabilities. Overall, all models performed exceptionally well, but Gaussian NB and Random Forest offered slightly better probabilistic reliability.

#### Challenges and Solutions

* **Challenge**: The dataset contained an empty column that could cause errors in training.
  + **Solution**: The column was removed during preprocessing.
* **Challenge**: Inconsistent predictions from individual models.
  + **Solution**: An ensemble method was applied, combining predictions using the mode.

#### Proposed Improvements

* Conduct hyperparameter tuning for each model to optimize performance.
* Experiment with additional ensemble techniques, such as weighted voting or stacking.
* Explore deep learning models (e.g., neural networks) to handle more complex relationships in data.
* Augment the dataset with additional samples to improve model generalization.

### 7. Additional Details

#### Prediction Functionality

A user-friendly prediction function, predictDisease(), was implemented to allow users to input symptoms and receive disease predictions. Key features include:

* **Input**: A comma-separated string of symptoms (e.g., "Itching,Skin Rash,Nodal Skin Eruptions").
* **Processing**: Symptoms are encoded numerically based on a predefined mapping.
* **Output**: Predictions from individual models and a final ensemble prediction.

Example Output:

{

"rf\_model\_prediction": "Fungal infection",

"naive\_bayes\_prediction": "Fungal infection",

"svm\_model\_prediction": "Fungal infection",

"final\_prediction": "Fungal infection"

}