Disease Prediction Using Machine Learning

Introduction

In this project, our goal was to develop a machine learning model capable of predicting one of 42 diseases based on 132 symptoms. The dataset provided for this task consists of two CSV files: Training.csv and Testing.csv. This report documents the detailed steps taken to prepare the data, train the model, and evaluate its performance.

Data Exploration and Cleaning

Loading the Data:

The first step involved loading the datasets Training.csv and Testing.csv from the Data folder. Using Python's pandas library, we read these files into DataFrames to inspect their structure and contents. This initial inspection was crucial for understanding the format and preparing the data for further processing.

Initial Inspection:

Upon loading, the Training.csv dataset contained 134 columns, with the last column being an unnamed one. This dataset had 133 columns dedicated to symptoms and one additional column, prognosis, which indicates the disease. The Testing.csv dataset initially included 133 columns, including the prognosis column, which is not expected in the test data.

Data Cleaning:

To prepare the data for model training and testing:

1. **Removing Unnecessary Columns:** The unnamed column in Training.csv was removed to ensure only relevant features and the target variable remained.
2. **Adjusting Testing Data:** The prognosis column was removed from Testing.csv, as it should only be present in the training data for prediction purposes.

Handling Missing Values:

While not explicitly covered in this report, it is crucial to address any missing values in the datasets. This involves checking for missing data and deciding on appropriate methods for handling them, such as imputation (filling missing values with mean, median, or mode) or removal of incomplete rows or columns. Proper handling of missing values ensures the reliability and completeness of the dataset for training and testing.

Data Preparation

Separating Features and Target Variable:

For the Training.csv dataset, we separated the features from the target variable:

- **Features (X_train):** This included the 132 columns representing symptoms.
- **Target Variable (y_train):** This was the prognosis column, representing the disease each set of symptoms corresponds to.

Separating these components is a crucial step in supervised learning, where the model learns to associate input features (symptoms) with the target variable (disease).

Encoding Categorical Data:

The prognosis column is categorical, with text labels for different diseases. Machine learning algorithms typically require numerical input, so we used LabelEncoder from sklearn.preprocessing to convert these categorical labels into numerical format. This transformation maps each disease to a unique integer, making it possible to train the model effectively.

Feature Scaling:

Feature scaling was performed to ensure that all features are on a similar scale. This step is vital as it helps the model perform better and converge faster. We used StandardScaler to standardise the features:

- **Training Features (X_train_scaled):** The training features were scaled to ensure consistency in model training.

- **Testing Features (X_test_scaled):** The same scaler was applied to the testing features to maintain uniformity.

Model Selection and Training

Logistic Regression Model:
To establish a baseline, we chose a Logistic Regression model, a simple yet effective classification algorithm. Logistic Regression models the probability of a categorical outcome and can be extended to handle multiple classes, such as our 42 diseases. We trained the model using the scaled training data (X_train_scaled and y_train_encoded).

Model Evaluation:
We evaluated the model’s performance using several metrics:

- **Accuracy:** The ratio of correctly predicted instances to the total number of predictions, providing a measure of the overall performance.
- **Confusion Matrix:** A detailed table showing the true positives, true negatives, false positives, and false negatives, offering insight into the model’s performance across various disease classes.
- **Classification Report:** This includes precision, recall, and F1-score for each disease, providing a comprehensive assessment of the model’s performance, especially in handling different disease classes.

Making Predictions:
After training the model, we used it to make predictions on the Testing.csv data. The predictions, initially in numerical format, were converted back to the original disease labels using the inverse transformation of the label encoder.

Submission Preparation
Creating the Submission File:
For submission, we created a DataFrame containing:

- **Id:** An index representing the test sample number.
- **Predicted:** The predicted disease labels for each test sample.

This DataFrame was then saved as submission.csv in the Data folder, following the project’s file management guidelines. This CSV file includes the predictions in a format suitable for review or further analysis.

Conclusion
This project demonstrated a methodical approach to applying machine learning techniques for disease prediction. Through careful data preparation, model training, and evaluation, we developed a predictive model capable of classifying diseases based on symptoms. The use of Logistic Regression provided a solid baseline, and future work could explore more sophisticated models, hyperparameter tuning, and additional features to enhance the model's accuracy and robustness.