Name: Areshay Asad
Roll no: 052
Section: BS-AI 3A

Stress Level Prediction Using Machine Learning

1. Introduction

Stress is a significant factor influencing mental and physical health in today's fast-paced world. The ability to predict stress levels can aid in early intervention, allowing individuals to manage their well-being more effectively. This project aims to create a machine learning-based system to predict the stress levels of individuals based on health and lifestyle data. The system uses various features such as sleep, physical activity, heart rate, and blood pressure to predict whether a person is likely to experience stress and to categorize the stress level.

2. Objective

The main objectives of this project are:

1. **Stress Level Prediction:** To predict an individual's stress level using health and lifestyle data through machine learning.
2. **Interactive System:** To create an interactive tool where users can input personal information, and the system predicts their likelihood of experiencing stress.
3. **Visual Insights:** To provide visual insights into the distribution of data and its correlation with stress levels.

3. Dataset Overview

The project uses a Sleep Health and Lifestyle Dataset.

Key Features:

- **Health Metrics:**
  - Blood Pressure (split into Systolic BP and Diastolic BP)
  - Heart Rate
  - BMI Category
- **Sleep-Related Data:**
  - Sleep Duration
  - Quality of Sleep
  - Sleep Disorder
- **Lifestyle Attributes:**
  - Age
  - Daily Steps
  - Physical Activity Level

- **Target Variable:**
    - Stress Level: Represents the severity of stress on a numerical scale (e.g., 3 to 8).

Dataset Challenges:

- Missing values in both numeric and categorical features.
- Mixed data types (numerical, categorical, and text-based).
- Class imbalance in the Stress Level target variable.

Data Information:

- The dataset contains 374 entries and 13 columns, including numeric and categorical data.
- Missing values are handled through median imputation for numeric columns and "Unknown" for categorical columns.
- Some columns, such as the **Blood Pressure** column, are processed to split into **Systolic BP** and **Diastolic BP**.

4. Project Workflow

The project follows several steps for preprocessing, modeling, and evaluation:

4.1 Data Preprocessing

- **Data Loading:** The dataset is loaded into a pandas DataFrame, and basic information (e.g., number of entries, columns, data types) is displayed.
- **Missing Values Handling:** Missing values are handled by replacing them with appropriate values. For example, numeric columns are filled with the median, while categorical columns are replaced with "Unknown."
- **Feature Engineering:** The **Blood Pressure** column is split into two new columns: **Systolic BP** and **Diastolic BP**. Additionally, categorical columns such as **BMI Category** and **Sleep Disorder** are encoded using **LabelEncoder** to convert them into numeric values.

4.2 Data Exploration

- **Visualizing Distributions:** Various histograms with Kernel Density Estimation (KDE) are plotted for numeric features like **Age**, **Sleep Duration**, **Physical Activity Level**, etc., to understand their distributions.
- **Categorical Feature Analysis:** Count plots are created for categorical features such as **Stress Level** and **BMI Category** to visualize their frequency distributions.
- **Correlation Analysis:** A heatmap of the correlation matrix is generated to identify relationships between numeric features, such as **Sleep Duration** and **Heart Rate**, and their potential impact on stress levels.

4.3 Model Development

- **Model Selection:** A **RandomForestClassifier** is selected for its ability to handle complex datasets with multiple features and its resistance to overfitting. It is ideal for this project as it can handle both numeric and categorical data.
- **Pipeline Creation:** A pipeline is created that includes a **StandardScaler** for feature scaling and the **RandomForestClassifier** for prediction. The pipeline ensures that data preprocessing and modeling are done in one unified step.
- **Hyperparameter Tuning: GridSearchCV** is used to search for the best hyperparameters for the RandomForest model. This process helps optimize the model's performance by selecting the best number of estimators, maximum depth, and the minimum number of samples required for splitting.

4.4 Model Evaluation

- **Accuracy Evaluation:** The best model, determined by **GridSearchCV**, is evaluated on the test set. The accuracy score is reported along with a classification report that includes precision, recall, F1-score, and support for each stress level.
- **Confusion Matrix:** A confusion matrix is used to visualize the performance of the model and identify the number of correctly and incorrectly classified samples across different stress levels.

5. Results

The best parameters found for the model were:

- **max_depth:** 10
- **min_samples_split:** 5
- **n_estimators:** 100

The model achieved an accuracy of **100%** on the test set, demonstrating its ability to predict stress levels accurately based on the features.

Classification Report:

- **Precision, Recall, and F1-Score** for each stress level were all 1.00, indicating that the model accurately predicted all stress levels.

Confusion Matrix:

- The confusion matrix showed no misclassifications across the test set, further confirming the accuracy of the predictions.

6. Tools and Technologies

The following tools and technologies were used in the development of this project:

- **Programming Language:** Python
- **Libraries:**
  - **pandas**: For data manipulation and analysis.
  - **NumPy**: For numerical computations.
  - **scikit-learn**: For machine learning algorithms, model evaluation, and preprocessing (e.g., StandardScaler, RandomForestClassifier).
  - **imblearn**: For handling imbalanced datasets using techniques such as SMOTE.
  - **matplotlib**: For data visualization.
  - **seaborn**: For advanced plotting and visual analysis.
- **Tools:**
  - **Jupyter Notebook**: For writing and running Python code interactively.
  - **GridSearchCV**: For hyperparameter tuning.

7. Challenges and Solutions

1. **Handling Missing Values**:
   - Solution: Used median for numeric columns and "Unknown" for categorical columns.
2. **Categorical Encoding**:
   - Solution: Used LabelEncoder for categorical variables.
3. **Class Imbalance**:
   - Solution: Applied class_weight='balanced' in the Random Forest Classifier.

8. Conclusion

The Stress Level Prediction System was successfully developed using machine learning techniques. It accurately predicts the likelihood of stress and determines the stress level of an individual based on their health and lifestyle factors. With further improvements, such as real-time data integration, this system can serve as a useful tool for stress management and intervention.