Mid-term project

Deadline: 23h59 - 05/12/2024

Note: suggest students try to understand the context of datasets, features meaning, or any relevant information before diving into the implementation.

Project 1: Mental Attention States Classification Using EEG Data

Objective:

Classify mental attention states (focused, unfocused, drowsy) based on EEG signals using machine learning techniques.

Dataset Details:

- Dataset URL: https://www.kaggle.com/datasets/inancigdem/eeg-data-for-mental-attention-state-detection/data
- Data was acquired from EMOTIV EEG devices during 34 experiments.
- EEG data is in channels 4 to 17 of the provided Matlab files.
- Sampling frequency: 128 Hz.

Requirements:

1. **Data Preprocessing**:
    - Extract and load the relevant EEG data channels (4–17) from the provided Matlab files.
    - Apply preprocess, normalize, scale, or any techniques to the data if you think it necessary for the task.
2. **Feature Engineering**:
    - Extract meaningful features from the EEG signals (e.g., frequency domain features like power spectral density, and statistical features).
    - Compare features across attention states to identify patterns.
3. **Model Development**:
    - Implement at least two classification models (e.g., Logistic Regression, SVM, Random Forest, or Neural Networks).
    - Evaluate their performance using accuracy, precision, recall, and F1 score.
    - Split data into training and test sets, ensuring balanced representation of attention states.
4. **Analysis and Visualization**:
    - Visualize EEG signals and derived features to explain classification differences.
    - Create confusion matrices and ROC curves for model evaluation.

5. **Report**:
    - Document data processing, feature engineering, model theory, model building, and performance evaluation.
    - Discuss challenges faced and potential ways to improve accuracy.

Deliverables:

- Source code (organized and well-commented).
- A 5-minute presentation summarizing findings.
- A detailed report (5–7 pages).

Project 2: Loan Application Approval Prediction

Objective:

Develop a machine learning model to predict loan application acceptance or rejection based on customer details.

Dataset Details:

- Dataset URL: https://www.kaggle.com/datasets/abhishek14398/loan-dataset/data
- Contains customer details for loan decisions (acceptance/rejection).
- Includes attributes such as income, credit history, employment type, and loan amount.

Requirements:

1. **Data Understanding and Cleaning**:
    - Explore the dataset for missing values, inconsistencies, and outliers.
    - Clean and preprocess the data (e.g., handle missing values, encode categorical variables, etc).
2. **Exploratory Data Analysis (EDA)**:
    - Visualize correlations between features and loan acceptance/rejection.
    - Analyze class distribution (balance of accepted vs. rejected loans).
    - Features analysis.
    - Try to present all the necessary information that shows insights from the data or affects your model-building strategy.
3. **Model Development**:
    - Train at least three different classification models (e.g., Decision Trees, Logistic Regression, Gradient Boosting, or Neural Networks).
    - Perform hyperparameter tuning, and feature selection to optimize models.
    - Evaluate models using metrics like accuracy, precision, recall, F1 score, and AUC-ROC.
4. **Explainability**:

    - Use techniques like SHAP values or feature importance to explain model predictions.
    - Identify which metrics are most important to the problem and explain them.
5. **Deployment Simulation (Optional)**:
    - Create a script or simple interface to predict loan decisions for new input data.
6. **Report**:
    - Document data preprocessing, EDA, model development, and interpretability.
    - Compare model performance and justify the final model choice.

Deliverables:

- Source code (organized and well-commented).a
- A 5-minute presentation summarizing findings and a demo of the prediction interface.
- A detailed report (5–7 pages).

Encouragement for Experimentation:

- Encourage students to experiment with different techniques on model evaluation, and hyperparameter tuning or beyond.
- Encourage collaboration and discussion (BUT NOT **Plagiarism**) among students to share insights and learn from each other's approaches.
- Encourage students to leverage platforms like Google Colab/ Kaggle Notebook to experiment with the implementation. Google Colab provides free access to resources (CPU/GPU), facilitating faster experimentation.
- Encourage students to seek help from teaching assistants during lab sessions if they encounter difficulties.

Plagiarism Warning:

- Students are strictly prohibited from copying or reproducing the solution code from their peers. Each submission must be the student's individual work. **Any instances of plagiarism or copying will result in a grade of 0 points for the assignment.**

Submission Guidelines:

- Zip file as .zip format (NOT .rar file) contains:
    - Jupyter notebook (.ipynb) file
    - Report (.pdf) file
- Please send me your work before the due date.
- You can download the jupyter-notebook file (*.ipynb) from Google Colab by the following steps:
    - *File -> Download -> Download .ipynb*
- Name your zip file and notebook by the following pattern:
    - PRML2024_Lab<LabID>_Group<GroupID>.zip
        - Example: PRML2024_Lab01_Group01.zip
    - PRML2024_Lab<LabID>_Group<GroupID>.ipynb
        - Example: PRML2024_Midterm_Group01.ipynb

- The code results have to be printed out in the notebook or else, it won't be accepted.
- Include comments explaining key parts of the code if possible.
- Submit the notebook at: **Submission link will be available 2 days before the due day**

There is **NO** acceptance for **cheating** or **copying**.