MOTOR IMAGERY BASED PHOTO VIEWER

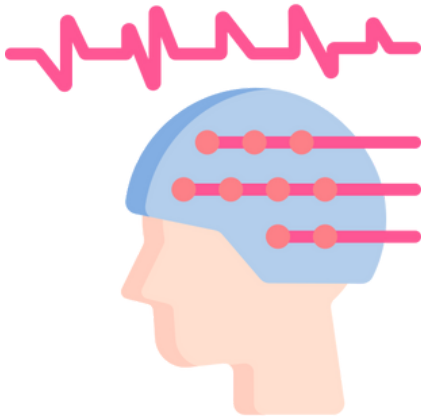
Project details and budget projections for website redesign

T/A:

Omar Sherief

Team ID:

SC_30



TEAM

Ali Mohamed Hassan Ahmed

20191700892

Fatma Ahmed Mohamed Eid Elnoby

20201700565

Mona Magdy Hassan Ahmed

20201701217

TABLE OF CONTENTS

DATA PREPARATION

- The data preparation process for EEG data from nine subjects aims to facilitate neuroscientific analysis, focusing on brain activity during tasks like motor imagery and eye movement. Key data characteristics include 25 electrodes, with 22 capturing EEG signals and 3 recording EOG signals for eye movement insights, sampled at 250Hz for high temporal resolution. The dataset covers recordings from nine subjects, enhancing statistical robustness.

- The data preparation involves loading raw EEG data, organizing it into dictionaries, and extracting relevant information for analysis. See the Python code snippet provided for detailed steps.

DATA PREPROCESSING

1. **Electrode Selection and Data Restructuring:**
    - Exclude non-essential Electrooculography (EOG) channels.
    - Transpose data to ensure consistent shape across subjects.
2. **Noise Reduction: Bandpass Filtering:**
    - mplement Butterworth bandpass filtering to isolate frequency bands of interest.
    - Filter EEG signals into multiple frequency bands (e.g., 4-8 Hz, 8-12 Hz, ..., 36-40 Hz).
3. **Epoch Extraction:**
    - Identify cue onset positions for left and right classes.
    - Extract epochs within specified time windows after cue onset for each class and frequency band.
4. **Data Splitting for Training and Testing:**
    - Define functions for splitting EEG data into training and testing sets for each class.
    - Split EEG data into training and testing sets for both left and right classes across all frequency bands and subjects.
5. **Implementation and Validation:**
    - Iterate through subjects and frequency bands to apply preprocessing steps uniformly.
    - Verify the integrity of the processed data through print statements or assertions.

FEATURE EXTRACTION

The Common Spatial Patterns (CSP) algorithm is a signal processing technique commonly used in the field of brain-computer interface (BCI). It's particularly useful for extracting discriminative features from EEG signals to differentiate between different mental states or tasks. Here's a detailed explanation of the CSP algorithm:

1.Objective:

- The primary goal of CSP is to find spatial filters that maximize the variance of EEG signals for one class while minimizing the variance for another class. This helps in enhancing the signal-to-noise ratio and extracting features that are most relevant for classification.

2.Covariance Calculation:

- CSP begins by computing the covariance matrices for each class of EEG signals (e.g., left hand movement vs. right hand movement).
- The covariance matrices capture the statistical relationships between different EEG channels or electrodes.

FEATURE EXTRACTION

3. Composite Covariance:

- The covariance matrices for each class are then averaged to create a composite covariance matrix.
- This composite covariance matrix represents the overall statistical properties of the EEG signals, combining information from both classes.

4. Whitening:

- Whitening matrices are computed from the composite covariance matrix to decorrelate the EEG signals.
- Whitening helps in transforming the data into a space where the covariance matrix becomes the identity matrix, simplifying subsequent computations.

5. Eigenvalue Decomposition:

- The whitened covariance matrix is decomposed into its eigenvectors and eigenvalues.
- Eigenvectors represent the spatial filters, while eigenvalues indicate the amount of variance captured by each filter.

FEATURE EXTRACTION

6. Spatial Filtering:

- Spatial filters are constructed using the eigenvectors obtained from the decomposition.
- These filters are designed to maximize the variance of EEG signals for one class while minimizing the variance for another class.

7. Feature Extraction:

- EEG signals are then projected onto the spatial filters to obtain spatially filtered signals.
- Features are extracted from the spatially filtered signals to capture discriminative information between classes.
- Common features include the logarithm of the variance of the filtered signals.

8. Application:

- The extracted features can be used as input to machine learning algorithms for classification tasks.
- CSP-derived features have been successfully applied in various BCI applications, such as motor imagery classification and mental state recognition.

PREPARING FOR CLASSIFICATION

1.Training Data:

- For each subject, the training data is prepared by extracting features from EEG signals recorded during different tasks or mental states.
- Features are extracted from all frequency bands except the last column, which contains labels indicating the corresponding class (e.g., left hand movement or right hand movement).
- Mutual information is employed to select the most informative features from the training data, ensuring that only relevant features are used for classification.
- The selected features and corresponding labels are stored in a new dictionary under the 'mutual' key, indicating that feature selection was performed based on mutual information.

PREPARING FOR CLASSIFICATION

2.Testing Data:

- Testing data preparation follows a similar process to training data.
- For each subject, features are extracted from EEG signals recorded during testing sessions for left and right hand movements.
- The previously computed CSP filters are applied to the testing data to obtain spatially filtered signals.
- Features are extracted from the spatially filtered signals, and mutual information is used to select the most informative features for classification.
- The selected features and corresponding labels are stored in a new dictionary under the 'mutual' key for testing data.

3.Randomization:

- Randomization is applied to both training and testing data to ensure that the order of samples does not introduce bias during model training or evaluation.
- The np.random.seed(42) statement fixes the random seed for reproducibility, ensuring consistent results across different runs.

CLASSIFICATION

Classifier	Parameters
Support Vector Machine (SVM)	Kernel: RBF, Gamma: Scale, Max Iterations: 3500
K-Nearest Neighbors (KNN)	Number of Neighbors: 50
Logistic Regression	Maximum Iterations: 3500, Regularization Parameter (C): 0.5
Decision Tree	Criterion: Gini, Max Depth: None

CLASSIFICATION

Subject	SVM Train Accuracy (%)	SVM Test Accuracy (%)	KNN Train Accuracy (%)	KNN Test Accuracy (%)	Logistic Regression Train Accuracy (%)	Logistic Regression Test Accuracy (%)	Decision Tree Train Accuracy (%)	Decision Tree Test Accuracy (%)
01	98.62	98.61	98.62	98.61	98.62	98.61	88.89	100.00
02	94.46	95.83	95.15	93.75	94.46	95.83	79.88	100.00
03	97.24	97.92	97.93	97.92	97.24	97.92	93.77	100.00
04	99.31	99.31	97.91	99.31	98.62	99.31	82.64	100.00
05	99.31	99.31	99.31	99.31	99.31	100.00	87.51	100.00
06	93.79	93.75	93.08	93.75	92.39	93.75	75.02	100.00
07	99.31	100.00	99.31	100.00	100.00	100.00	91.72	100.00
08	97.24	98.61	98.62	98.61	98.62	98.61	95.84	100.00
09	92.32	90.97	88.87	92.36	90.96	92.36	88.10	100.00

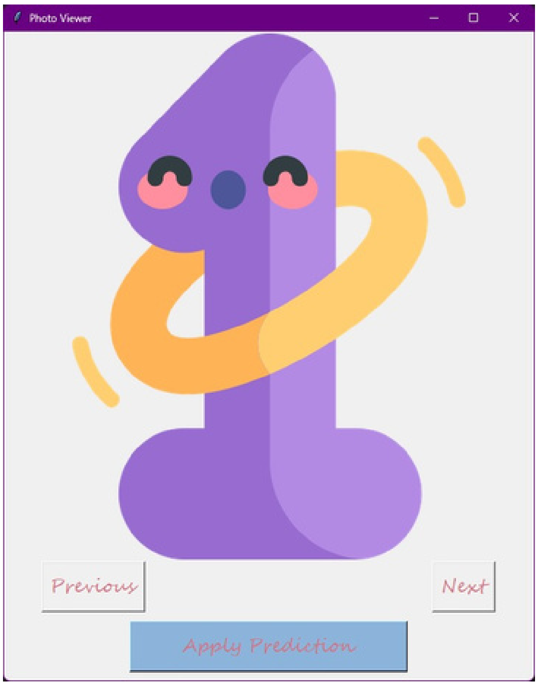
CLASSIFICATION

Comparisons:

- Across all classifiers, the training accuracies tend to be higher than the testing accuracies, indicating some degree of overfitting in the models.
- SVM and Logistic Regression models generally exhibit higher accuracies compared to KNN and Decision Tree models for most subjects.
- SVM, Logistic Regression, and KNN models achieve perfect or near-perfect testing accuracies for some subjects, while Decision Tree model shows varied performance across subjects.
- Subject 7 consistently shows high accuracy across all classifiers, while Subject 6 exhibits relatively lower accuracy compared to others.
- Decision Tree model achieves perfect testing accuracy for all subjects, which could indicate either overfitting or a perfect fit to the data.

USER INTERFACE

1. When you start the app viewer, a window pops up. To see what the app predicts, just click "Apply Prediction."



2. In the prediction scenario, the button's color alters when it matches the predicted outcome(next), and here's the result.

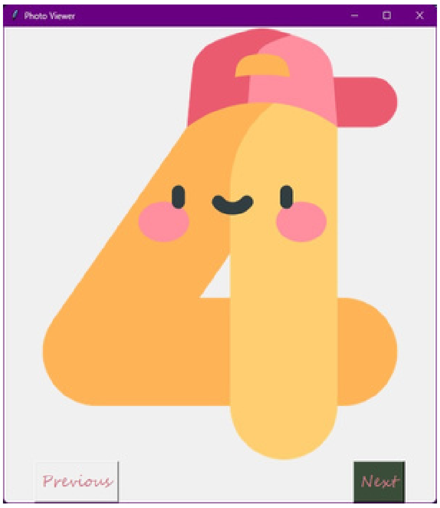
USER INTERFACE

3. And so on like the previous.

4. the button's color alters when it matches the predicted outcome(next), and here's the result.

USER INTERFACE

5. And so on like the previous.



THANK YOU!