Info File

Feature Extraction

➢ *check_timestamps(path)*

iterates over each csv in the given directory, and check the homogenity of "timestamps" column prints out every sampling period that was observed in each dataframe

attributes

path : the path of the data (csv) directory

return:

None

➢ *interpolate_dataset(path, dt)*

interpolate every channel of every dataframe at the given directory to create homogenous sampling the function stores the interpolated signals at a directory called "interpolated_signals" created at the same relative path of the script if the directory exist then the files will be overwritten

attributes

path : the path of the data (csv) directory dt : sampling period

return:

None

➢ *create_eeg_features_empty_dataset(path)*

creates an instance of the dataset header with the header of the dataframe

attributes

path : the path of the data (csv) directory

return

an empty pandas dataframe, wih only the headers

➢ *extract_features(dataset,path)*

creates an instance of the dataframe header with the header of the dataframe

attributes

dataset : a dataframe initialized by the function
create_eeg_features_empty_dataset(path) path : the path of the data (csv) directory

return:

the dataset with all the features and records

➢ *export_to_csv(dataset)*

save the dataframe as csv file with name "dataset.csv" for later uses

attributes

dataset : a dataframe initialized by the function
create_eeg_features_empty_dataset(path)

return:

None

➢ *power_spectral_density_features(signal, fs=1/0.004)*

calculate the features related to the power spectral density

attributes

signal : numpy array of the signal fs : sampling frequency

return:

entropy , sef (Spectral edge frequency), peak_freq (Spectral peak frequency)

➢ *calculate_plv(signal1, signal2, fs)*

calculate the Phase synchronization which is a measure of the synchronization of the phase angles of two EEG signals at different electrode sites

attributes

signal1 : numpy array of the first signal signal2 : numpy array of the second signal fs : sampling frequency

return:

array of size 5, the synchronization of each brain wave frequency band {α, β, θ, δ, γ}

Feature Selection and Model Performance

➢ encodes categorical:

transform_categorical(data): function transforms categorical variables in a pandas DataFrame to numeric values using scikit-learn's LabelEncoder class. We use it to transfer the result column from (relaxed, neutral, concentrating) to (0, 1, 2)

➢ crossValidation(model,x,y,model_name)

Description: function performs k-fold cross-validation on a given machine learning model and returns the mean accuracy of the model across the cross-validation folds and prints the model metrics to help in selecting and evaluation model.

Input Parameters:

model: The machine learning model to be evaluated

x: The feature matrix containing the input data

y: The target vector containing the labels for the input data

model_name: A string representing the name of the model being evaluated

Output:

Prints the different metric score including: accuracy, precision, recall, f1

Print the fit time of the model

The function returns the mean accuracy of the model across the cross-validation 10 folds

➢ get_all_models_cross_validation_scores(models,x,y)

Description: function performs k-fold cross-validation on a list of machine learning models and returns a dictionary containing the mean accuracy of each model across the cross-validation folds.

Input Parameters:

models: A list of tuples, where each tuple contains a string representing the name of the model and the machine learning model object

x: The feature matrix containing the input data

y: The target vector containing the labels for the input data

Output:

The function returns a dictionary containing the mean accuracy of each model across the cross-validation folds

➢ remove_constant_features(X)

Description: function removes constant features from a pandas dataframe containing the input features.

Input Parameters:

X: A pandas dataframe containing the input features

Output:

The function returns a modified version of the input dataframe with constant features removed

➢ remove_Quasi_Constant_Features(X)

Description: This function removes constant features from a pandas dataframe containing the input features.

Input Parameters:

X: A pandas dataframe containing the input features

Output:

The function returns a modified version of the input dataframe with constant features removed

➢ remove_duplicated_features(X)

Description: This function removes duplicated features from a pandas dataframe containing the input features.

Input Parameters:

X: A pandas dataframe containing the input features

Output:

The function returns a modified version of the input dataframe with duplicated features removed

➢ remove_correlated_features(X,threshold):

Description: This function removes the correlated features from a pandas dataframe containing the input features.

Input Parameters:

X: A pandas dataframe containing the input features

Output:

The function returns a modified version of the input dataframe with correlated features removed

➢ select_K_mutual_info_classif(K,X,y)

Description: This function selects the top K features based on mutual information gain from a pandas dataframe containing the input features and the target variable.

Input Parameters:

K: An integer value that specifies the number of features to select

X: A pandas dataframe containing the input features

y: A pandas dataframe containing the target variable

Output:

The function returns a pandas dataframe containing the top K features.

➢ select_top_roc_values(X,y ,K)

Description: This function selects the top K features based on ROC AUC score from a pandas dataframe containing the input features and the target variable.

Input Parameters:

K: An integer value that specifies the number of features to select

X: A pandas dataframe containing the input features

y: A pandas dataframe containing the target variable

Output:

The function returns a pandas dataframe containing the top K features.

➢ apply_combined_filters(X,y)

Description: This function applies a combination of feature selection filters to a pandas dataframe containing the input features and the target variable.

Input Parameters:

X: A pandas dataframe containing the input features

y: A pandas dataframe containing the target variable

Output:

The function returns a pandas dataframe containing the features that have passed all of the filters.

➢ confusion matrix:

plotCM (model,model_name) : function that generates a heatmap of the confusion matrix of a machine learning model given the model and it's name.

➢ evaluation metrics:

evaluation (model,model_name): function that computes various evaluation metrics as (precision, f1, sensitivity, specificity, negative_predicitive_value, accuracy) for a machine learning model and prints them where it given the model and it's name from predefined

➢ ROC:

The plot_ROC(model, model_name, X_train, y_train, X_test, y_test) :function that generates a plot of the receiver operating characteristic (ROC) curves and area under the curve (AUC) scores for a multiclass classification model.

To use the provided functions

if its only for exploration it would be recommended to run "feature extraction.ipynb" cell by cell

or to use the provided functions as follows

```
if __name__ == "main":
    check_timestamps(path)
    interpolate_dataset(path, dt)
    dataset = create_eeg_features_empty_dataset(path)
    dataset = extract_features(dataset,path)
    export_to_csv(dataset)
df=pd.read_csv('./dataset.csv')
X = df.drop("result", axis = 1)
```

```
y = df["result"]
X_new = apply_combined_filters(X,y)
get_all_models_cross_validation_scores(models, X_new ,y)
```
