AttentionPoints Namespace Reference

Functions

def	**load_traindata** ()
def	**load_testdata** ()
def	**correlation_matrix** ()
def	**pair_plot** ()
def	**data_normalize** ()
def	**split_data** ()
def	**build_model** ()
def	**train_model** ()
def	**model_evaluate** ()

Variables

list	**features** = ['object1position_x','object1position_y','object1position_z','object1scaling_x','object1scaling_y','object1scaling_z','object2position_x','object2position_y','object2position_
list	**output_label** = ['attentionposition_x','attentionposition_y','attentionposition_z']
	ax = plt.axes(projection='3d')
def	**zdata** = load_traindata()['object1position_z']
def	**xdata** = load_traindata()['object1position_x']
def	**ydata** = load_traindata()['object1position_y']
def	**z** = load_traindata()['object2position_z']
def	**x** = load_traindata()['object2position_x']
def	**y** = load_traindata()['object2position_y']
	cmap

Detailed Description

```
The following libraries are imported to build the model.

1.  sklearn

2.  keras.utils.normalize

3.  keras.models.sequential

4.  keras.layers.Dense

5.  keras.layers.Dropout

6.  keras.optimizers.Adam

7.  numpy

8.  matplotlib.pyplot

9.  sklearn.preprocessing

10. sklearn.model_selection.train_test_split

11. sklearn.preprocessing.MinMaxScaler

12. sklearn.metrics.mean_squared_error

13. pickle

14. seaborn
```

Function Documentation

◆ build_model()

def AttentionPoints.build_model ( )

```
Multilayer perceptron model is built using Adam optimizer and mean squared error as the loss function.
Returns the built model.
```

◆ correlation_matrix()

def AttentionPoints.correlation_matrix ( )

```
Heatmap and corr functions are used to plot the correlation matrix.
```

◆ data_normalize()

def AttentionPoints.data_normalize ( )

```
Normalizes the input features and label using MinMaxScaler.
Returns the normalized data.
```

◆ load_testdata()

def AttentionPoints.load_testdata ( )

```
Loads the test data from train_data.csv file.
```

◆ load_traindata()

def AttentionPoints.load_traindata ( )

```
Loads the train data from train_data.csv file.
```

◆ model_evaluate()

def AttentionPoints.model_evaluate ( )

```
The built model is de-serialized using pickle and evaluated.
```

◆ pair_plot()

def AttentionPoints.pair_plot ( )

```
Pair plot is used to plot the pairwise relationships in a dataset.
```

◆ split_data()

def AttentionPoints.split_data ( )

```
The input data is split into train and validation data.
Returns the split data.
```

◆ train_model()

def AttentionPoints.train_model ( )

```
The built model is compiled and serialized using pickle.
Returns the compiled model.
```