A picture containing text, wall, person, indoor

Description automatically generated

**ShaoXing 绍兴**

**Package for Post-tuning and Model Evaluation**

*20/02/2023*

**Background**

ShaoXing, the native city of Chinese Revolutionary and Leader Zhou Enlai (周恩来), is what this Post-tuning and Model Evaluation Package is named after. The purpose of this package is to streamline the model analysis project and provide a wholesome framework for post-model tuning actions.

The naming was particularly chosen as Model Evaluation is often the uninteresting but necessary part of the project, and Zhou served faithfully as the right hand man of Paramount Leader Mao Zedong after the founding of the PRC, carrying out the day-to-day running of the nation to free up Mao to transform China according to his vision.

Functionalities of ShaoXing include building models based on the best hyperparameters, making future predictions, calculating more detailed statistics than the tuning outputs of i.e. JiXi and YangZhou, as well as cross validation statistics. It is also capable of producing residual plots for regressions and confusion matrices for classifications.

It can also help visualise feature importance for a range of different classes of models

**Class**

|  |  |
| --- | --- |
| Class | Purpose |
| ShaoXing | Object that stores full, train, val, test and potentially future data, as well as feature column names, label name, model class, best hyperparameters to perform analysis and prediction based on the best model  Able to get:  -extensive model train val test evaluation statistics  -cross validated evaluation statistics  -residual plots/evaluation matrices  -feature importance data and plot |

**Class: ZhongShan**

**Methods:**

|  |  |
| --- | --- |
| Methods | Purpose |
| *ShaoXing()* |  |
| read\_in\_features\_label(self, features, label) | Reads in values feature column name list and label  Parameters:  features – list  label - str |
| read\_in\_full\_train\_test\_data(full\_data, train\_data, val\_data, test\_data) | Reads in Full, Training, Validated and Test data along with features list, auto transforming to X, y  Parameters:  full\_data – DataFrame  train\_data – DataFrame  val\_data – DataFrame  test\_data - DataFrame |
| read\_in\_future\_data(future\_data) | Read in Future data (X only – for predictions)  Parameters:  future\_data - DataFrame |
| read\_in\_untrained\_model(model\_class, best\_params, type, model\_class\_type, model\_name) | Reads in underlying model object for tuning, as well as what type of model it is and its name (arbitrary)  Parameters:  model\_class – class object  Best\_params – dictionary of str:int (or str)  type – str – either ‘Classification’ or ‘Regression’  model\_class\_type - str– either ‘Sklearn Tree’ or ‘LGB’ or ‘OLS LR’ or ‘Sklearn linear kernel’  model\_name - str |
| read\_in\_fitted\_model(fitted\_model, type, model\_class\_type, model\_name) | Reads in a fitted model  Parameters:  fitted\_model – model object  type - str – either ‘Classification’ or ‘Regression’  model\_class\_type - str– either ‘Sklearn Tree’ or ‘LGB’ or ‘OLS LR’ or ‘Sklearn linear kernel’  model\_ name - str |
| fit\_model() | Fit the model based on train x and trian y for model class with previously inputted hyperparameters |
| export\_model(model\_export\_address) | Export the model as a pickle  Parameters:  model\_export\_address – str – does not need to include ‘.pickle’ |
| predict\_using\_future\_data(self, return\_pred = False) | Make predictions for future x data, with option for function to output result (instead of just storing internal dataframe)  Parameters:  return\_pred – bool – optional, default False |
| export\_future\_data\_and\_predictions(future\_data\_and\_pred\_saving\_address) | Export future predictions (joined up to future x)  Parameters:  Future\_data\_and\_pred\_saving\_address – str – does not need to include ‘.csv’ |
| view\_future\_data\_and\_predictions(return\_df = False) | View the Dataframe of future X and future labels, with option for function to output result (instead of just just displaying)  Parameters:  return\_df – bool – optional, default False |
| get\_analysis(address = None) | Get all model statistics, with option to save the first two dataframes.  For clf\_type == ‘Regression’  -r2, RMSE, 4 quantile r2, 4 quantile RMSE, 10 quantile r2, 10 quantile rmse (Train and Val and Test)  -Cross validated aforementioned stats (Train and Test)  -how well does X predict residuals? (r2, corr and pvals)  -how well does each xi predict residuals (r2, corr and pvals)  -Residual plots  For clf\_type == ‘Classification’  -accuracy, balanced accuracy, precision, recall, f1 (last three weighted, macro, micro) (Train and Val and Test)  -Cross validated aforementieond stats (Train and Test)  -Confusion Matrix  Saving: will save first two dataframe as ‘*address*\_regular.csv’ and ‘*address*\_CV.csv’  Parameters:  address – str – optional, does not need to include ‘.csv’ |
| export\_analysis\_dfs(address) | Export the analysis data  Parameters:  address – str – does not need to include ‘.csv’ |
| export\_residual\_plot(feature, data\_type, address) | Saves the specified residual plot as a png  Parameters:  feature – str  data\_type – str – either ‘Train’ or ‘Val’ or ‘Test’  address – str – does not need to include ‘.png’ |
| display\_model\_analysis\_stats() | Displays the model analysis statistics and relevant plots (if exists) |
| get\_feature\_importance(address = None) | Calculates and displays feature importance bar graphs, with option to save the bar graph.  Parameters:  address- str – optional, does not need to include ‘.png’ |

**Objects:**

|  |  |
| --- | --- |
| Objects | Purpose |
| full\_data | DataFrame |
| train\_data | DataFrame |
| val\_data | DataFrame |
| test\_data | DataFrame |
| features | list |
| label | str |
| train\_x | DataFrame |
| train\_y | DataFrame |
| val\_x | DataFrame |
| val\_y | DataFrame |
| test\_x | DataFrame |
| test\_y | DataFrame |
| future\_data | DataFrame |
| train\_analyse\_df | DataFrame |
| val\_analyse\_df | DataFrame |
| test\_analyse\_df | DataFrame |
| model | Model object |
| future\_pred | np.array |
| future\_data\_and\_pred | DataFrame |
| regular\_stats\_df | DataFrame |
| CV\_stats\_df | DataFramae |
| RESIDUAL\_PLOT\_OBJECTS | Dict – str:dict of str:matplotlib.figures |
| feature\_importance\_df | DataFrame |

**Dependencies**

pandas

numpy

matplotlib