

---

# **ML Evalution Ploting Library**

***Release 0.0.0***

**Spencer Barton**

**Sep 19, 2018**



---

## Contents:

---

<b>1</b>	<b>Installation</b>	<b>3</b>
1.1	API Reference . . . . .	3
	<b>Python Module Index</b>	<b>7</b>



mlplot is a Python data visualization library for machine learning evaluation plots. It is based on [matplotlib](#) and [sklearn](#).

The code live in [Github](#).



```
pip install mlplot
```

## 1.1 API Reference

### 1.1.1 Classification Plots

Plots to evaluate classification models.

Module containing all classification model evaluation plots

`mlplot.classification.calibration` (*y\_true*, *y\_pred*, *ax=None*, *n\_bins='auto'*)

Plot a calibration plot

Calibration plots are used to determine how well the predicted values match the true value.

This plot is as found in [sklearn](#).

#### Parameters

- **y\_true** (*np.array of str or int*) – A vector of size N that contains the true labels. There should be two labels of type string or numeric.
- **y\_pred** (*np.array of float*) – A vector of size N that contains predictions as floats from 0 to 1.
- **class\_labels** (*dict, optional*) – A dictionary mapping from labels in *y\_true* to class names.  
Ex: {0: 'not dog', 1: 'is dog'}
- **ax** (*matplotlib.axes.Axes, optional*)
- **n\_bins** (*int or string*) – The number of bins to group *y\_pred*. See [numpy.histogram](#)

`mlplot.classification.confusion_matrix` (*y\_true*, *y\_pred*, *class\_labels=None*, *threshold=0.5*,  
*ax=None*)

Plot a heatmap for the confusion matrix

An example of this heatmap can be found on [sklean](#).

#### Parameters

- **y\_true** (*np.array of str or int*) – A vector of size N that contains the true labels. There should be two labels of type string or numeric.
- **y\_pred** (*np.array of float*) – A vector of size N that contains predictions as floats from 0 to 1.
- **class\_labels** (*dict, optional*) – A dictionary mapping from lables in y\_true to class names. Ex: {0: 'not dog', 1: 'is dog'}
- **ax** (*matplotlib.axes.Axes, optional*)
- **threshold** (*float*) – Defines the cutoff to be considered in the asserted class

`mlplot.classification.population_histogram(y_true, y_pred, class_labels=None, ax=None)`  
Plot histograms of the predictions grouped by class

#### Parameters

- **y\_true** (*np.array of str or int*) – A vector of size N that contains the true labels. There should be two labels of type string or numeric.
- **y\_pred** (*np.array of float*) – A vector of size N that contains predictions as floats from 0 to 1.
- **class\_labels** (*dict, optional*) – A dictionary mapping from lables in y\_true to class names. Ex: {0: 'not dog', 1: 'is dog'}
- **ax** (*matplotlib.axes.Axes, optional*)

`mlplot.classification.precision_recall(y_true, y_pred, x_axis='recall', ax=None)`  
Plot the precision-recall curve

An example of this plot can be found on [sklean](#).

#### Parameters

- **y\_true** (*np.array of str or int*) – A vector of size N that contains the true labels. There should be two labels of type string or numeric.
- **y\_pred** (*np.array of float*) – A vector of size N that contains predictions as floats from 0 to 1.
- **class\_labels** (*dict, optional*) – A dictionary mapping from lables in y\_true to class names. Ex: {0: 'not dog', 1: 'is dog'}
- **ax** (*matplotlib.axes.Axes, optional*)
- **x\_axis** (*str 'recall' or 'threshold'*) – Specify the x axis of the plot. Precision recall tends to come in 2 flavors, one precision vs recall and the other precion and recall vs threshold.

`mlplot.classification.report_table(y_true, y_pred, class_labels=None, ax=None)`  
Generate a report table containing key stats about the dataset

#### Parameters

- **y\_true** (*np.array of str or int*) – A vector of size N that contains the true labels. There should be two labels of type string or numeric.
- **y\_pred** (*np.array of float*) – A vector of size N that contains predictions as floats from 0 to 1.



- **class\_labels** (*dict, optional*) – A dictionary mapping from lables in `y_true` to class names.  
Ex: `{0: 'not dog', 1: 'is dog'}`
- **ax** (*matplotlib.axes.Axes, optional*)

`mlplot.classification.roc_curve(y_true, y_pred, ax=None)`

Reciever operating curve

#### Parameters

- **y\_true** (*np.array of str or int*) – A vector of size N that contains the true labels. There should be two labels of type string or numeric.
- **y\_pred** (*np.array of float*) – A vector of size N that contains predictions as floats from 0 to 1.
- **class\_labels** (*dict, optional*) – A dictionary mapping from lables in `y_true` to class names.  
Ex: `{0: 'not dog', 1: 'is dog'}`
- **ax** (*matplotlib.axes.Axes, optional*)



**m**

`mlplot.classification`, [3](#)



### C

`calibration()` (in module `mlplot.classification`), [3](#)  
`confusion_matrix()` (in module `mlplot.classification`), [3](#)

### M

`mlplot.classification` (module), [3](#)

### P

`population_histogram()` (in module `mlplot.classification`),  
[4](#)  
`precision_recall()` (in module `mlplot.classification`), [4](#)

### R

`report_table()` (in module `mlplot.classification`), [4](#)  
`roc_curve()` (in module `mlplot.classification`), [5](#)