Package find projections

Comprehensively searches for informative projection boxes (2-d projection boxes) in the raw feature space which separate out homogeneous data points.

Returns all projection boxes which match search criteria (support, purity).

For discrete output, the algorithm tries to find 2-d projection boxes which can separate out any class of data from the rest with high purity.

For numeric output, the algorithm tries to find 2-d projection boxes which can separate out data points with low variance.

Classes:

SearchHyperParameters *in find_projections.search_projections*

binsize: No. of data points for binning each feature. Should be a positive integer. Default value is 10.

support: Minimum number of data points to be present in a projection box for evaluation. Should be a positive integer. Default value is 100

purity: Minimum purity (class proportion) in a projection box. Should be in the range 0.0 - 1.0. Default value is 0.9.

num_threads: No. of threads for multi-threaded operation. Should be a positive integer. Default value is 1.

validation_size: Proportion of training data which is held out for validation purposes. Should be in the range 0.0 - 0.5. Default value is 0.1. This is used for fit().

Search *in find_projections.search_projections*

Class to search for 2-d projection boxes in raw feature space for discrete (categorical) output (for classification problems).

For discrete output, the algorithm tries to find 2-d projection boxes which can separate out any class of data from the rest with high purity.

__init__()

Constructor.

Methods:

search_projections()

Comprehensively evaluates all possible pairs of 2-d projections in the data

Returns all projection boxes which match search criteria

Returns FeatureMap instance containing all the projection boxes found meeting the search criteria

find_easy_explain_data()

Learns decision list of projection boxes for easy-to-explain data (for classification). Returns FeatureMap instance containing all the projection boxes found meeting the search criteria.

Parameters

Returns FeatureMap instance containing all the projection boxes found meeting the search criteria

fit()

Calls find_easy_explain_data() and stores the resultant FeatureMap instance.

set_training_data(inputs, outputs)

Sets input and output feature space.

Parameters

inputs: 2-d numpy array of floats (dense, no missing values)

outputs: 1-d numpy array of floats (dense)

SearchNumericHyperParameters *in find_projections.search_numeric_projections*

binsize: No. of data points for binning each feature. Should be a positive integer. Default value is 10.

support: Minimum number of data points to be present in a projection box for evaluation. Should be a positive integer. Default value is 100

num_threads: No. of threads for multi-threaded operation. Should be a positive integer. Default value is 1.

validation_size: Proportion of training data which is held out for validation purposes. Should be in the range 0.0 - 0.5. Default value is 0.1. This is used for fit().

mode: Valid values are 0, 1, 2. 1 for high mean, 2 for low mean and 0 for low variance boxes.

SearchNumeric *in find_projections.search_numeric_projections*

Class to search for 2-d projection boxes in raw feature space for numeric output (for regression problems).

For numeric output, the algorithm tries to find 2-d projection boxes which can separate out any class of data from the rest with high purity. The box search would depend on the ‘mode’ value.

mode: Valid values are 0, 1, 2.

1 for high mean, 2 for low mean and 0 for low variance boxes.

__init__()

Constructor.

Methods:

search_projections()

Comprehensively evaluates all possible pairs of 2-d projections in the data

Returns all projection boxes which match search criteria

Returns FeatureMap instance containing all the projection boxes found meeting the search criteria

find_easy_explain_data()

Learns decision list of projection boxes for easy-to-explain data (for regression). Returns FeatureMap instance containing all the projection boxes found meeting the search criteria.

Parameters

Returns FeatureMap instance containing all the projection boxes found meeting the search criteria

fit()

Calls find_easy_explain_data() and stores the resultant FeatureMap instance.

set_training_data(inputs, outputs)

Sets input and output feature space.

Parameters

inputs: 2-d numpy array of floats (dense, no missing values)

outputs: 1-d numpy array of floats (dense)

FeatureMap *in find_projections.feature_map*

Container class containing projection boxes found from search operations

Methods:

get_num_projections():

Returns the total number of projection boxes in this container object

get_projection(i)

Retrieve the i'th projection-box. This can be an instance of discrete_projection or numeric_projection class depending on the problem.

projection *in find_projections.libfind_projections*

Represents 2-d subset of data, bounded like a rectangular box. Each projection box is defined by a pair of attributes and a range of values for each of the attributes.

Methods:

get_total()

Returns the total number of data points contained in the projection box.

get_att1()

Returns the index of the first attribute

get_att2()

Returns the index of the second attribute

get_att1_start()

Returns the starting value for the range of the first attribute.

get_att1_end()

Returns the ending value for the range of the first attribute.

get_att2_start()

Returns the starting value for the range of the second attribute.

get_att2_end()

Returns the ending value for the range of the second attribute.

get_projection_metric()

Returns the projection metric. If discrete_projection, returns the class label index(majority class).

If numeric_projection, returns the mean of output of data points in the box.

point_lies_in_projection(ds, row)

Returns boolean whether the row'th point in ds data lies inside the projection.

pprojection()

Prints out the contents of the projection.

For a discrete_projection, prints out the box contents in the following order-

Class_label_index, Att1, Att2, Att1_start, Att1_end, Att2_start, Att2_end, Positives, Negatives, Purity of the box.

For a numeric_projection, prints out the box contents in the following order-

Att1, Att2, Att1_start, Att1_end, Att2_start, Att2_end, Total no. of points, mean of output of points and sum-squared-error of points in the box.

discrete_projection **(derives from projection class)** *in find_projections.libfind_projections*

Represents a projection box learnt from data with discrete output (multi-class data).

Methods:

get_class()

Returns the index of the majority class of data points inside the projection box. This will be referred to as positive class and other class points will be referred to as negatives.

get_pos()

Returns the number of positive data points in the projection box.

get_neg()

Returns the number of negative data points in the projection box.

numeric_projection **(derives from projection class)** *in find_projections.libfind_projections*

Represents a projection box learnt from data with numeric output (output for regression).

Methods:

get_mean()

Returns the arithmetic mean of the data points contained within the projection box.

get_sum_sq_error()

Returns the number sum-of-squared-error for the points contained within the projection box.