graphlab.nearest_neighbor_classifier.create

graphlab.nearest_neighbor_classifier.**create**(*dataset, target, features=None, distance=None, verbose=True*)

Create a NearestNeighborClassifier. This model predicts the class of a query point by finding the most common class among the query's nearest neighbors.

Warning

The 'dot_product' distance is deprecated and will be removed in future versions of GraphLab Create. Please use 'transformed_dot_product' distance instead, although note that this is more than a name change; it is a *different* transformation of the dot product of two vectors. Please see the distances module documentation for more details.

Parameters:

dataset : SFrame

Dataset for training the model.

target : string

Name of the column containing the target variable. The values in this column must be of string or integer type.

features : list[string], optional

Name of the columns with features to use in comparing records. 'None' (the default) indicates that all columns except the target variable should be used. Please note: if *distance* is specified as a composite distance, then that parameter controls which features are used in the model. Each column can be one of the following types:

- *Numeric*: values of numeric type integer or float.
- *Array*: array of numeric (integer or float) values. Each array element is treated as a separate variable in the model.
- *Dictionary*: key-value pairs with numeric (integer or float) values. Each key indicates a separate variable in the model.
- *String*: string values.

Please note: if *distance* is specified as a composite distance, then that parameter controls which features are used in the model.

distance : string, function, or list[list], optional

Function to measure the distance between any two input data rows. This may be one of two types:

- *String*: the name of a standard distance function. One of 'euclidean', 'squared_euclidean', 'manhattan', 'levenshtein', 'jaccard', 'weighted_jaccard', 'cosine', 'dot_product' (deprecated), or 'transformed_dot_product'. Please see the `distances` module for more details.
- *Function*: a function handle from the `distances` module. Please see the documentation for that module for specific distance functions.
- *Composite distance*: the weighted sum of several standard distance functions applied to various features. This is specified as a list of distance components, each of which is itself a list containing three items:
  1. list or tuple of feature names (strings)
  2. standard distance name (string)
  3. scaling factor (int or float)

Note that for sparse vectors, missing keys are assumed to have value 0.0. If distance is left unspecified or set to 'auto', then a composite distance is constructed automatically based on feature types.

verbose : bool, optional

If True, print progress updates and model details.

Returns:

out : NearestNeighborClassifier

A trained model of type `NearestNeighborClassifier`.

See also

`NearestNeighborClassifier`, `graphlab.toolkits.nearest_neighbors`, `graphlab.toolkits.distances`

References

- Wikipedia - nearest neighbors classifier
- Hastie, T., Tibshirani, R., Friedman, J. (2009). The Elements of Statistical Learning. Vol. 2. New York. Springer. pp. 463-481.

Examples

```
>>> sf = graphlab.SFrame({'species': ['cat', 'dog', 'fossa', 'dog'],
...                       'height': [9, 25, 20, 23],
...                       'weight': [13, 28, 33, 22]})
...
>>> model = graphlab.nearest_neighbor_classifier.create(sf, target='species')
```

As with the nearest neighbors toolkit, the nearest neighbor classifier accepts composite distance functions. >>> my_dist = [[('height', 'weight'), 'euclidean', 2.7], ... [('height', 'weight'), 'manhattan', 1.6]] ... >>> model = graphlab.nearest_neighbor_classifier.create(sf, target='species', ... distance=my_dist)