K Nearest Neighbours

Dr. Ashish Tendulkar

IIT Madras

Machine Learning Practice

Nearest neighbor classifier

- It is a type of instance-based learning or non-generalizing learning
  - does not attempt to construct a model
  - simply stores instances of the training data

- Classification is computed from a simple majority vote of the nearest neighbors of each point.

- Two different implementations of nearest neighbors classifiers are available.

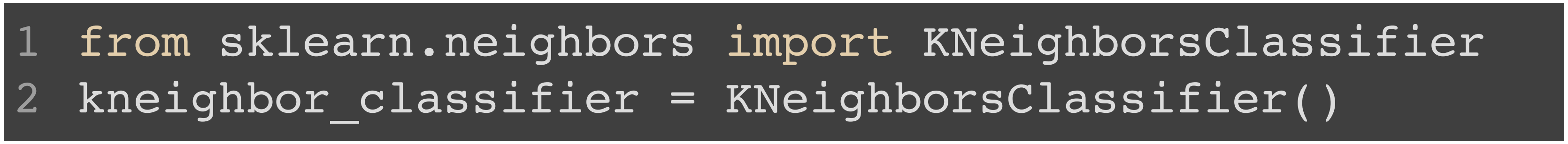
1. KNeighborsClassifier
2. RadiusNeighborsClassifier

How are KNeighborsClassifier and RadiusNeighborsClassifier different?

KNeighborsClassifier	**RadiusNeighborsClassifier**
• learning based on the k nearest neighbors	• learning based on the number of neighbors within a fixed radius *r* of each training point
• most commonly used technique	• used in cases where the data is not uniformly sampled
• choice of the value *k* is highly data-dependent	• fixed value of *r* is specified, such that points in sparser neighborhoods use fewer nearest neighbors for the classification

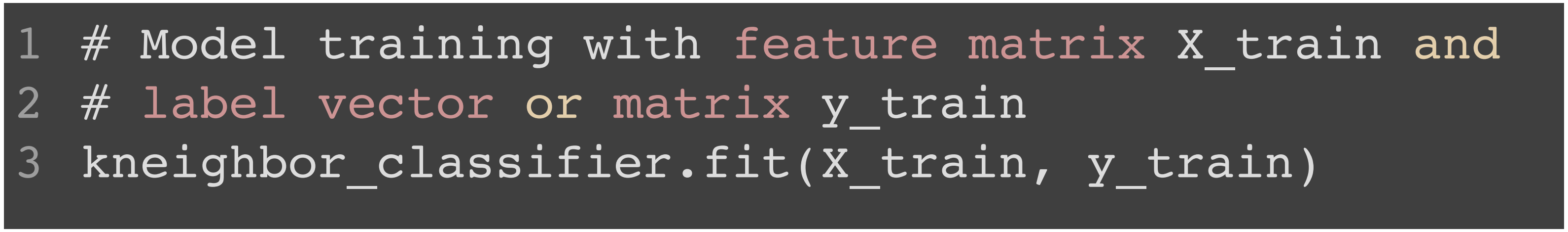
How do you apply KNeighborsClassifier?

Step 1: Instantiate a KNeighborsClassifer estimator without passing any arguments to it to create a classifer object.



```
from sklearn.neighbors import KNeighborsClassifier
kneighbor_classifier = KNeighborsClassifier()
```

Step 2: Call fit method on KNeighbors classifier object with training feature matrix and label vector as arguments.



```
# Model training with feature matrix X_train and
# label vector or matrix y_train
kneighbor_classifier.fit(X_train, y_train)
```

How do you specify the number of nearest neighbors in KNeighborsClassifier?

- Specify the number of nearest neighbors *K* from the training dataset using `n_neighbors` parameter.
  - value should be int.

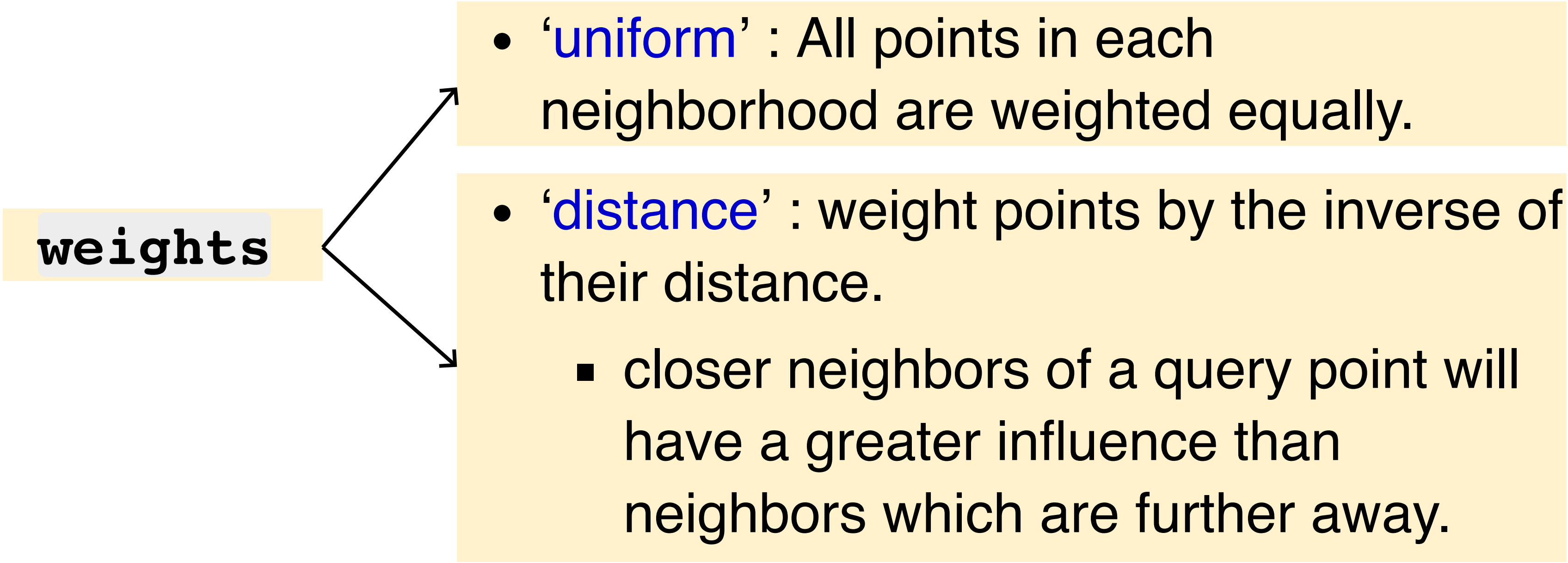
```
kneighbor_classifier = KNeighborsClassifier(n_neighbors = 3)
```

What is the default value of *K*?

`n_neighbors = 5`

How do you assign weights to neighborhood in KNeighborsClassifier?

- It is better to weight the neighbors such that nearer neighbors contribute more to the fit.



`weights`

- ‘uniform’ : All points in each neighborhood are weighted equally.

- ‘distance’ : weight points by the inverse of their distance.
  - closer neighbors of a query point will have a greater influence than neighbors which are further away.

Default:

```
kneighbor_classifier = KNeighborsClassifier(weights= 'uniform')
```

Can we define our own weight values for KNeighborsClassifier?

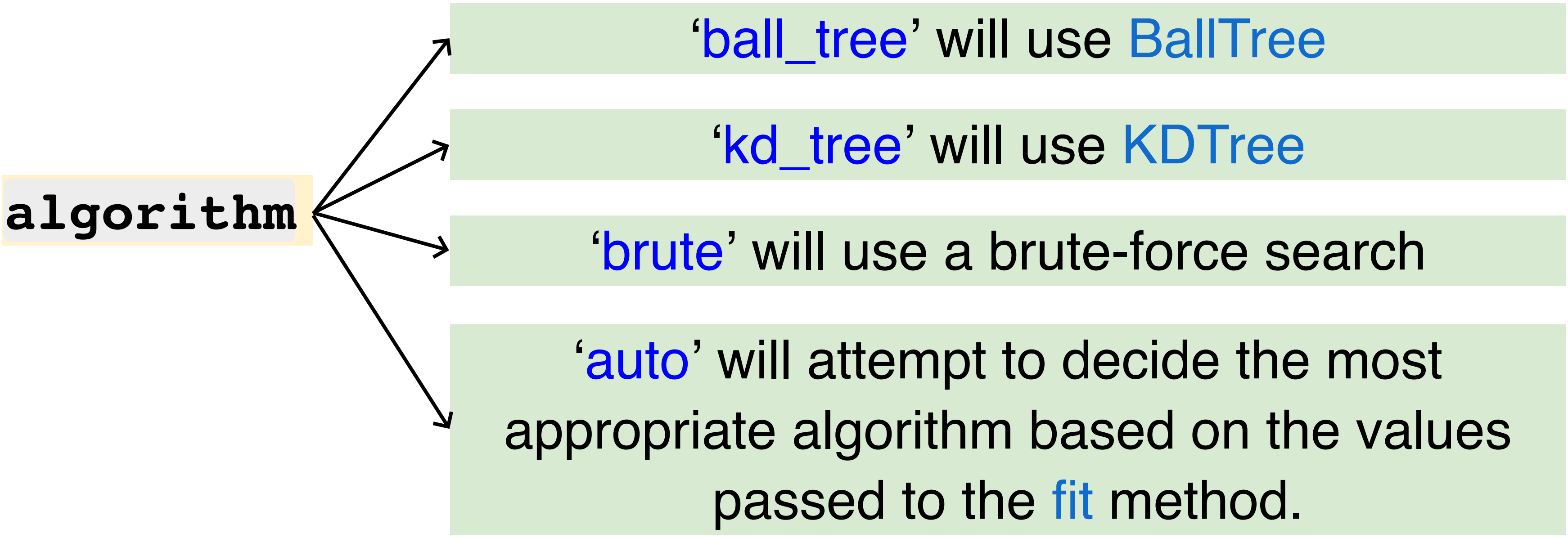
- Yes, it is possible if you have an array of distances.
- **`weights`** parameter also accepts a user-defined function which takes an array of distances as input, and returns an array of the same shape containing the weights.

Example:

```
def user_weights(weights_array):
    return weights_array

kneighbor_classifier = KNeighborsClassifier(weights=user_weights)
```

Which algorithm is used to compute the nearest neighbors in KNeighborsClassifier?



`algorithm`

‘ball_tree’ will use BallTree

‘kd_tree’ will use KDTree

‘brute’ will use a brute-force search

‘auto’ will attempt to decide the most appropriate algorithm based on the values passed to the fit method.

Default:

```
kneighbor_classifier = KNeighborsClassifier(algorithm='auto')
```

Some additional parmeters for tree algorithm in KNeighborsClassifier?

For 'ball_tree' and 'kd_tree' algorithms, there are some other parameters to be set.

`leaf_size`

- can affect the speed of the construction and query, as well as the memory required to store the tree
- default = 30

`metric`

- Distance metric to use for the tree
- It is either string or callable function
  - some metrics are listed below:
    - “euclidean”, “manhattan”, “chebyshev”, “minkowski”, “wminkowski”, “seuclidean”, “mahalanobis”
- default = 'minkowski'

`p`

- Power parameter for the Minkowski metric.
- default = 2

How do you apply RadiusNeighborsClassifier?

Step 1: Instantiate a RadiusNeighborsClassifer estimator without passing any arguments to it to create a classifer object.

```
from sklearn.neighbors import RadiusNeighborsClassifier
radius_classifier = RadiusNeighborsClassifier()
```

Step 2: Call fit method on RadiusNeighbors classifier object with training feature matrix and label vector as arguments.

```
# Model training with feature matrix X_train and
# label vector or matrix y_train
radius_classifier.fit(X_train, y_train)
```

How do you specify the number of neighbors in RadiusNeighborsClassifier?

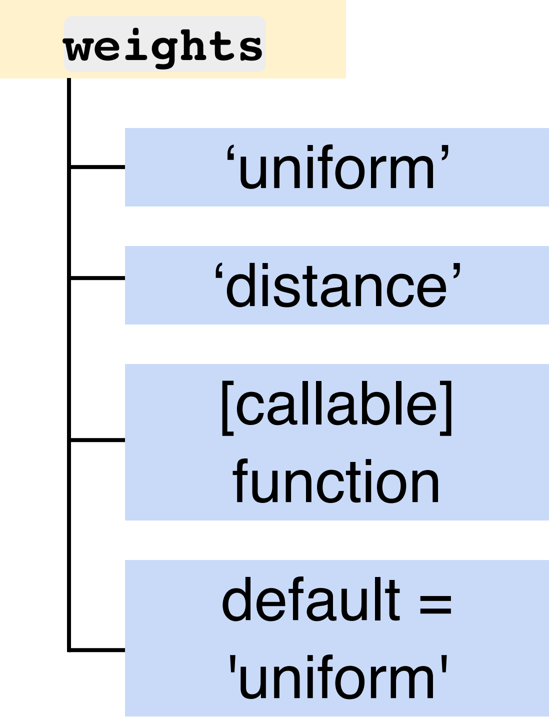
- The number of neighbors is specified within a fixed radius *r* of each training point using `radius` parameter.
- *r* is a float value.

```
radius_classifier = RadiusNeighborsClassifier(radius=1.0)
```

What is the default value of *r* ?

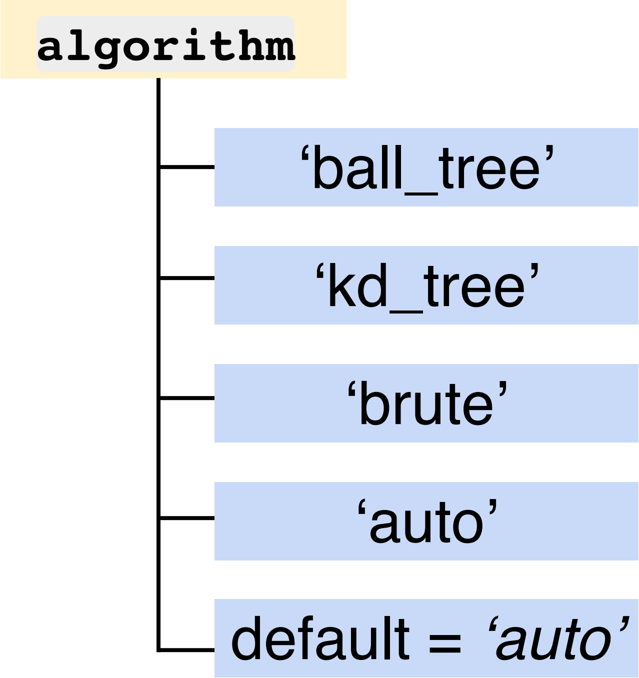
`r = 1.0`

Parameters for RadiusNeighborsClassifier



`weights`

- ‘uniform’
- ‘distance’
- [callable] function
- default = 'uniform'



`algorithm`

- ‘ball_tree’
- ‘kd_tree’
- ‘brute’
- ‘auto’
- default = *‘auto’*

`leaf_size`

default = 30

`metric`

default = 'minkowski'

`p`

default = 2