Computer Exercise 4

王永浩 WangYonghao 2016011420

Contents

1 k-NN

1.1 k-NN package used in the experiment

Source: A k-NN package from sklearn module:sklearn.neighbors

Platform: scikit-learn is python package which is a simple and efficient tool used in mechine learning. It can be used on a various of platroms such as Windows,linux and Mac.

Algorithm used for searching nearest neighbors: The algorithms used in the k-NN are optional,which we have 3 choices: Brute Force,K-D Tree and Ball Tree.

Brute Force: The most naive neighbor search implementation involves the brute-force computation of distances between all pairs of points in the dataset: for N samples in D dimensions, this approach scales as $o(DN^2)$. Efficient brute-force neighbors searches can be very competitive for small data samples. However, as the number of N grows, the brute-force approach quickly becomes infeasible.

K-D Tree: The KD tree is a binary tree structure which recursively partitions the parameter space along the data axes, dividing it into nested orthotropic regions into which data points are filed. The construction of a KD tree is very fast: because partitioning is performed only along the data axes, no D-dimensional distances need to be computed. Once constructed, the nearest neighbor of a query point can be determined with only $o(log(N))$distance computations. Though the KD tree approach is very fast for low-dimensional ($D < 20$) neighbors searches, it becomes inefficient as Dgrows very large: this is one manifestation of the so-called "curse of dimensionality" .

Ball True: A ball tree recursively divides the data into nodes defined by a centroid C and radius r, such that each point in the node lies within the hypersphere defined by r and C. The number of candidate points for a neighbor search is reduced through use of the triangle inequality:$|x + y| \leq |x| + |y|$.With this setup, a single distance calculation between a test point and the centroid is sufficient to determine a lower and upper bound on the distance to all points within the node. Because of the spherical geometry of the ball tree nodes, it can out-perform a KD-tree in high dimensions, though the actual performance is highly dependent on the structure of the training data.

Hyper-parameters and options users need to choose:

- n_neighbors : Number of neighbors to use by default for k neighbors queries.

- weights : str or callable, optional (default = ‘uniform’):
  - ‘uniform’ : uniform weights. All points in each neighborhood are weighted equally.
  - ‘distance’ : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
  - [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.
- algorithm : ‘auto’ , ‘ball_tree’ , ‘kd_tree’ , ‘brute’ , optional(has been explained above)
- metric : string or callable, default ‘minkowski’ ; the distance metric to use for the tree. The default metric is minkowski, and with p=2 is equivalent to the standard Euclidean metric. See the documentation of the DistanceMetric class for a list of available metrics.

1.2 Experiment Design

In this section I perform the experiments based on the data used in Exercise 2. I evaluate the method under different settings to justify the results.

1.2.1 Parameter Setting

Parameters of k-NN I have set are 'weight' and 'n_neighbors'. For 'weight' I have tried 'uniform' and 'distance', 'n_neighbors' is set to 5,10,15,20,25,30.

1.3 Experiment Result

The classification is trained on trainingset-1 and trainingset-2 then tested on TestSet and TestSet-2, since trainingset-2 is the combination of samples in Trainingset-1 and TestSet-2, we don't use TestSet-2 to test the model when trained on trainingset-2. The results are shown as the Table 1 below.

1.3.1 Discussions on the results of the experiment

The influence of 'weights': We can find that if we set the 'weights' to 'distance', the k-NN model always perfoms better than it was assigned to 'uniform' when 'n_neighbors' doesn't change, both training and test accuracy are improved. That's because under some circumstances, it is better to weight the neighbors such that nearer neighbors contribute more to the fit, which is more reasonable.

The influence of the value of 'n_neighbors': The optimal value of 'n_neighbors'(we denote it as k) is highly data-dependent: in general a larger k suppresses the effects of noise, but makes the classification boundaries less distinct. In this experiment we can see that when we train the model on trainingset-1, the best performence occured at $k = 5$ and it would decrease its accuracy rapidly with the increase of k, but when we use trainigset-2, the model would get a better performence at around $k = 15$ and its accuracy changed little with the variety of k.

2 Random Forest

trainingdata	testdata	weights	n-neighbors	training accuracy	test accuracy
trainingset-1	testset	uniform	5	0.925	0.88
trainingset-1	testset	uniform	10	0.8	0.8
trainingset-1	testset	uniform	15	0.75	0.73
trainingset-1	testset	uniform	20	0.65	0.57
trainingset-1	testset	uniform	25	0.65	0.55
trainingset-1	testset	uniform	30	0.65	0.55
trainingset-1	testset	distance	5	1.0	0.87
trainingset-1	testset	distance	10	1.0	0.83
trainingset-1	testset	distance	15	1.0	0.83
trainingset-1	testset	distance	20	1.0	0.71
trainingset-1	testset	distance	25	1.0	0.61
trainingset-1	testset	distance	30	1.0	0.57
trainingset-1	testset-2	uniform	5	0.925	0.8403041825095057
trainingset-1	testset-2	uniform	10	0.8	0.7414448669201521
trainingset-1	testset-2	uniform	15	0.75	0.688212927756654
trainingset-1	testset-2	uniform	20	0.65	0.5741444866920152
trainingset-1	testset-2	uniform	25	0.65	0.5513307984790875
trainingset-1	testset-2	uniform	30	0.65	0.5513307984790875
trainingset-1	testset-2	distance	5	1.0	0.8365019011406845
trainingset-1	testset-2	distance	10	1.0	0.7984790874524715
trainingset-1	testset-2	distance	15	1.0	0.7376425855513308
trainingset-1	testset-2	distance	20	1.0	0.6844106463878327
trainingset-1	testset-2	distance	25	1.0	0.5855513307984791
trainingset-1	testset-2	distance	30	1.0	0.5551330798479087
trainingset-2	testset	uniform	5	0.9471947194719472	0.92
trainingset-2	testset	uniform	10	0.9174917491749175	0.91
trainingset-2	testset	uniform	15	0.9372937293729373	0.93
trainingset-2	testset	uniform	20	0.9207920792079208	0.93
trainingset-2	testset	uniform	25	0.9306930693069307	0.92
trainingset-2	testset	uniform	30	0.9207920792079208	0.91
trainingset-2	testset	distance	5	1.0	0.93
trainingset-2	testset	distance	10	1.0	0.93
trainingset-2	testset	distance	15	1.0	0.93
trainingset-2	testset	distance	20	1.0	0.93
trainingset-2	testset	distance	25	1.0	0.92
trainingset-2	testset	distance	30	1.0	0.92

Table 1. k-NN result