Nearest Neighbors

compiled by Alvin Wan from Professor Efros's lecture

For nearest neighbors: Given a query instance x_q, locate nearest training example x_n, and take x_n's value, so estimate $\hat{f}(x_q) \leftarrow f(x_n)$. For k-nearest nearest neighbors, take a vote from k nearest neighbors.

This is also known as instance-based learning, memory-based, exemplar-based learning, or lazy learning. At training time, we simply do nothing, but we need to carry all data points with us, to make a prediction. Nearest neighbors is known as a non-parametric algorithm.

1 Parametric v. Non-Parametric Algorithms

In a parametric algorithm, the number of parameters is independent of n. For non-parametric algorithms, the number of parameters increases with N.

1. Logistic Regression: parametric
2. Nearest Neighbor: non-parametric, N parameters
3. K-Nearest Neighbor: non-parametric, $\frac{N}{k}$ parameters
4. Linear Discriminant Analysis: parametric
5. SVM: parametric
6. Kernel SVM: non-parametric (Remember, there is an α for each data point)
7. Deep Learning: neither

(Professor Benjamin Recht does not think this distinction is useful.)

2 Voronoi Diagrams

A K-nearest neighbors instance results in a Voronoi diagram. A **Voronoi cell** is comprised of all points closer to x than to any other instance in S. The **region of class** C is the union of Voronoi cells of insatnces of C in S. Let us now examine the behavior of nearest neighbors in the limit. Cover and Hart (1967) prove that with infinite data, nearest neighbors (NN) has at most twice the error of the optimal solution.

With $\epsilon^*(x)$ (error of optimal prediction) and $\epsilon_{NN}(x)$ (error of nearest neighbors):

$$\lim_{n\to\infty} \epsilon_{NN} \leq 2\epsilon^*$$

For K-nearest neighbors (kNN),

$$\lim_{n\to\infty, k\to\infty, k/n\to 0} \epsilon_{kNN} = \epsilon^*$$

3 Advantages, Disadvantages of NN

The advantages of nearest neighbors include the following:

1. Training is fast (no training).
2. Learn complex target functions easily, as we make no assumptions about the distribution of the data.
3. Don't lose information. With parametric algorithms, the expressiveness of your model is limited. With non-parametric algorithms, you are not limited.

For non-parametric models in general, we don't need predetermined categories. With lazy label transfer (kNN), we can easily add new features for our data, such as elevation or population density.

The disadvantages include:

1. Slow at query time
2. Lots of storage
3. Easily fooled by irrelevant attributes. This is highly contingent on the quality of the distance metric.

4 Curse of Dimensionality

We have a concept called **curse of dimensionality**; NN is easily misled in high dimensions. Easy problems in low dimensions can be hard in high dimensions, and intuitions for a low-dimensional space don't apply to a space with high dimensions. Some examples include:

- Points on hyper-grid
- Hypersphere
- Hypercube v. hypersphere
- High-dimensional Gaussian

4.1 Anomalies

Consider a sphere with radius R. Consider a thin boundary of ϵ distance; calculate the ratio of the interior to the entire sphere, in high dimensions.

$$(\frac{R-\epsilon}{R})^d = (1-\epsilon)^k$$

As k approaches infinity, this ratio approaches 0; the boundary assumes more and more of the total volume. Consider a similar problem, with the hypersphere-hypercube ratio. In 2 dimensions, there are two corners that lie outside of the hypersphere. In 3 dimensions, there are now 4. This trend continues, with the ratio approaching 0.

In high dimensions, we see that a Gaussian does not retain the 2-dimensional form we expect.

4.2 Solutions

To remedy the situation, we can choose to either:

1. get more data (make N larger) or
2. make better features or a better distance metric (make D smaller)
    - Rescale the image, and make it more invariant to small perturbations.
    - Histograms: marginal histograms or joint histograms

We can also *learn* better features. Much of machine learning can be thought of as learning features and then applying a simple algorithm such as kNN or linear classifiers.

In sum, don't underestimate NN. It works remarkably well, but it is **always a good baseline.** The larger issue is that we are still just interpolating training data and need to extrapolate, generalize.