Notes on Statistical and Machine Learning

Prototype Methods and Nearest-Neighbors

Chapter: *18* **Prepared by:** *Chenxi Zhou*

This note is prepared based on *Chapter 13, Prototype Methods and Neatest-Neighbors* in Hastie, Tibshirani, and Friedman (2009).

I. Prototype Methods

1. **Setup:** Let $\{(\mathbf{x}_i, g_i)\}_{i=1}^n$ be n pairs of training data, where g_i is a class label taking values in $\mathcal{W} := \{1, 2, \cdots, W\}$ so that we have W classes in total.

2. **Prototype Methods:** *Prototype methods* represent the training data by a set of points in the feature space. Each prototype has an associated class label, and classification of a query point $\mathbf{x}$ is made to the class of the closest prototype.

   Remark. The "closedness" between $\mathbf{x}$ and the prototypes is usually measured by the Euclidean distance, even though other distance measurements are also possible.

3. **General Comments:**

   (a) Prototype methods can be very *effective* if the prototypes are well-positioned to capture the distribution of each class. Irregular class boundaries can be represented, with enough prototypes in the right places in feature space.

   (b) The main challenges are to figure out

   - how many prototypes to use, and
   - where to put them.

4. **K-means Clustering for Unlabeled Data:**

   (a) *Overview: K-means clustering* is a method for finding clusters and cluster centers for a set of *unlabeled* data.

   (b) *Main Idea:* One chooses the desired number of cluster centers, denoted by K, and iteratively moves the centers to minimize the total within cluster variance.

   (c) *Procedure:* Given an initial set of centers, K-means clustering algorithm alternates between the following two steps until convergence:

   i. For each center, identify the subset of training points (cluster of points) that is closer to it than any other centers;

   ii. Compute the means of each feature for the data points in each clusters, and this mean vector becomes the new center for that cluster.

(d) *Initialization of Centers:* Typically, initial centers are K randomly chosen observations from the training data.

5. K-means Clustering for Labeled Data:

(a) *Procedure:*

i. Apply K-means clustering to the training data in each class *separately*, using K prototypes per class;

ii. Assign a class label to each of the $W \times K$ prototypes;

iii. Classify a new feature vector $\mathbf{x}$ to the class of the closest prototype.

(b) *Comments:* It is possible that a number of prototypes are near the class boundaries, leading to potential misclassification error for points near these boundaries.

6. Learning Vector Quantization:

(a) *Overview:* Learning vector quantization (LVQ) is an *online* algorithm — observations are processed one at a time.

(b) *Main Idea:* Training points attract prototypes of the correct class and repel other prototypes. When iterations settle down, prototypes should be close to the training points in their class.

(c) *Algorithm:* The complete algorithm is shown in Algorithm 1.

Algorithm 1 Learning Vector Quantization

1: Choose K initial prototypes for each class: $\mathbf{c}_1(w), \mathbf{c}_2(w), \cdots, \mathbf{c}_K(w)$, for all $w = 1, 2, \cdots, W$ (for example, by sampling K training points at random from each class);

2: Sample a training point $\mathbf{x}_i$ randomly (with replacement), and let (j, w) index the closest prototype $\mathbf{c}_j(w)$ to $\mathbf{x}_i$;

i. If $g_i = w$ (i.e., they are in the same class), move the prototype towards the training point:

$$\mathbf{c}_j(w) \quad \leftarrow \quad \mathbf{c}_j(w) + \alpha(\mathbf{x}_i - \mathbf{c}_j(w)),$$

where $\alpha > 0$ is the learning rate;

ii. If $g_i \neq w$ (i.e., they are in different classes), move the prototype away from the training point:

$$\mathbf{c}_j(w) \quad \leftarrow \quad \mathbf{c}_j(w) - \alpha(\mathbf{x}_i - \mathbf{c}_j(w));$$

3: Repeat the preceding step, decreasing the learning rate α with each iteration towards zero.

(d) *Comments:*

i. Advantage: The prototypes tend to move away from the decision boundaries and away from prototypes of competing classes. This fixes the problem of the K-means clustering for labeled data.

ii. Drawback: The procedure of learning vector quantization is defined by an algorithm rather than the optimization of some fixed criterion. It is hard to understand its properties.

7. Gaussian Mixtures:

(a) *Overview:* Each cluster is described in terms of a Gaussian density, which has a centroid (as in K-means clustering) and a covariance matrix.

(b) *Procedure:* We use the EM algorithm to fit the Gaussian mixture model:

i. In the E-step, each observation is assigned a responsibility or weight for each cluster, based on the likelihood of each of the corresponding Gaussians;

ii. In the M-step, each observation contributes to the weighted means (and covariances) for every cluster.

(c) *Comparison to K-means Clustering:* The Gaussian mixture model is often referred to as a *soft* clustering method, while K-means is a *hard* one.

(d) *Classification Rule:* Suppose we have class labels. When Gaussian mixture models are used to represent the feature density in each class, it produces smooth posterior probabilities

$$\big(\hat{p}_1(\mathbf{x}), \hat{p}_2(\mathbf{x}), \cdots, \hat{p}_W(\mathbf{x})\big)^\top$$

for classifying $\mathbf{x}$. This can be viewed as a *soft classification rule*. We classify $\mathbf{x}$ to Class $\arg\max_{w=1,2,\cdots,W} \hat{p}_w(\mathbf{x})$.

II. k-Nearest-Neighbor Classifiers

1. Main Idea: The k-nearest-neighbor classifiers are *memory-based* and require *no* model to be fit. Given a query point $\mathbf{x}_0$,

(a) find the k-training points $\mathbf{x}_{(r)}$, for $r = 1, \cdots, k$, closest in distance to $\mathbf{x}_0$, and

(b) classify using the majority vote among the k neighbors.

When there are ties, we break them at random.

2. Effectiveness: k-nearest-neighbors is very successful when

(a) each class has many possible prototpes, and

(b) the decision boundary is very irregular.

3. Relationship Between k-Nearest-Neighbors and Prototype Methods: In 1-nearest-neighbor classification, each training point is a prototype.

Remark. For the 1-nearest-neighbor classification, the *bias* of estimate is often low, but the variance is high, since it uses only the training point closest to the query point.

4. **Error Rate:** Asymptotically, the error rate of the 1-nearest-neighbor classifier is never more than twice the Bayes error rate. Here, we assume that the query point coincides with one of the training points, so that the *bias* of the 1-nearest-neighbor classifier is zero.

   At the point $\mathbf{x}$, let w^* be the dominant class, and $p_w(\mathbf{x})$ be the true conditional probability for Class w. Then, conditional on $\mathbf{x}$, we have

$$\begin{aligned} \text{Bayes error} &= 1 - p_{w^*}(\mathbf{x}), \\ \text{1-nearest-neighbor error} &= \sum_{w=1}^{W} p_w(\mathbf{x})(1 - p_w(\mathbf{x})) \geq 1 - p_{w^*}(\mathbf{x}). \end{aligned}$$

   Thus, the 1-nearest-neighbor error is always at least equal to the Bayes error rate. Then,

   (a) if $W = 2$, we have

$$2p_{w^*}(\mathbf{x})(1 - p_{w^*}(\mathbf{x})) \leq 2(1 - p_{w^*}(\mathbf{x})); \tag{1}$$

   that is, the asymptotic error rate of 1-nearest-neighbor classifier is never more than twice the Bayes error rate;

   (b) in general, the following inequality holds

$$\sum_{w=1}^{W} p_w(\mathbf{x})(1 - p_w(\mathbf{x})) \leq 2(1 - p_{w^*}(\mathbf{x})) - \frac{W}{W-1}(1 - p_{w^*}(\mathbf{x}))^2. \tag{2}$$

III. Adaptive Nearest-Neighbor Methods

1. **Motivation:** When nearest-neighbor classification is carried out in a high-dimensional feature space, the nearest neighbors of a point can be very far away, causing bias and degrading the performance of the rule.

2. **Example:** Consider n data points uniformly distributed in the unit cube $[-\frac{1}{2}, \frac{1}{2}]^p$. Let R be the radius of a 1-nearest-neighborhood centered at the origin. Then, we show that

$$\text{median}(R) = v_p^{-\frac{1}{p}} \left(1 - 2^{-\frac{1}{n}}\right)^{\frac{1}{p}}, \tag{3}$$

   where v_p is the constant such that the volume of a ball of radius r in p-dimensional space is $v_p r^p$.

   Let $X \sim [-\frac{1}{2}, \frac{1}{2}]^p$. Then,

$$\mathbb{P}(X \notin B_r(\mathbf{0}_p)) = 1 - v_p r^p,$$

where $B_r(\mathbf{0}_p)$ denotes the ball centered at the origin with radius $r \in (0, \frac{1}{2})$. Then, with $X_1, X_2, \cdots, X_n \sim [-\frac{1}{2}, \frac{1}{2}]^p$, the probability that the 1-nearest-neighborhood of origin, i.e., the point that is the closest to the origin, falls outside of the ball $B_r(\mathbf{0}_p)$ is

$$\mathbb{P}(X_1, X_2, \cdots, X_n \notin B_r(\mathbf{0}_p)) = (1 - v_p r^p)^n.$$

If we let R_i be the distance of X_i to the origin and $R := \min_{i=1,2,\cdots,n} R_i$, we have

$$\mathbb{P}(R > r) = \mathbb{P}(X_1, X_2, \cdots, X_n \notin B_r(\mathbf{0}_p)) = (1 - v_p r^p)^n.$$

Therefore, median(R) must satisfy

$$(1 - v_p \text{median}(R)^p)^n = \frac{1}{2},$$

from which we can obtain (3).

In particular, we can notice that the median radius quickly approaches 0.5, the distance to the edge of the cube.

3. **Discriminant Adaptive Nearest-Neighbor (DANN):** Discriminant adaptive nearest-neighbor (DANN) is an approach to adapt the metric used in the nearest-neighbor classification.

   (a) *Main Idea:* At each query point $\mathbf{x}_0 \in \mathbb{R}^p$, a neighborhood of k points is formed, and the class distribution among these points is used to decide how to adapt the metric. The adapted metric is then used in a nearest-neighbor rule at the query point.

   Remark. At each query point, a potentially different metric is used.

   (b) *DANN Metric:* The *DANN metric* at a query point $\mathbf{x}_0 \in \mathbb{R}^p$ is defined by

   $$D(\mathbf{x}, \mathbf{x}_0) := (\mathbf{x} - \mathbf{x}_0)^\top \mathbf{\Sigma} (\mathbf{x} - \mathbf{x}_0), \tag{4}$$

   where

   $$\mathbf{\Sigma} := \mathbf{W}^{-\frac{1}{2}} (\mathbf{W}^{-\frac{1}{2}} \mathbf{B} \mathbf{W}^{-\frac{1}{2}} + \varepsilon \mathbf{I}_p) \mathbf{W}^{-\frac{1}{2}},$$

   and $\mathbf{W} := \sum_{w=1}^{W} \pi_w \mathbf{W}_w$ is the pooled within-class covariance matrix, $\mathbf{B} := \sum_{w=1}^{W} \pi_w (\bar{\mathbf{x}}_w - \bar{\mathbf{x}})(\bar{\mathbf{x}}_w - \bar{\mathbf{x}})^\top$ is the between class covariance matrix, $\bar{\mathbf{x}}_w$ is the mean vector of the w-th class, and $\bar{\mathbf{x}}$ is the overall mean vector.

   Note that here all quantities are computed from the k observations close to the query point $\mathbf{x}_0$.

   Remark. If all k points belong to a single class, we have $\mathbf{B} = \mathbf{0}_{p \times p}$ and $\mathbf{\Sigma}$ reduces to $\varepsilon \mathbf{W}^{-1}$.

   (c) *DANN:* After computation of the metric, it is used in a nearest-neighbor rule at $\mathbf{x}_0$.

References

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The Elements of Statistical Learning*. Vol. 1. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.