Estimating the Total Variation Distance using Two Samples

Tom Dietterich, October 30, 2014, version 3

Objective: Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability distributions over $\mathfrak{R}^n$. Given a set of iid samples $X = \{x_1, \dots, x_N\}$ and a set of iid samples $Y = \{y_1, \dots, y_M\}$ generated from unknown probability distributions $\mathbb{P}$ and $\mathbb{Q}$ respectively, estimate the distance between the distributions $\mathbb{P}$ and $\mathbb{Q}$,

$$\delta(\mathbb{P}, \mathbb{Q}) = \int_{\mathfrak{R}^n} |\mathbb{P}(x) - \mathbb{Q}(x)| dx \qquad (1)$$

Solution: For a finite alphabet, eq. (1) is equivalent to

$$\delta(\mathbb{P}, \mathbb{Q}) = \frac{1}{2} \sum_x |\mathbb{P}(x) - \mathbb{Q}(x)|, \qquad (2)$$

which is half the L1 distance between the corresponding discrete probability mass functions.

Suppose we define a partition of $\mathfrak{R}^n$ into sets $\{S_1, \dots, S_K\}$. Let $S_k(X)$ denote the number of points from X that fall in cell S_k. Then we can approximate δ by

$$\sum_k \left| \frac{S_k(X)}{N} - \frac{S_k(Y)}{M} \right|.$$

The below algorithm does exactly this by growing a kd-tree on the given data samples X and Y. At each node, the kd-tree splits on the median and stops when further splitting create leaves that are too small.

```
BuildTree(Data, minLeaf, NA, NB, NDim)
% the data. Data.A data set A; Data.B is data set B.
% NA = number of points in data set A
% NB = number of points in data set B
% NDim = number of dimensions
       if (Data.A.numPoints() < 2*minLeaf || Data.B.numPoints() < 2*minLeaf) {
              % not enough points to split, so compute the variation distance at this leaf
              return 0.5*(abs(Data.A.numPoints() / NA – Data.B.numPoints()/NB))
              }
       else {
              % select a dimension at random
              d = floor(random(NDim))
              % Compute the median of the union of the two data sets along this dimension
              med = median(SetUnion(Data.A[d], Data.B[d]))  % [] returns only d'th dimension
              % split the data according to the median
              LeftData.A = all elements of Data.A <= med
              LeftData.B = all elements of Data.B <= med
              RightData.A = all elements of Data.A > med
              RightData.B = all elements of Data.B > med
              % combine the results of the left and right recursive calls
              return BuildTree(LeftData, minLeaf, NA, NB, NDim) +
                                BuildTree(RightData, minLeaf, NA, NB, NDim)
              }
```

We can call this function k (≈ 100) times and take the average as an approximate measure of total variation δ.