kNN, Decision Tree, Random Forest

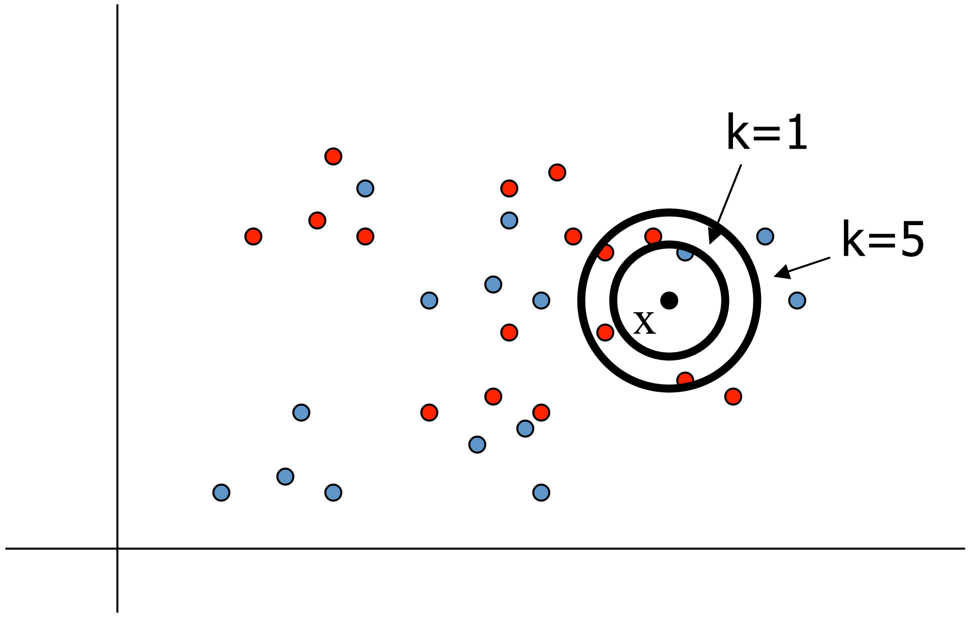
Mikhail Lipkovich
06/25/2018

Nearest Neighbor Algorithm

- Learning Algorithm:
  - Store training examples
- Prediction Algorithm:
  - To classify a new example **x** by finding the training example ($\mathbf{x}^i$,y^i) that is *nearest* to **x**
  - Guess the class $y = y^i$

K-Nearest Neighbor Methods

- To classify a new input vector x, examine the k-closest training data points to x and assign the object to the most frequently occurring class



common values for k: 3, 5

Nearest Neighbor

When to Consider

- Instance map to points in R^n
- Less than 20 attributes per instance
- Lots of training data

Advantages

- Training is very fast
- Learn complex target functions
- Do not lose information

Disadvantages

- Slow at query time
- Easily fooled by irrelevant attributes

Issues

- Distance measure
  - Most common: Euclidean
- Choosing k
  - Increasing k reduces variance, increases bias
- For high-dimensional space, problem that the nearest neighbor may not be very close at all!
- Memory-based technique. Must make a pass through the data for each classification. This can be prohibitive for large data sets.

Distance

- Notation: object with p measurements

$$x^i = (x^i_1, x^i_2, \ldots, x^i_p)$$

- Most common distance metric is *Euclidean* distance:

$$d_E(x^i, x^j) = \left(\sum_{k=1}^{p} (x^i_k - x^j_k)^2 \right)^{\frac{1}{2}}$$

- ED makes sense when different measurements are commensurate; each is variable measured in the same units.

- If the measurements are different, say length and weight, it is not clear.

Standardization

When variables are not commensurate, we can standardize them by dividing by the sample standard deviation. This makes them all equally important.

The estimate for the standard deviation of x_k :

$$\hat{\sigma}_k = \left(\frac{1}{n} \sum_{i=1}^{n} \left(x_k^i - \bar{x}_k \right)^2 \right)^{\frac{1}{2}}$$

where x_k is the sample mean:

$$\bar{x}_k = \frac{1}{n} \sum_{i=1}^{n} x_k^i$$

Weighted Euclidean distance

Finally, if we have some idea of the relative importance of each variable, we can weight them:

$$d_{WE}(i,j) = \left(\sum_{k=1}^{p} w_k (x_k^i - x_k^j)^2 \right)^{\frac{1}{2}}$$

Non-numerical Data

How to calculate distance for categorical or text data?

Non-numerical Data

How to calculate distance for categorical or text data?

- Categorical data:
  c1 = c2 => dist = 0
  c1 != c2 => dist = 1

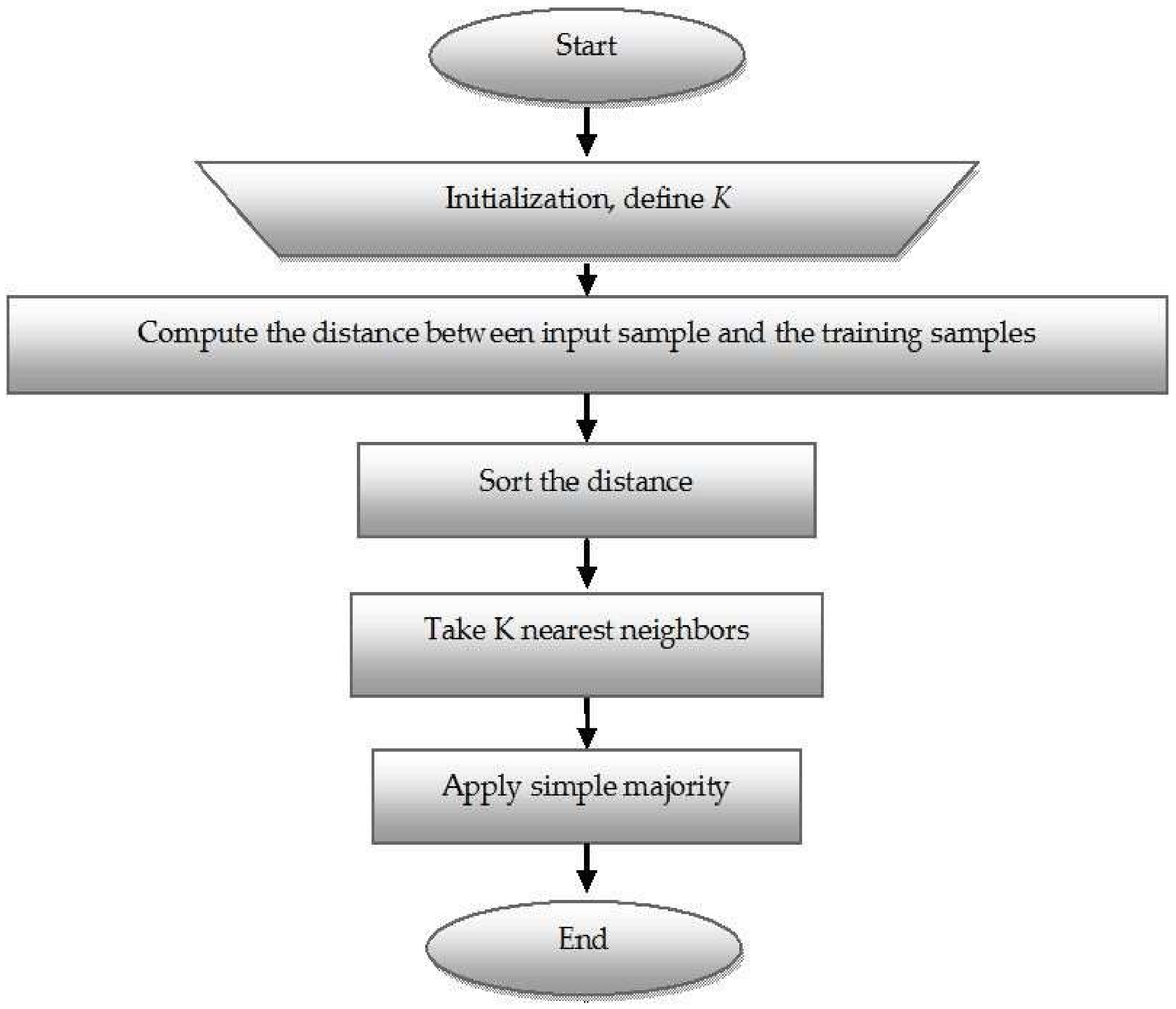
Non-numerical Data

How to calculate distance for categorical or text data?

- Categorical data:
  c1 = c2 => dist = 0
  c1 != c2 => dist = 1
- Text data:

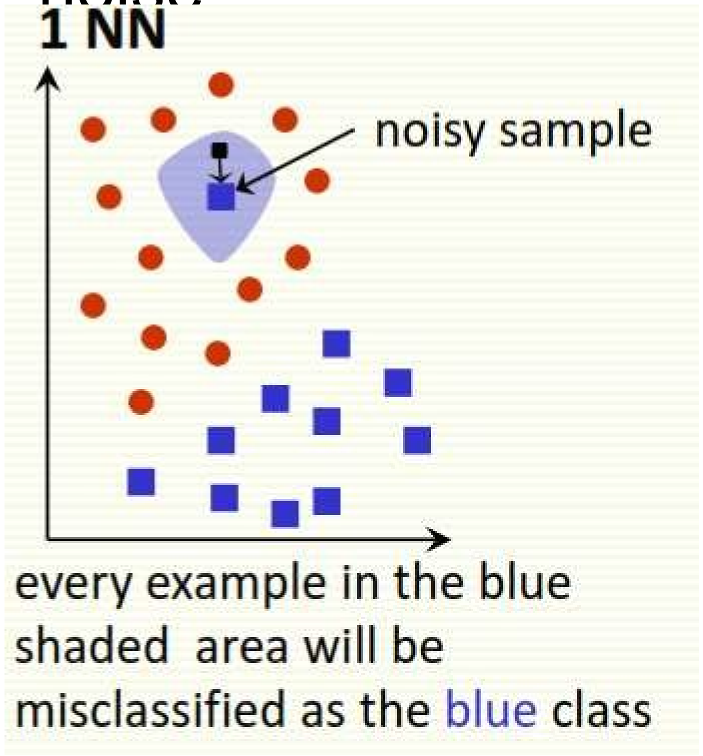
  d(Анау, Мынау) = 2
  d(Сергей, Серик) = 3

KNN Classifier Algorithm

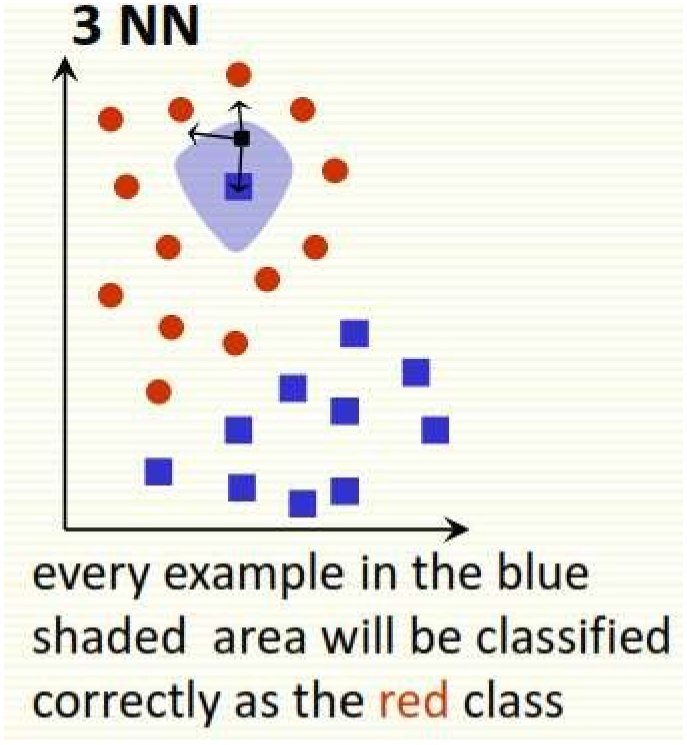

Start
Initialization, define K
Compute the distance betw een input sample and the training samples
Sort the distance
Take K nearest neighbors
Apply simple majority
End

How to choose K?

- If infinite number of samples available, the larger is k, the better is classification.
- k = 1 is often used for efficiency, but sensitive to "noise"



every example in the blue shaded area will be misclassified as the blue class



every example in the blue shaded area will be classified correctly as the red class

KNN Advantages

- Easy to program
- No optimization or training required
- Classification accuracy can be very good; can outperform more complex models