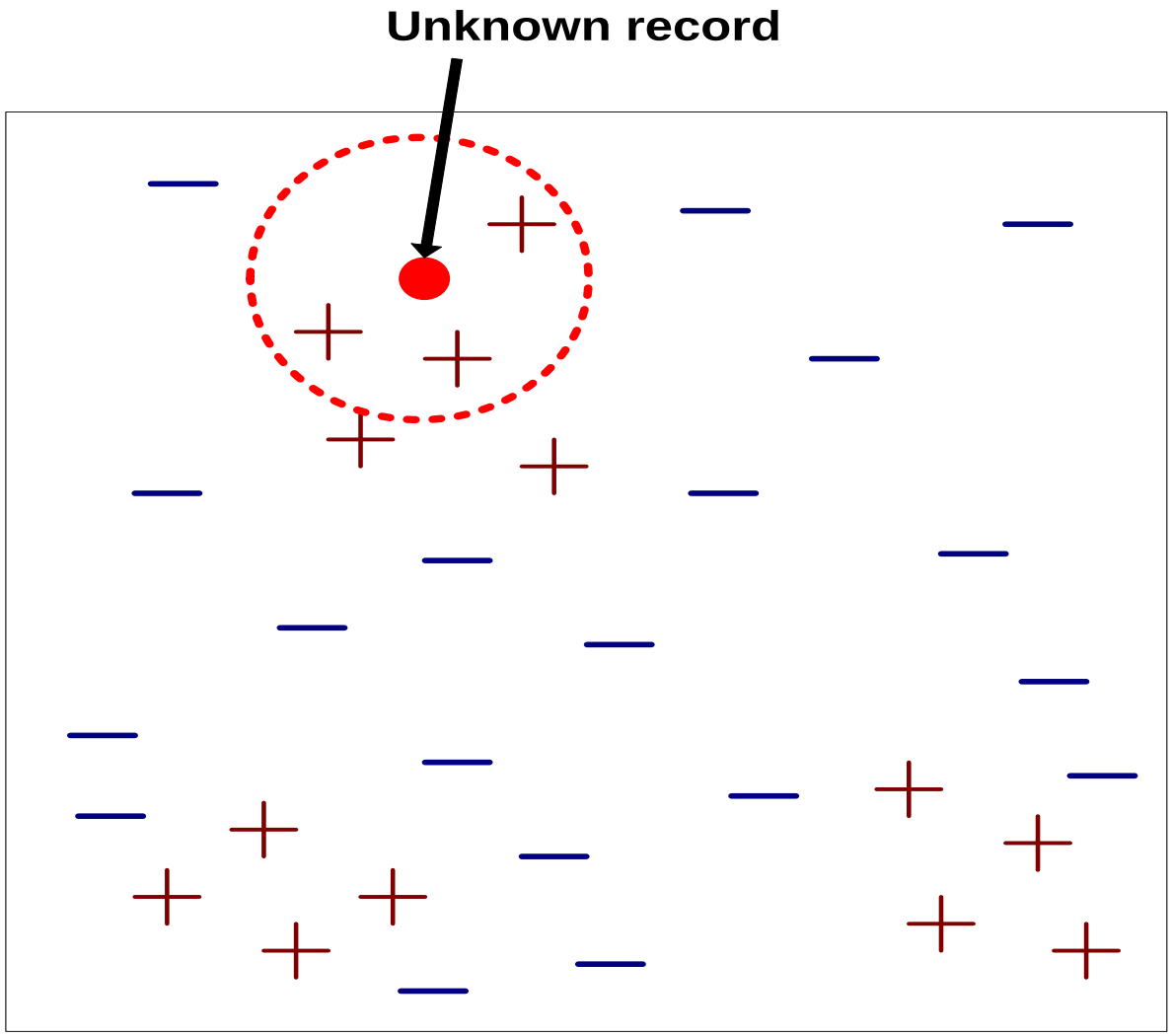
K-Nearest Neighbor Model

K-Nearest Neighbor Model

- Given a test point $\underline{x}'$, compute the distance between $\underline{x}'$ and each input data point

- Find the closest neighbors in the training data
  - For classification, Assign $\underline{x}'$ the class label of the most frequent neighbor
  - For Regression, assign calculate the average of neighbors

KNN Model



K-Nearest Neighbor Model

- If Euclidean distance is used as the distance measure (the most common choice)
  - The nearest neighbor classifier results in piecewise linear decision boundaries
  - The nearest neighbor regressor results in fluctuating time series curve

Parameter Learning for kNN

- There is no objective function like other ML models.
- Normalized training data is just remembered by model
- k(tunable parameter) can be discovered using validation data with cross-validation techniques

The kNN Classifier

- The kNN classifier often works very well.
- Easy to implement.
- Easy choice if characteristics of your problem are unknown.

- Can be sensitive to the choice of distance metric.
  - Often normalize feature axis values, e.g., z-score or [0, 1]
    - E.g., if one feature runs larger in magnitude than another
  - Categorical feature axes are difficult, e.g., Color as Red/Blue/Green
    - Maybe use the absolute differences of their wavelengths?
    - But what about French/Italian/Thai/Burger?
    - Often used: delta(A,B) = {IF (A=B) THEN 0 ELSE 1}

- Can encounter problems with sparse training data.

- Can encounter problems in very high dimensional spaces.
  - Most points are corners.
  - Most points are at the edge of the space.
  - Most points are neighbors of most other points.

Picking K (and $w_1, \ldots, w_d$)

- Given training examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
- Use N fold cross validation
  - Search over K = (1,2,3,…,*Kmax*). Choose search size *Kmax* based on compute constraints
  - Calculated the average error for each K:
    - Calculate predicted class $\hat{y}_i$ for each training point

      $$(\mathbf{x}_i, y_i), \quad i = 1, \ldots, N$$

      (using all other points to build the model)
    - Average over all training examples
- Pick K to minimize the cross validation error