Classification

CS 580
Intro to AI

A new problem

What if the function we want to learn outputs **labels** instead of numbers?

Sum of squared errors is no longer appropriate

ϕ()

h() = “Cat”

h(<5.1, 3.5, 1.4, 0.2>) = “*Iris setosa*”

A new loss function

Since we can assign a set of discrete labels to a set of numbers, why can't we just still use regression?

Sum of squared errors implies that **distance** from the correct answer is important

For most classification problems there is no obvious distance between class labels

Misclassification Loss:

$$\mathcal{J}_S(h) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left[h(\mathbf{x}^{(i)}) \neq y^{(i)}\right]$$

A simple non-parametric classifier

k-Nearest Neighbors

h(x) = “Out of the *k* closest training points return the class with the most votes”

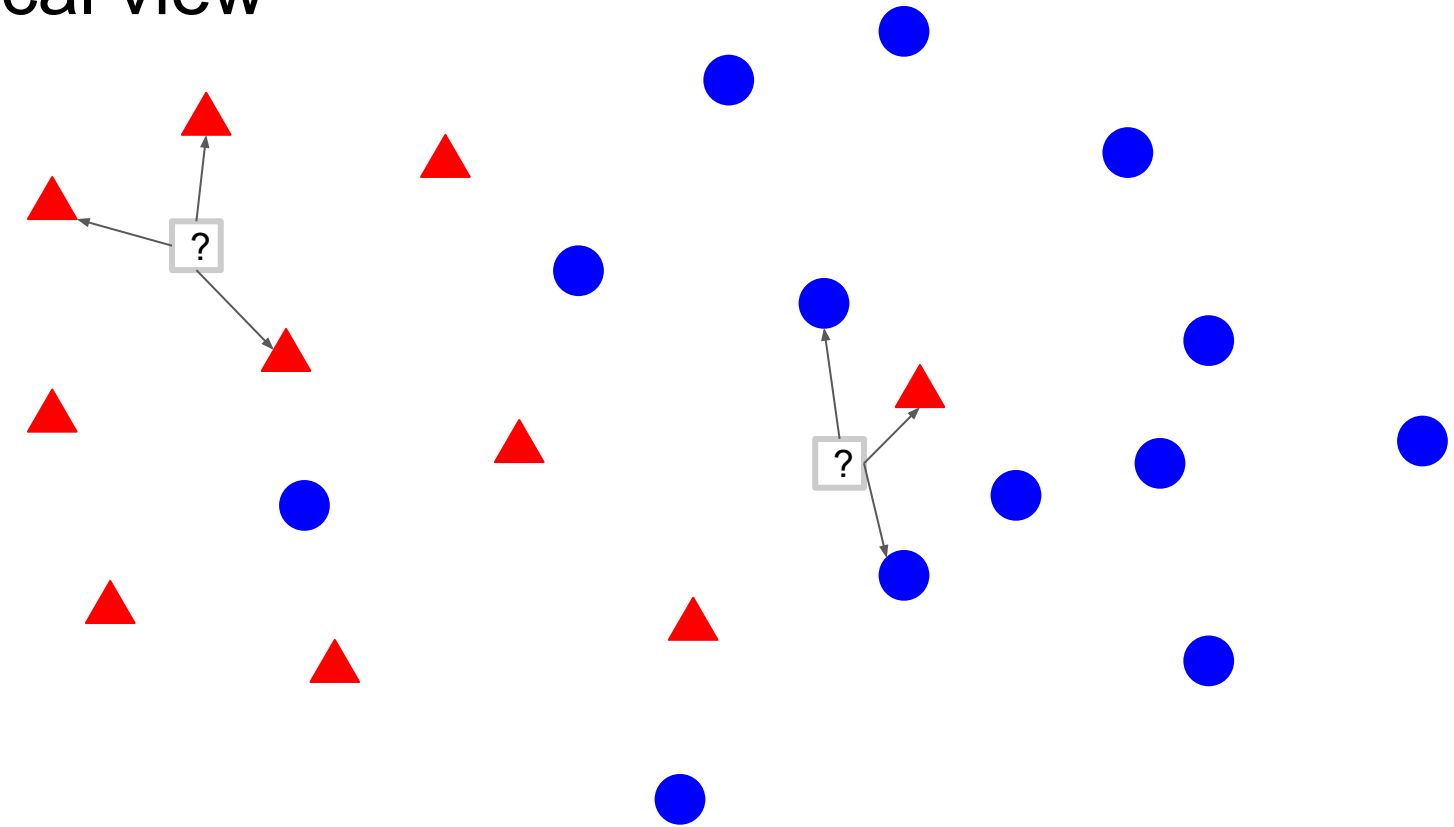
$$\mathcal{H} = \left\{ h_{k,d} : h_{k,d}(\mathbf{x}) = \arg\max_{c \in \mathcal{Y}} \sum_{n_i \in N_k(\mathbf{x}, S, d)} \mathbb{I}\left(y^{(n_i)} = c\right) \right\}$$

$$N_k(\mathbf{x}, S, d) = \{n_i\}_{i=1}^{k}, \quad \text{such that} \quad d(\mathbf{x}, \mathbf{x}^{(n_1)}) \leq d(\mathbf{x}, \mathbf{x}^{(n_2)}) \leq d(\mathbf{x}, \mathbf{x}^{(n_3)}) \leq \dots$$

Probabilistic interpretation

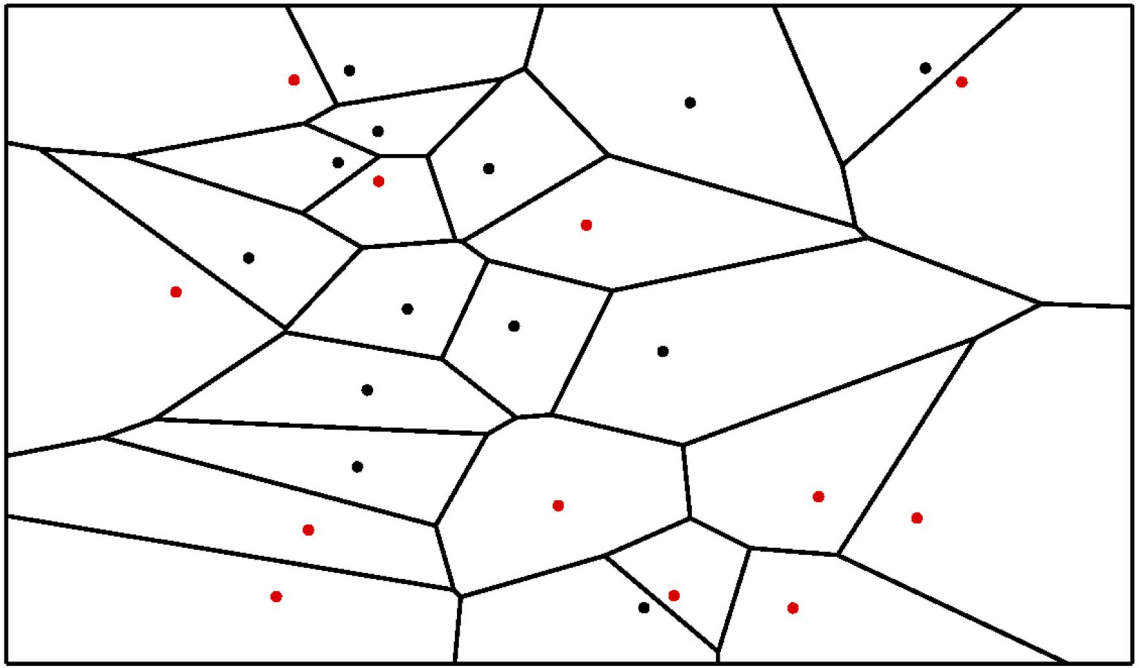
$$p(y = c \mid \mathbf{x}, S, h_{k,d}) = \frac{1}{k} \sum_{n_i \in N_k(\mathbf{x}, S, d)} \mathbb{I}\left(y^{(n_i)} = c\right)$$

Graphical view

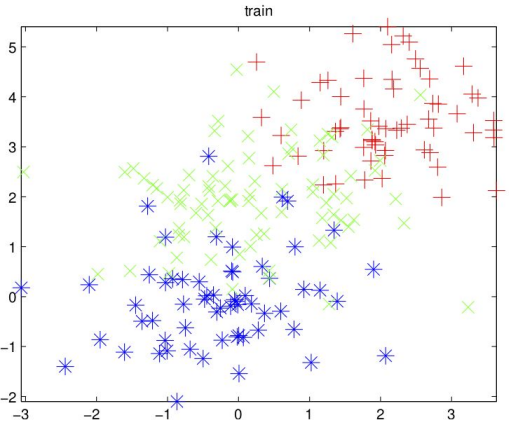


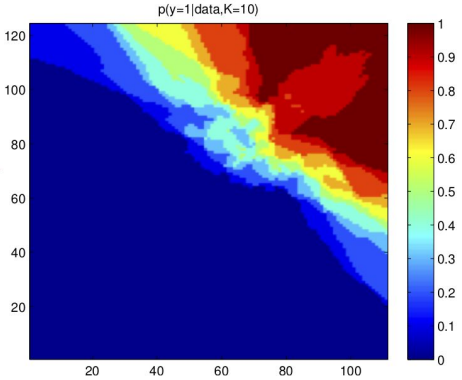
Voronoi tessellation

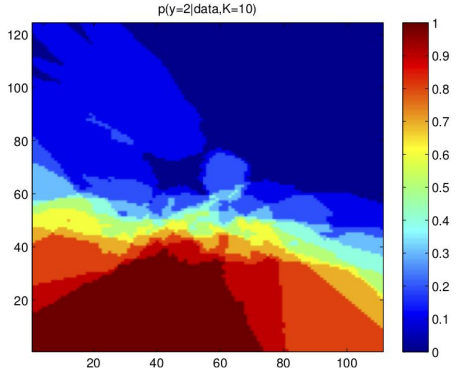
kNN with k=1 splits the input space into cells around the training data, **inducing** a Voronoi tessellation

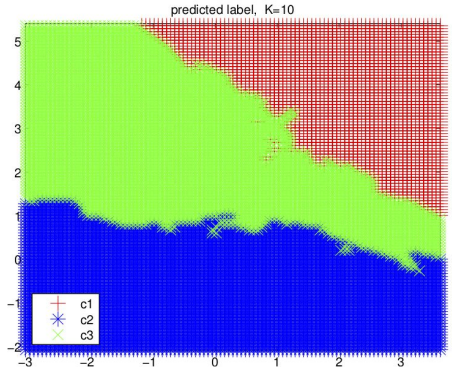


Synthetic example









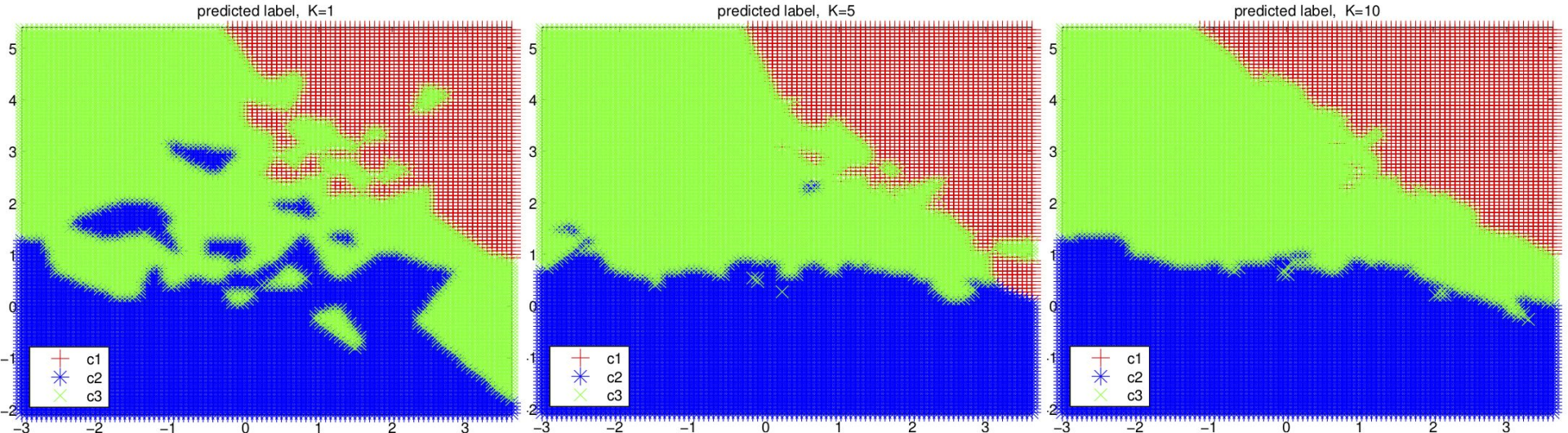
Choosing k

Different choices for k lead to different hypotheses:

Larger k: “Smoother” boundaries, more training error

Smaller k: “Complex” boundaries, less training error

This is the bias-variance tradeoff again!



Choosing the distance function

We can also use different functions for distance

$$d_{\text{euc}}(\mathbf{x}, \mathbf{x}') = \sqrt{\sum_{j=1}^{D}(x_j - x'_j)^2} = \left((\mathbf{x} - \mathbf{x}')^\top(\mathbf{x} - \mathbf{x}')\right)^{\frac{1}{2}}$$

$$d_{\text{taxi}}(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{D}|x_j - x'_j|$$

$$d_{\max}(\mathbf{x}, \mathbf{x}') = \max_d(|x_d - x'_d|)$$

$$d_p(\mathbf{x}, \mathbf{x}') = \left(\sum_{j=1}^{D}|x_j - x'_j|^p\right)^{\frac{1}{p}}$$

Mahalanobis distance

What if the features of your data have vastly different scales?

Solution: Center and normalize

$$\tilde{\mathbf{x}} = \mathbf{x} - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}^{(i)}$$

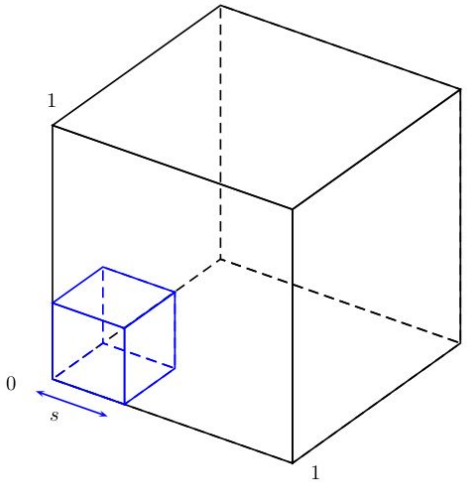
$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & \cdots & \sigma_D \end{bmatrix}$$

Notice: We could also define a distance for some arbitrary matrix here...

$$d_\Sigma(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')^\top \Sigma^{-1} (\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')}$$

The curse of dimensionality

As the dimensionality of x grows larger, the k-nearest neighbors can get further and further away. The “nearest neighbors” may not be very near at all!

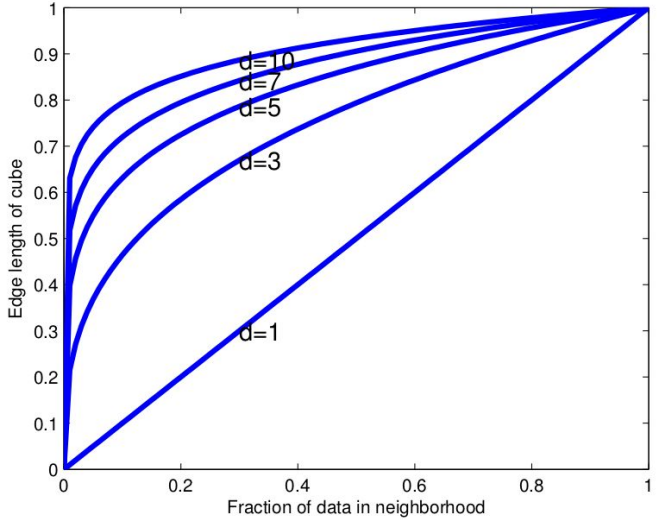


$$f = \frac{V_s}{V_B}$$

$$f = \frac{e^D}{1^D}$$

$$f = e^D$$

$$e = f^{\frac{1}{D}}$$



Summary and preview

Wrapping up

- Classification problems use a **different loss** from regression problems
- kNN is a **non-parametric** classifier that returns the most common class of nearby points
- Both **number of neighbors** and the **distance function** determine the hypothesis class and its complexity

Preview

- Decision Trees and Random Forests