K Nearest Neighbor

Fraida Fund

Contents

In this lecture

- Parametric vs. non-parametric models
- Nearest neighbor
- Model choices
- The Curse of Dimensionality
- Bias and variance of KNN

Parametric vs. non-parametric models

So far…

All of our models have looked like

$$\hat{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

Flexible models

A model is more flexible if it can produce more possibilities for $f(\mathbf{x})$.

- Some possibilities for $f(x_1, \dots, x_p) = \beta_0 + \beta_1 x_1$
- More possibilities for $f(x_1, \dots, x_p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
- Even more possibilities for $f(x_1, \dots, x_p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$

A way to get more flexible models is with a **non-parametric** approach.

Parametric models

- A particular model class is assumed (e.g. linear)
- Number of parameters fixed in advance

Non-parametric models

- Minimal assumptions about model class
- Model structure determined by data

Nearest neighbor

- A kind of non-parametric model.
- Basic idea: Find labeled samples that are “similar” to the new sample, and use their labels to make prediction for the new sample.

1-NN

- Given training data $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ and a new sample $\mathbf{x}_0$
- Find the sample in the training data $\mathbf{x}_{i'}$ with the least distance to $\mathbf{x}_0$.

$$i' = \operatorname*{argmin}_{i=1,\dots,N} d(\mathbf{x}_i, \mathbf{x}_0)$$

- Let $\hat{y}_0 = y_{i'}$

1NN - decision boundaries

K nearest neighbors

Instead of 1 closest sample, we find K closest samples.

KNN for classification (1)

Idea: Estimate conditional probability for class as fraction of points among neighbors with that class label.

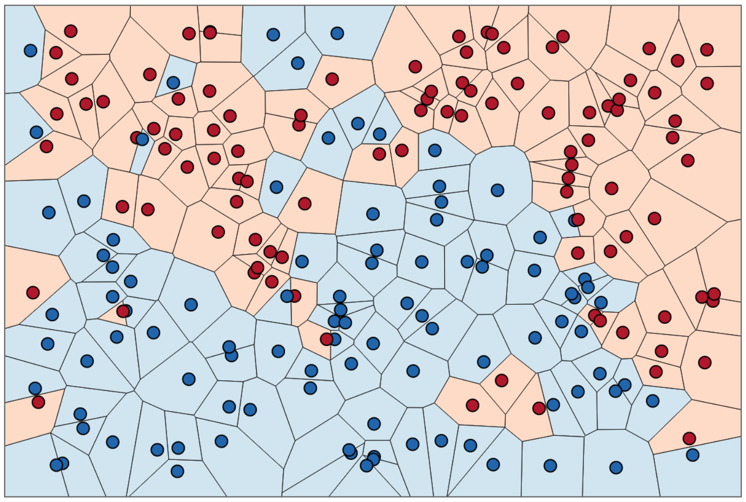

Figure 1: 1NN decision boundaries - Nearest neighborhoods for each point of the training data set. Note that there will be zero error on training set!

KNN for classification (Illustration)

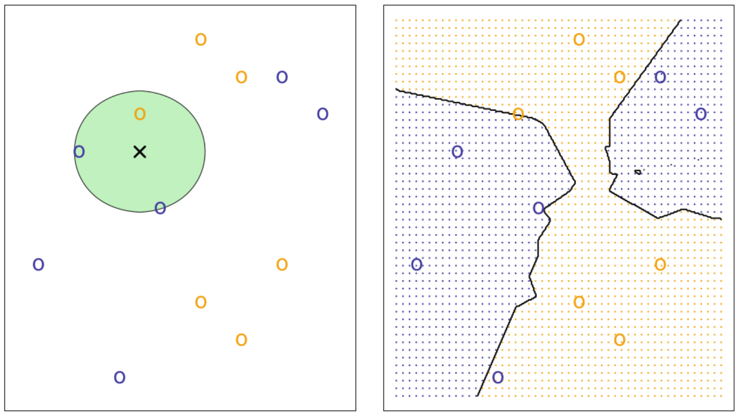

Figure 2: 3-NN classification. Image from ISLR.

KNN for classification (2)

Let N_0 be the set of K points in the training data that are closest to $\mathbf{x}_0$. Then, for each class $m \in M$, we can estimate the conditional probability

$$P(y = m|\mathbf{x_0}) = \frac{1}{K} \sum_{(\mathbf{x}_i, y_i) \in N_0} I(y_i = k)$$

where $I(y_i = m)$ is an indicator variable that evaluates to 1 if for a given observation $(\mathbf{x}_i, y_i) \in N_0$ is a member of class m, and 0 otherwise.

KNN for classification (3)

- We can then select the class with the highest probability.

- Practically: select the most frequent class among the neighbors.

KNN for regression

Idea: Use the the combined label of the K nearest neighbors.

Let N_0 be the set of K points in the training data that are closest to $\mathbf{x}_0$. Then,

$$\hat{y}_0 = \frac{1}{K} \sum_{(\mathbf{x}_i, y_i) \in N_0} y_i$$

Model choices

- What value of K?
- What distance measure?
- How to combine K labels into prediction?

What value of K?

- K must be specified, can be selected by CV. (In general: larger K, less complex model.)
- Often cited "rule of thumb": use $K = \sqrt{N}$
- Alternative: Radius-based neighbor learning, where fixed radius r is specified, can be selected by CV. (Number of neighbors depends on local density of points.)

What value of K? Illustration (1NN)

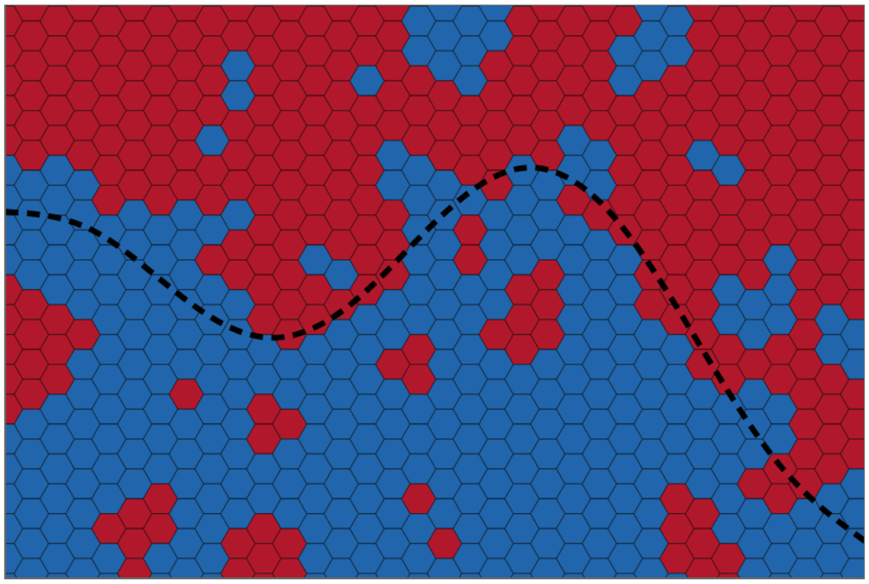

Figure 3: 1NN

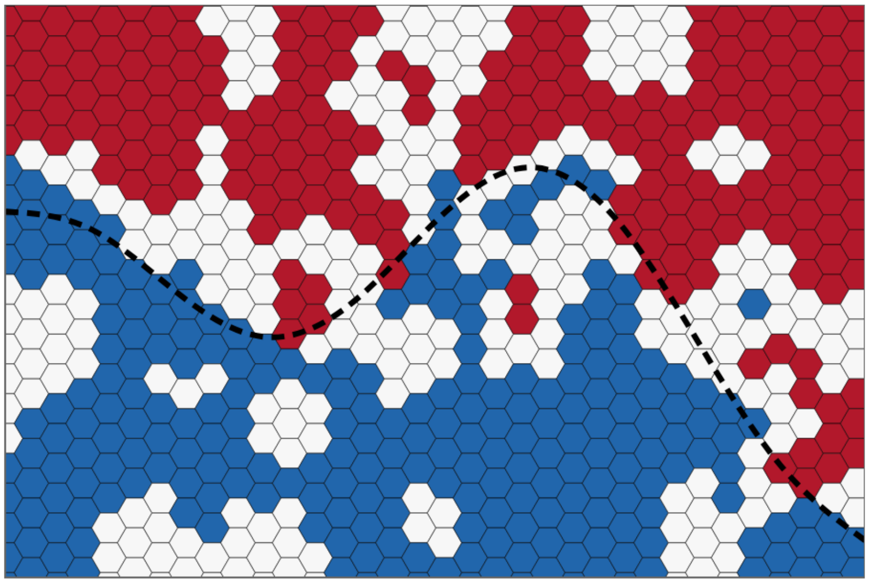

Figure 4: 2NN

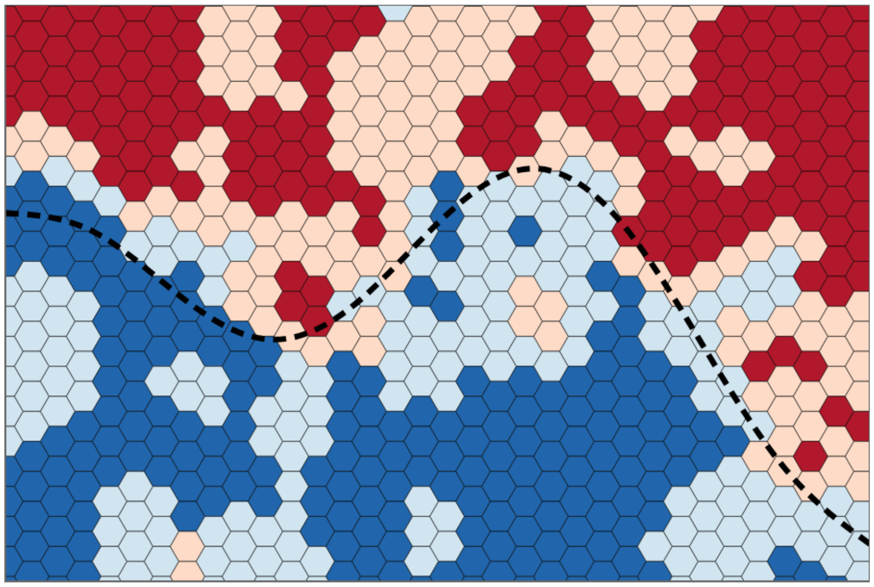

Figure 5: 3NN

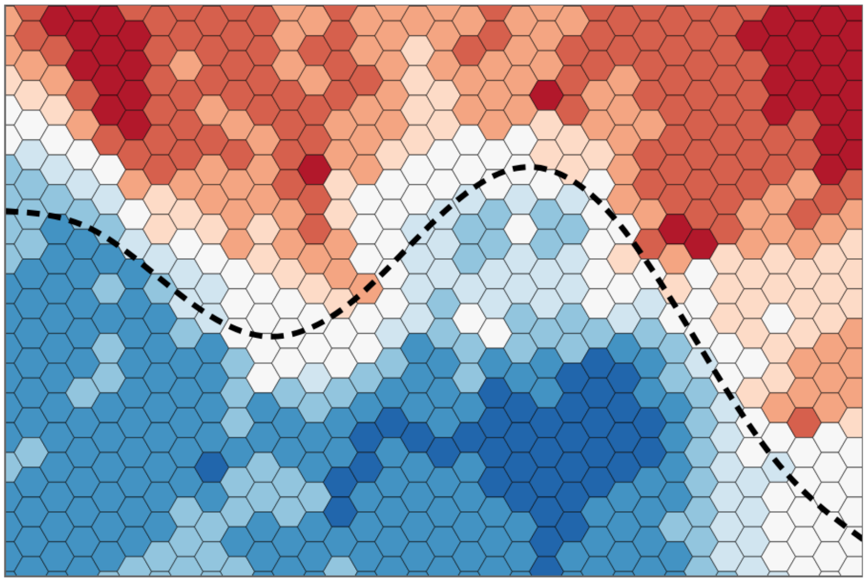

Figure 6: 9NN

What value of K? Illustration (2NN)

What value of K? Illustration (3NN)

What value of K? Illustration (9NN)

What distance measure? (1)

- Euclidean (L2): $\sqrt{\sum_{i=1}^{k}(a_i - b_i)^2}$
- Manhattan (L1): $\sum_{i=1}^{k}|a_i - b_i|$
- Minkowski (generalization of both): $\left(\sum_{i=1}^{k}(|a_i - b_i|)^q\right)^{\frac{1}{q}}$

(L2 distance prefers many medium disagreements to one big one.)

What distance measure? (2)

- Feature weight depends on scale
- KNN implicitly weights all features equally

Standardization (1)

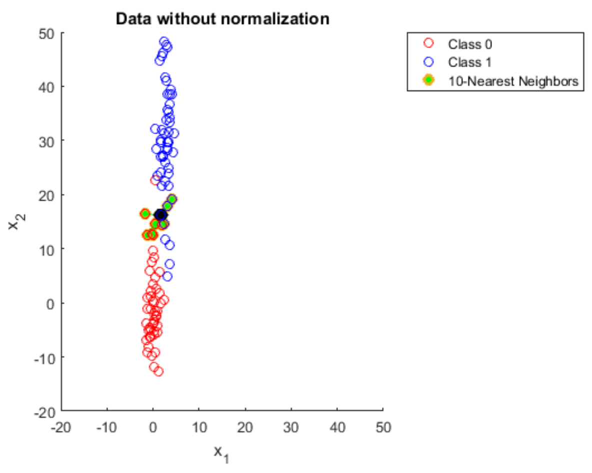

Figure 7: Without standardization, via https://stats.stackexchange.com/a/287439/

Standardization (2)

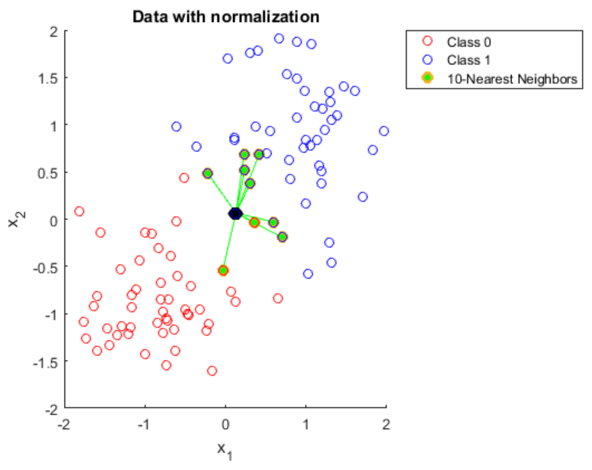

Figure 8: With standardization, via https://stats.stackexchange.com/a/287439/

What distance measure? (3)

- Alternative to equal weighted features: assign feature weights

$$d(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{k} (w_i |a_i - b_i|)^q \right)^{\frac{1}{q}}$$

How to combine labels into prediction?

- Basic voting: use mode of neighbors for classification, mean or median for regression.
- Distance-weighted: weight of vote inversely proportional to distance from the query point. ("More similar" training points count more.)

Nearest neighbor in `sklearn`

- `KNeighborsClassifier`
- `KNeighborsRegressor`

The Curse of Dimensionality

Space grows exponentially with dimension

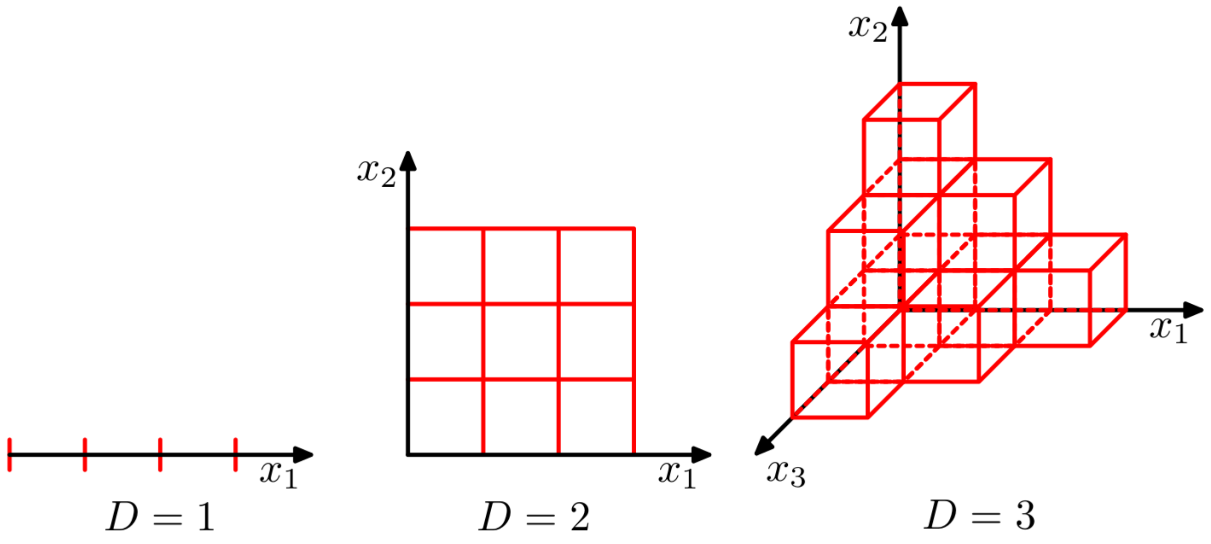

Figure 9: Number of cells grows exponentially with dimension. From Bishop PRML, Fig. 1-21

KNN in 1D

- Consider a dataset $(x_1, y_1), \dots, (x_N, y_N)$ with N=100
- x takes on values in the range [0,1] with uniform distribution
- On average, one data point is located every 1/100 units along 1D feature axis.
- To find 3NN, would expect to cover 3/100 of the feature axis.

KNN in 2D

- Now consider a dataset with two features: $([x_{1,1}, x_{1,2}], y_1), \dots, ([x_{N,1}, x_{N,2}], y_N)$ with N=100
- Each feature takes on values in the range [0,1] with uniform distribution
- To find 3NN, would expect to cover $0.03^{\frac{1}{2}}$ of the unit rectangle.

Density of samples decreases with dimensions

To get 3NN,

- need to cover 3% of space in 1D
- need to cover 17% of space in 2D
- need to cover 70% of space in 10D. At this point, the nearest neighbors are not much closer than the rest of the dataset.

Density of samples decreases with dimensions - general

The length of the smallest hyper-cube that contains all K-nearest neighbors of a test point:

$$\left(\frac{K}{N}\right)^{\frac{1}{d}}$$

for N samples with dimensionality d.

Density of samples decreases with dimensions - illustration

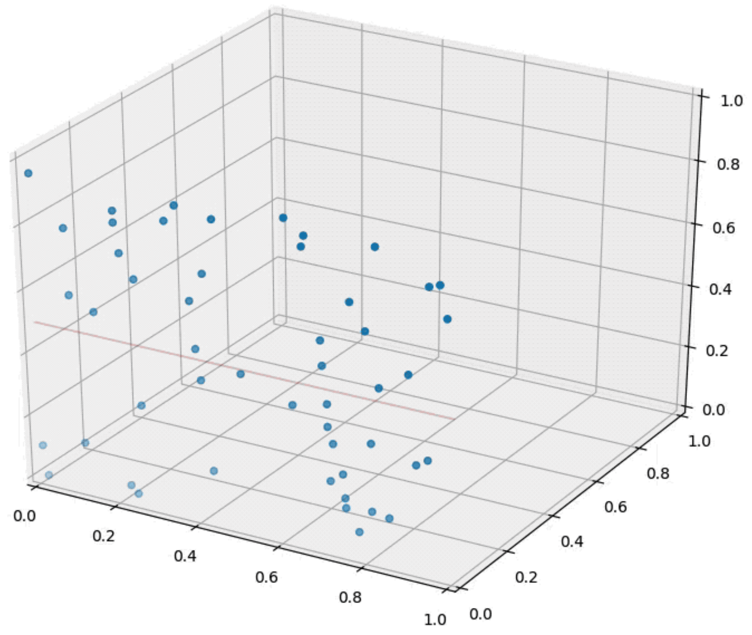

Figure 10: Image source: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote02_kNN.html

Solutions to the curse (1)

Add training data?

$$\left(\frac{K}{N}\right)^{\frac{1}{d}}$$

As number of dimensions increases linearly, number of training samples must increase exponentially to counter the "curse".

Solutions to the curse (2)

Reduce d?

- Feature selection (and in previous lab)
- Dimensionality reduction: a type of unsupervised learning that *transforms* high-d data into lower-d data.

Bias and variance of KNN

True function

Suppose data has true relation

$$y = f_0(\mathbf{x}) + \epsilon, \quad \epsilon \sim N(0, \sigma_\epsilon^2)$$

and our model predicts $\hat{y} = f(\mathbf{x})$.

Expected loss

We will use least square loss function, so that the expected error at a given test point $\mathbf{x}_{test}$ is:

$$MSE_y(\mathbf{x}_{test}) := E[y - \hat{y}]^2$$

$$\begin{aligned} &= E[f_0(\mathbf{x}_{test}) + \epsilon_0 - f(\mathbf{x}_{test})]^2 \\ &= E[f_0(\mathbf{x}_{test}) - f(\mathbf{x}_{test})]^2 + \sigma_\epsilon^2 \end{aligned}$$

which we know can be further decomposed into squared bias, variance, and irreducible error.

KNN output

The output of the KNN algorithm at a test point is

$$f(\mathbf{x}_{test}) = \frac{1}{K} \sum_{\ell \in K_x} f_0(\mathbf{x}_\ell) + \epsilon_\ell$$

where K_x is the set of K nearest neighbors of $\mathbf{x}_{test}$. (We assume that these neighbors are fixed.)

Bias of KNN

When we take the expectation of bias over all test samples, the ϵ values disappear because it has zero mean, so squared bias becomes

$$\begin{aligned} Bias^2 &= \left(f_0(\mathbf{x}_{test}) - E[f(\mathbf{x}_{test})]\right)^2 \\ &= \left(f_0(\mathbf{x}_{test}) - E\left(\frac{1}{K} \sum_{\ell \in K_x} f_0(\mathbf{x}_\ell) + \epsilon_\ell\right)\right)^2 \\ &= \left(f_0(\mathbf{x}_{test}) - \frac{1}{K} \sum_{\ell \in K_x} f_0(\mathbf{x}_\ell)\right)^2 \end{aligned}$$

(We also rely on the assumption that neighbors are fixed.)

Variance of KNN (1)

$$\begin{aligned}
Var(\hat{y}) &= Var\left(\frac{1}{K}\sum_{\ell \in K_x} f_0(\mathbf{x}_\ell) + \epsilon_\ell\right) \\
&= \frac{1}{K^2}\sum_{\ell \in K_x} Var\left(f_0(x_\ell) + \epsilon_\ell\right) \\
&= \frac{1}{K^2}\sum_{\ell \in K_x} Var\left(f_0(x_\ell)\right) + Var\left(\epsilon_\ell\right)
\end{aligned}$$

Variance of KNN (2)

$$\begin{aligned}
&= \frac{1}{K^2}\sum_{\ell \in K_x} Var\left(\epsilon_\ell\right) \\
&= \frac{1}{K^2} K \sigma_\epsilon^2 \\
&= \frac{\sigma_\epsilon^2}{K}
\end{aligned}$$

where $Var(f_0(x_\ell)) = 0$ only if we assume that the neighbors are fixed.

MSE of KNN

Then the MSE of KNN is

$$MSE(\mathbf{x}_{test}) = \left(f_0(\mathbf{x}_{test}) - \frac{1}{K}\sum_{\ell \in K_x} f_0(\mathbf{x}_\ell)\right)^2 + \frac{\sigma_\epsilon^2}{K} + \sigma_\epsilon^2$$

where K_x is the set of K nearest neighbors of $\mathbf{x}_{test}$.

Bias variance tradeoff

- Variance decreases with K
- Bias likely to increase with K, if function $f_0()$ is smooth. (Few closest neighbrs to test point will have similar values, so average will be close to $f_0(\mathbf{x})$; as K increases, neighbors are further way, and average of neighbors moves away from $f_0(\mathbf{x})$.)

Bias variance tradeoff - selected problems

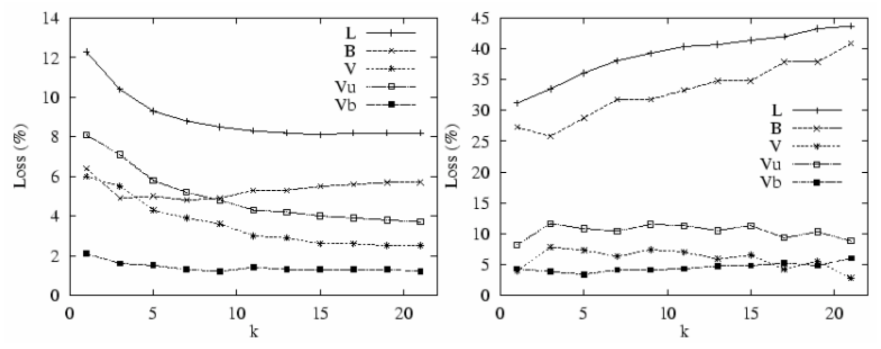

Figure 11: Via Domingos 2000.

Summary of NN method

NN learning

Learning:

- Store training data
- Don't do anything else until you have a new point to classify

NN prediction

Prediction:

- Find nearest neighbors using distance metric
- Classification: predict most frequently occuring class among nearest neighbors
- Regression: predict mean value of nearest neighbors

The good and the bad (1)

Good:

- Good interpretability
- Fast "learning" (*memory-based*)
- Works well in low dimensions for complex decision surfaces

The good and the bad (2)

Neutral:

- Assumes similar inputs have similar outputs

The good and the bad (3)

Bad:

- Slow prediction (especially with large N)
- Curse of dimensionality

Illustration

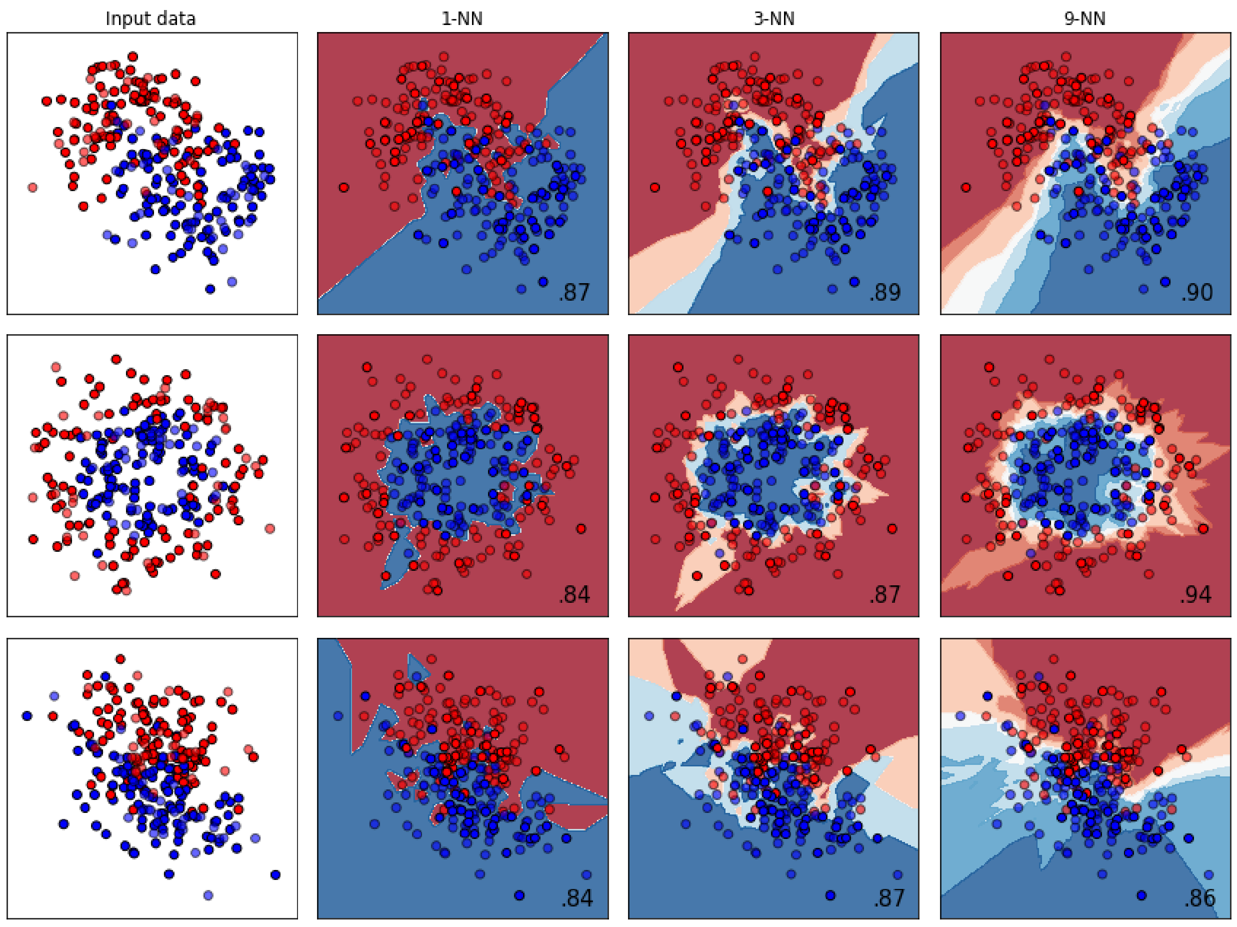

Figure 12: 1NN, 3NN, 9NN comparison on three types of data, with accuracy on test set shown in the corner.