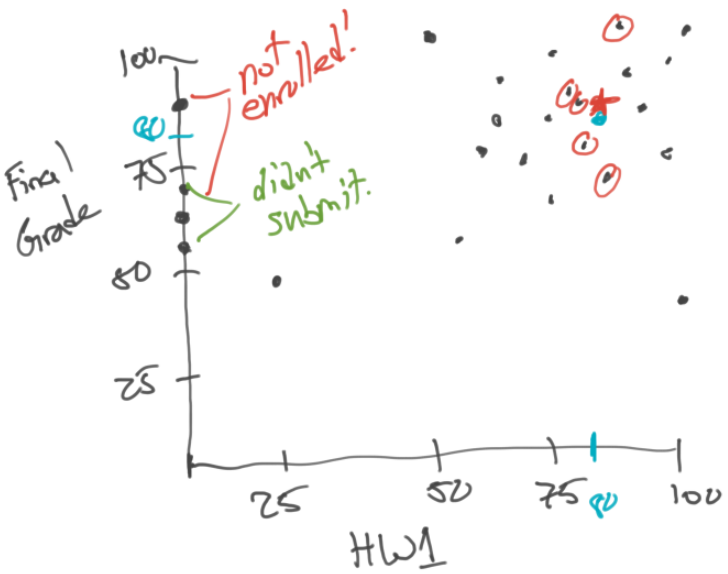


Lecture 2: Supervised Learning Part 1

- k-Nearest Neighbor
- Linear regression
- Optimization perspective of regression

Data: (row = 1 data point)

Person number	HW1	Final Grade
1	83	95
2	94	87
3	71	62
⋮	⋮	⋮
~~80~~	77	89

n = number of points. n = 80.

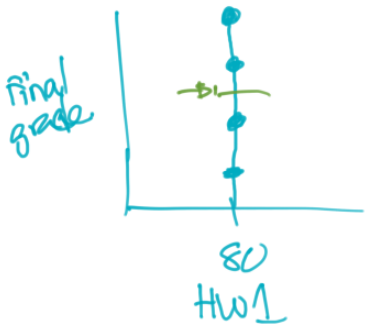
Input: 80

Output: 80

Idea: Find the point with closest HW1 score (break ties randomly), predict same final score.

Pros
- Simple
- sometimes all you need!
- Efficient
  - $O(\log(n))$ average
    $O(n)$ worst case.
    - KD-Tree.

Cons
- Not always accurate, even with tons of data.



- Improvements
  - ☆ 1) Average values for all points tied for "closest"
    - What if inputs are continuous.
  - 2) Average the values for the k nearest points.
    - k is called a hyperparameter.
    - Called k-nearest neighbor
      k-NN.

Hyperparameter: a parameter whose value is used to control the learning process

3) Assign a weight to the nearest neighbors based on their distance, and use a weighted average.
   - weighted k-NN.

Notation: Training data is n input-label pairs: ~~Dataset~~ "Data set"

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

input, label

Shorthand: $(x_i, y_i)_{i=1}^{n}$

Each input is a vector (array) of m features.

$$x_i = (x_{i,1}, x_{i,2}, x_{i,3}, \dots, x_{i,m}), \quad x_{i,j} \in \mathbb{R}$$

i: which point; j: feature number.

$x_{i,j}$ is real numbers.

$$x_i \in \mathbb{R} \times \mathbb{R} \times \dots \times \mathbb{R}$$

$$x_i \in \mathbb{R}^m$$

individual measurable property or characteristic of a phenomenon being observed.

Each label is a real number.

$$\forall i \in \{1, \ldots, n\}, \; y_i \in \mathbb{R}$$

Let $\hat{y}_i$ be the agent's prediction of y_i from x_i.

"hat" → estimate or approximation of same symbol without the hat.

"Regression problem"

~~weighted~~ K-NN

Input: $(x_i, y_i)_{i=1}^n$, x_{n+1}, $k \in \mathbb{N}_{>0}$, $\sigma \in \mathbb{R}_{>0}$, distance measure "dist"

↑ input to predict label for.

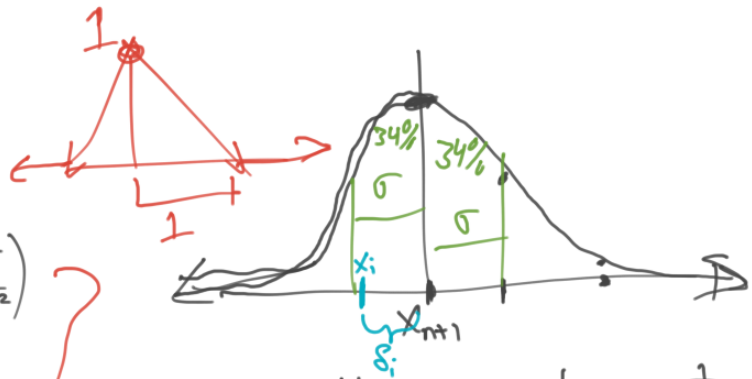
$\mathbb{N}$ = natural numbers. $\mathbb{N}_{>0}$ → don't include 0. $\mathbb{N}_{\geq 0}$ → do include 0.

Output: $\hat{y}_{n+1}$

1) for $i = 1$ to n

$d_i = \text{dist}(x_i, x_{n+1})$

$w_i = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{\delta_i^2}{\sigma^2}\right)$ ← $w_i = 1$



2) Sort $(x_i, y_i, d_i, w_i)_{i=1}^n$ by d_i, smallest to largest.

(KD-Tree! (more efficient).)

3) $k' = \min(n, k)$

4) $c = \sum_{i=1}^{k'} w_i$

5) $\hat{y}_{n+1} = \sum_{i=1}^{k'} \frac{w_i}{c} y_i$

6) Return $\hat{y}_{n+1}$.

Non-weighted.

$w_i = \begin{cases} 1 - \delta_i & \text{if } \delta_i \leq 1 \\ 0 & \text{otherwise.} \end{cases}$

$\mathbb{R}$ $\mathbb{N}$ $\mathbb{R}$

math bold black board.

$\mathbb{R}$ $\mathbb{E}$

How pick hyperparams

~ Art!

k, σ, dist?

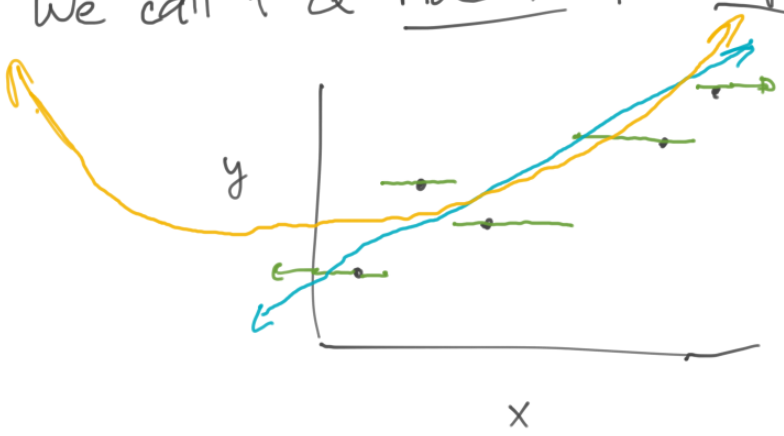
↓

how compute weights from distances?

Let f be the function mapping inputs to label predictions.

$\hat{y}_i = f(x_i)$

We call f a model in supervised learning.



1-NN

linear model $\hat{y} = mx + b$ (slope, y-intercept)

polynomial model $\hat{y} = ax^2 + bx + c$

k-NN is nonparametric. ~ it doesn't assume a particular parametric form for the model.

Parametric model forms:

$\hat{y} = mx + b$

$\hat{y} = ax^2 + bx + c$

(model) parameters: m, b

a, b, c.

↳ often called "weights", $w = (w_1, w_2, \dots, w_{|w|})$

w

$|w|$

↳ 1.1 absolute value

↳ 1.1 cardinality of the set.

↳ length of vector.

Linear Regression

- Assume linear parametric form of the model.

$$f_w(x_i) = w_1 x_{i,1} + w_2 x_{i,2} + \dots + w_m x_{i,m}$$

$$f(x, w) = \sum_{j=1}^{m} w_j x_{i,j}$$

$$= w \cdot x_i \quad \text{(dot product)}$$

$$= \boxed{w^T x_i}$$