KNN - K nearest neighbors

↳ Hyper parameter

→ Non parametric

↳ No weights are trained

(Lazy)

→ Prediction → $x_q \Rightarrow y$ (P_{new})

↳ take k nearest neigh

Underfitting	Overfitting
High bias	High variance
Low variance	Low bias

Data → general pattern + noise

KNN

Noise

Complexity: k → very small ⇒ k=1 → non linear boundary

x_q → [illegible] → Noise

Underfitting: Time k very high → [illegible] / high bias

x_q → [illegible] → Noise

↳ Time Complexity ↓ bias ↑

k ↓ Perfect k (underfitting)

↳ (overfitting) ↳ (underfitting)

Data (overfitting)

- Train (overfitting) → Hyper parameter
- Val (Hyper parameter)
- Test → Unknown data → Prediction
  - Hyper [illegible] published → Prediction
  - ↳ Cannot see

10

↳ (1, 2, 3 ..... 50)

k=1 Train model (Train dataset) → Val set to find

k=2 . . . . . .

k=3 . . . . . .

k=50 . . . . . .

Calculate acc of Train

Calculate acc of Val

k=1 acc = 100%

D tuples (n data points)

d dimensions

Train	Prediction (x_q)
Store the data	to find distance x_q all n points $O(nd)$
Time complexity $O(1)$	to sort in distance $O(n \log n)$
space $O(n \times d)$	to pick top k $O(k)$

k << n

↓ ↓

log n

Prediction time complexity $= O(nd) + O(n \log n) + O(k) \approx O(nd + n \log n)$

$x_1 \; d_1 \; d_2 \; d_3 \cdots d_d$ $\quad x_2 \; d_1 \; d_2 \cdots d_d$

$\sqrt{(d_1 - d_1)^2 + (d_2 - d_2)^2 + (d_3 - d_3)^2 + \cdots (d_d - d_d)^2}$

$O(nd)$ → distance for d dimensions

No distance space complexity $O(n)$

---

features in KNN

Bad group

categorical → encode

4d → 11d

KNN d

d → d + 7777

categorical feature

- Small category indicator → Dummy encoding
- Large category → Target encoding
  - Target
  - Target encoding

Curse of dimensionality

↳ distance introduce if there are very too many dimensions → very small

euclidean dist

d → very high

distance is similar in all points

↳ d ⇒ x_q dimensions

↳ increase time

Manhattan dist

$|x_a - x_b| + |y_a - y_b|$

euclidean

↳ more Better

Manhattan dist

$\left(\sum_{j=1}^{d} |x_i^{(j)} - x_j^{(j)}|\right)^{1/1}$

euclidean dist

$\left(\sum_{j=1}^{d} (x_i^{(j)} - x_j^{(j)})^2\right)^{1/2}$

Generalized

$\left(\sum_{j=1}^{d} (x_i^{(j)} - x_j^{(j)})^p\right)^{1/p}$

minkowski dist (p)

p=1 → manhattan

p=2 → euclidean

---

text/image → embedding

cosine

d →

euclidean | manhattan | cosine

Business use

Outlier / Big data

Cosine → manhattan

↳ euclidean

dot (map)

$\bar{1}, \bar{2}, \bar{3}$ $\quad \bar{1}, \bar{2}, \bar{3}$

x_1 $\quad x_2$

$1 + |1-3|$

$+ (3-5)$

$2 + 2 = 4$

Putting KNN in production

prediction $O(nd + n \log n)$

$\approx n$

LSH (Local sensitivity hashing)

	p_1	p_2	p_3	hash $h(x)$	x_q
x^1	1	10	5	0,1,0	
x^2	-1	10	-10	0,1,0	$\bar{v}$ = tiger ≈ $\bar{v}$
x^3	-20	20	20	0,1,0	
x^4	-100	20	20	1,1,1	$\bar{v} = (\bar{x}^1 + \bar{x}^2 + \bar{x}^3)$

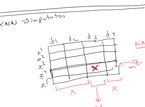
0,1,0 x^1, x^2, x^3

1,1,1 x^4

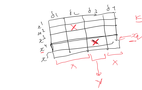
x_q → $h(x)$ → 1,1,1

↳ $\bar{v}$ ↳ 0,1,0

KNN → implementation



x_q x^{4}

K NN → average (continuous)

↳ majority (Category)