CPSC 340/540 Tutorial 2

Winter 2024 Term 1

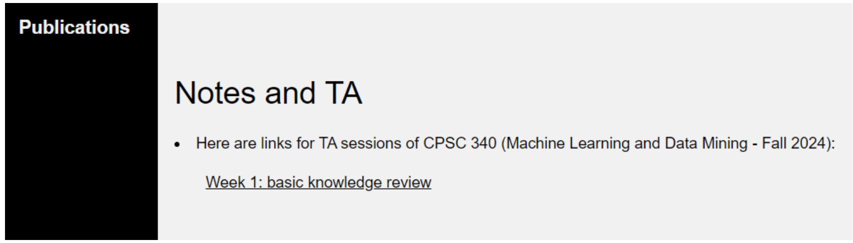
T1A: Tuesday 16:00-17:00;
T1C: Thursday 10:00-11:00;
Office Hour: Wednesday 15:00-16:00

Slides can be found at Piazza and my personal page after T1C.

Yi (Joshua) Ren

https://joshua-ren.github.io/
renyi.joshua@gmail.com
PhD with Danica

Machine Learning:
Learning dynamics, LLM, Compositional Generalization



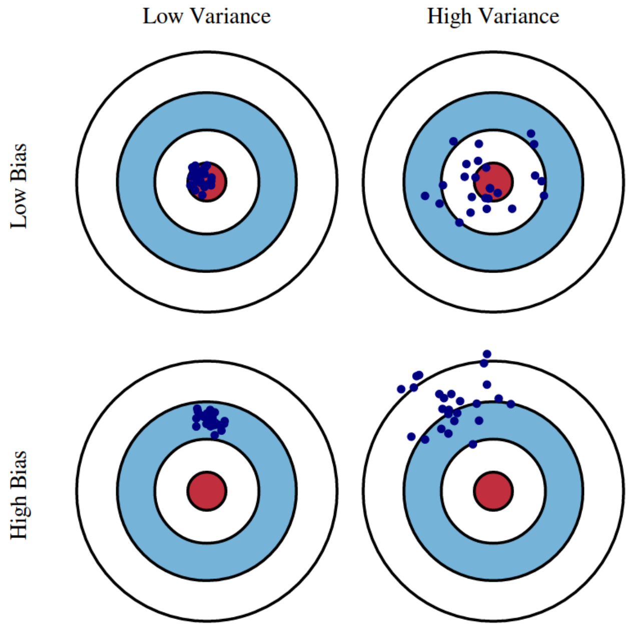
More helpful on theory

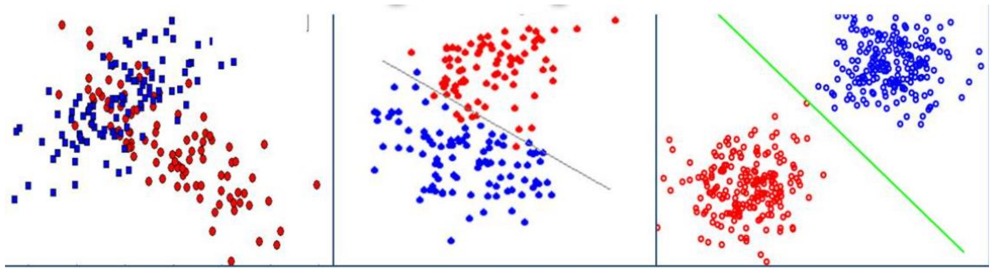
Less helpful on coding

Slides Credit: To various pervious TA’s of this course

- Variance-bias trade-off
- KNN
- Naive Bayes

Variance-bias trade-off (traditional discussion)





The “noise” is becoming smaller.

- Expected squared test error in this setting is

$$\mathbb{E}[(\tilde{y}_i - \hat{y}_i)^2] = \mathbb{E}[(\hat{y}_i - \bar{y}_i)]^2 + (\mathbb{E}[\hat{y}_i^2] - \mathbb{E}[\hat{y}_i]^2) + \sigma^2$$

"test squared error" "bias" "variance" "noise"

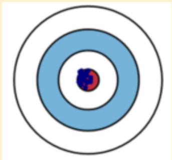
  - Where expectations are taken over possible training sets of ‘n’ examples.
  - Bias is expected error due to having wrong model.
  - Variance is expected error due to sensitivity to the training set.
  - Noise (irreducible error) is the best can hope for given the noise (E_{best}).

- Some learning theory results use E_{best} to further decompose E_{test}:

$$E_{test} = \underbrace{(E_{test} - E_{train})}_{E_{gap}} + \underbrace{(E_{train} - E_{best})}_{E_{model}} + \underbrace{E_{best}}_{\text{"noise"}}$$

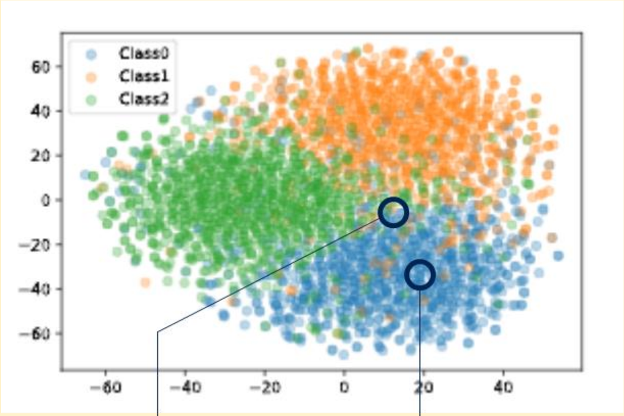
- E_{gap} measures *how sensitive we are to training data.*
- E_{model} measures *if our model is complicated enough to fit data.*
- E_{best} measures how low can **any** model make test error.
  - E_{best} does not depend on what model you choose.

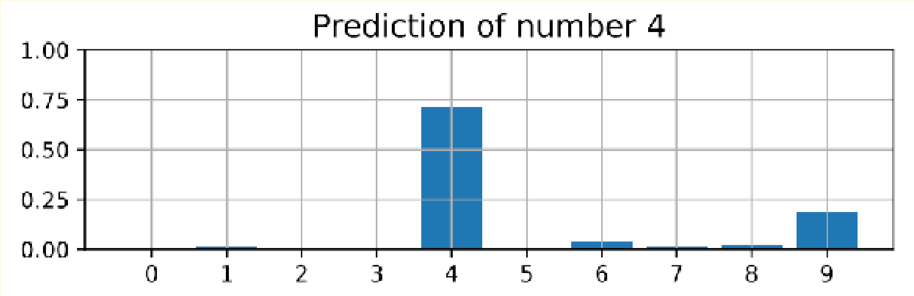
Variance-bias trade-off (but when data is non-seperable)



Not always the best!

- Network's **calibration**: low variance (high confidence) is not always good
  - Fact: most of the time, our model gives a probabilistic prediction, e.g., spam-filter, MNIST, ...
  - Different samples with the same label can be different.
  - Then, we want **confidence** aligns well with **facts**.





$p^*(y|x) \approx [1, 0, 0]$

$p^*(y|x) \approx [1, 0, 0]$?

$p^*(y|x) = [0.98, 0.01, 0.01]$

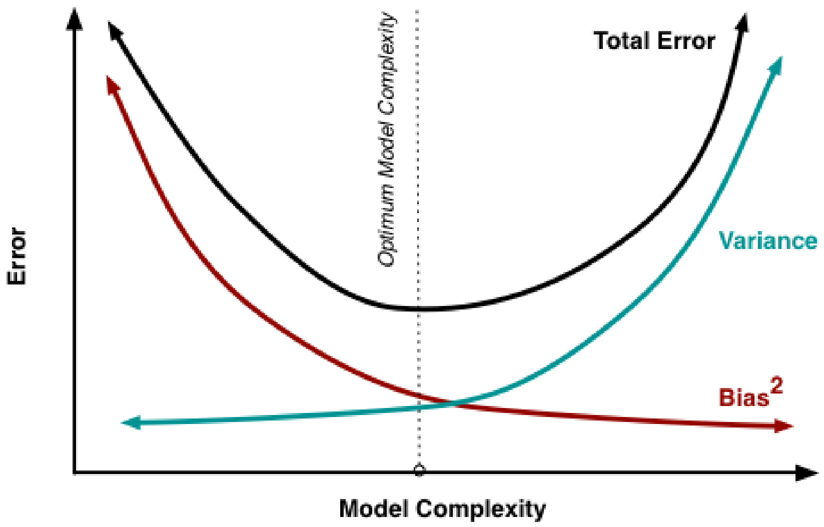
$\mathbf{e}^T_{y_n} = [1, 0, 0]$

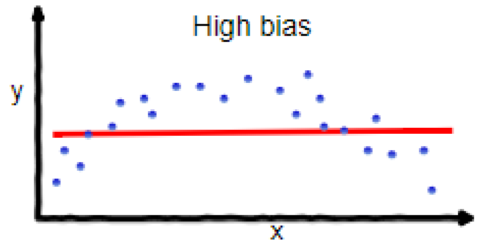
$p^*(y|x) = [0.50, 0.30, 0.20]$

$\mathbf{e}^T_{y_n} = [1, 0, 0]$

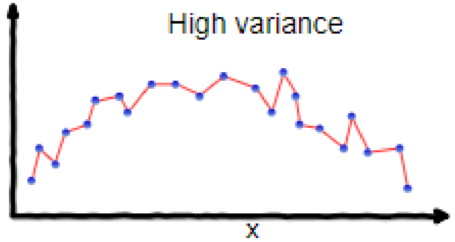
Guo, Chuan, et al. "On calibration of modern neural networks." *ICML*, 2017. (6k+ citations)
Ren, Yi, Shangmin Guo, and Danica J. Sutherland. "Better supervisory signals by observing learning paths." ICLR 2022

Variance-bias trade-off (traditional)

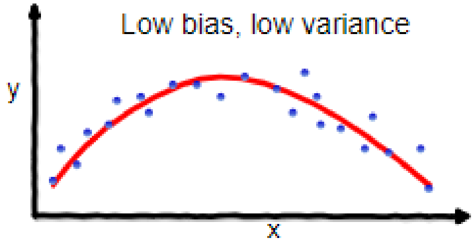




underfitting



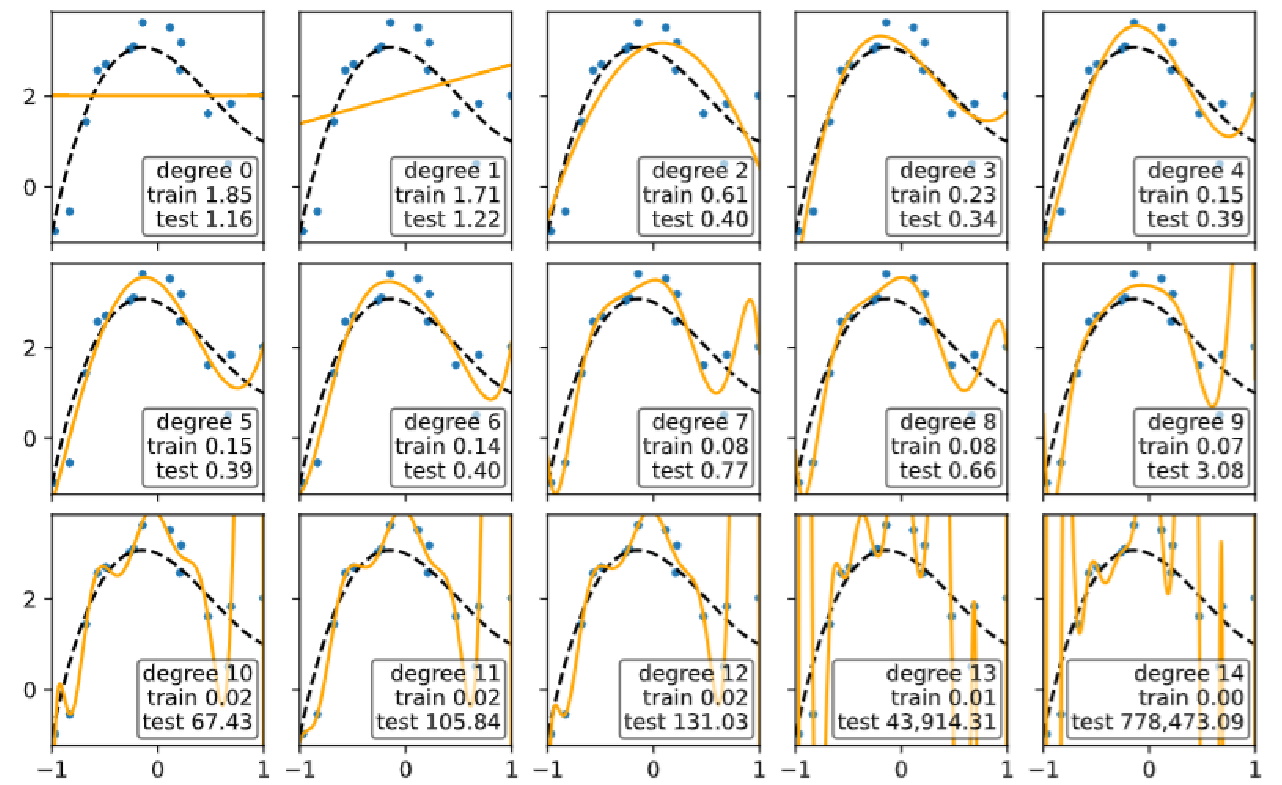
overfitting



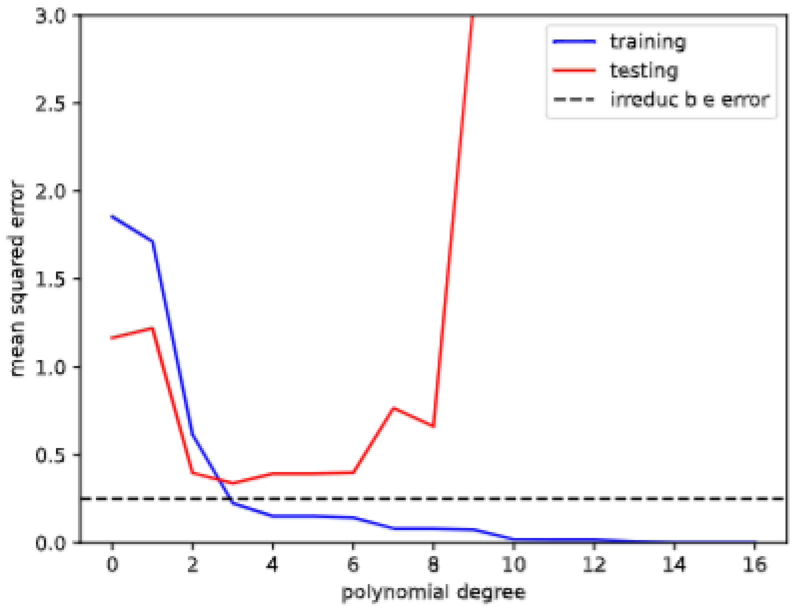
Good balance

What we usually see in textbooks

Variance-bias trade-off (double descent, benign overfitting)



(a) Polynomial regression, $h(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_k x^k$, for increasing k, to data points shown in blue. ERM fits are in orange; dashed black lines show $\mathbb{E}[y \mid x]$, a cubic function. Text gives mean squared error for training and testing sets.



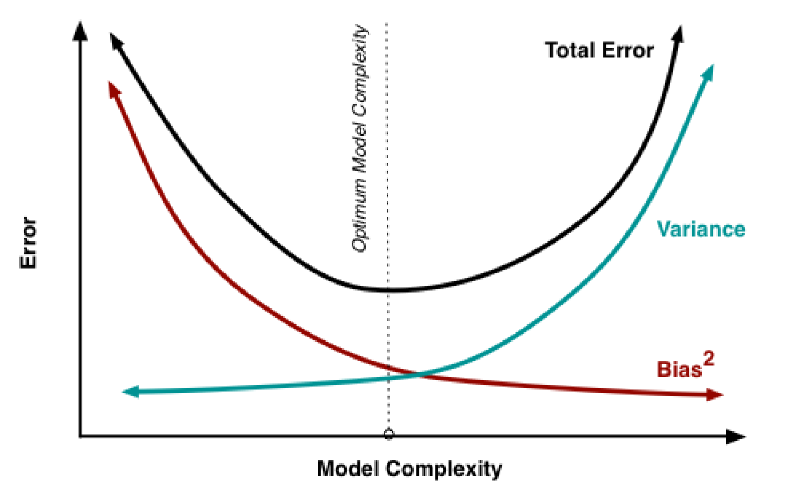
(b) Training and test errors from Figure 1.1a.

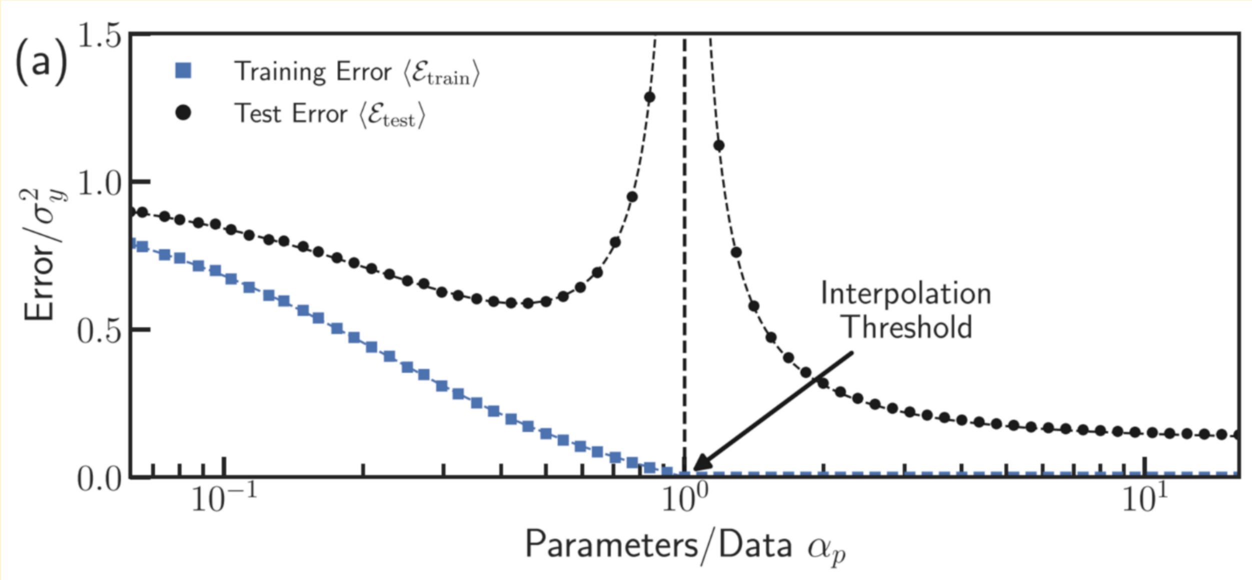
What happens when d=2000?

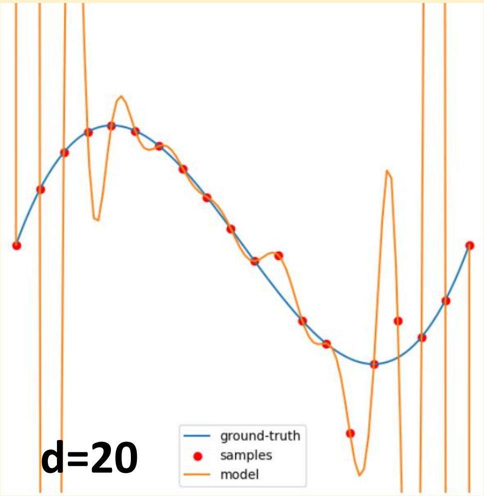
Figures from: https://www.cs.ubc.ca/~dsuth/532D/24w1/notes/1-intro-erm.pdf

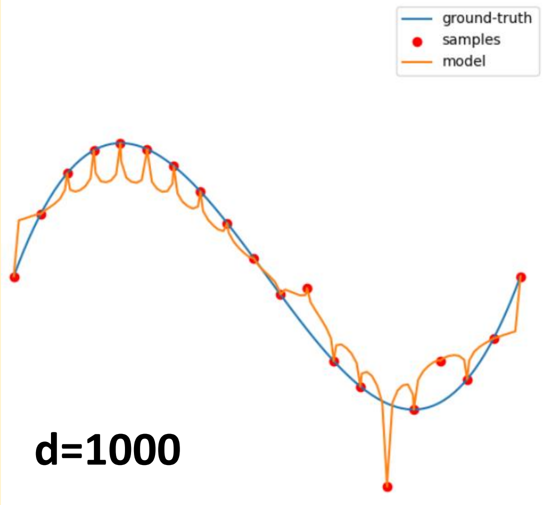
https://colab.research.google.com/drive/1UJYKXj317aJGeIgwV0qqL3MhUnHf9VJK#scrollTo=PrA4y-mEJZZW

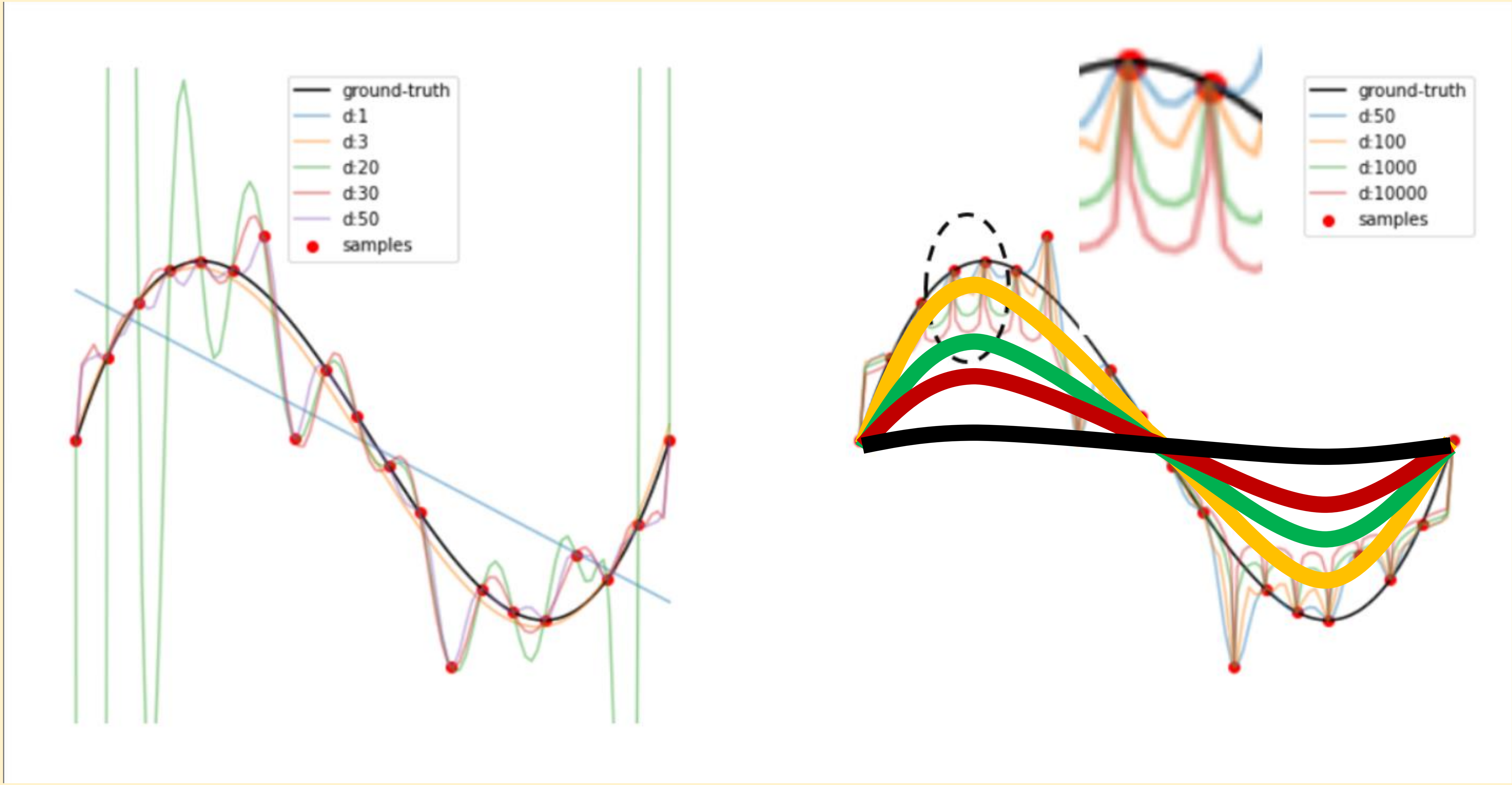
Variance-bias trade-off (double descent (DNN), benign overfitting)







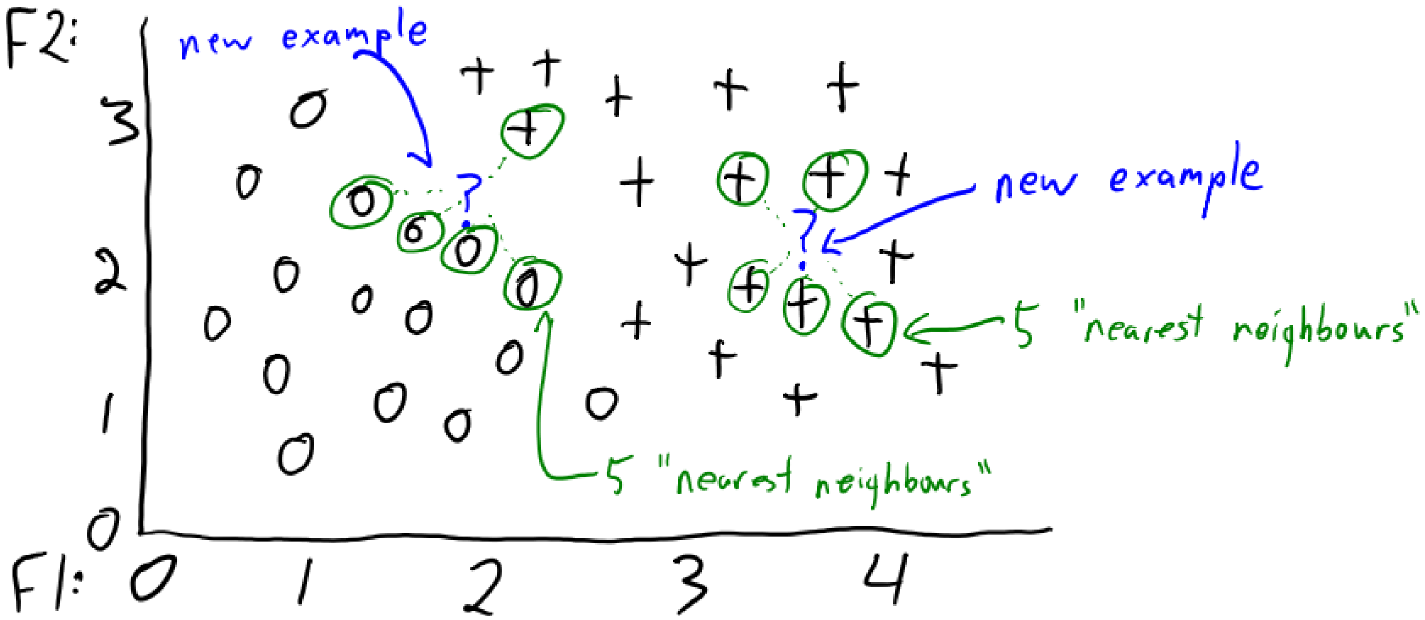



ground-truth
d:1
d:3
d:20
d:30
d:50
samples
ground-truth
d:50
d:100
d:1000
d:10000
samples

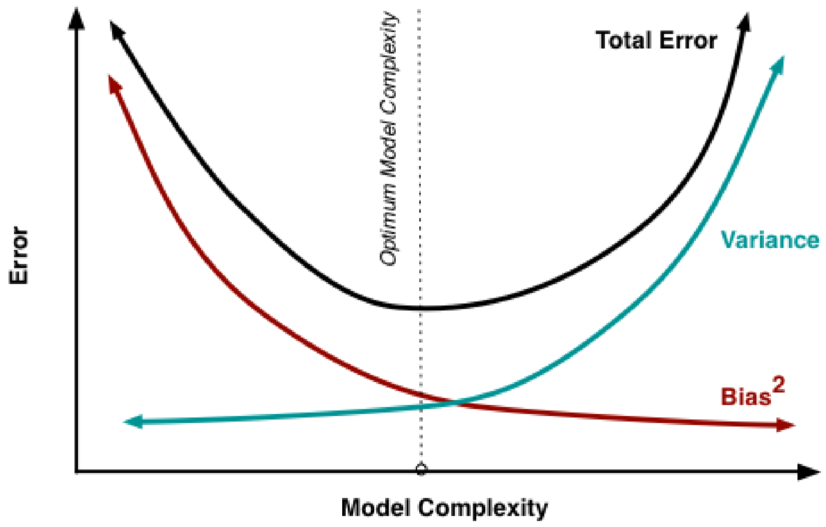
KNN (Algorithm and implementation)

- To classify an example $\tilde{x}_i$:
  0. Define distance
  1. Find the ‘k’ training examples x_i that are “nearest” to $\tilde{x}_i$.
  2. Classify using the most common label of “nearest” training examples.

F1	F2	Label
1	3	O
2	3	+
3	2	+
2.5	1	O
3.5	1	+
…	…	…



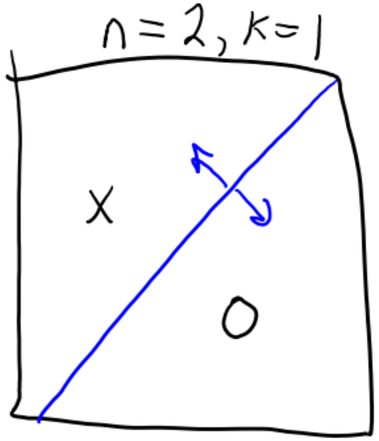
KNN (bias-variance trade-off)

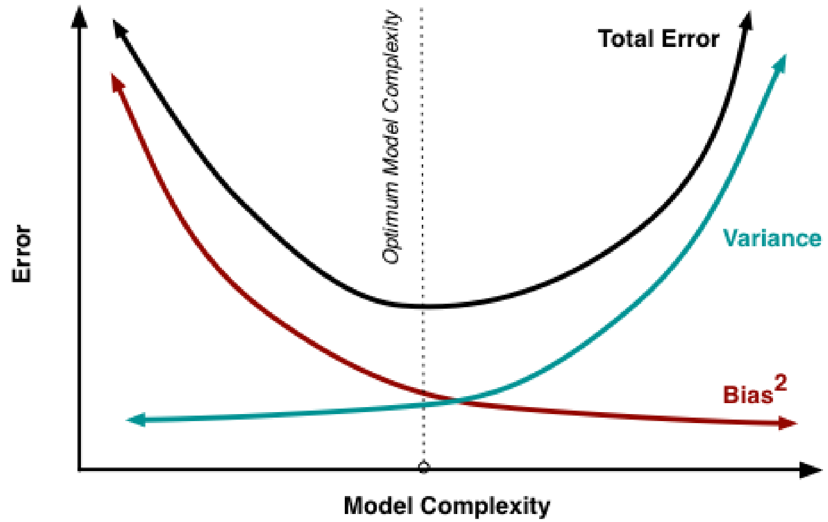


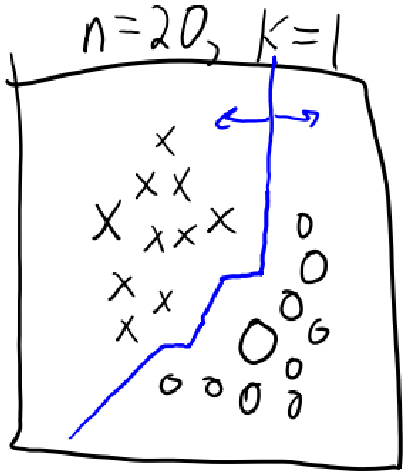
Q: how to put the value of “n” and “k” in this diagram?

KNN (bias-variance trade-off)

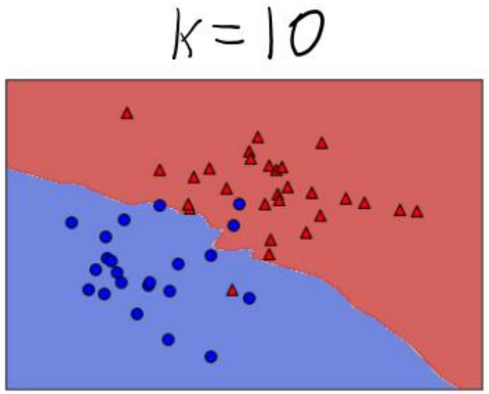
$$f = WX + k|W|_2^2$$
$$= W\begin{bmatrix} x \\ \dots \\ x^{n} \end{bmatrix} + \boldsymbol{k}|W|_2^2$$

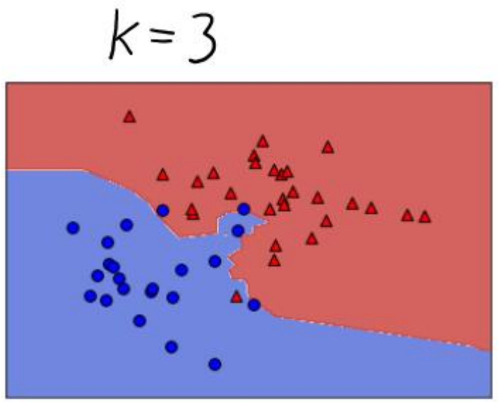


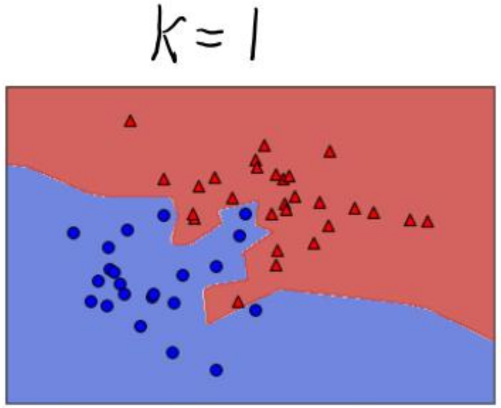




A: **larger n** means higher model complexity,
larger k behaves like stronger regularization



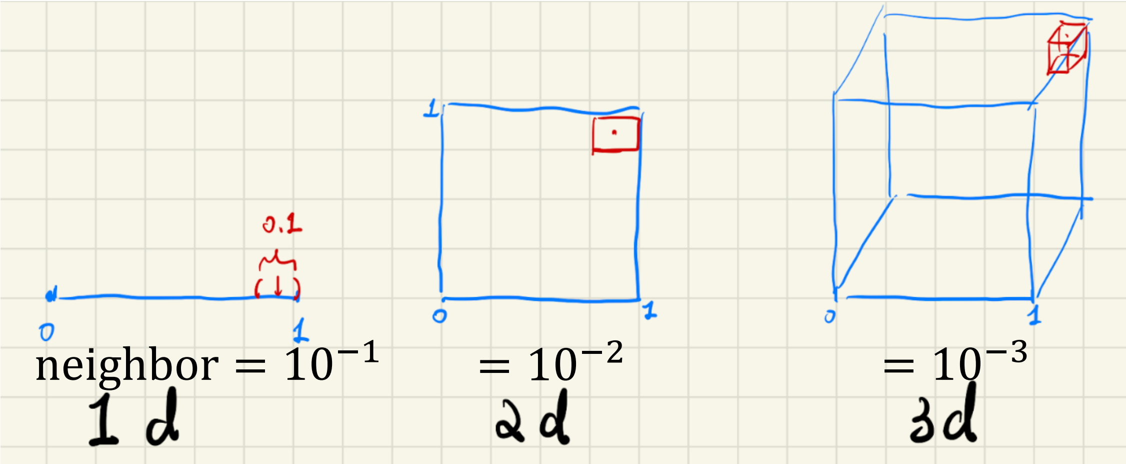




KNN (Curse of Dimensionality)

- Fact 1: need exponential more examples to get reasonable good neighbours
  - Volume of space grows exponentially with dimension.
    - Circle has area O(r^2), sphere has area O(r^3), 4d hyper-sphere has area O(r^4),...
  - Need exponentially more points to 'fill' a high-dimensional volume.
    - "Nearest" neighbours might be really far even with large 'n'.

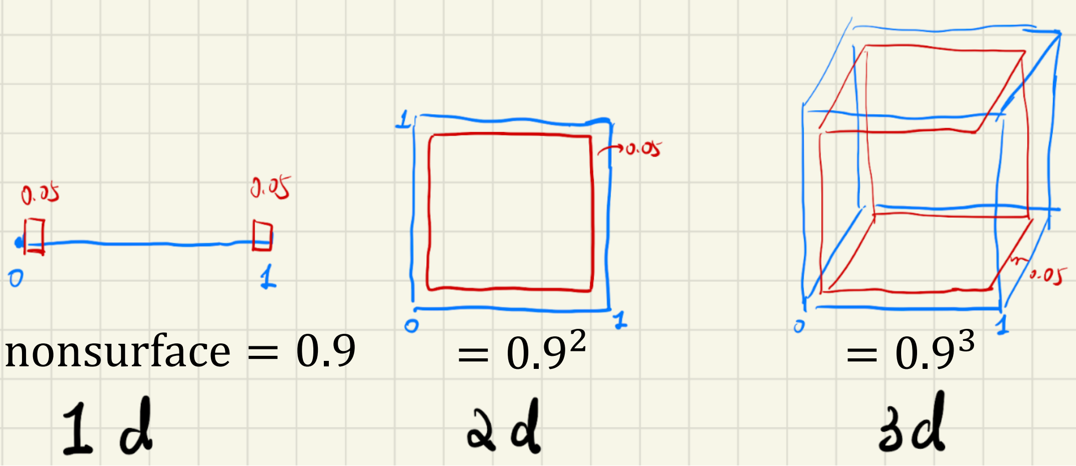
-- Assume $r < 0.05$ is a reasonable choice on unit ball



- **That is why many learning methods want DENSE representations and low-rank manifold**

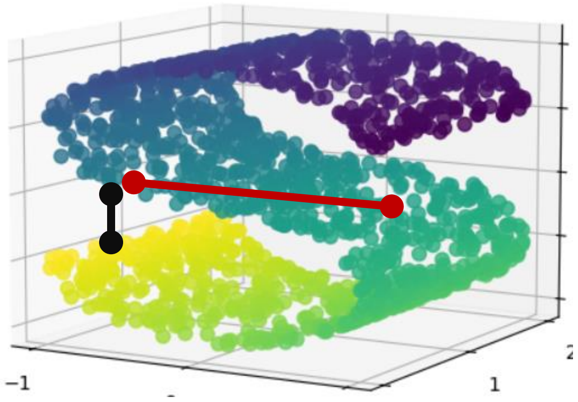
KNN (Curse of Dimensionality)

- Fact 2: if samples are **uniformly generated**, most samples are on "surface"

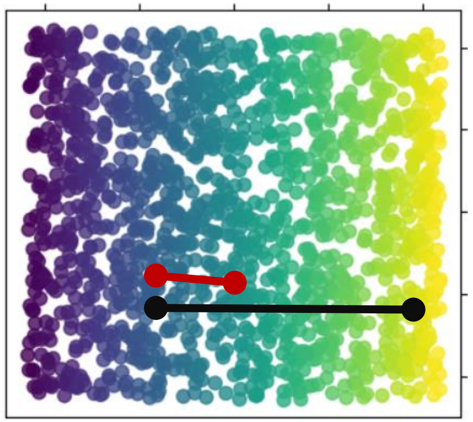


- Split the whole space into several 0.1*0.1*0.1 ... blocks.
- Random select one
- Higher prob. that block comes from the "surface"

- Support 3: since samples are **NOT uniformly generated**, they are on "**low-dim manifold**"



Think about their distance



Naive Bayes (Algorithm and implementation)

- Use bag of words to create features, gets users to label them

$	Hi	CPSC	340	Vicodin	Offer	...
1	1	0	0	1	0	...
0	0	0	0	1	1	...
0	1	1	1	0	0	...
...	...	...	...	...	...	...

$\mathbf{x}_1 = [110010]$

$\mathbf{x}_2 = [000011]$

$\mathbf{x}_3 = [011100]$

Spam?
1
1
0
...

$\mathbf{y}_1 = 1$

$\mathbf{y}_2 = 1$

$\mathbf{y}_3 = 0$

- Intuition:

if $p(\mathbf{y}_i = 1|\mathbf{x}_i) > p(\mathbf{y}_i = 0|\mathbf{x}_i)$

- return “spam”

else

- return “not spam”

Naive Bayes (Algorithm and implementation)

- Supervise learning usually model $p(\mathbf{y}_i|\mathbf{x}_i)$ directly, but here we use Bayes to decompose that:

$$p(\mathbf{y}_i|\mathbf{x}_i) = \frac{p(\mathbf{x}_i|\mathbf{y}_i)p(\mathbf{y}_i)}{p(\mathbf{x}_i)} = \frac{p(\mathbf{x}_{i1},\mathbf{x}_{i2},\dots,\mathbf{x}_{iB}|\mathbf{y}_i)p(\mathbf{y}_i)}{p(\mathbf{x}_i)}$$

- $p(\mathbf{x}_i)$ is usually hard to calculate, because we might not have enough data when ”Bag size” B is large (Recall curse of dimensionality)

$$p(\mathbf{y}_i = 1|\mathbf{x}_i) > p(\mathbf{y}_i = 0|\mathbf{x}_i)$$

$$\frac{p(\mathbf{x}_i|\mathbf{y}_i = 1)p(\mathbf{y}_i = 1)}{p(\mathbf{x}_i)} > \frac{p(\mathbf{x}_i|\mathbf{y}_i = 0)p(\mathbf{y}_i = 0)}{p(\mathbf{x}_i)}$$

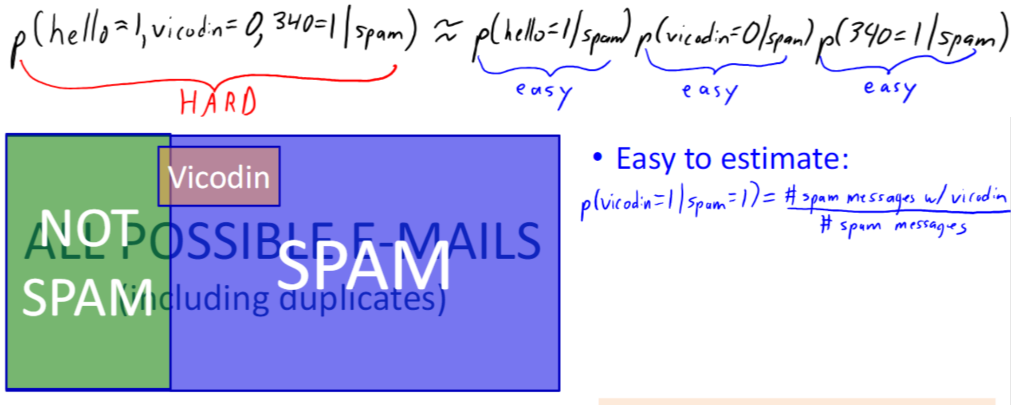
$$p(\mathbf{x}_i|\mathbf{y}_i = 1)p(\mathbf{y}_i = 1) > p(\mathbf{x}_i|\mathbf{y}_i = 0)p(\mathbf{y}_i = 0)$$

- Then, $p(\mathbf{x}_i|\mathbf{y}_i)$ is also hard to calculate due to similar reason. (Recall curse of dimensionality)
  We then assume the independence (might introduce bias, but generally OK)

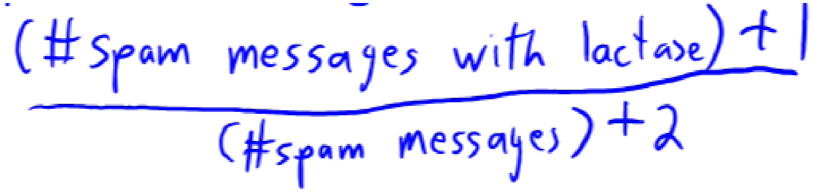
$$p(\mathbf{x}_{i1},\mathbf{x}_{i2},\dots,\mathbf{x}_{iB}|\mathbf{y}_i) \approx \prod_{b=1}^{B} p(\mathbf{x}_{ib}|\mathbf{y}_i)$$

Naive Bayes (Algorithm and implementation)

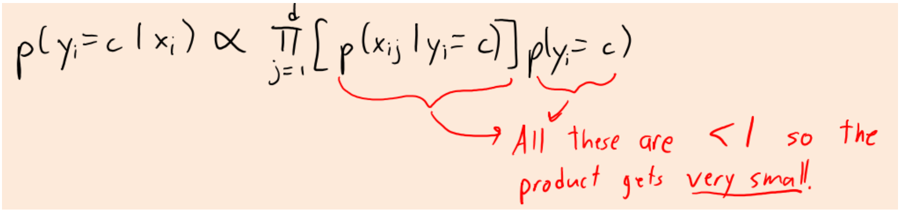
- Now, the task is to estimate $p(\mathrm{x}|\mathrm{y})$ for each possible x and y; and the margin prob $p(\mathrm{y})$ for each y



- Label smoothing: what happen if any term in $\prod_{b=1}^{B} p(\mathbf{x}_{ib}|\mathbf{y}_i)$ is zero?



- Avoid probability underflow: use log-prob instead



Thanks for your time!
Questions?