INTRODUCTION TO MACHINE LEARNING COMPSCI 4ML3

LECTURE 5

HASSAN ASHTIANI

THE TRADE-OFF

- A powerful/flexible curve-fitting method
  - Small training error
  - Requires more training data to generalize
  - Otherwise large test error
- A less flexible curve-fitting Method
  - Larger training error
  - Requires less training data
  - Smaller difference between training and test error
- The so-called "bias-variance" trade-off

THE CASE OF MULTIVARIATE POLYNOMIALS

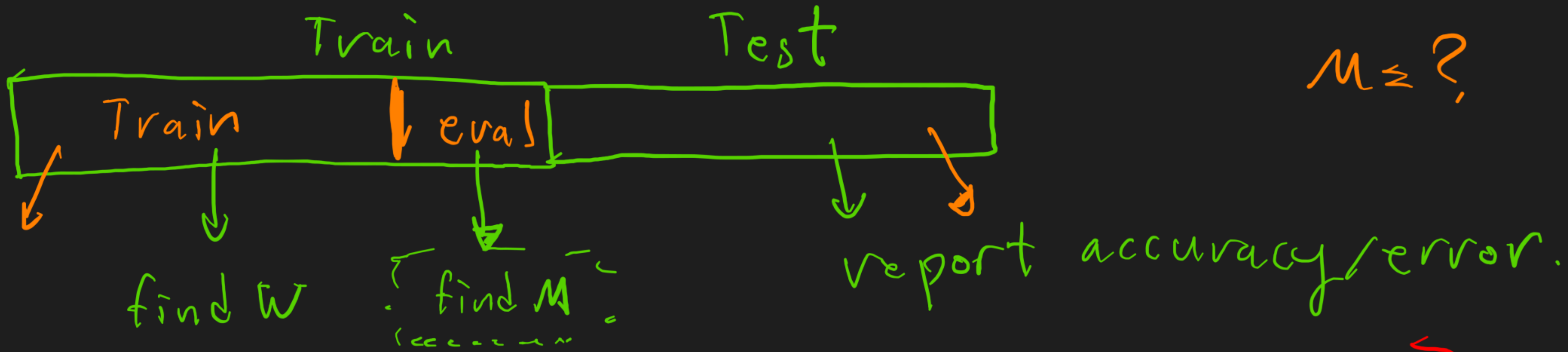
- Assume $M \gg d$
- Number of terms (monomials): $\approx (\frac{M}{d})^d$
- #training samples $\approx$ #parameters $\approx (\frac{M}{d})^d$
  - #training samples should increase exponentially with d
  - Susceptible to over-fitting…
  - An example of **curse of dimensionality**!
- We can say **sample complexity** of learning multivariate polynomials is exponential in d
  - Orthogonal to computational complexity

MODEL SELECTION: HOW TO AVOID OVERFITTING?

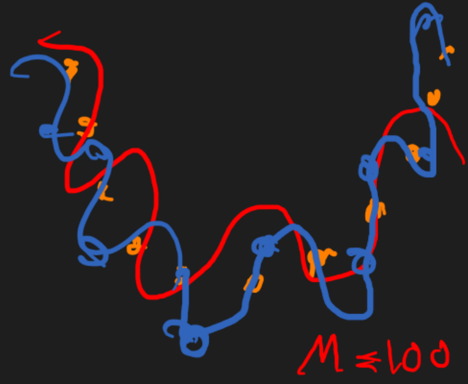
- SELECTING M (THE COMPLEXITY OF THE MODEL)
  - BASED ON d (DIMENSION) AND n (NUMBER OF SAMPLES)
- MORE PRACTICALLY, TRY SEVERAL OPTIONS FOR M
  - USE A **HOLDOUT (EVALUATION) SAMPLE**
  - NEVER USE TEST DATA TO TUNE PARAMETERS!

T.

* M is a Hyperparmeter



M = ?



* try $M = 1$, $M = 5$, $M = 10$
  * for each one, find W
  * compare the 3 models on eval, choose the best.
* Report error based on Test.

||w||

AVOID OVERFITTING WITH REGULARIZED LEAST SQUARES

d = 1000000

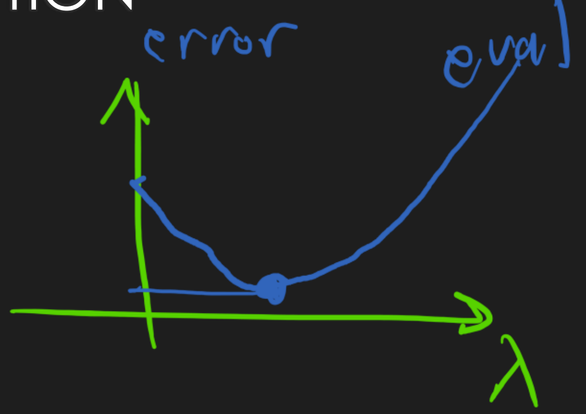
$$\min_{W \in \mathcal{R}^d} \|XW - Y\|_2^2 + \lambda \|W\|_2^2$$

λ = 1000000

w = 0 → 100

w = [0.1, 0.1, 0, 0]

- ENCOURAGE A SOLUTION WITH A SMALLER NORM
- $W^{RLS} = (X^T X + \lambda I)^{-1} X^T Y$
  - EXERCISE: PROVE THAT THIS IS THE OPTIMAL SOLUTION
- DOES THE INVERSE ALWAYS EXIST?
  - YES! (EXERCISE: PROVE) λ > 0
- HOW TO CHOOSE λ?



$w = (0.01, 0.01, 0.1, 0.01) \rightarrow$ small $\underbrace{\text{norms}}_{}$ ℓ_2

$w = (0, 0, 0.1, 0) \rightarrow$ sparse vector

POLYNOMIAL CURVE-FITTING REVISITED

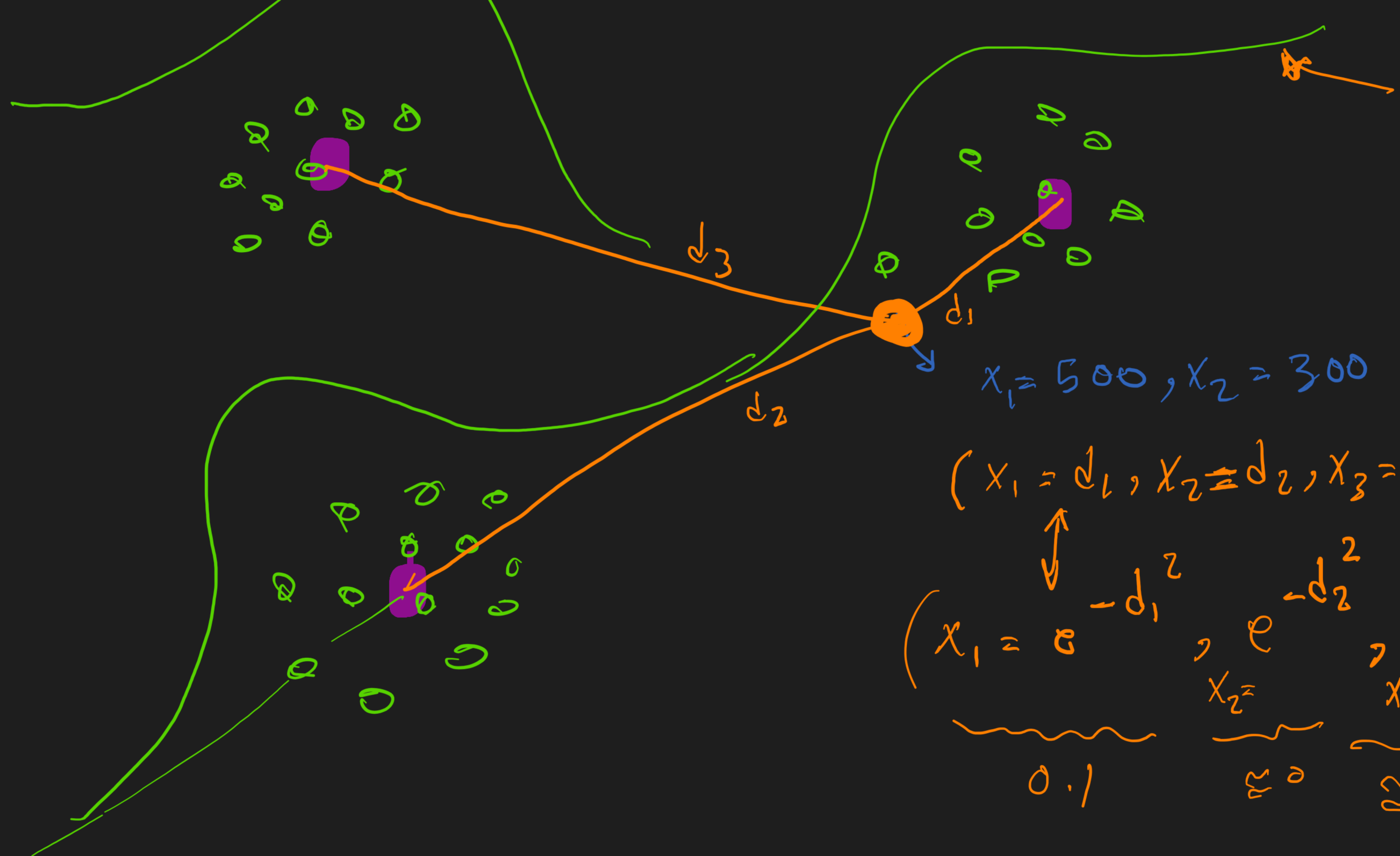
- MAP THE INPUTS x^i TO A HIGHER DIMENSIONAL SPACE
  - A KIND OF "PRE-PROCESSING" THE DATA
- DO LINEAR REGRESSION ON THE HIGH-DIMENSIONAL SPACE
  - EQUIVALENT TO PERFORMING NON-LINEAR REGRESSION IN THE ORIGINAL SPACE
- MAP $\boxed{\phi(x)}: \mathrm{R}^{d_1} \longmapsto \mathrm{R}^{d_2}$ WHERE $d_2 \gg d_1$
- $\phi(x) = \begin{pmatrix} \phi_1(x) \\ \dots \\ \phi_{d_2}(x) \end{pmatrix}$ IS NONLINEAR, E.G., $x \in R$ AND $\phi(x) = \begin{pmatrix} x \\ x^2 \\ x^3 \\ \dots \\ x^{d_2} \end{pmatrix}$
- WHAT IF d_2 IS MUCH LARGER THAN THE NUMBER OF SAMPLES?

CURVE-FITTING WITH BASIS FUNCTIONS

- FEATURE MAP: $\phi(x): \mathrm{R}^{d_1} \longmapsto \mathrm{R}^{d_2}$ $\quad d_2 \gg d_1$
- $\Phi_{n \times d_2} = [\phi(x^1) \quad \dots \quad \phi(x^n)]^T$ $\longleftrightarrow$ $X_{n \times d_1}$
- TRAINING
  - $W^* = \min_W \|\Phi W - Y\|_2^2 + \lambda \|W\|_2^2$
  - $W^* = (\Phi^T \Phi + \lambda I)^{-1} \Phi^T Y$
- PREDICTION
  - $\hat{y} = < W^*, \phi(x) > = {W^*}^T \phi(x)$

OTHER CHOICES OF $\phi(x)$

- PICK A FIXED (NONLINEAR) $\Phi(x)$
  - ENCODES YOUR PRIOR KNOWLEDGE ABOUT THE DATA
  - FEATURE ENGINEERING!
- POLYNOMIAL BASIS FUNCTIONS
- GAUSSIAN BASIS FUNCTIONS:
  - $\boxed{\phi_i(x) = e^{-\frac{\|x-\mu_i\|_2^2}{2\sigma^2}}}$
- DFT (FFT), WAVELET FOR TIME SERIES
- IS IT POSSIBLE TO LEARN THE MAPPING $\phi_i(x)$ ITSELF?
  - LATER, E.G., NEURAL NETWORKS



$x_1 = 500, \; x_2 = 300$

$$(x_1 = d_1, \; x_2 = d_2, \; x_3 = d_3)$$

$$\updownarrow$$

$$\left(\underbrace{x_1 = e^{-d_1^2}}_{0.1}, \; \underbrace{x_2 = e^{-d_2^2}}_{\approx 0}, \; \underbrace{x_3 = e^{-d^3}}_{\approx 0}\right)$$

COMPUTATIONAL COMPLEXITY OF NAÏVE RLS

- Training: Calculate $\mathrm{w}^{\mathrm{RLS}} = (\phi^T \phi + \lambda I)^{-1} \phi^T Y$
- Bottleneck: Matrix inversion
  - How many operations?
- Prediction: $\hat{y} =< \phi(x), w^{RLS} >$
  - How many operations?
- Regularization allows us to go into high-dimensional space without overfitting, but it does not solve the computational problem

COMPUTATIONAL COMPLEXITY

- Matrix multiplication (n-by-n matrices)
  - Native method: $O(N^3)$
  - Strassen's Algorithm: $O(N^{2.8074})$
  - Coppersmith–Winograd-like algorithms [Current best $O(N^{2.3728639})$]
- Matrix inversion
  - Gaussian Elimination: $O(N^3)$
  - Possible to reduce it to multiplication

THE COMPUTATIONAL PROBLEM

- CAN WE SOLVE THE REGULARIZED LEAST SQUARES IN R^{d_2} **WITHOUT EXPLICITLY MAPPING THE DATA** INTO R^{d_2}?
  - $W^* = \min_{W \in R^{d_2}} \|\Phi W - Y\|_2^2 + \lambda \|W\|_2^2$
- SOMETHING LIKE MULTIPLICATION USING FFT
- IF SO, WE COULD EVEN MAP THE DATA TO AN INFINITE DIMENSIONAL SPACE!!

FFT AND MULTIPLICATION