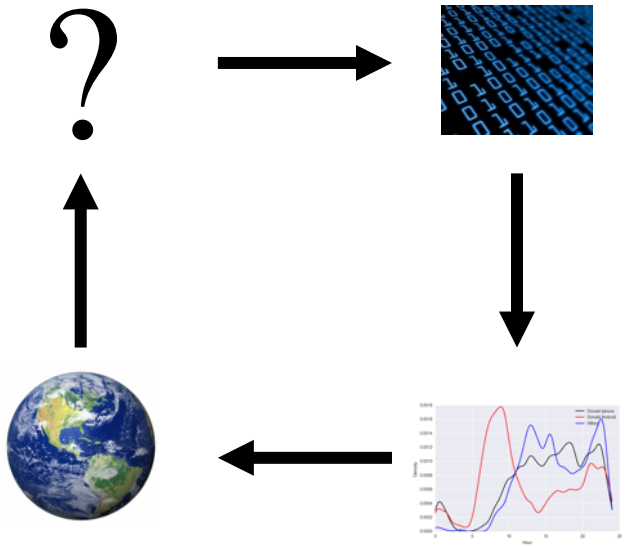
Review

DS 100, Spring 2017

Slides by:

Joseph Gonzalez

jegonzal@berkeley.edu

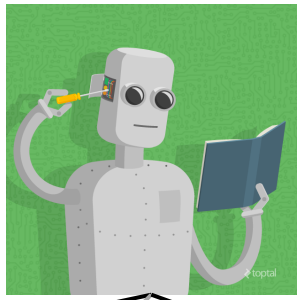


Topics Covered in This Deck

- Feature Engineering
- Over-fitting and Cross Validation
- Regularization and the Bias Variance Tradeoff
- Classification (Logistic Regression)

The Taxonomy of Machine Learning Problems

Taxonomy of Machine Learning



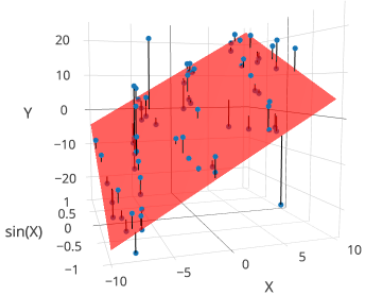
Labeled Data

Unlabeled Data

Supervised Learning

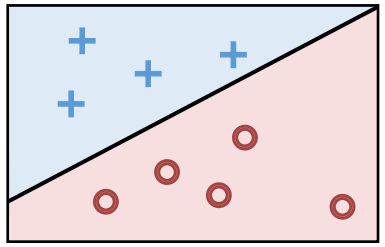
Continuous Response

Regression

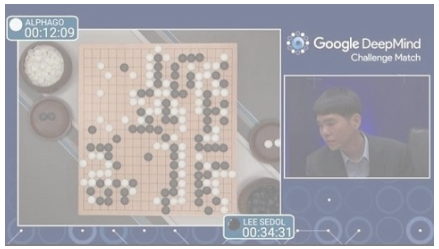


Categorical Response

Classification



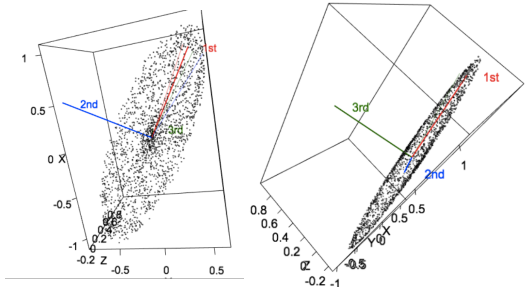
Reinforcement Learning (not covered)



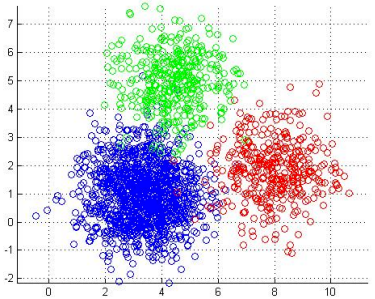
Alpha Go

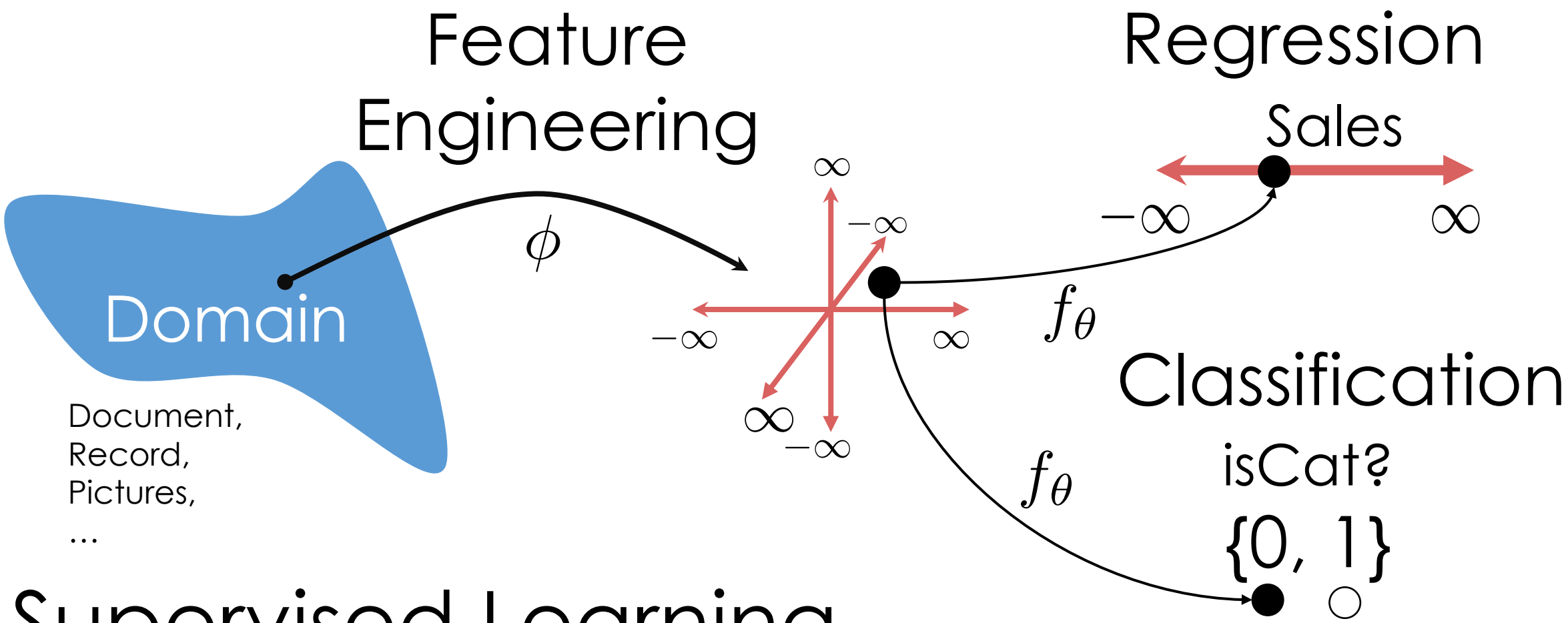
Unsupervised Learning

Dimensionality Reduction



Clustering





Supervised Learning

Given pairs of (Features, Label) learn a function from *feature* → *label*

- Continuous Label → Regression
- Categorical Label → Classification

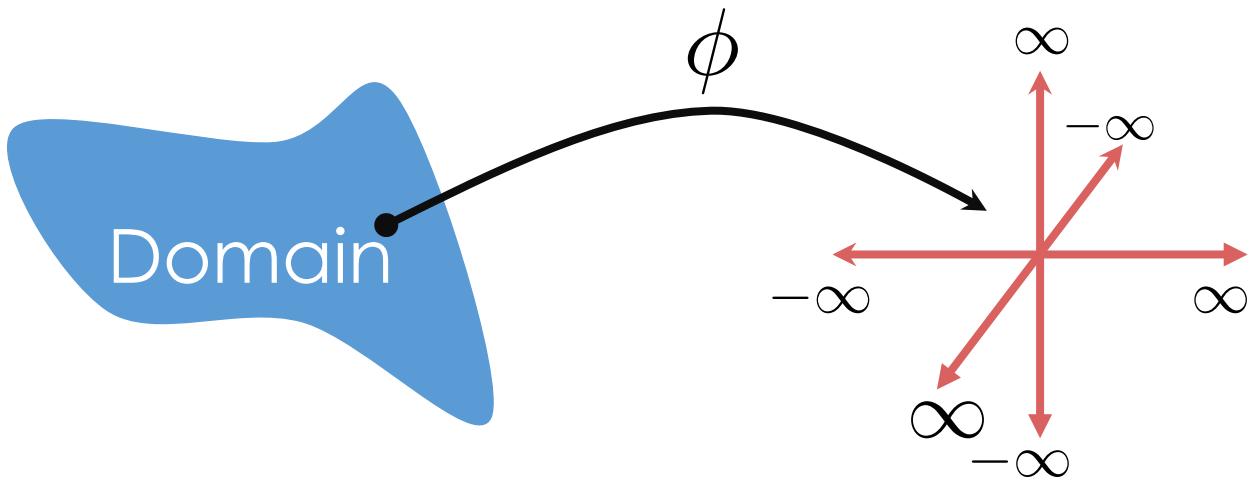
$$f_\theta\left(\phi\left(\texttt{features}\right)\right) \rightarrow \texttt{label}$$

Recap: Feature Engineering

- Linear models with feature functions:

$$f_\theta(x) = \sum_{j=1}^{p} \theta_j \phi_j(x)$$

- Feature Functions: $\phi : \mathcal{X} \to \mathbb{R}^p$



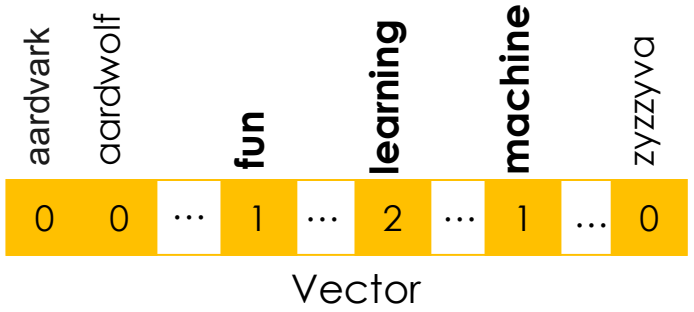
➢ **One-hot encoding:** *Categorical Data*

state
NY
WA
CA

AL	…	CA	…	NY	…	WA	…	WY
0	…	0	…	1	…	0	…	0
0	…	0	…	0	…	1	…	0
0	…	1	…	0	…	0	…	0

➢ **Bag-of-words & N-gram:** *Text Data*

"Learning about machine learning is fun."

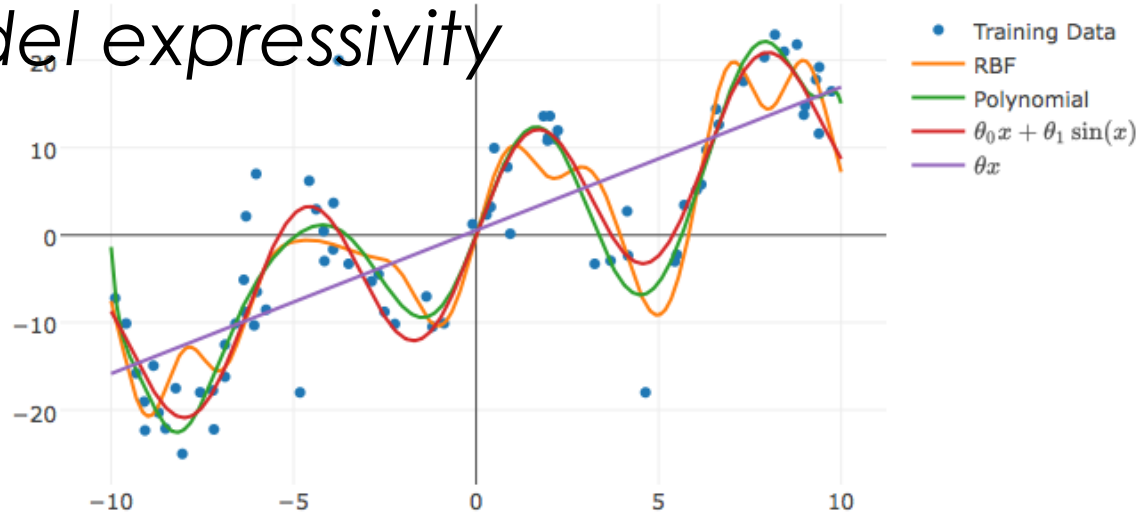


➢ **Custom Features:** *Domain Knowledge*

$$\phi(\text{lat}, \text{lon}, \text{amount}) = \frac{\text{amount}}{\textbf{Stores}[\textbf{ZipCode}[\text{lat}, \text{lon}]]}$$

- **Generic Features:** *increase model expressivity*
  - **Polynomial and trigonometric**

$$\phi(x) = [x, x^2, \ldots, x^p]$$



  - **RBF: Radial Basis Functions**

- **Hyper-parameters:** parameters that are selected using prior knowledge or through *cross-validation*.
  - Example: Poly. deg., μ_i, and σ_i

- **How to Study**
  - Review Notebooks on Feature Engineering Part 1, Part 2
  - What don't I need to know:
    - You do not need to know about RBFs

Overfitting, Cross Validation, and the Bias-Variance Tradeoff

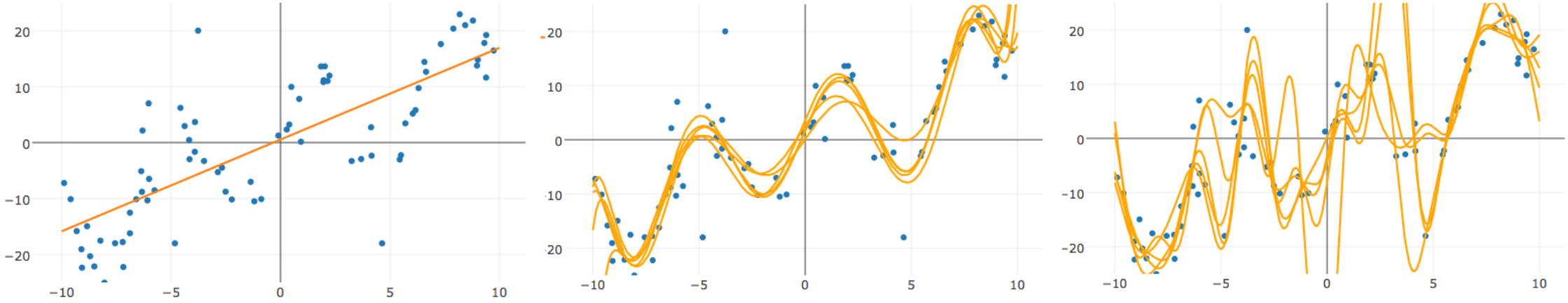
Fundamental Tension in *Learning*

- ***Fit* the Data**
  - Provide an explanation for what we observe
- ***Generalize* to the World**
  - Predict the future
  - Explain the unobserved

Is this cat grumpy or are we overfitting to human faces?

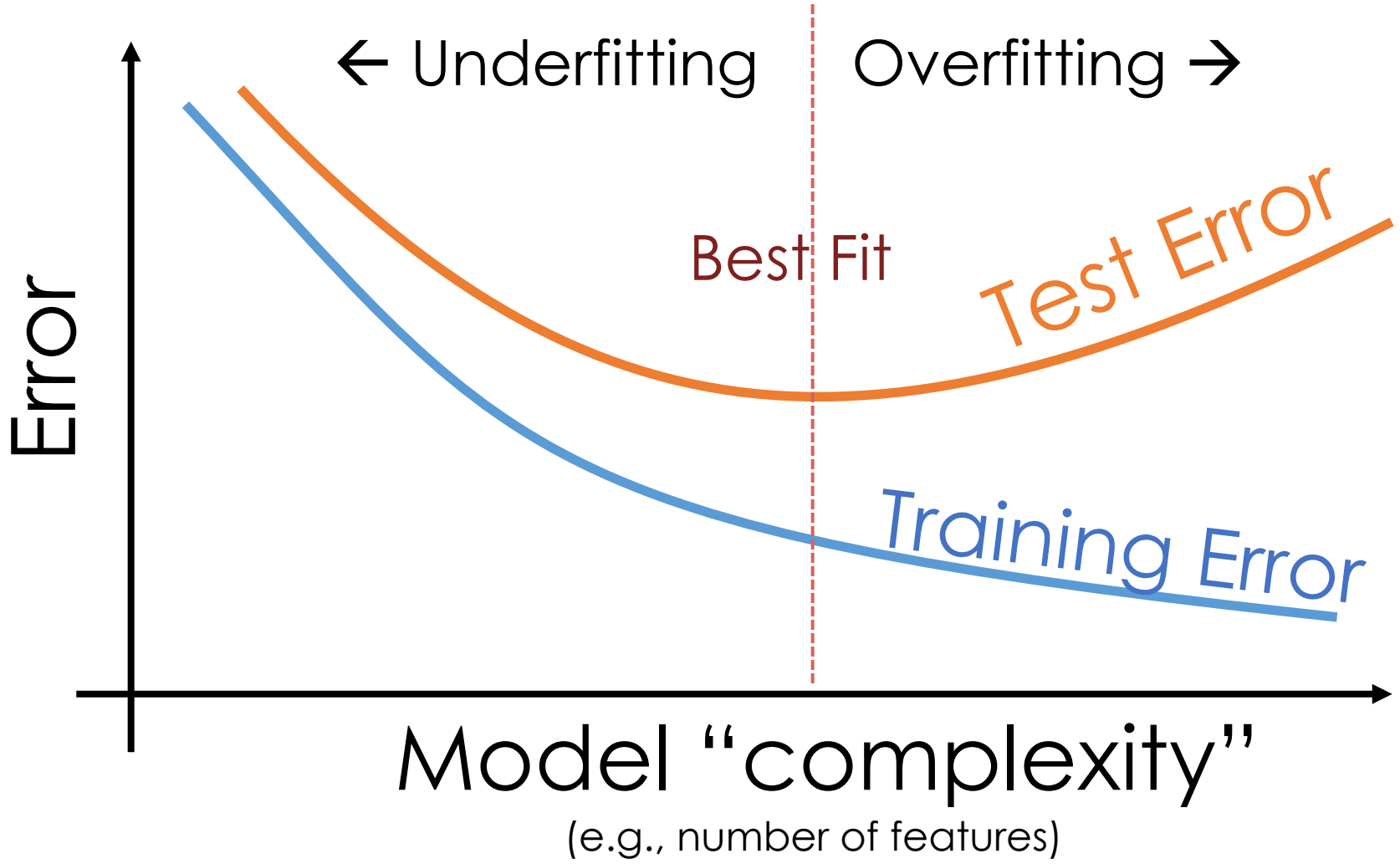
The Bias-Variance Tradeoff

Overfitting



Underfitting

Training vs **Test** Error



Typically

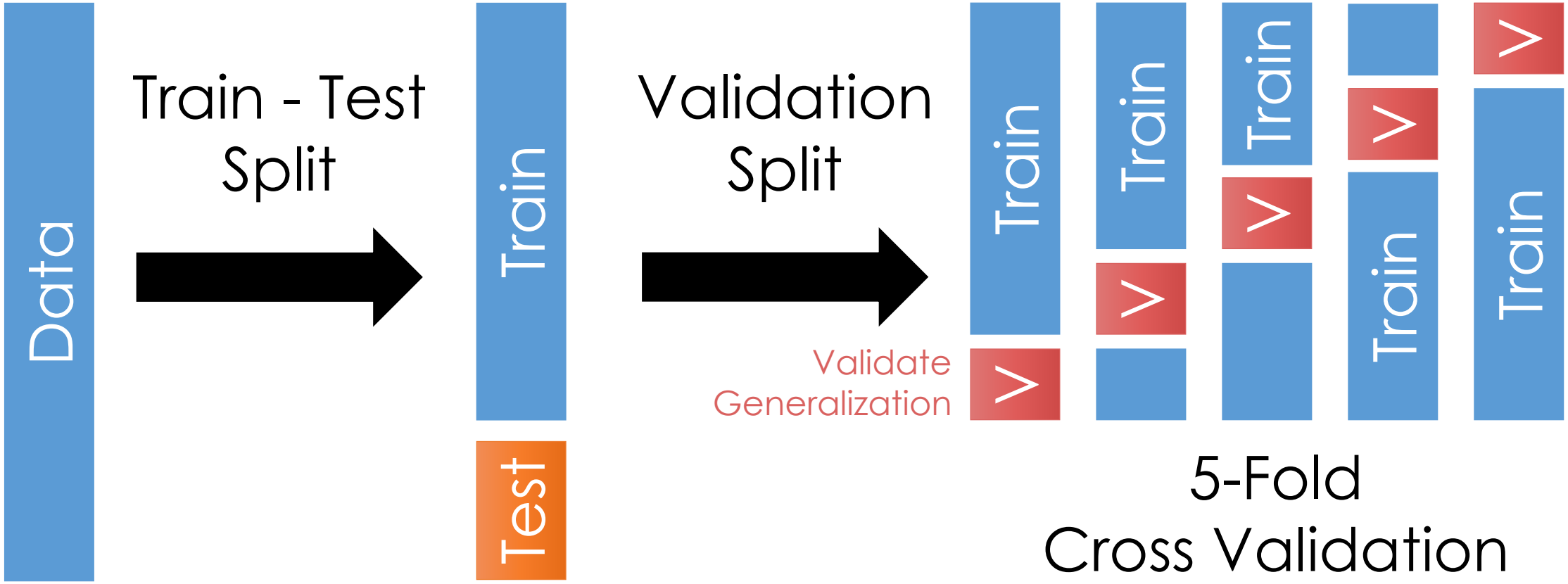
Training Error < Test Error

Why

- Training minimizes training error

Be able to explain everything on this slide!

Train-Test Split & Cross Validation



Cross validation simulates multiple train test-splits on the training data.

You can only use the test dataset once after deciding on the model.

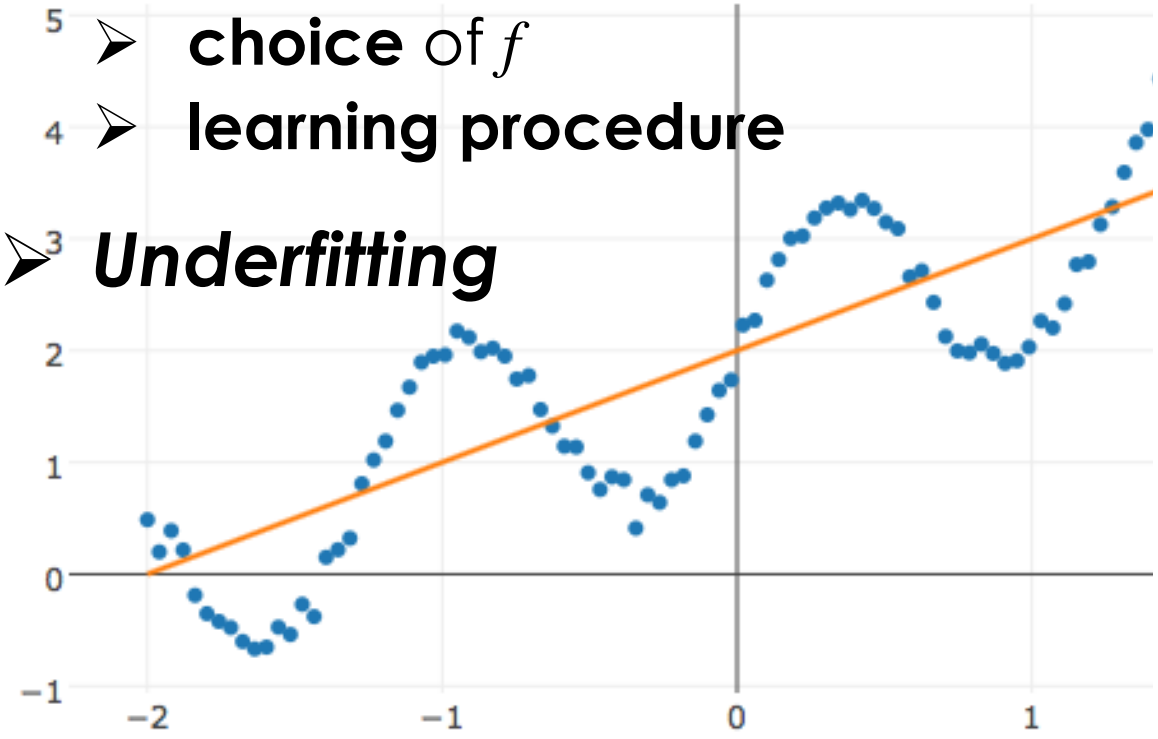
Source of Prediction Error

- **Noise:** the intrinsic variability in the process we are trying to model
- **Bias:** the expected deviation between the predicted value and the true value
- **Variance:** variability between the estimated value and the true value across different training datasets

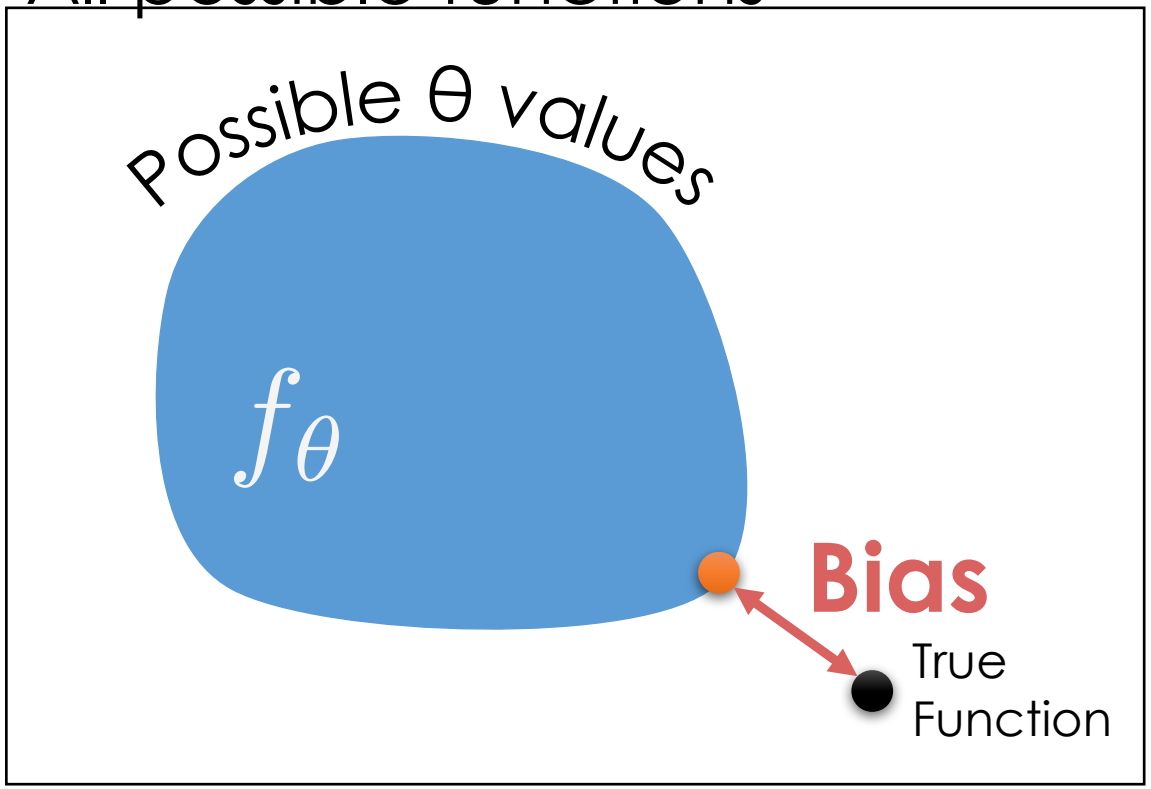
Bias

The expected deviation between the predicted value and the true value

- Depends on both the:
  - **choice** of f
  - **learning procedure**
- ***Underfitting***



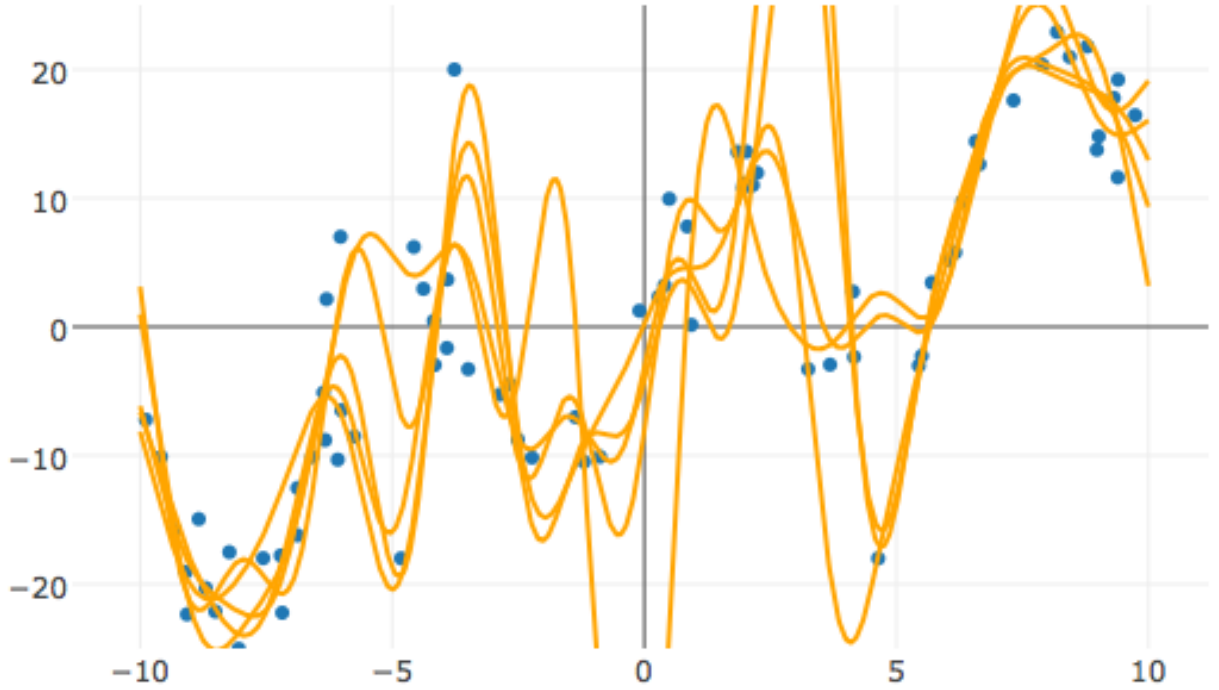
All possible functions



Variance

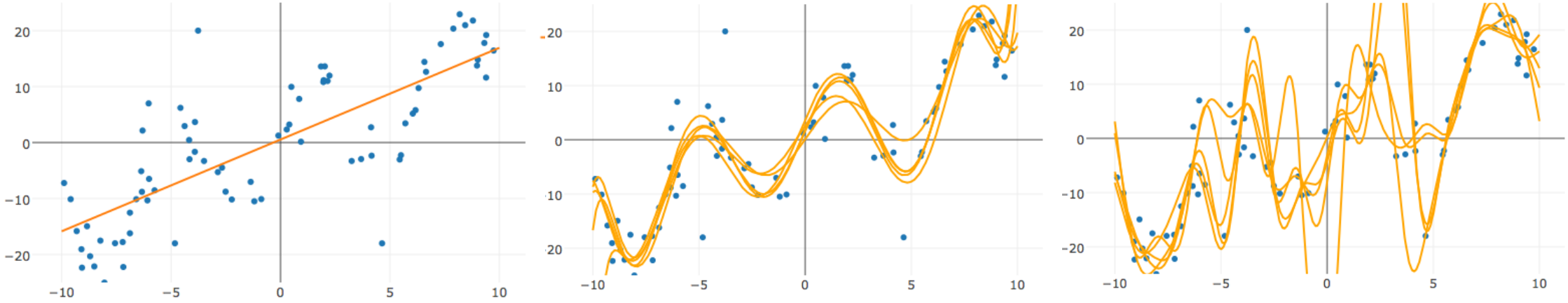
variability between the estimated value and the true value across different training datasets

- Sensitivity to variation in the training data
- Poor generalization
- **Overfitting**



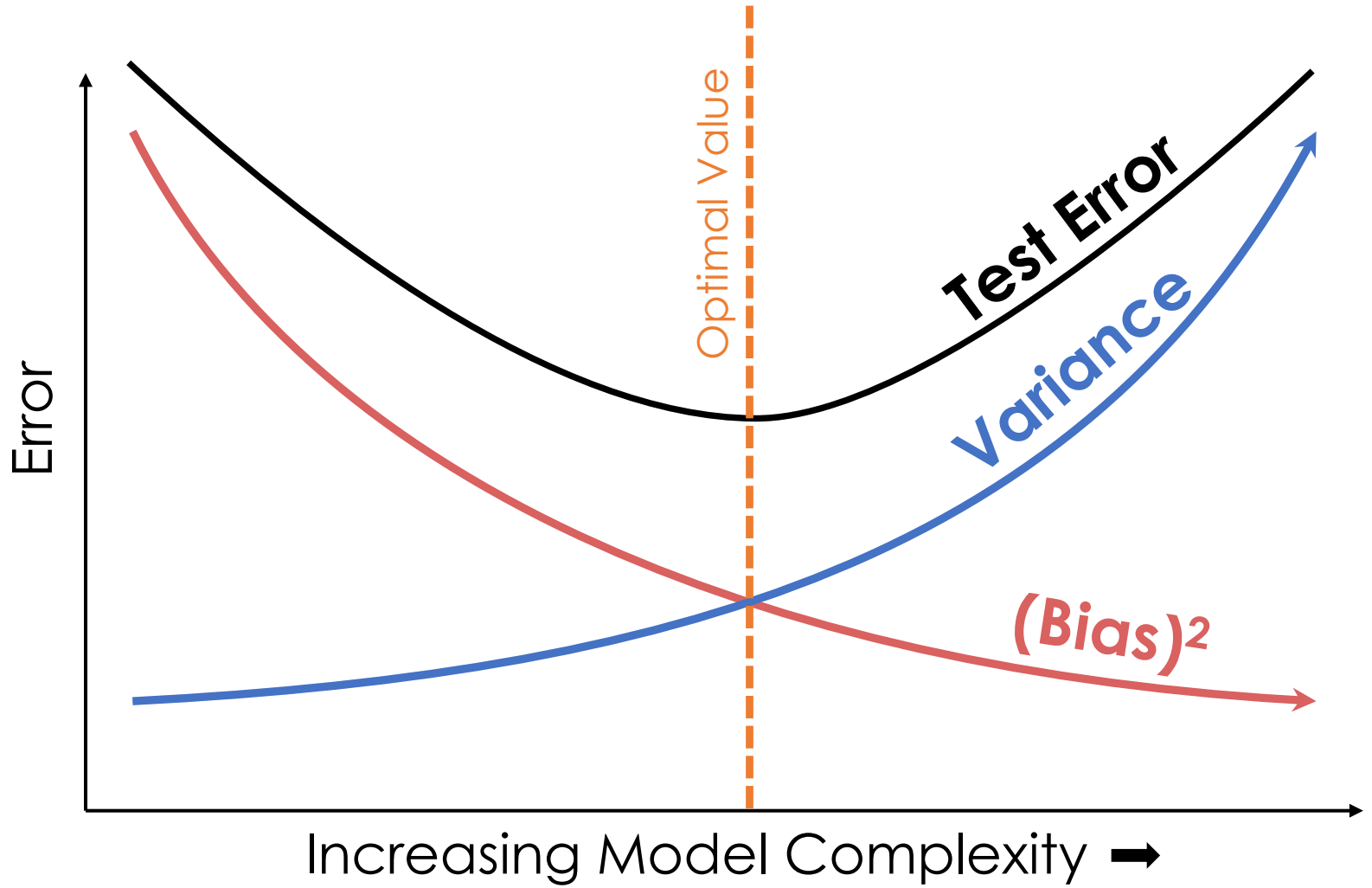
The Bias-Variance Tradeoff

Variance



Bias

Bias Variance Plot

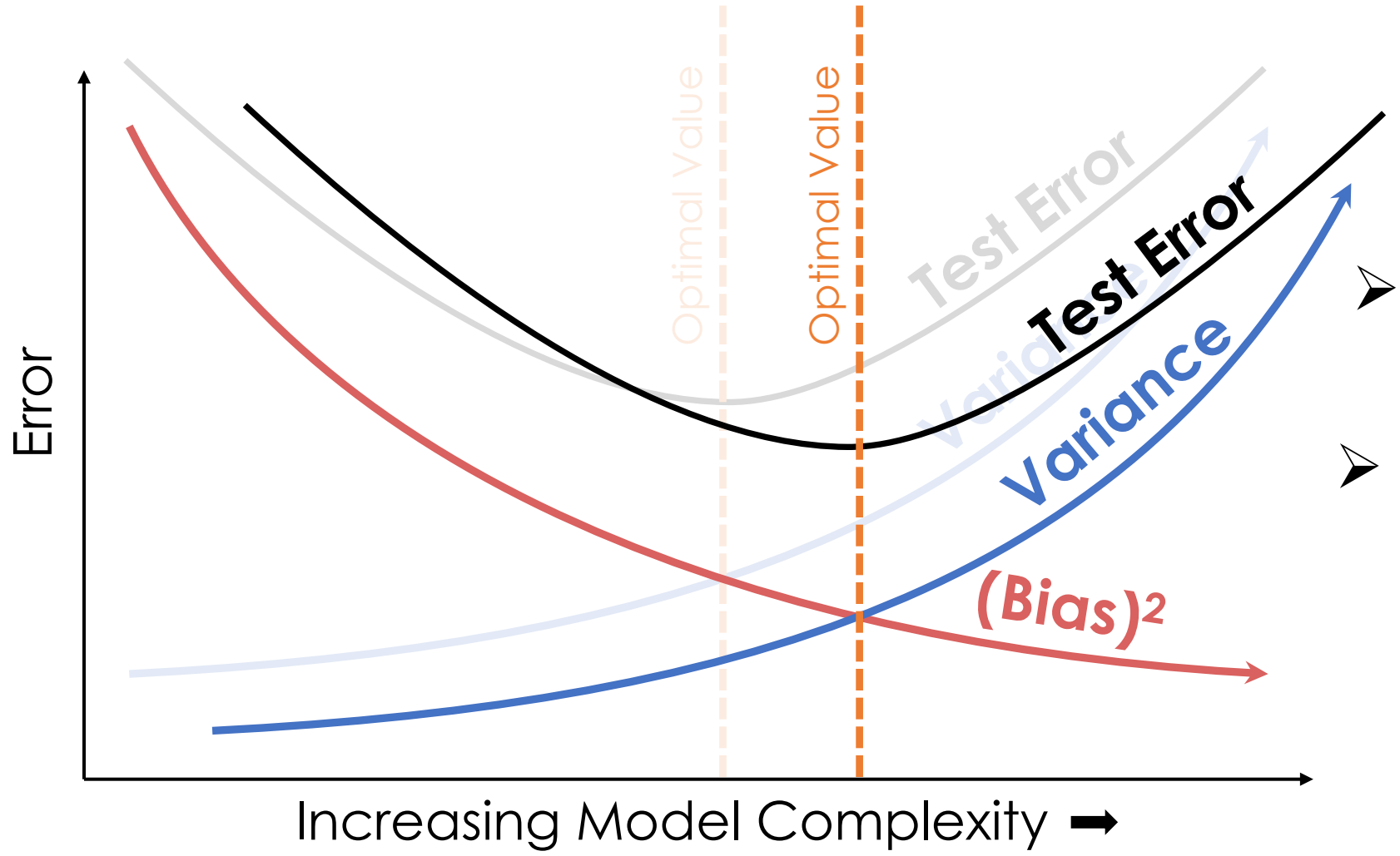


How do we compute test error?

- Use test set to pick best model?
  - No no no no noooo!
- Cross validation?
  - Yes!!!!

- What happens with more data?

Bias Variance Plot



How do we compute test error?

- Use test set to pick best model?
  - No no no no noooo!
- Cross validation?
  - Yes!!!!

- What happens with more data?
- How do we tune the model complexity?
  - Number of features
  - Types of features
  - Different models
  - **Regularization**

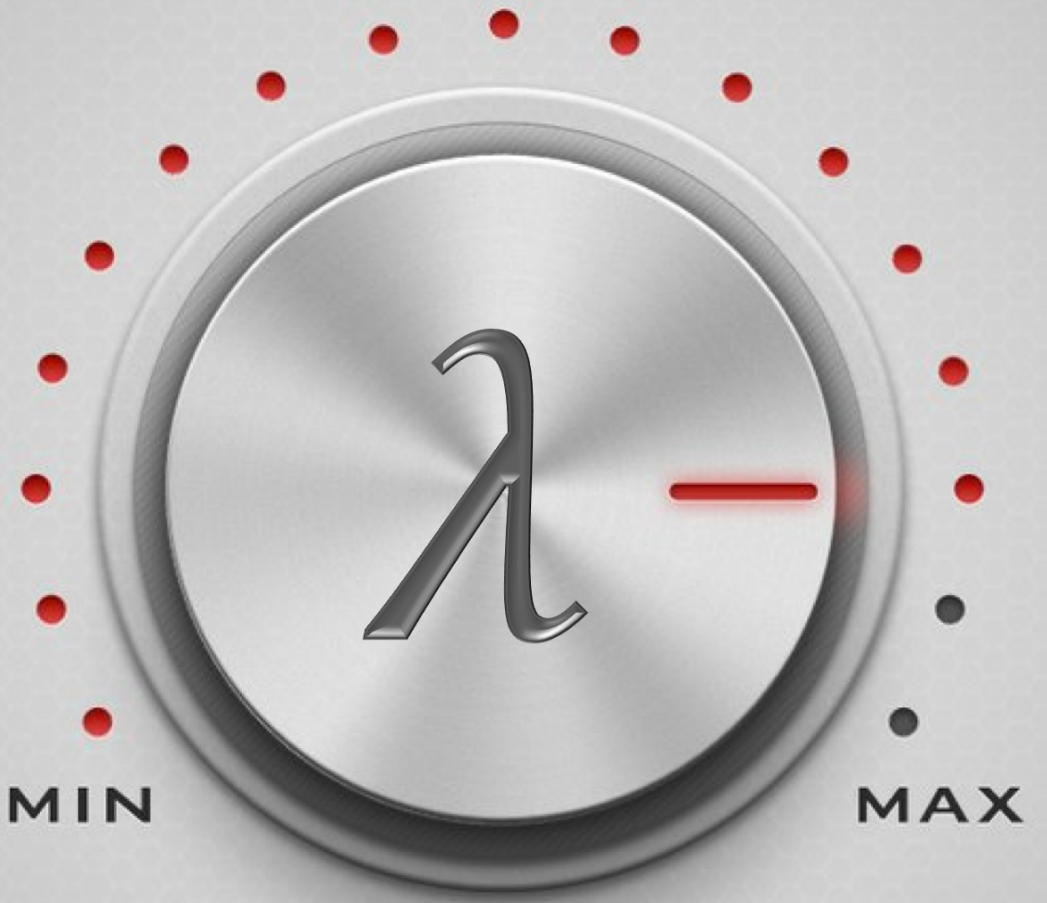
Study Suggestions

- You should know
  - Training vs Test error
  - Train, test, validation splits
  - Cross validation
  - Sources of prediction error and meaning of bias & variance
- What is not on the exam:
  - The derivation of the Bias Variance Tradeoff
- How to study
  - Review lecture materials
  - *Try not to over fit!*

Regularization

Managing Overfitting

- Tradeoff:
  - **Increase bias**
  - **Decrease variance**
- In DS100 we studied parametric regularization
  - L1 and L2 penalty
  - regularization parameter λ



Regularization and Loss Minimization

Fit the Data

Penalize Complex Models

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathbf{Loss}\left(y_i, f_\theta(x_i)\right) + \lambda \mathbf{R}(\theta)$$

Regularization Parameter

- How should we define **R(θ)**?
- How do we determine λ?

Common Regularization Functions

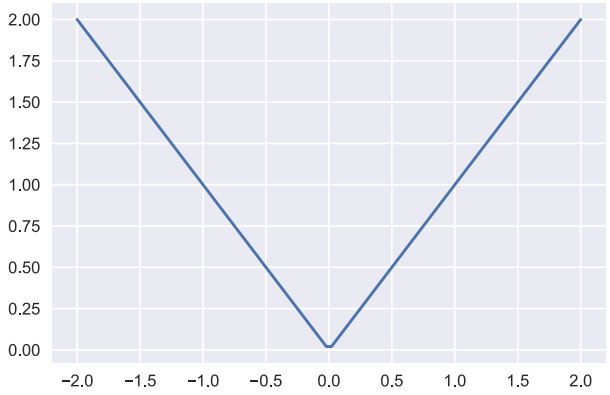
Ridge Regression (L2-Reg)

$$R_{\text{Ridge}}(\theta) = \sum_{i=1}^{d} \theta_i^2$$

- Distributes weight across related features (robust)
- Analytic solution (easy to compute)
- Does not encourage sparsity → small but non-zero weights.

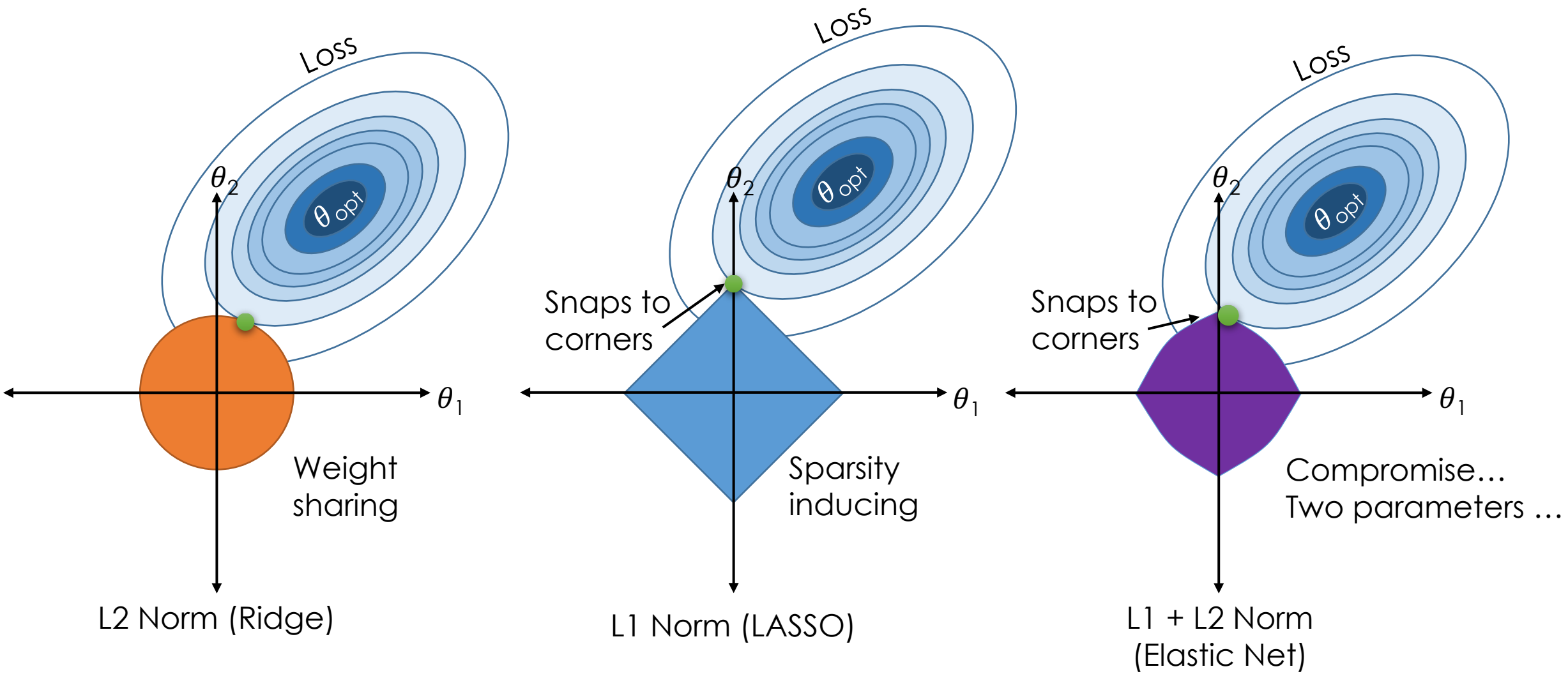
LASSO (L1-Reg)

$$R_{\text{Lasso}}(\theta) = \sum_{i=1}^{d} |\theta_i|$$



- **Encourages sparsity** by setting weights = 0
  - Used to select informative features
- Does not have an analytic solution → numerical methods

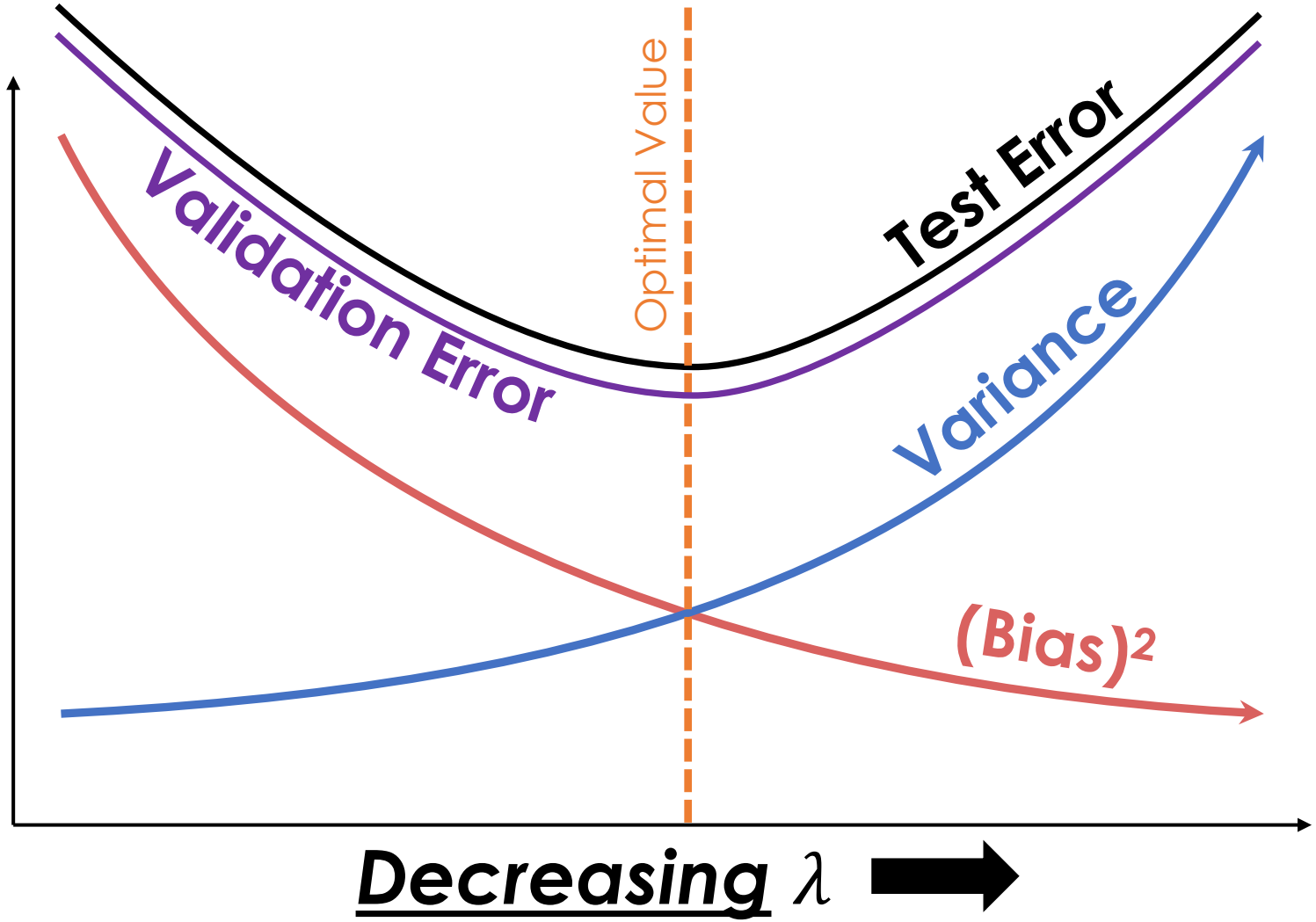
Regularization and Norm Balls



Determining the Optimal λ

How do we determine λ?

- Try a range of values
- Use cross validation to estimate **generalization error**



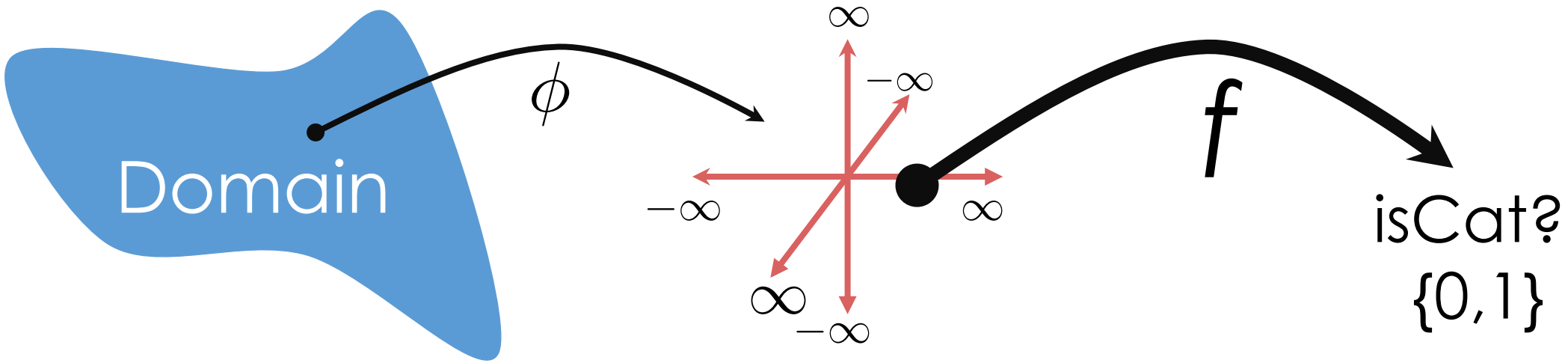
- Value of λ determines bias-variance tradeoff
  - Larger values → more regularization → more bias → less variance

How to study

- Review lecture slides and python notebooks
- You should know:
  - Role of λ and regularization penalty
  - Affect of λ on bias and variance
- How to determine λ (via cross validation)
  - Try range of values and take minimum
- Compare L1 and L2 regularization

Classification & Logistic Regression

Classification

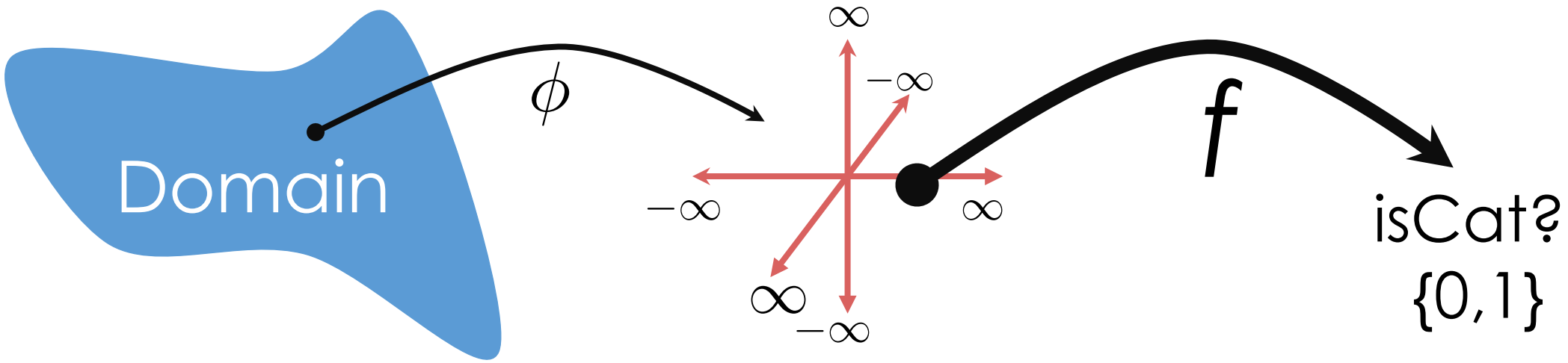


Predicting the value of a categorical variable:

- **Binary Classification:** 2-classes (e.g., {Spam, Ham})
- **Multiclass Classification:** k-classes (e.g., {Happy, Sad, Angry …})

Could treat binary case as least squares regression → **but don't**!

Classification



Can we just use least squares?

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2 + \lambda \mathbf{R}(\theta)$$

- Yes … for binary ca
- Ne
  (e.
- Diff to interpret model …

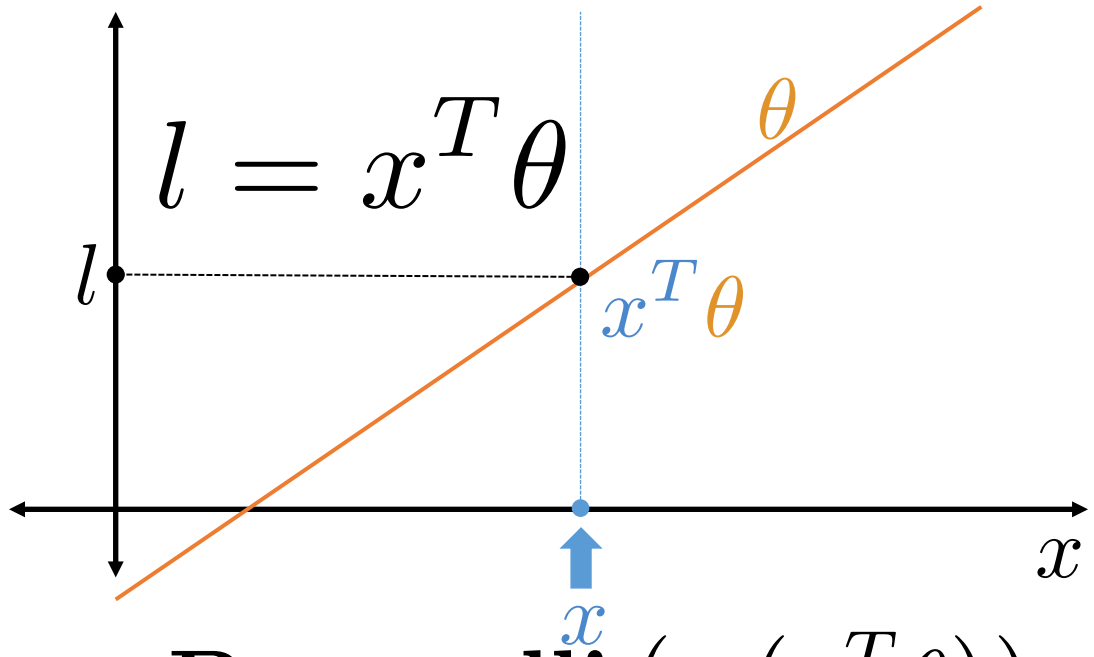
Don't use Least Squares for Classification

Many Classification Models

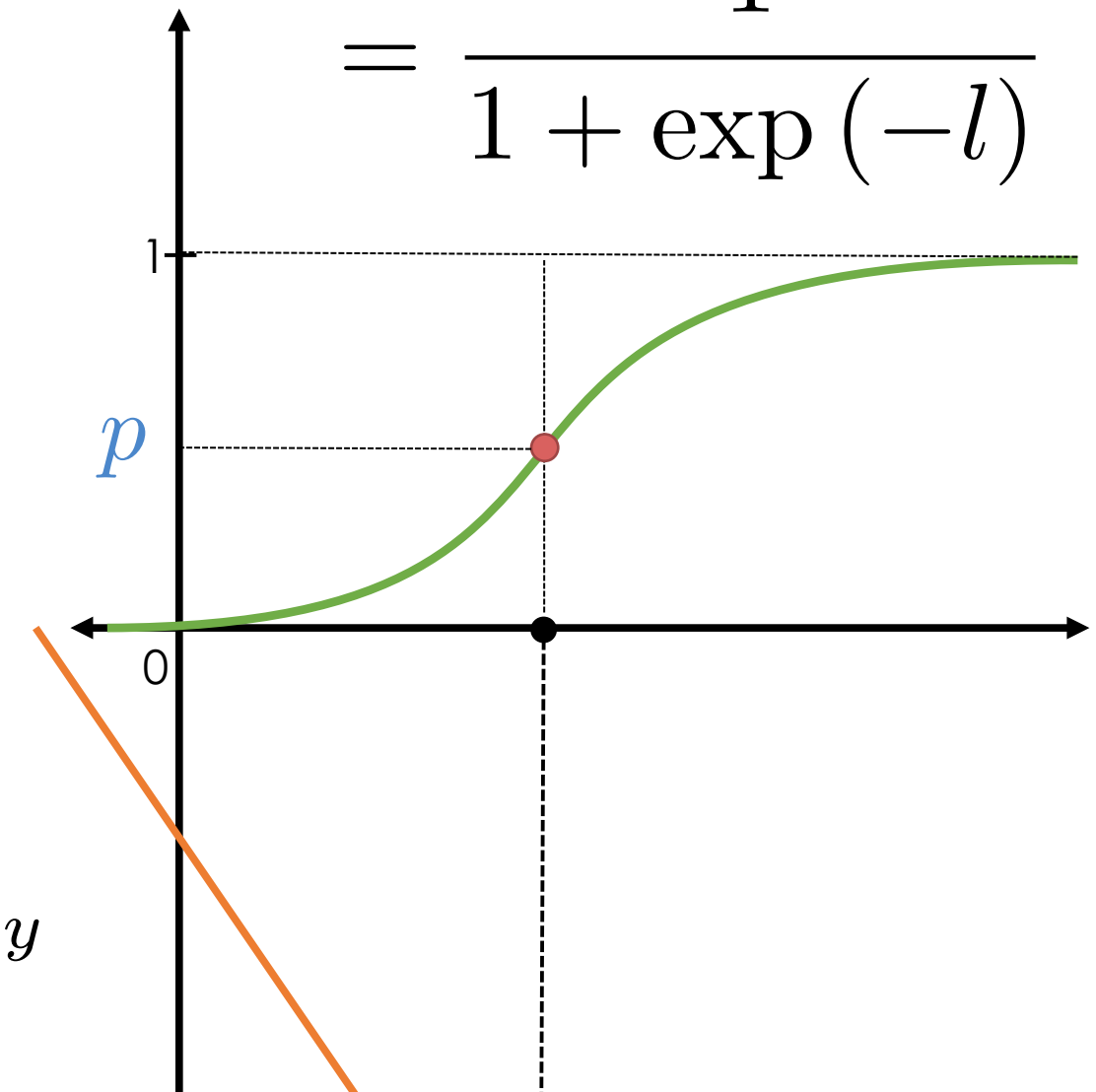
- **Logistic Regression** [the one you need to know]
  - Widely used and shares a lot in common with other models
- Neural Networks & Deep Learning
  - Covered in lab … (not on exam)
- Support Vector Machines
- Decision Trees
- Naïve Bayes
- …

Logistic Regression Model

$$p = \sigma(l)$$



$$= \frac{1}{1 + \exp(-l)}$$



$$y \sim \textbf{Bernoulli}\left(\sigma\left(x^T\theta\right)\right)$$

$$\mathbf{P}\left(y \,|\, x\right) = p^y(1-p)^{(1-y)}$$

$$= \sigma\left(x^T\theta\right)^y \left(1 - \sigma\left(x^T\theta\right)\right)^{1-y}$$

The Logistic Regression **Model**

In what sense is this a model?

Single Data Point Model (x, y)

$$l = x^T\theta$$

Sigmoid Function
Not standard dev.

$$p = \sigma(l)$$

$$= \frac{1}{1 + \exp(-l)}$$

$$y \sim \textbf{Bernoulli}(p)$$

$$y \sim \textbf{Bernoulli}\left(\sigma\left(x^T\theta\right)\right)$$

$$\mathbf{P}(y \mid x) = p^y(1-p)^{(1-y)}$$

$$= \sigma\left(x^T\theta\right)^y\left(1 - \sigma\left(x^T\theta\right)\right)^{1-y}$$

How do we find the "best" θ?

Maximum Likelihood Method

What do we need?

The Maximum Likelihood Method

- Procedure for estimating model parameters

Steps of Maximum Likelihood

1. Define *likelihood* of the data (often assume IID → Prod. …)
2. Take the log of the likelihood function (simplifies math: Prod. → Sum)
3. Maximize *Log-Likelihood*
   1. Take the derivative
   2. Set equal to 0
   3. Solve
      - *if you can't solve use gradient descent*

Useful Log Identities:

$$\log(ab) = \log(a) + \log(b)$$
$$\log(a/b) = \log(a) - \log(b)$$
$$\log(a^b) = b\log(a)$$
$$\log(e) = 1$$

$$\mathcal{L}(\theta) = \prod_{i=1}^{n} \sigma\left(x_i^T\theta\right)^{y_i} \left(1 - \sigma\left(x_i^T\theta\right)\right)^{(1-y_i)}$$

Taking the log:

$$\log \mathcal{L}(\theta) = \sum_{i=1}^{n} y_i \log \sigma\left(x_i^T\theta\right) + (1 - y_i) \log\left(1 - \sigma\left(x_i^T\theta\right)\right)$$

Take the gradient (you **will need** to be able to take **derivatives** on the final!)

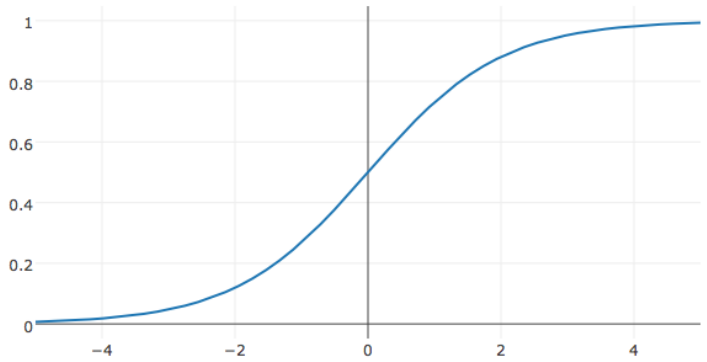
$$\nabla_\theta \log \mathcal{L}(\theta) = \sum_{i=1}^{n} \left(y_i - \sigma\left(x_i^T\theta\right)\right) x_i$$

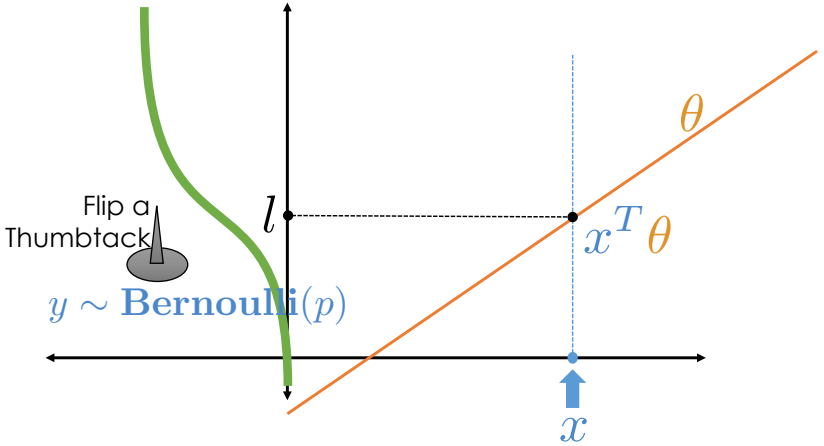
Couldn't solve for gradient = 0!! → Gradient Descent …

Recap: Logistic Regression

- Defined a model: $y \sim \textbf{Bernoulli}\left(\sigma\left(x^T\theta\right)\right)$

$$\sigma(t) = \frac{1}{1+e^{-t}}$$





- Constructed a likelihood function:

$$\mathcal{L}\left(\theta\right) = \prod_{i=1}^{n} \sigma\left(x_i^T\theta\right)^{y_i} \left(1-\sigma\left(x_i^T\theta\right)\right)^{(1-y_i)}$$

- Constructed a likelihood function:

$$\mathcal{L}(\theta) = \prod_{i=1}^{n} \sigma\left(x_i^T \theta\right)^{y_i} \left(1 - \sigma\left(x_i^T \theta\right)\right)^{(1-y_i)}$$

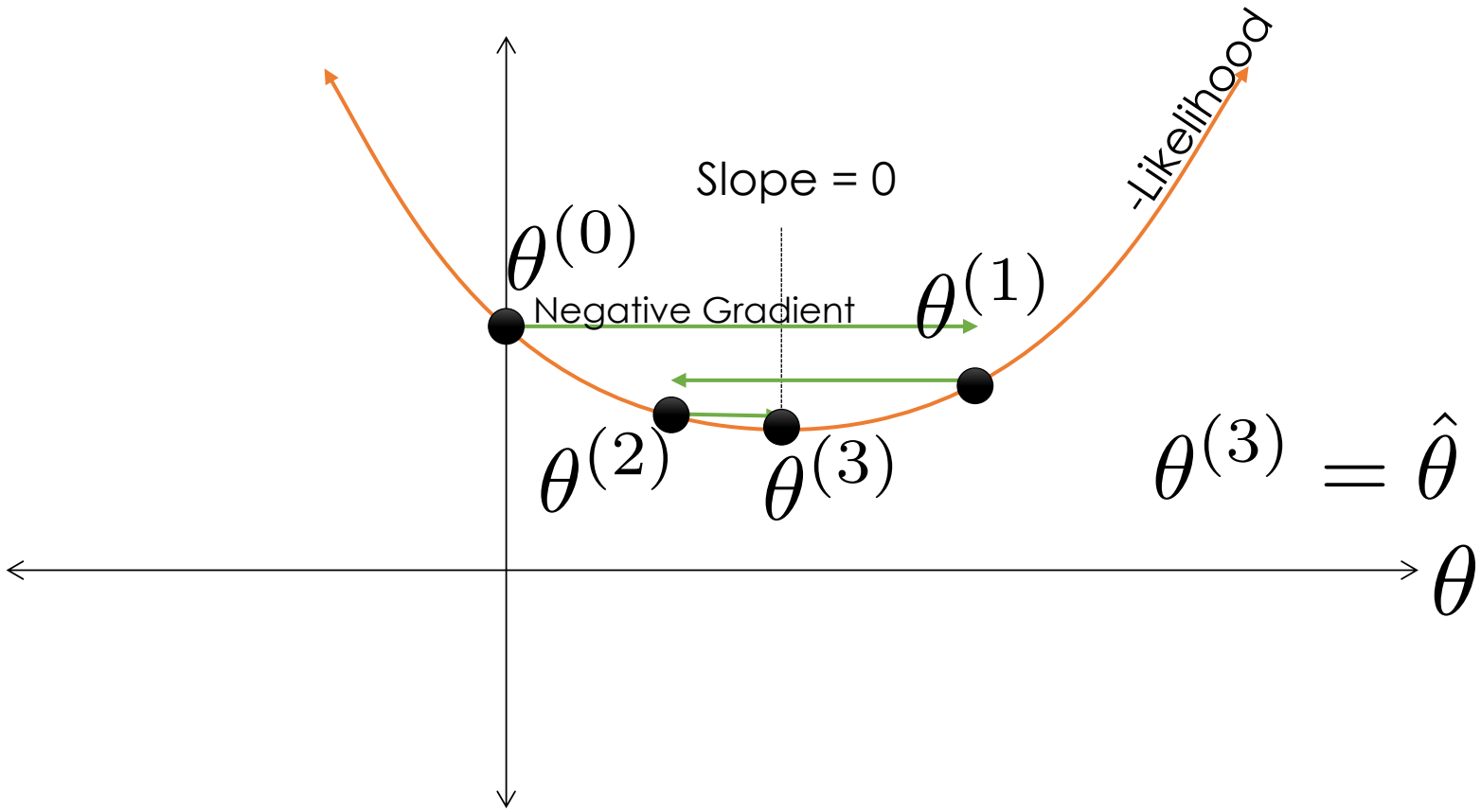
- Computed the gradient of the log-likelihood

$$\nabla_\theta \log \mathcal{L}(\theta) = \sum_{i=1}^{n} \left(y_i - \sigma\left(x_i^T \theta\right)\right) x_i$$

  - Set equal to 0 and solve … no analytic solution ☹

- Gradient Descent

Gradient Descent Intuition



Simple, fast, and works well in high dimensions.

- Gradient Descent

$\theta^{(0)} \leftarrow$ random initial vector

For τ from 0 to convergence

$$\theta^{(\tau+1)} \leftarrow \theta^{(\tau)} - \rho(\tau)\nabla_\theta(-\log \mathcal{L}(\theta))\Big|_{\theta^{(\tau)}}$$

Evaluated at $\theta^{(\tau)}$

Maximum Likelihood → Minimize ***Negative*** Log-likelihood

- The learning rate $\rho(\tau)$ determines the step size.
  - Typically:

$$\rho(\tau) = \frac{1}{\tau}$$

  - Should be a positive decreasing function

➢ Gradient Descent

$\theta^{(0)} \leftarrow$ random initial vector

For τ from 0 to convergence

$$\theta^{(\tau+1)} \leftarrow \theta^{(\tau)} - \rho(\tau)\nabla_\theta(-\log \mathcal{L}(\theta))\Big|_{\theta^{(\tau)}}$$

Evaluated at $\theta^{(\tau)}$

➢ Stochastic Gradient Descent (SGD)

$\theta^{(0)} \leftarrow$ random initial vector

For τ from 0 to convergence

$\mathcal{B} \leftarrow$ select *k* random indices

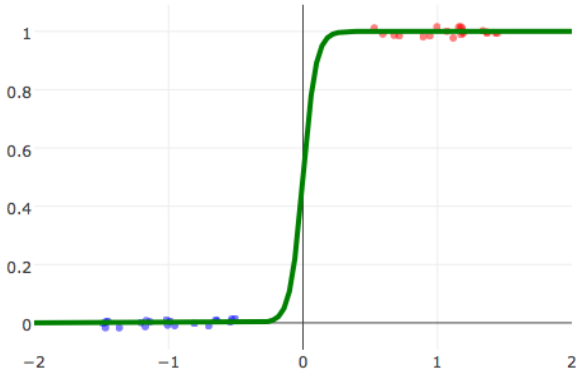
$$\theta^{(\tau+1)} \leftarrow \theta^{(\tau)} - \rho(\tau)\frac{n}{k}\sum_{i\in\mathcal{B}}\nabla_\theta(-\log \mathcal{L}(\theta \mid x_i, y_i)\Big|_{\theta^{(\tau)}}$$

Eval. at $\theta^{(\tau)}$

Estimate the gradient by **sampling** the data

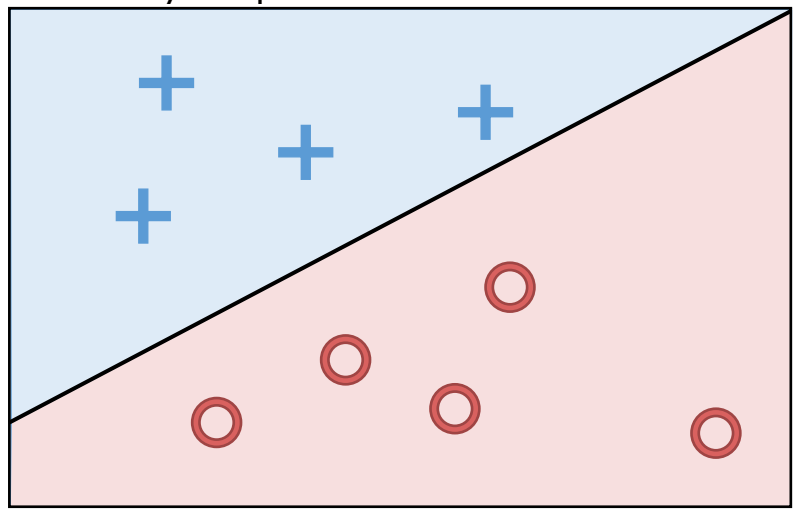
Linearly Separable Data

- A classification dataset is said to be linearly separable if there exists a hyperplane that separates the two classes.
- If data is linearly separable, logistic regression requires regularization

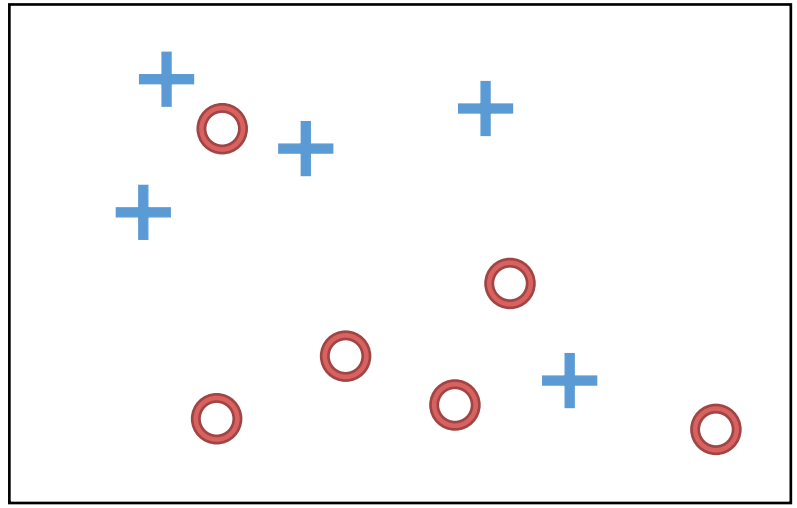


Otherwise weights go to infinity!

Linearly Separable Data

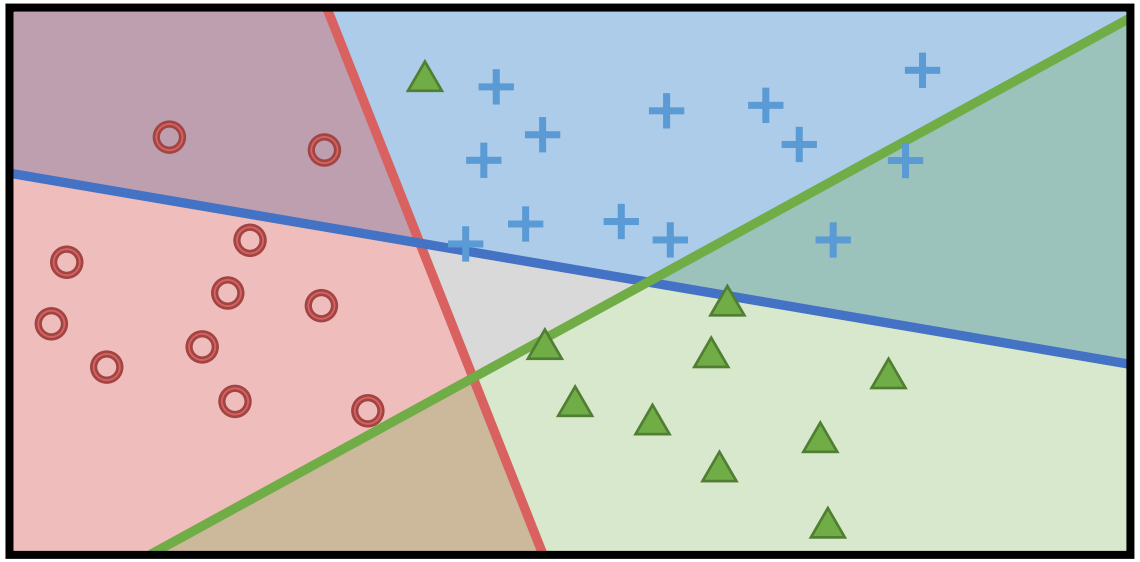


Not Linearly Separable Data



Multiclass (more than 2) Classification

- **One-vs-rest** train separate binary classifiers for each class
  - Class with highest confidence wins
  - Need to address class imbalance issue

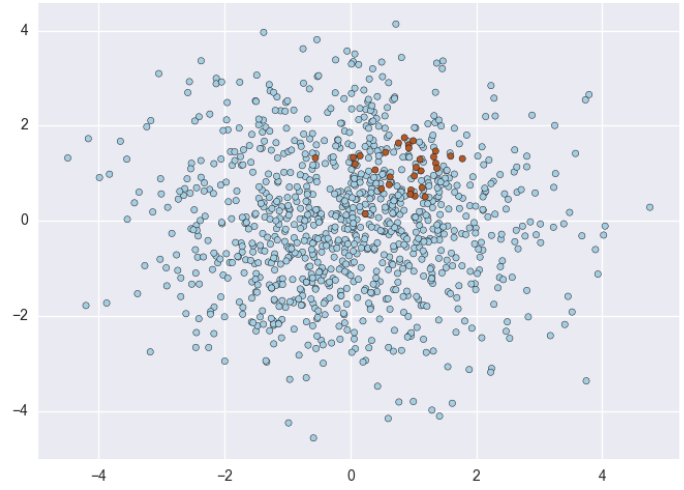


- **Soft-Max** multiclass classification
  - Covered in lab but not on exam

Dealing with Class Imbalance

Suppose 95% of your data are from one class (e.g., Spam)

- What if you always predict the majority class?
  - *f*(email) = Spam
  - Accuracy: 95% (not bad! or is it?)
  - The majority class proportion is a good baseline …
- How to address Class Imbalance
  - collect more data ← ideal
  - resample existing data ← *commonly used*
  - Adjust loss function … ← can be complicated



How to Study

- Review lecture notes **including the python notebooks**
- Be familiar with key concepts:
  - When to apply classification vs regression techniques
  - The maximum likelihood method
  - Logistic regression likelihood function
  - Gradient descent algorithm
  - Linear Separability
  - Class Imbalance
- Not on the exam:
  - deep-learning & soft-max loss