# ECE 5984: Introduction to Machine Learning

Topics:

- (Finish) Model selection
- Error decomposition
  - Bias-Variance Tradeoff
- Classification: Naïve Bayes

Readings: Barber 17.1, 17.2, 10.1-10.3

Dhruv Batra

Virginia Tech

# Administrativia

- HW2
  - Due: Friday 03/06, 11:55pm
  - Implement linear regression, Naïve Bayes, Logistic Regression

- Need a couple of catch-up lectures
  - How about 4-6pm?

# Administrativia

- Mid-term
  - When: March 18, class timing
  - Where: In class

  - Format: Pen-and-paper.
  - Open-book, open-notes, closed-internet.
    - No sharing.

  - What to expect: mix of
    - Multiple Choice or True/False questions
    - “Prove this statement”
    - “What would happen for this dataset?”

  - Material
    - Everything from beginning to class to (including) SVMs

# Recap of last time

# Regression

# Polynomial regression

- Consider 1D for simplicity:

$$f(x; \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_m x^m.$$

- No longer linear in $x$ – but still linear in $\mathbf{w}$!

# Polynomial regression

- Consider 1D for simplicity:

$$f(x; \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_m x^m.$$

- No longer linear in $x$ – but still linear in $\mathbf{w}$!
- Define $\phi(\mathbf{x}) = [1, x, x^2, \ldots, x^m]^T$
- Then, $f(x; \mathbf{w}) = \mathbf{w}^T \phi(\mathbf{x})$ and we are back to the familiar simple linear regression. The least squares solution:

$$\hat{\mathbf{w}} = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{y}, \text{ where } \mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^m \\ 1 & x_2 & x_2^2 & \ldots & x_2^m \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & x_N & x_N^2 & \ldots & x_N^m \end{bmatrix}$$

# General additive regression models

$$f(\mathbf{x};\mathbf{w}) = w_0 + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \ldots + w_m\phi_m(\mathbf{x}),$$

- Still the same ML estimation technique applies:

$$\hat{\mathbf{w}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

where $\mathbf{X}$ is the *design matrix*

$$\begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \ldots & \phi_m(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \ldots & \phi_m(\mathbf{x}_2) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \ldots & \phi_m(\mathbf{x}_N) \end{bmatrix}$$

(for convenience we will denote $\phi_0(\mathbf{x}) \equiv 1$)

# What you need to know

- Linear Regression
  - Model
  - Least Squares Objective
  - Connections to Max Likelihood with Gaussian Conditional
  - Robust regression with Laplacian Likelihood
  - Ridge Regression with priors
  - Polynomial and General Additive Regression

# Plan for Today

- (Finish) Model Selection
  - Overfitting vs Underfitting
  - Bias-Variance trade-off
    - aka Modeling error vs Estimation error tradeoff

- Naïve Bayes

# New Topic: Model Selection and Error Decomposition

# Example for Regression

- Demo
  - http://www.princeton.edu/~rkatzwer/PolynomialRegression/
- How do we pick the hypothesis class?

# Model Selection

- How do we pick the right model class?

- Similar questions
  - How do I pick magic hyper-parameters?
  - How do I do feature selection?

# Errors

- Expected Loss/Error

- Training Loss/Error
- Validation Loss/Error
- Test Loss/Error

- Reporting Training Error (instead of Test) is CHEATING

- Optimizing parameters on Test Error is CHEATING

# Cross-validation

- The improved holdout method: $k$-fold *cross-validation*
  - Partition data into $k$ roughly equal parts;
  - Train on all but $j$-th part, test on $j$-th part

# Cross-validation

- The improved holdout method: $k$-fold *cross-validation*
  - Partition data into $k$ roughly equal parts;
  - Train on all but $j$-th part, test on $j$-th part

# Cross-validation

- The improved holdout method: $k$-fold *cross-validation*
  - Partition data into $k$ roughly equal parts;
  - Train on all but $j$-th part, test on $j$-th part

# Cross-validation

- The improved holdout method: $k$-fold *cross-validation*
  - Partition data into $k$ roughly equal parts;
  - Train on all but $j$-th part, test on $j$-th part

# Cross-validation

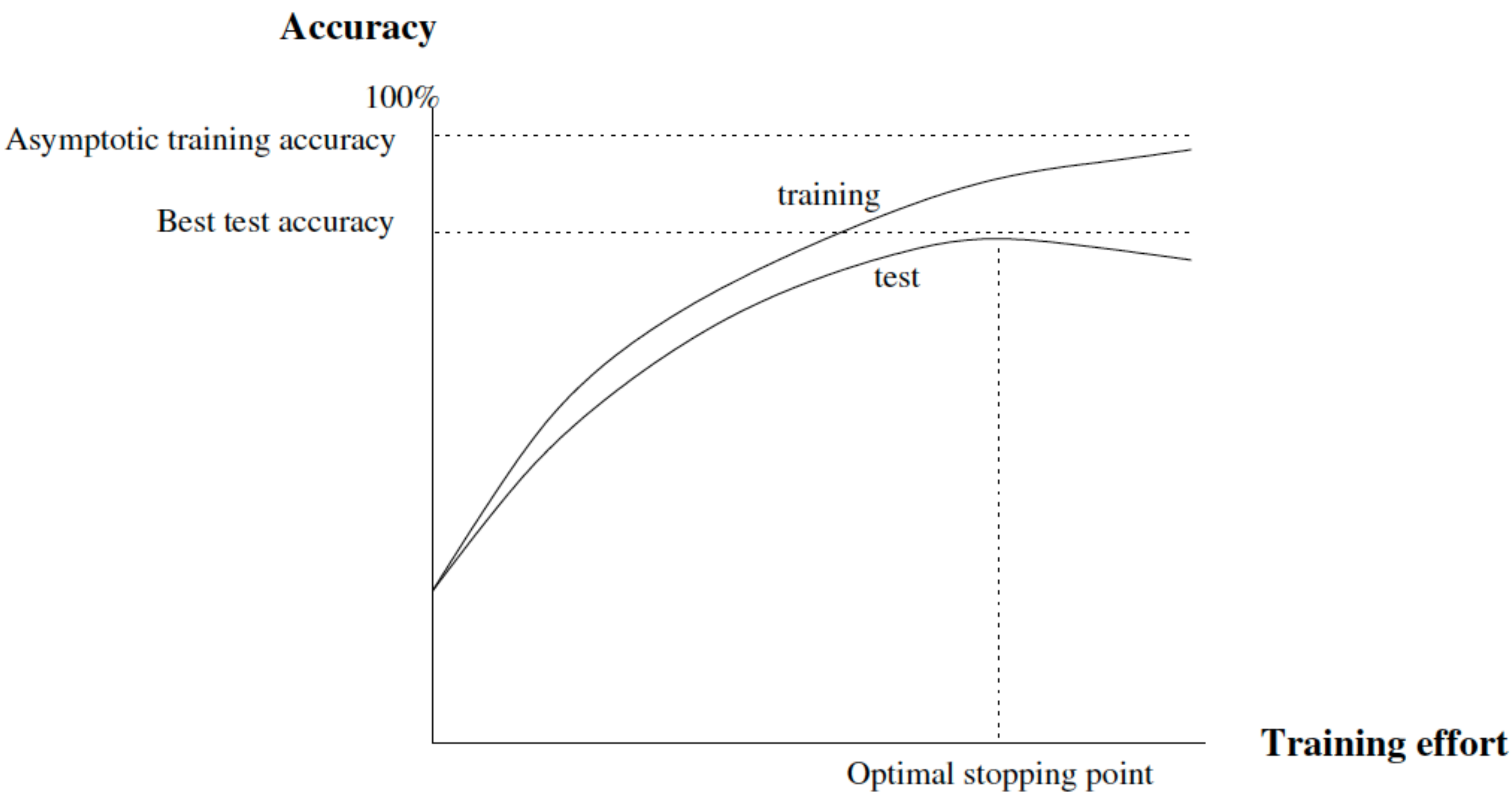
- The improved holdout method: $k$-fold *cross-validation*
  - Partition data into $k$ roughly equal parts;
  - Train on all but $j$-th part, test on $j$-th part


- An extreme case: *leave-one-out* cross-validation

$$\hat{L}_{\text{cv}} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i; \hat{\mathbf{w}}_{-i})\right)^2$$

where $\hat{\mathbf{w}}_{-i}$ is fit to all the data but the $i$-th example.

# Typical Behavior

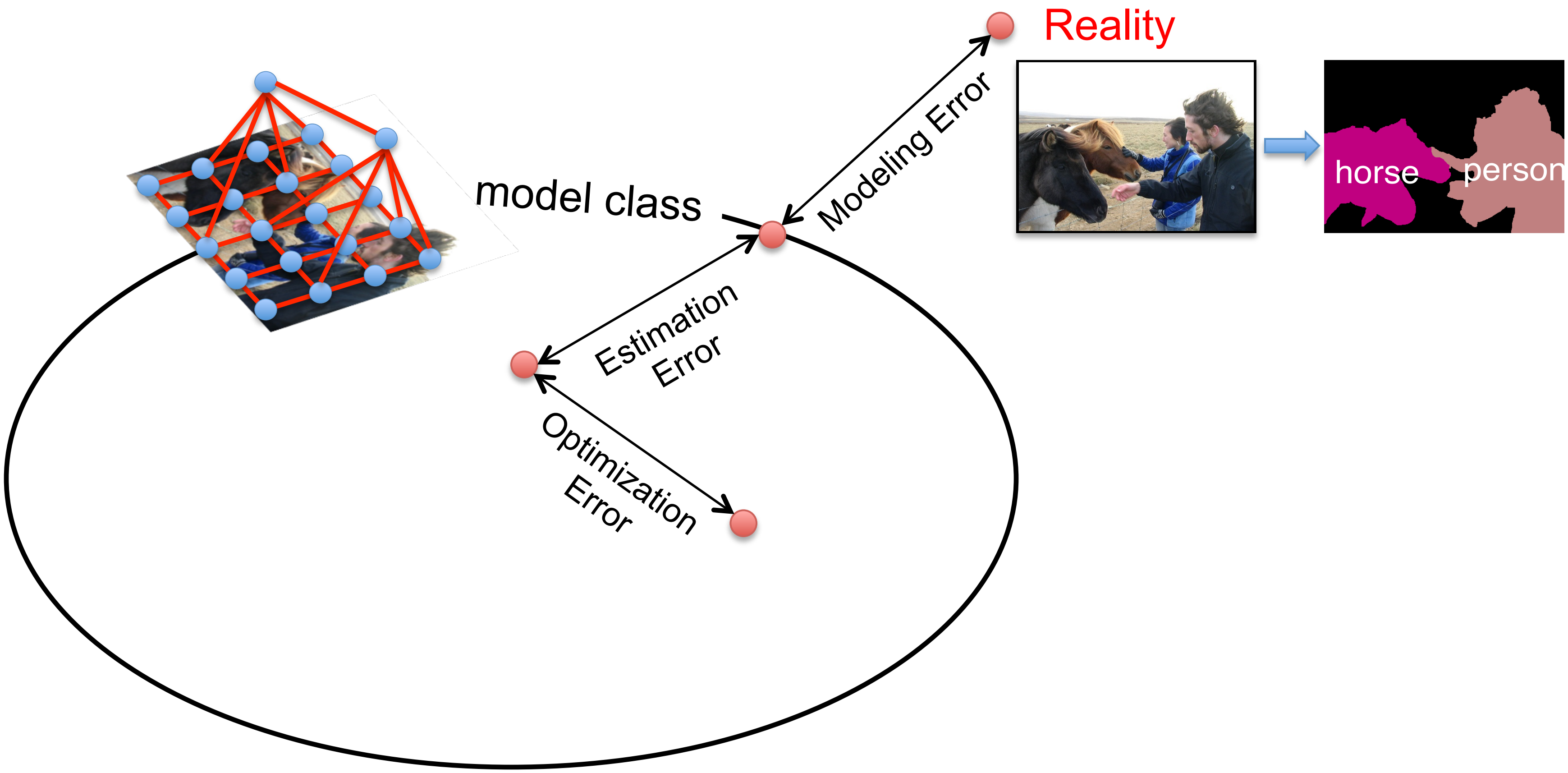


# Overfitting

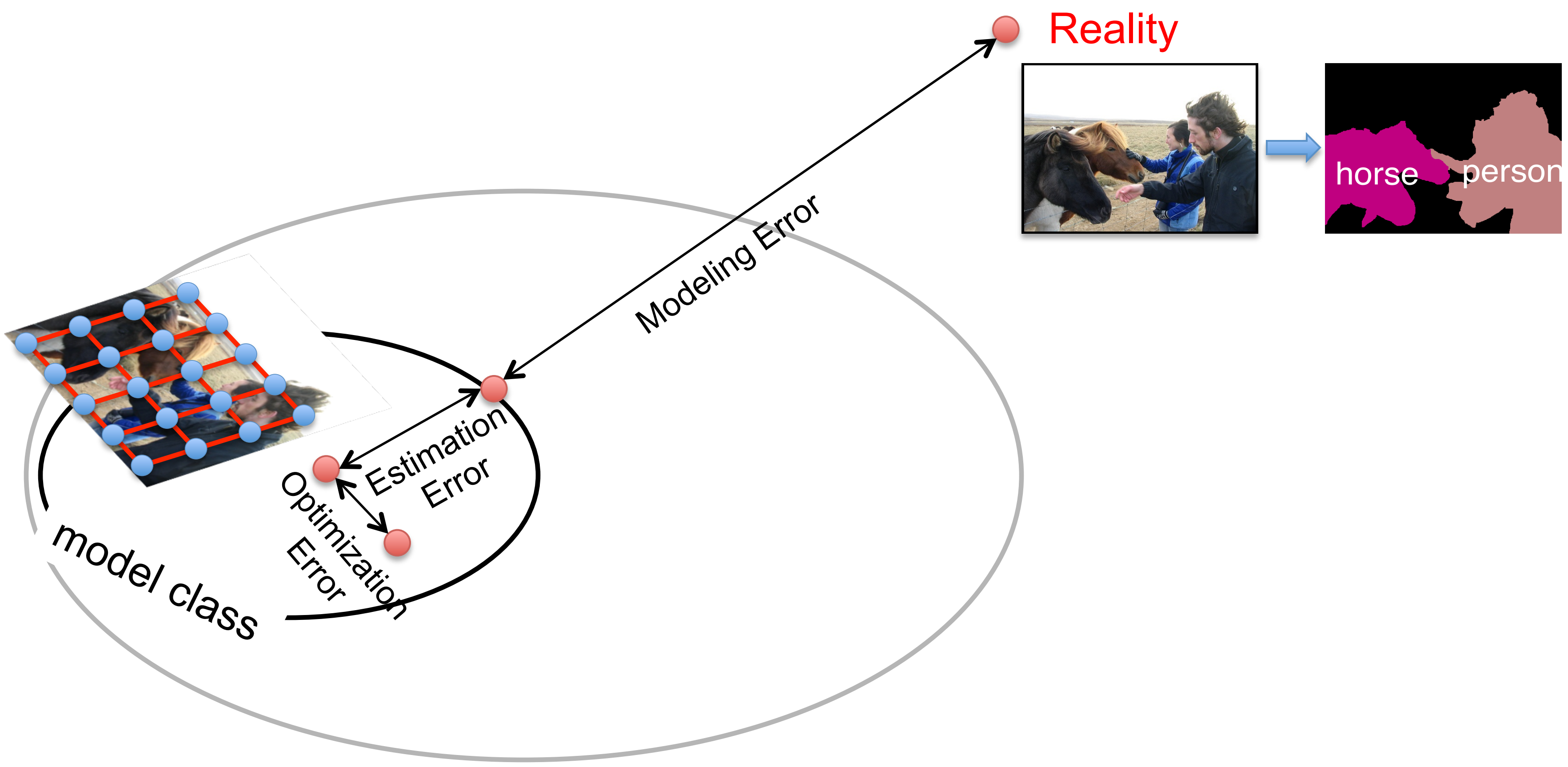
- **Overfitting:** a learning algorithm overfits the training data if it outputs a solution **w** when there exists another solution **w'** such that:

$$[error_{train}(\mathbf{w}) < error_{train}(\mathbf{w}')] \wedge [error_{true}(\mathbf{w}') < error_{true}(\mathbf{w})]$$

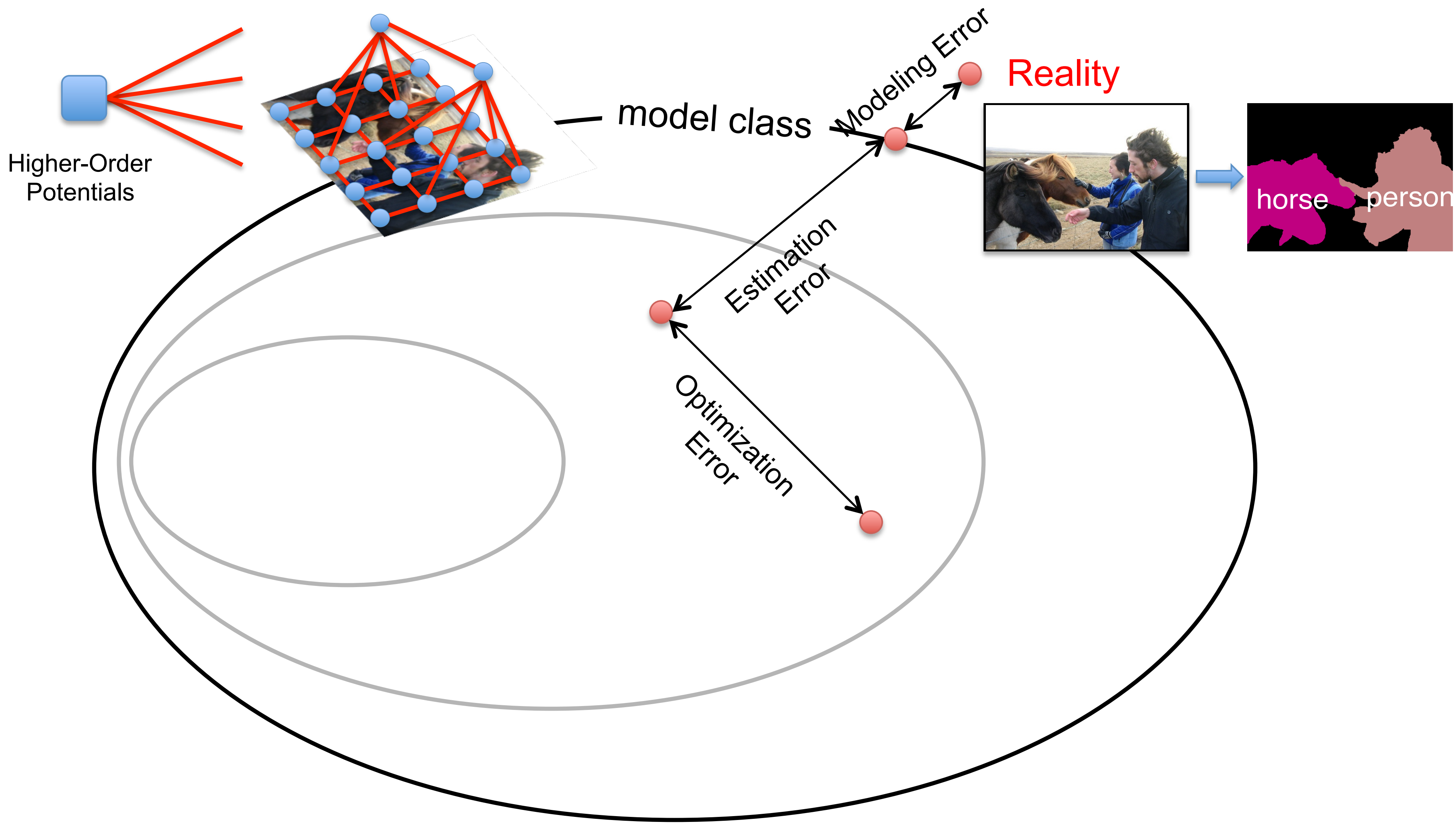
# Error Decomposition



# Error Decomposition



# Error Decomposition



# Error Decomposition

- Approximation/Modeling Error
  - You approximated reality with model
- Estimation Error
  - You tried to learn model with finite data
- Optimization Error
  - You were lazy and couldn't/didn't optimize to completion
- (Next time) Bayes Error
  - Reality just sucks

# Bias-Variance Tradeoff

- **Bias:** difference between what you expect to learn and truth
  - Measures how well you expect to represent true solution
  - Decreases with more complex model

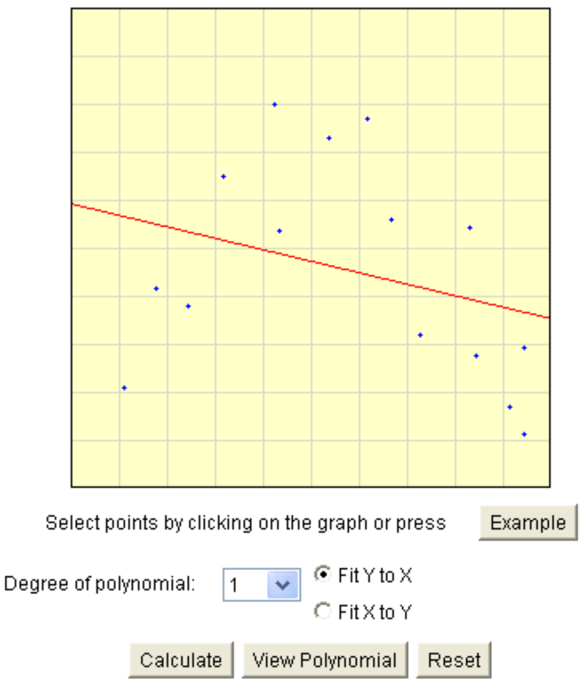
- **Variance:** difference between what you expect to learn and what you learn from a from a particular dataset
  - Measures how sensitive learner is to specific dataset
  - Increases with more complex model

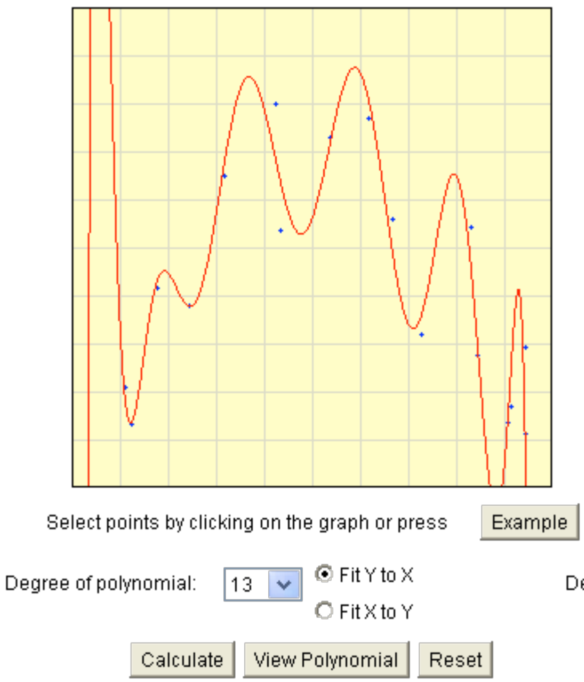
# Bias-Variance Tradeoff

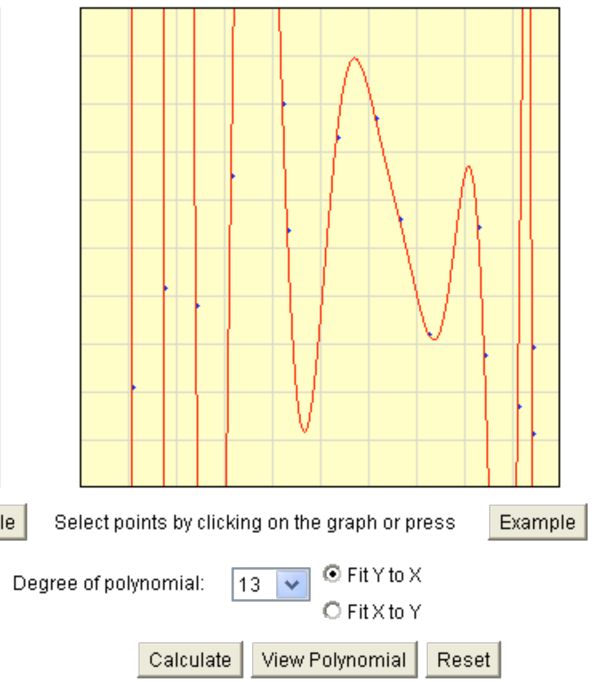
- Matlab demo

# Bias-Variance Tradeoff

- Choice of hypothesis class introduces learning bias
  - More complex class → less bias
  - More complex class → more variance

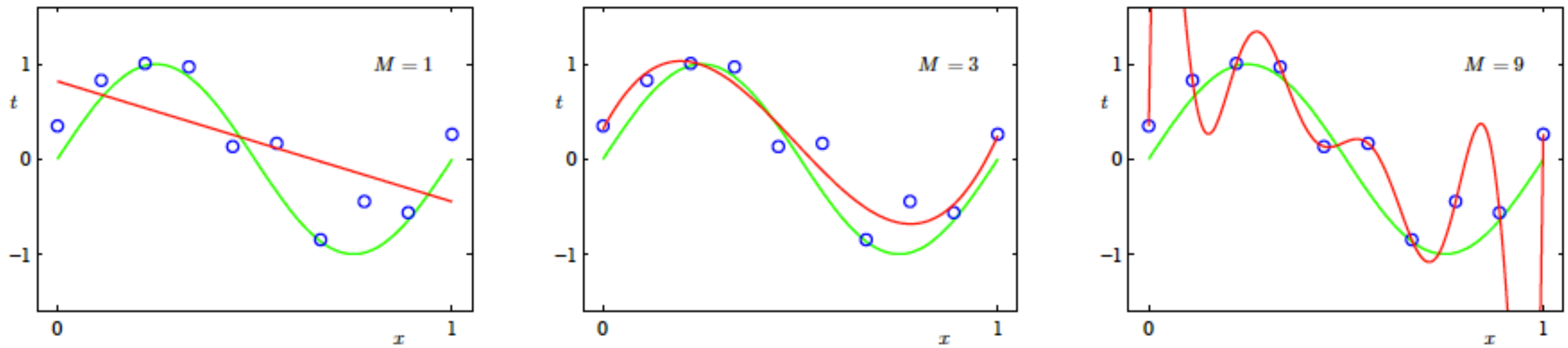


# Linear regression

- Example: polynomial regression, true [from Bishop, Ch. 1]




- Value of the optimal (ML) regression coefficients:

| | $m=0$ | $m=1$ | $m=3$ | $m=9$ |
|---|---|---|---|---|
| $w_0^*$ | 0.19 | 0.82 | 0.31 | 0.35 |
| $w_1^*$ | | -1.27 | 7.99 | 232.37 |
| $w_2^*$ | | | -25.43 | -5321.83 |
| $w_3^*$ | | | 17.37 | 48568.31 |
| $w_4^*$ | | | | -231639.30 |
| $w_5^*$ | | | | 640042.26 |
| $w_6^*$ | | | | -1061800.52 |
| $w_7^*$ | | | | 1042400.18 |
| $w_8^*$ | | | | -557682.99 |
| $w_9^*$ | | | | 125201.43 |

# Learning Curves

- Error vs size of dataset

- On board
  - High-bias curves
  - High-variance curves

# Debugging Machine Learning

- My algorithm does work
  - High test error

- What should I do?
  - More training data
  - Smaller set of features
  - Larger set of features
  - Lower regularization
  - Higher regularization

# What you need to know

- Generalization Error Decomposition
  - Approximation, estimation, optimization, bayes error
  - For squared losses, bias-variance tradeoff

- Errors
  - Difference between train & test error & expected error
  - Cross-validation (and cross-val error)
  - NEVER EVER learn on test data

- Overfitting vs Underfitting

# New Topic:
# Naïve Bayes
# (your first probabilistic classifier)

# Classification

- **Learn**: h:**X** $\mapsto$ Y
  - **X** – features
  - Y – target classes

- Suppose you know P(Y|**X**) exactly, how should you classify?
  - Bayes classifier:

- **Why?**

# Optimal classification

- **Theorem:** Bayes classifier $h_{Bayes}$ is optimal!
  - That is $error_{true}(h_{Bayes})) \leq error_{true}(h), \ \forall h(\mathbf{x})$

- **Proof**:

$$p(error_h) = \int_x p(error_h|x)p(x)dx$$

Slide Credit: Carlos Guestrin

# Generative vs. Discriminative

- Using Bayes rule, optimal classifier

$$h^*(\mathbf{x}) = \underset{c}{\operatorname{argmax}} \{\log p(\mathbf{x}|y=c) + \log p(y=c)\}$$

- Generative Approach
  - Estimate p(x|y) and p(y)
  - Use Bayes Rule to predict y

- Discriminative Approach
  - Estimate p(y|x) directly OR
  - Learn “discriminant” function h(x)

# Generative vs. Discriminative

- Generative Approach
  - Assume some functional form for P(X|Y), P(Y)
  - Estimate p(X|Y) and p(Y)
  - Use Bayes Rule to calculate P(Y| X=x)
  - Indirect computation of P(Y|X) through Bayes rule
  - But, **can generate a sample**, $P(X) = \sum_y P(y) P(X|y)$

- Discriminative Approach
  - Estimate p(y|x) directly OR
  - Learn "discriminant" function h(x)
  - Direct but cannot obtain a sample of the data, because P(X) is not available

# Generative vs. Discriminative

- Generative:
  - Today: Naïve Bayes

- Discriminative:
  - Next: Logistic Regression

- NB & LR related to each other.

# How hard is it to learn the optimal classifier?

- Categorical Data

- How do we represent these? How many parameters?
  - Class-Prior, P(Y):
    - Suppose Y is composed of *k* classes

  - Likelihood, P(**X**|Y):
    - Suppose **X** is composed of *d* binary features

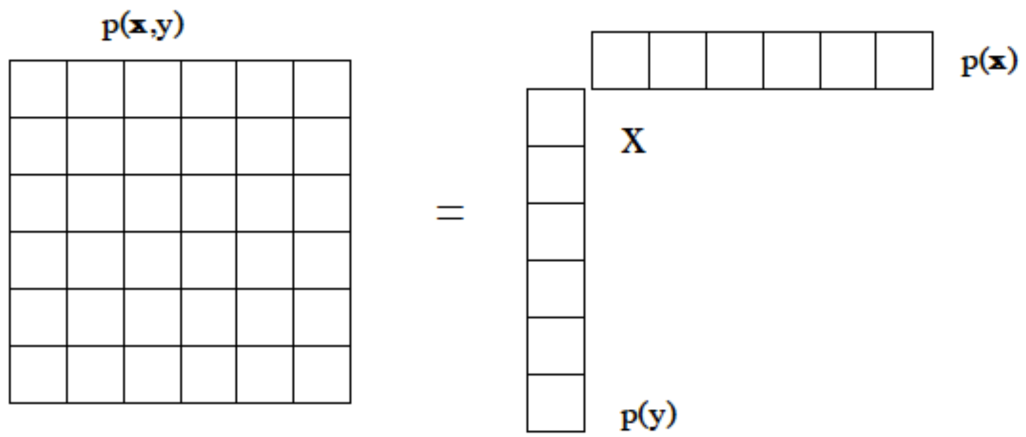
- Complex model → High variance with limited data!!!

Slide Credit: Carlos Guestrin

# Independence to the rescue

- Two variables are independent iff their joint factors:

$$p(x, y) = p(x)p(y)$$




- Two variables are conditionally independent given a third one if for all values of the conditioning variable, the resulting slice factors:

$$p(x, y|z) = p(x|z)p(y|z) \qquad \forall z$$

# The Naïve Bayes assumption

- Naïve Bayes assumption:
  - Features are independent given class:

$$P(X_1, X_2|Y) = P(X_1|X_2, Y)P(X_2|Y)$$
$$= P(X_1|Y)P(X_2|Y)$$

  - More generally:

$$P(X_1 \ldots X_d|Y) = \prod_i P(X_i|Y)$$

- How many parameters now?
  - Suppose **X** is composed of *d* binary features

# The Naïve Bayes Classifier

- Given:
  - Class-Prior P(Y)
  - *d* conditionally independent features **X** given the class Y
  - For each $X_i$, we have likelihood $P(X_i|Y)$
- Decision rule:

$$
\begin{aligned}
y^* = h_{NB}(\mathbf{x}) &= \arg\max_y P(y) P(x_1, \ldots, x_n \mid y) \\
&= \arg\max_y P(y) \prod_i P(x_i|y)
\end{aligned}
$$

- If assumption holds, NB is optimal classifier!