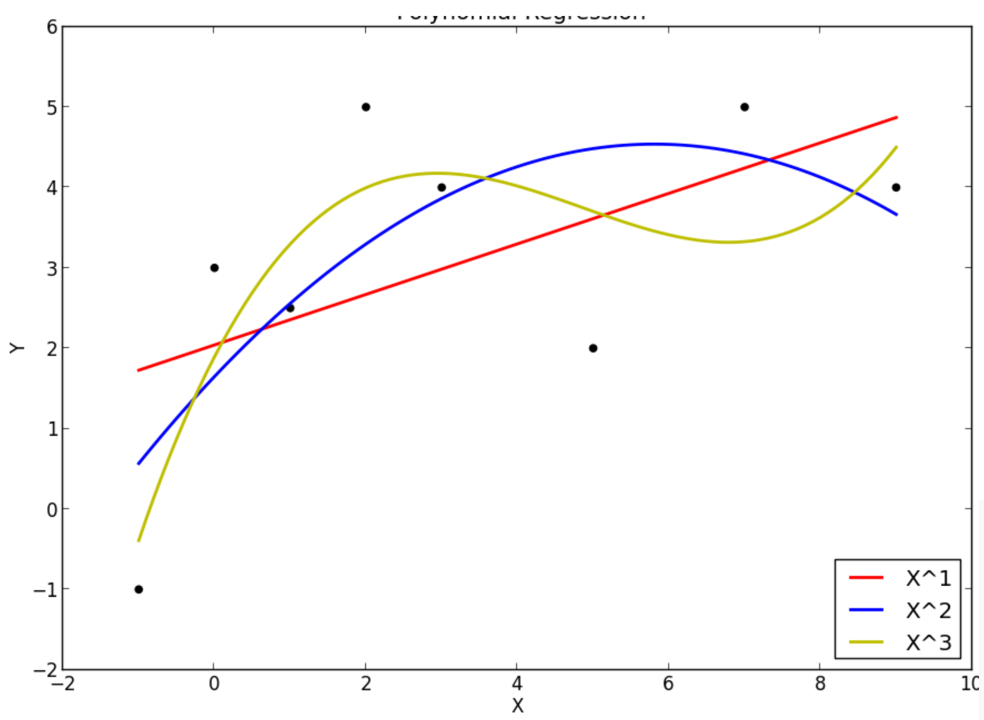
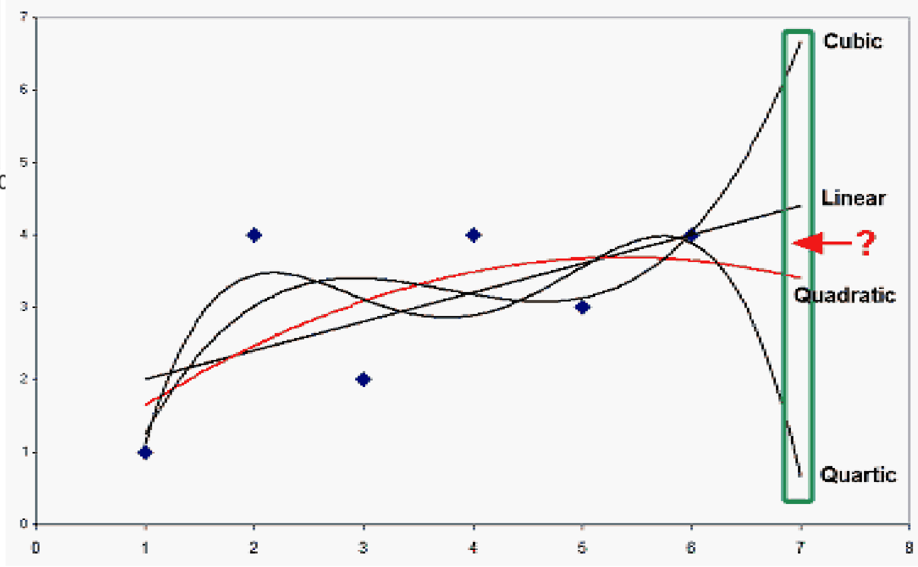
Comparing Models

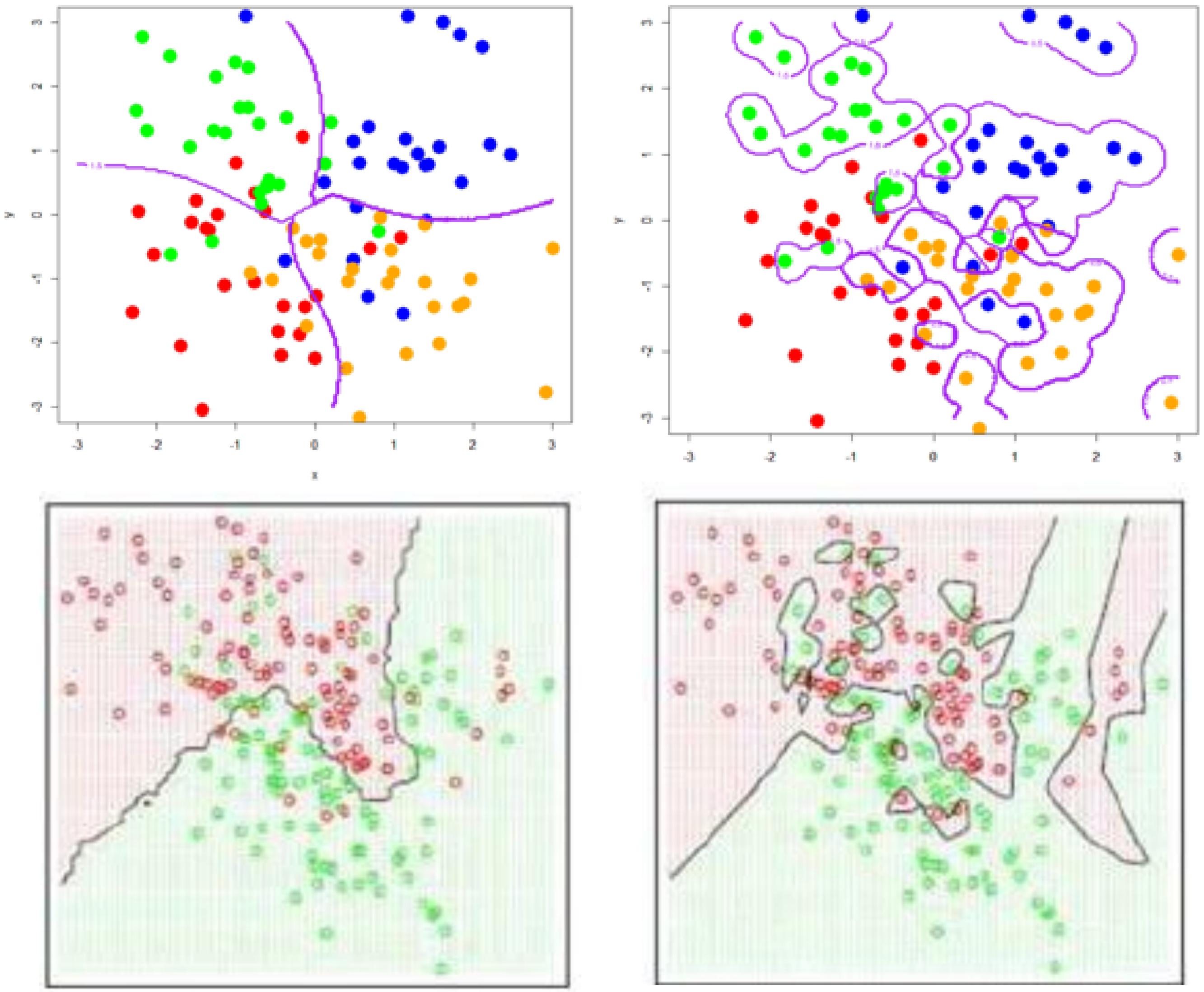
Model Complexity, Bias-Variance, Generalization Error, Overfitting, Hyperparameters vs. Parameters

Reducing error… at what cost? | Regression





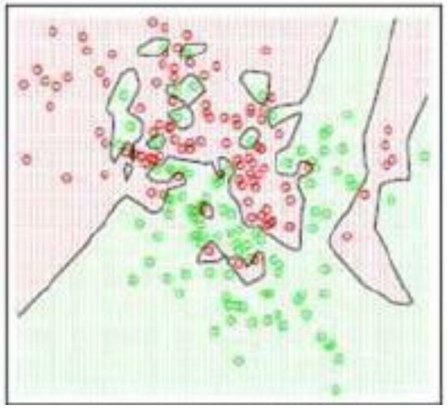
Reducing error… at what cost? | Classification

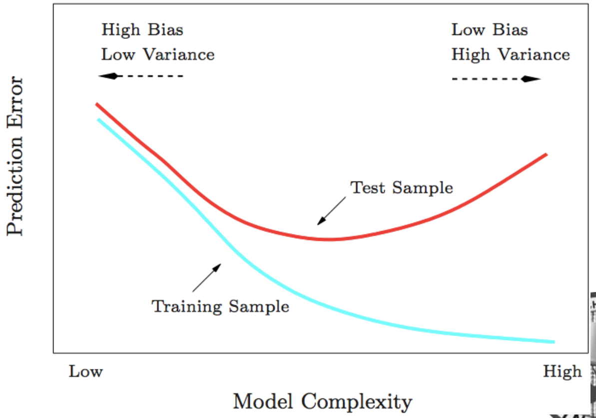


Model Evaluation : Error vs. Complexity

- Intuition
  - Some models are "un-necessarily complex"
  - Some models tend to "over fit" the given data
  - Does a model "overfit"?
    - Visual inspection not always feasible
    - High dimensional data *(too many variables, features)*
- Approach: Constrain the complexity of the model
  - Define statistic on the data *(statistical approach)*
  - Adjusted R2 : Explained Variance normalized with DoF
  - AIC / BIC / Cp : penalizes number of parameters in model
- Approach: Measure model performance on "new" data
  - Split available data
    - Learn model using "Training data; Evaluate on "Test data"
  - Train vs. Test Data : Train vs. Test Error
  - Try it out on test data *(computational approach)*

- BIG Idea: Generalization Error
  - How does model perform on data it did not learn from?
  - Model Complexity / Flexibility vs. Model Performance
  - Lower Training error does not always imply Lower Test Error!
- Equivalence
  1. Model Overfits
  2. Model reduces training error with an over-complex model
  3. Model reduces training error but test error increases

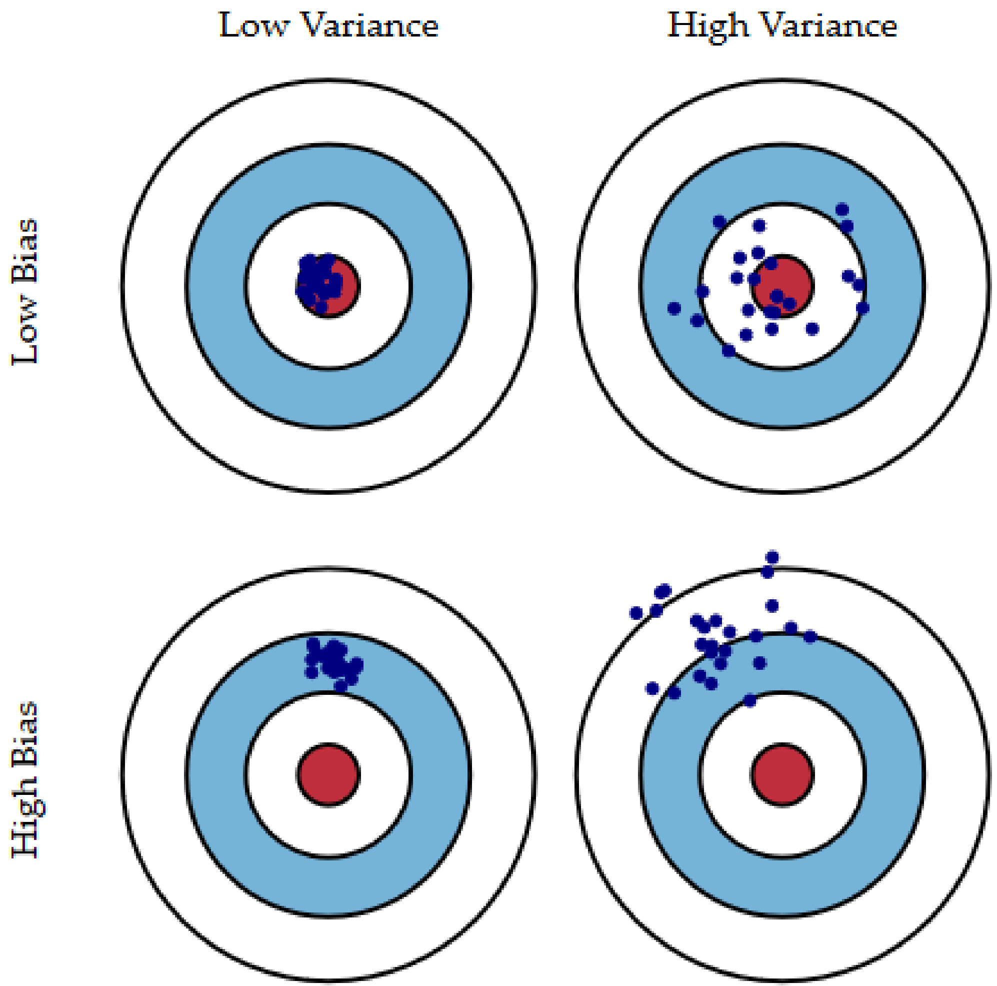




Model Evaluation : Bias vs. Variance tradeoff

- Model Bias
  - Error due to the assumptions (limitations) of the model
  - E.g. linearity, continuous functions.
  - High bias ➔ Look for a different class of functions
    - more "flexible"
    - More complex

- Model Variance
  - How much does the model change with a change in sample?
  - Sensitivity to change in sample (training data)
  - High variance ➔



Bias vs. Variance Tradeoff

- Function Approximation framework
  - Learn a function from the data (which minimizes some error)

$$y = f(x) + \varepsilon$$
$$\varepsilon \sim \mathrm{N}(0, \sigma)$$

$$\hat{y} = \hat{f}(x) + 0$$
$$\mathrm{P}(y \mid x)$$

- Error
  - Depends on the sample
  - Depends on the choice of the model family

$$\mathrm{E}\left[\left(y - \hat{f}(x)\right)^2\right]$$

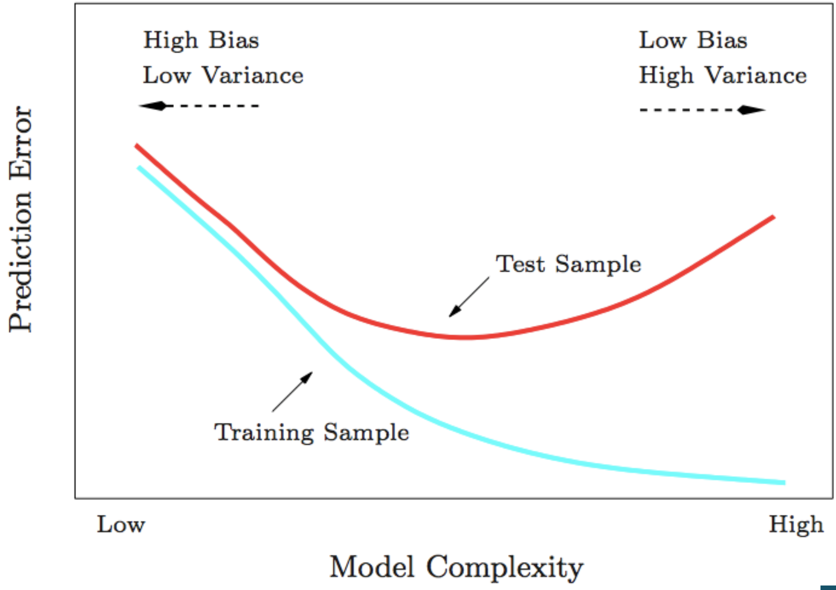
$$\mathrm{E}\left[\left(y - \hat{f}(x)\right)^2\right] = \mathrm{Bias}\left[\hat{f}(x)\right]^2 + \mathrm{Var}\left[\hat{f}(x)\right] + \sigma^2$$

- Bias-vs-Variance Tradeoff
  - Increase complexity to reduce bias
  - ➔ Make it more sensitive to the data
  - ➔ Make it more sensitive to the training data (sample)
  - ➔ Increase Variance

$$\mathrm{Bias}\left[\hat{f}(x)\right] = \mathrm{E}\left[\hat{f}(x) - f(x)\right]$$

$$\mathrm{Var}\left[\hat{f}(x)\right] = \mathrm{E}[\hat{f}(x)^2] - \mathrm{E}[\hat{f}(x)]^2$$

What is a good model : Summary



- Model Complexity & Overfitting
  - Trying to reduce training error with a more complex model
  - More degrees of freedom (More variables, features)
  - Error can be reduced with more complex models: When is it overfiitting?
  - Lower Training error does not always imply Lower Test Error!

- Bias Variance Tradeoff
  - Bias: Error introduced due to simplifying the real world with a “simple” model.
  - Variance: How much does the model vary if we train it on a different training set?
  - Tradeoff: Increasing Complexity ➔ Lower Bias but may lead to overfitting (higher variance)

- Approaches for model evaluation
  - Validation Set, LOOCV, K-fold
  - Given Data = Training + Test
  - Given Data = Training + Calibration + Test (Later)

Complexity-aware Model Evaluation

Validation Set

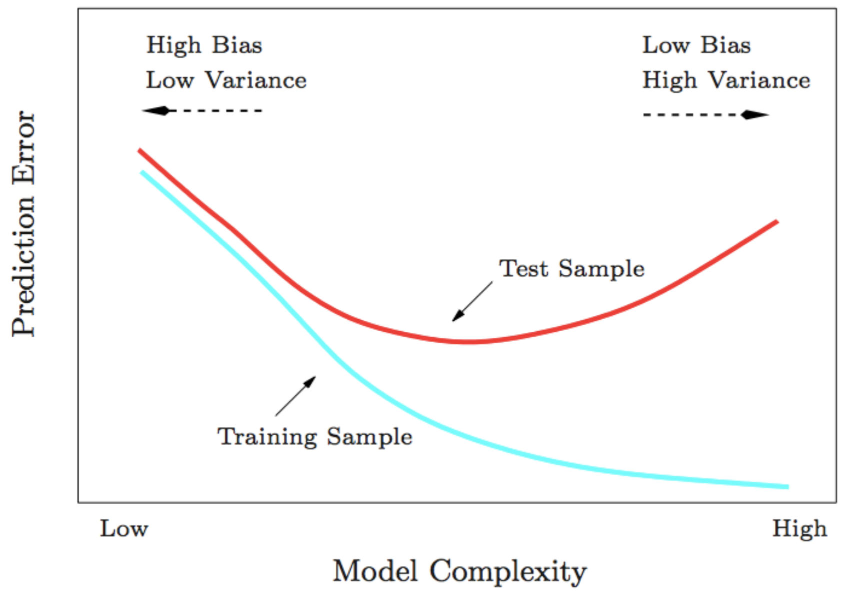
- Key Idea : Assume you have less data available than you actually have
- Split your data into training & test (validation)
- Learn the model on training set. Evaluate (Test) it on validation

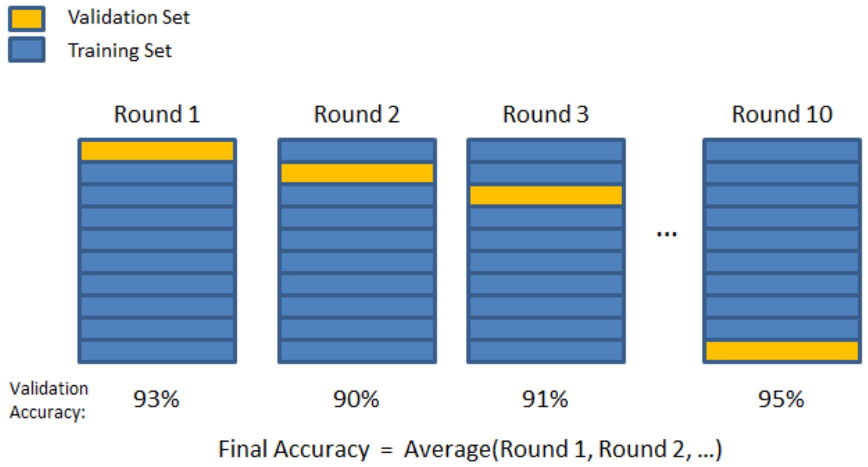
LOOCV

- Validation Set = 1 instance
- Learn the model on training set. Evaluate (Test) it on validation
- Repeat (Go to step-1)

K-Fold CV

- Validation Set = 1 sub-set
- Learn the model on training set. Evaluate (Test) it on validation
- Repeat (Go to step-1)
- Gold Standard :
    - More stable than validation set;
    - Less computationally intensive than LOOCV





Q?

Praphul Chandra

Statistical Decision Theory

Praphul Chandra

Statistical Decision Theory

- Framework
  - Function Approximation
  - Joint Probability Distribution
  - Loss Function

Function Approximation: $Y = f(X)$
Joint Distribution: $\mathbb{P}(X, Y)$
Loss Function: $L(Y, f(X))$

- Loss Variants
  - L2 (Squared Error Loss)
  - L1 Loss

$$L(Y, f(X)) = (Y - f(X))^2$$

- Expected Prediction Error
  - Choosing the "best" function
  - Depends on choice of loss function
  - **L2**: The best prediction of Y at an point X=x is the conditional **mean**.
  - **L1**: The best prediction of Y at an point X=x is the conditional **median**

$$\begin{aligned} EPE(f) &= \mathbb{E}[(Y - f(X))^2] = \int [y - f(x)]^2 \mathbb{P}(dx, dy) \\ &= \mathbb{E}_X \mathbb{E}_{Y|X}[(Y - f(X))^2 | X] \end{aligned}$$

$$\begin{aligned} f(x) &= \arg\min_c \mathbb{E}_{Y|X}[(Y - c^2 | X = x] \\ &= \mathbb{E}[Y | X = x] \end{aligned}$$

The best prediction of Y at an point X=x is….

$$\text{Loss Function}: \sum_{i=1}^{n} L(y_i, c) = \sum_{i=1}^{n} (y_i - c)^2$$

$$\text{Minimize Loss}: \frac{d}{dc}\sum_{i=1}^{n} (y_i - c)^2 = 0$$

$$-1 \times 2 \times \sum_{i=1}^{n} (y_i - c) = 0 \Rightarrow \sum_{i=1}^{n} (y_i - c) = 0$$

$$\sum_{i=1}^{n} y_i = nc \Rightarrow c = \frac{1}{n}\sum_{i=1}^{n} y_i$$

c is the mean of y_i

$$\text{Loss Function}: \sum_{i=1}^{n} L(y_i, c) = \sum_{i=1}^{n} |y_i - c|$$

$$\text{Minimize Loss}: \frac{d}{dc}\sum_{i=1}^{n} |y_i - c| = 0$$

$$-sign \sum_{i=1}^{n} |y_i - c| = 0$$

Derivate vanishes when there is the same number of positive and negative terms among the y_i - c which (roughly speaking) arises when cis the median of the y_i.

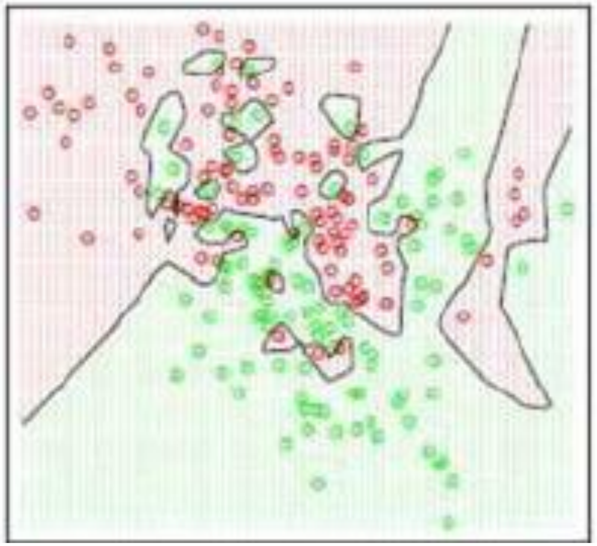
K-Nearest Neighbor

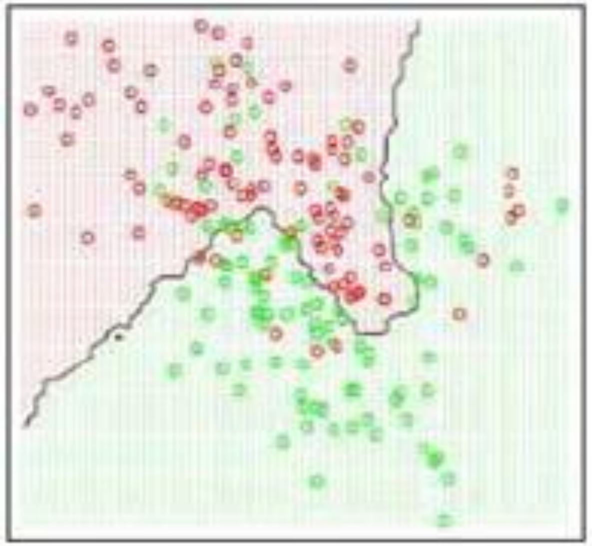
- Statistical Decision Theory
  - The best prediction of Y at an point X=x is the conditional mean. (L2 loss)
  - knn: At each point x, approximate y by averaging all y_i with input x_i near x

$$f(x) = \mathbb{E}[Y|X = x]$$

$$\hat{f}(x) = Ave(y_i|x_i \in N_k(x))$$

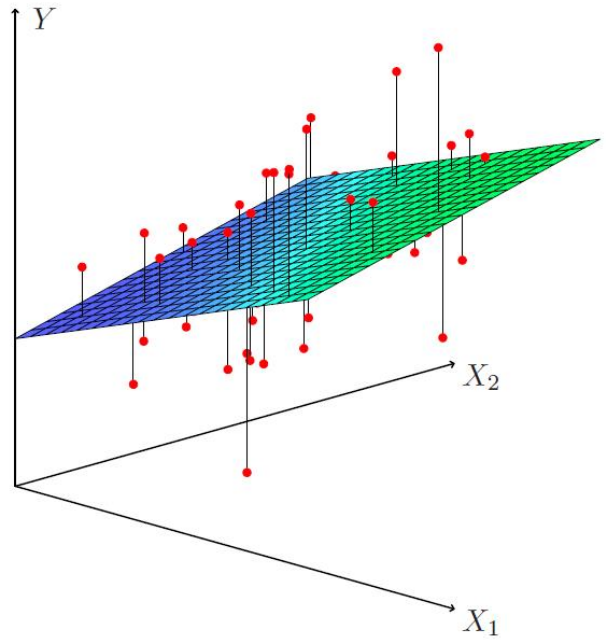
- Two approximations
  - Expectation is approximated by averaging over sample data.
  - Conditioning at a point x is relaxed to conditioning on some region "close" to x
- Note
  - Model Free *(No assumption on form of f)*
  - Computational Complexity *(Time, Space)*
  - Locally constant
- Behavior
  - Large k : Smoother boundaries
  - Large N : Large storage req. (space complexity)
  - Large p : lower accuracy (curse of dimensionality)





Linear Regression

- Statistical Decision Theory
  - The best prediction of Y at an point X=x is the conditional mean. (L2 loss)
  - LR : Find a linear function which minimizes the total loss (sum of least squares) across x
- Two approximations
  - Global function
  - Linearity
- Note
  - Model Based *(f() is Globally Linear)*
  - Computational Complexity *(Time, Space)*
- Behavior
  - Large N : Larger training time (computational complexity)
  - Large p : potentially lower accuracy (linearity in higher dimensions)
  - Larger k?? *(Feature Expansion – Later)*



knn: Summary

- The best prediction of Y at an point X=x is the conditional mean. (L2 loss)
- At each point x, approximate y by averaging all y_i with input x_i near x

- Lazy | Model Free *(No assumption on form of f)*
- Computational Complexity *(Time, Space)*

- Distance based algorithm
  - Scaling attributes is important
  - Attributes with larger range can dominate e.g., Age versus Salary
  - May not be suitable for high dimensional data
- Categorical variables and Ordinal variables need to be appropriately measured in distance
  - Think distance w.r.t the target

Statistical Decision Theory: Summary $Y = f(X)$

f	(X)	$L(Y, f(X))$

• Constant
• Linear
• Non-Linear
  • Polynomial
• Piecewise
  • Splines & Kinks
• Additive
	• Global
	• Local
	• Kernel
	• Basis Transformation
	  • Expansion
	  • Reduction
	  • Learn (Dictionary)
	• Manifold
		• Distance Measure
		  • L2, L1, etc.
		  • Hinge Loss
		• Overfitting
		  • Regularization
		  • Penalize roughness