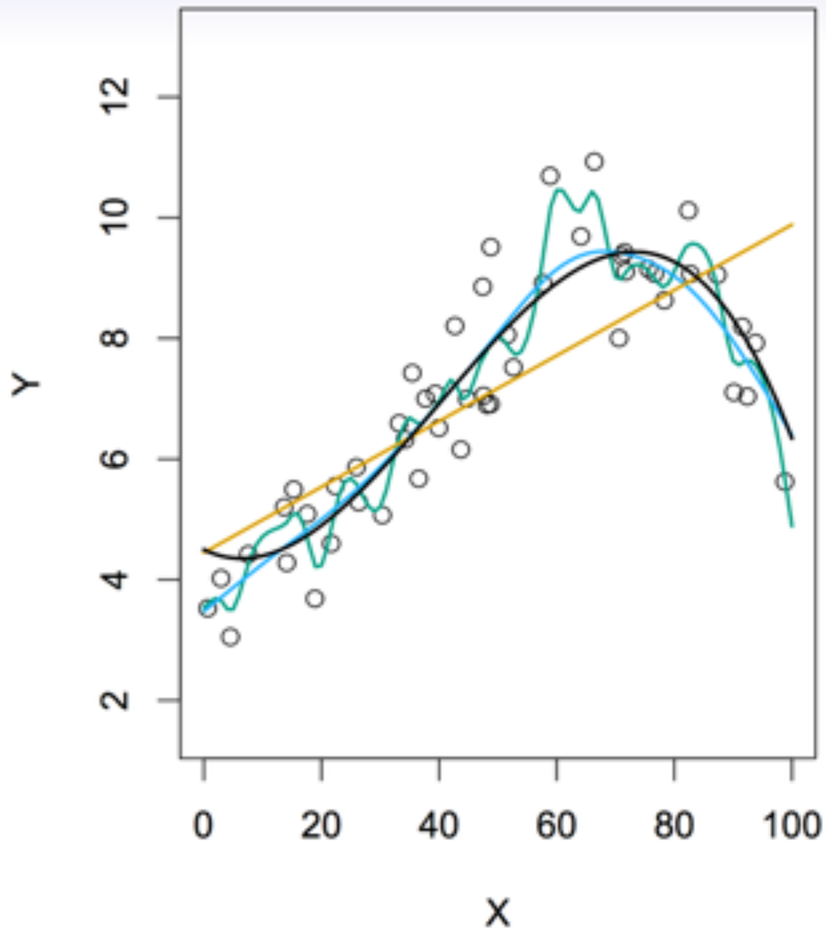
Galvanize Lightning Review Week 4

Bias-Variance Tradeoff



$$\mathrm{Var}(\hat{f}(x_0))$$

Amount by which $\hat{f}$ would change if estimated it using a different training dataset

$$\mathrm{Bias}(\hat{f}(x_0))] = E[\hat{f}(x_0)] - f(x_0)$$

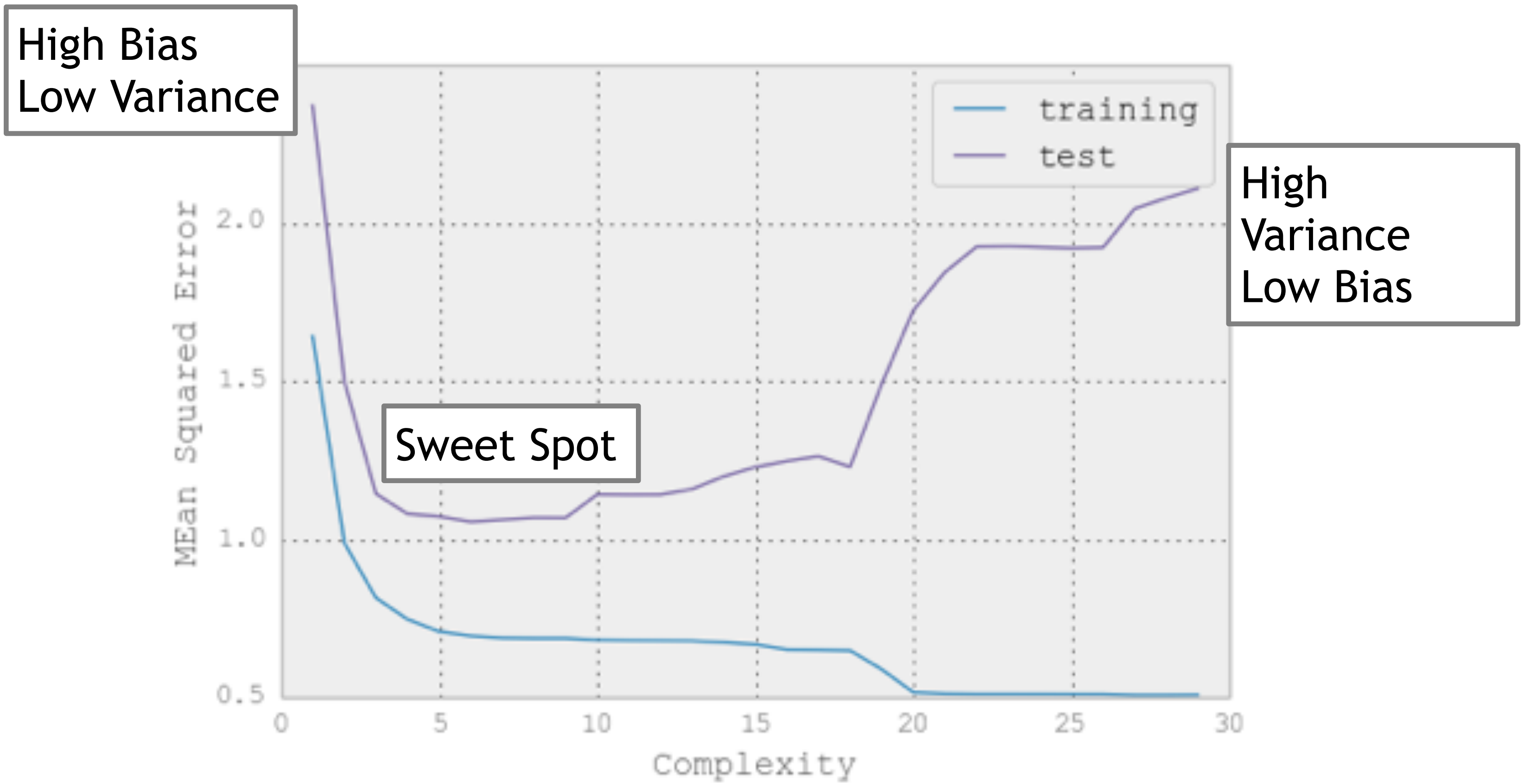
Difference between expected prediction of our model and correct value we are trying to predict

$$\mathrm{Var}(\epsilon)$$

Simply because $Y = f(X) + \epsilon$

Generally speaking, the *more flexible* the model, the *greater the variance*.

Model Framework - Evaluation

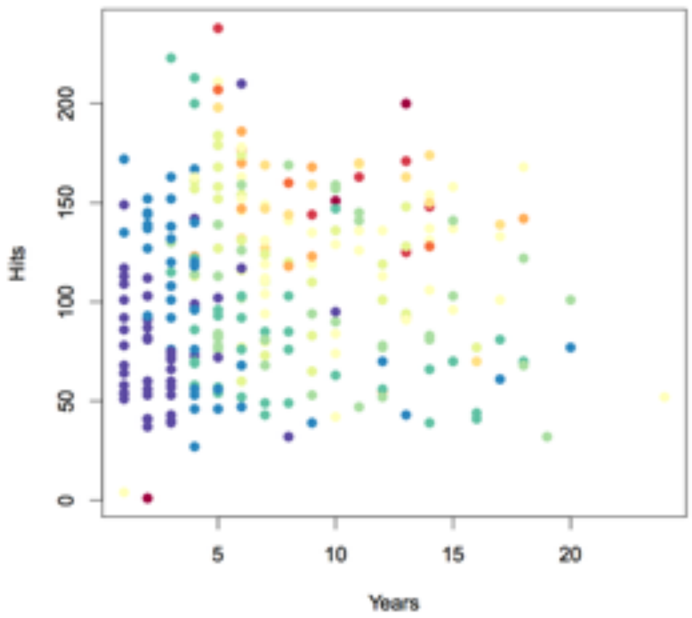


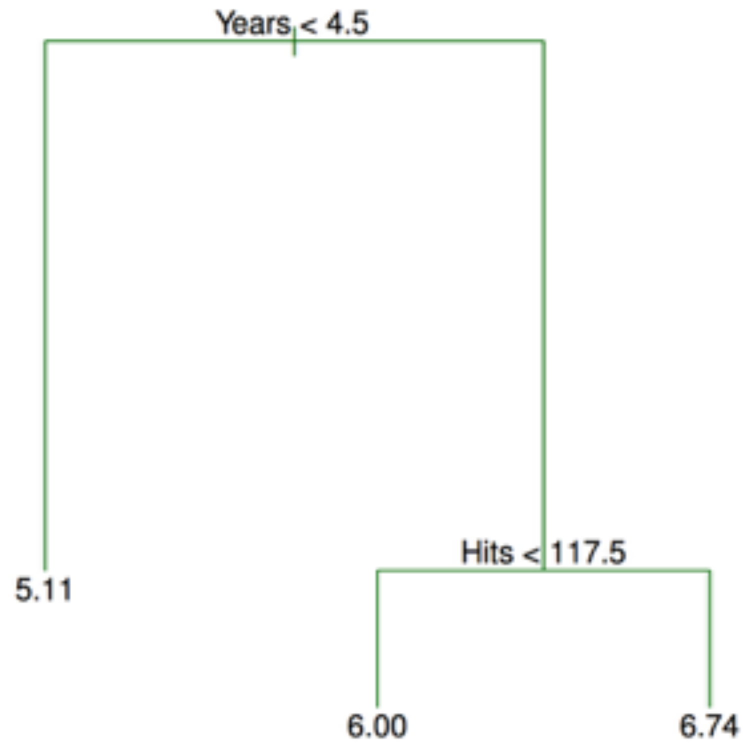
- Can break this complexity tradeoff into what we call “bias” and “variance”

Decision Trees – Regression

Baseball salaries:

(Blue, Green) for low salaries
(Yellow, Red) for high salaries





Bagging

- Bootstrap data going into each tree

- Grow many large “bushy” trees and average away the variance (*central limit theorem*) by growing lots of trees (*bootstrapping*)!

- Decision by voting (classification) or average (regression)

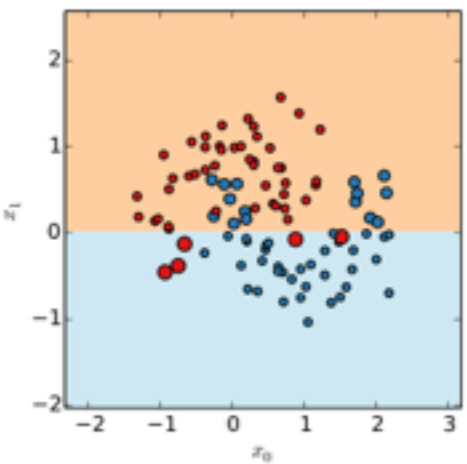
Random Forest

- Bootstrap data going into each tree
- Grow many large “bushy” trees and average away the variance (*central limit theorem*) by growing lots of trees (*bootstrapping*)!
- Decision by voting (classification) or average (regression)
- Subset the features available at each split sqrt(features) for classification and features / 3 for regression

Boosting

AdaBoost

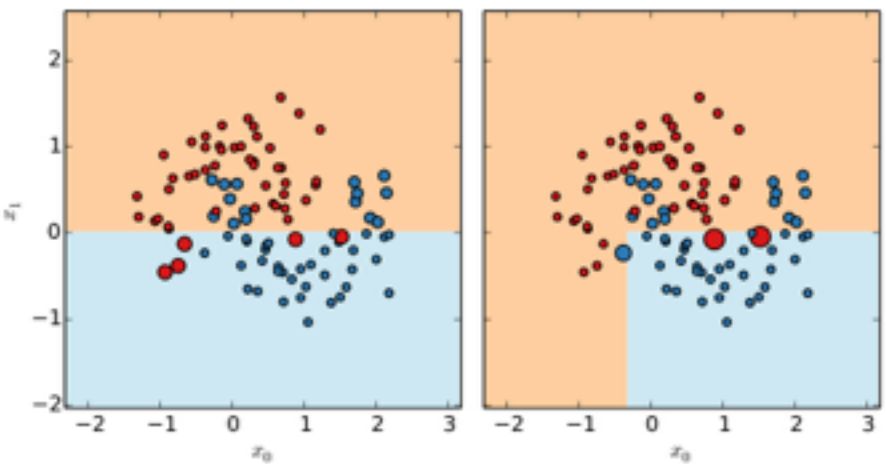
- Each tree is expert on attacking errors of predecessor
- Iteratively re-weights observations based on errors



Boosting

AdaBoost

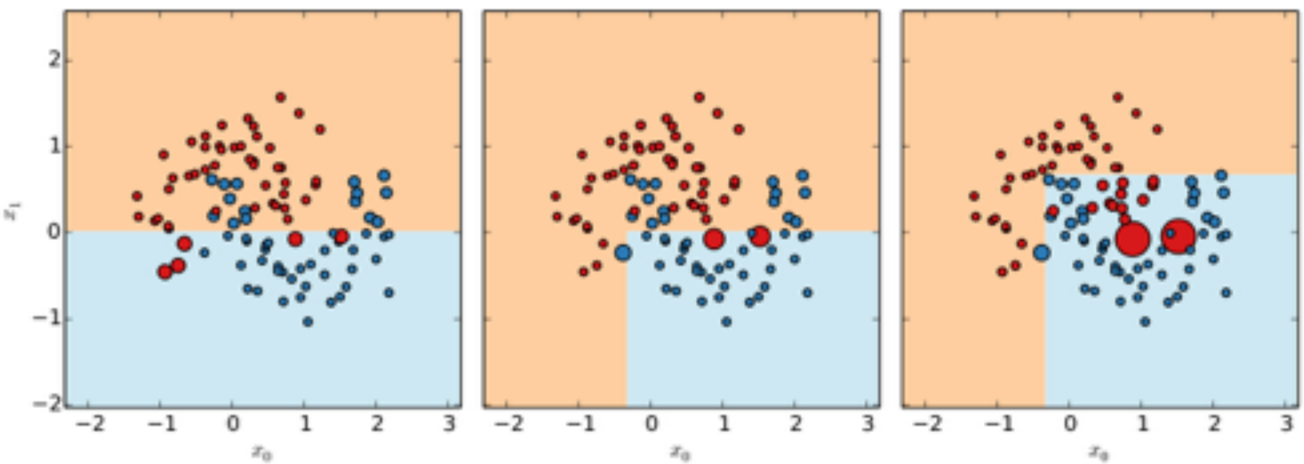
- Each tree is expert on attacking errors of predecessor
- Iteratively re-weights observations based on errors



Boosting

AdaBoost

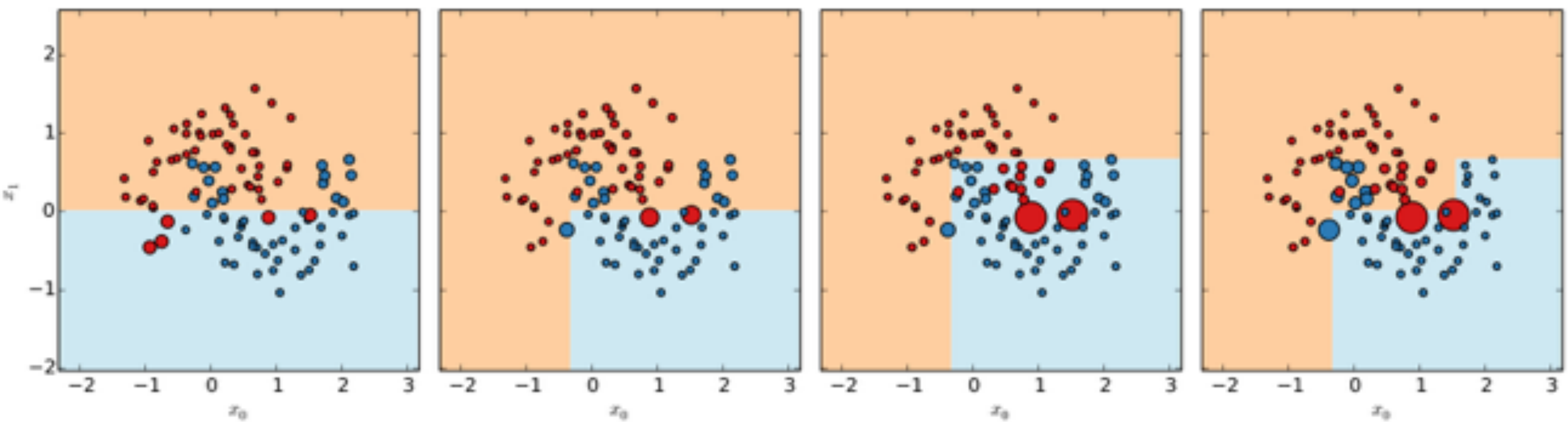
- Each tree is expert on attacking errors of predecessor
- Iteratively re-weights observations based on errors



Boosting

AdaBoost

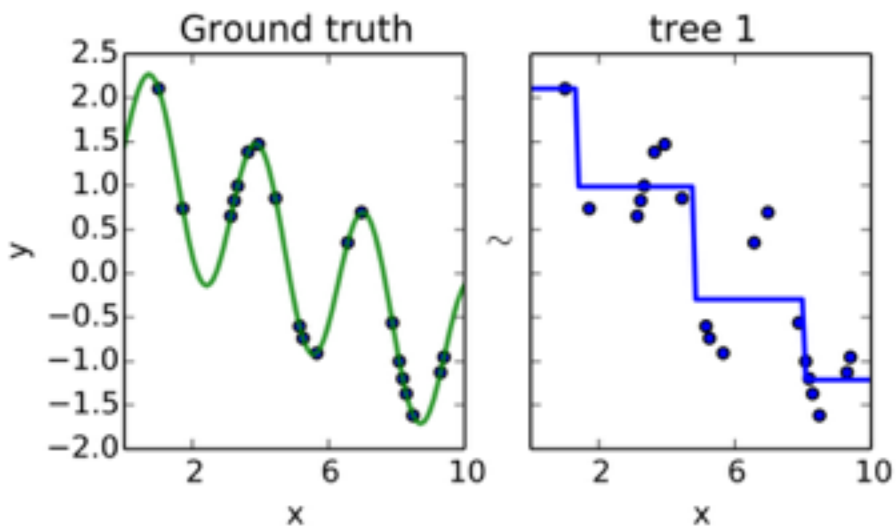
- Each tree is expert on attacking errors of predecessor
- Iteratively re-weights observations based on errors



Boosting

Gradient Boosted Regression Trees

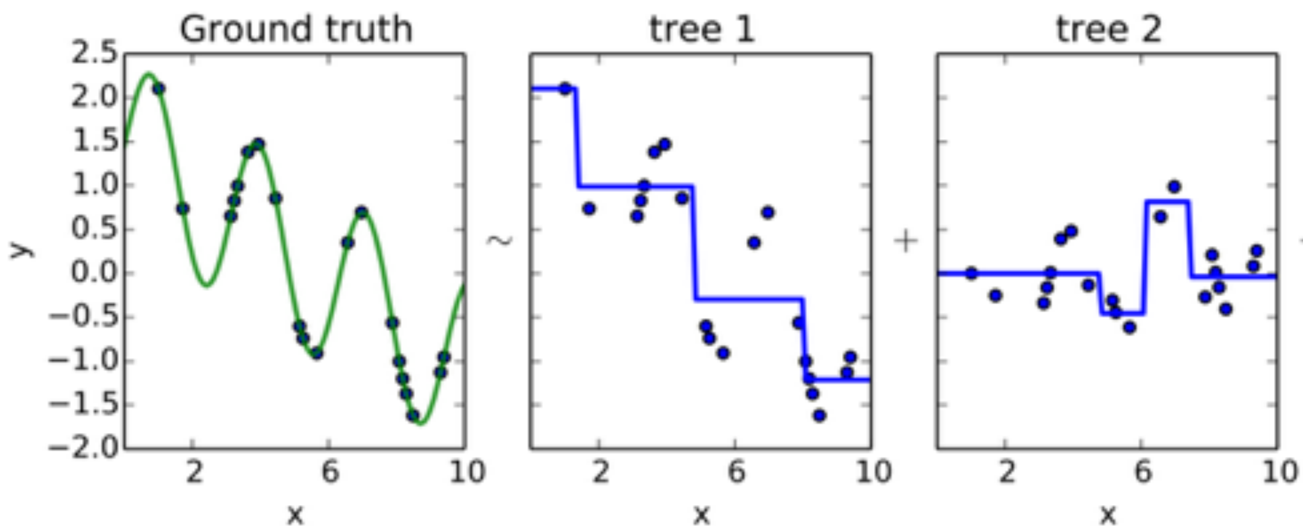
- Instead of fitting to reweighted training observations, fit residuals to of previous tree



Boosting

Gradient Boosted Regression Trees

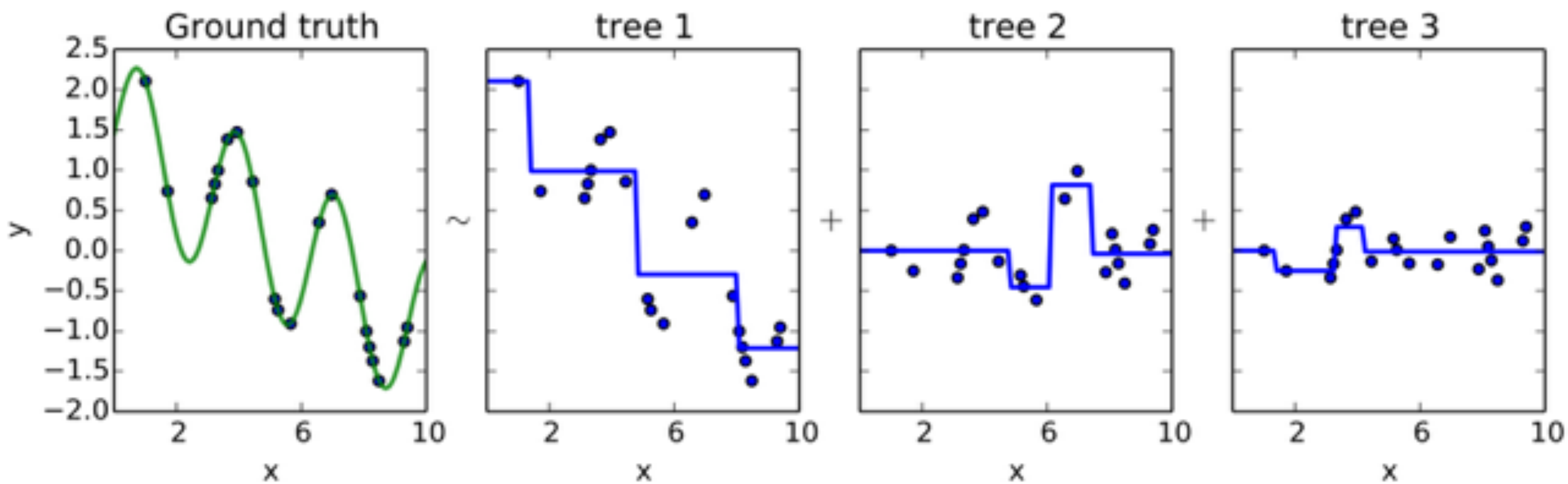
- Instead of fitting to reweighted training observations, fit residuals to of previous tree



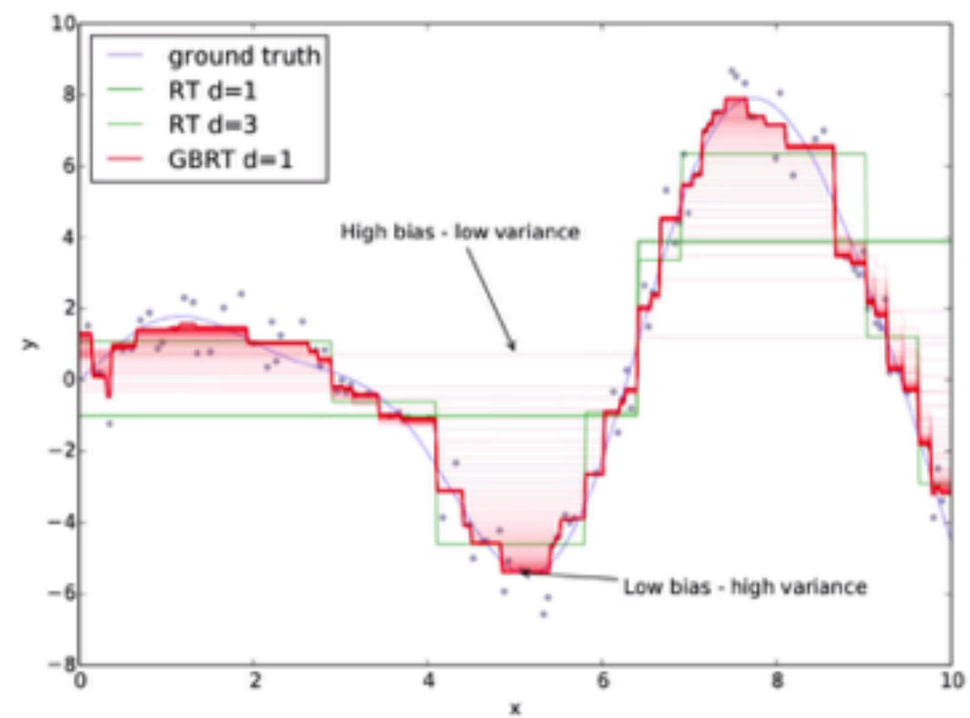
Boosting

Gradient Boosted Regression Trees

- Instead of fitting to reweighted training observations, fit residuals to of previous tree



Boosting

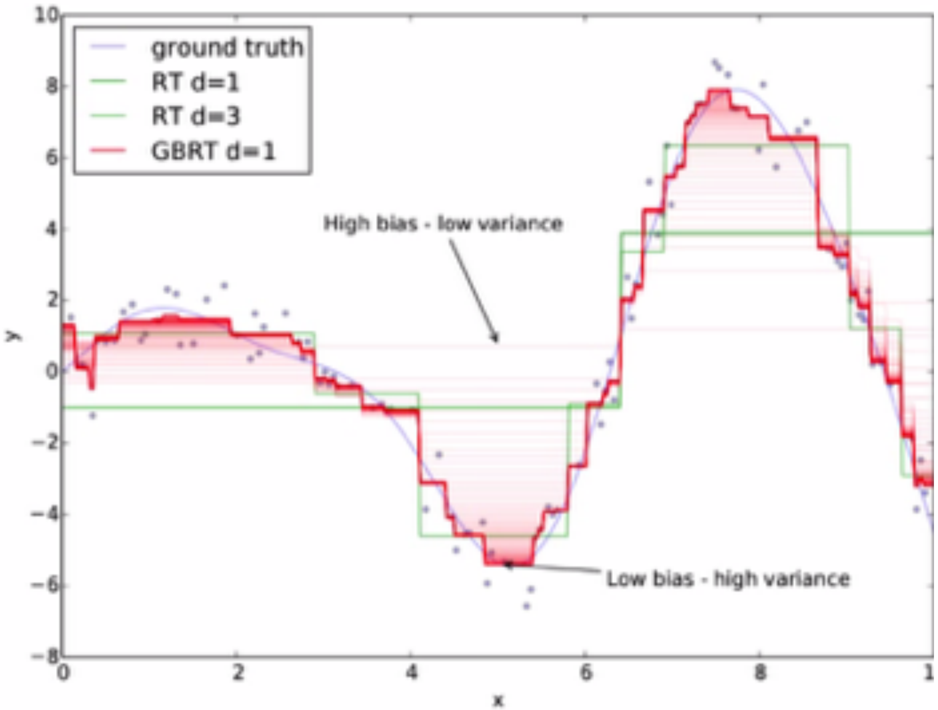


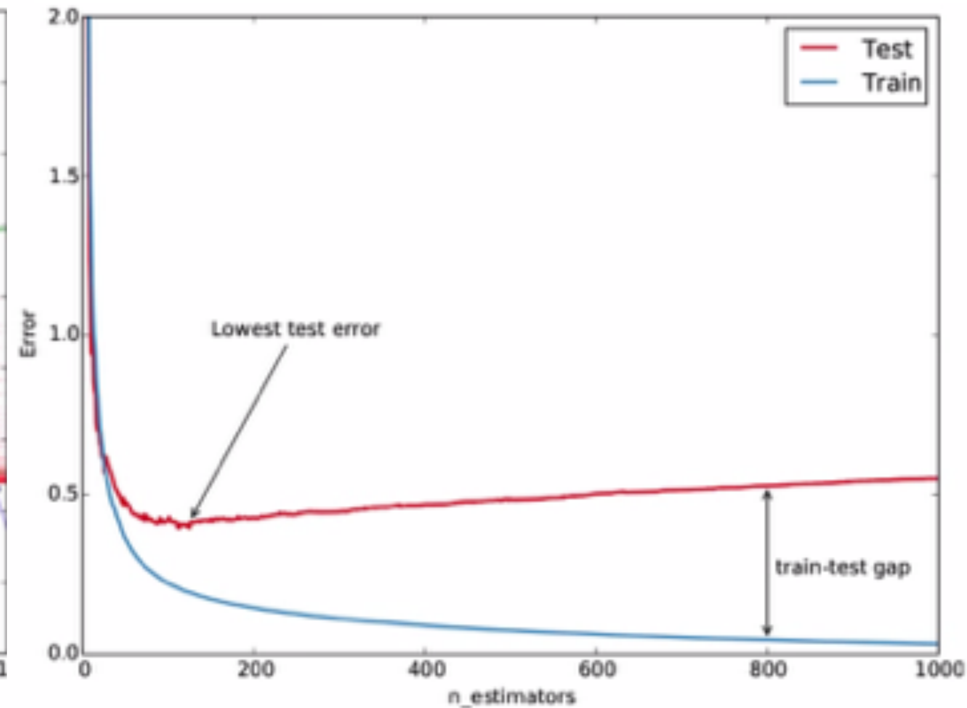
A single short (boosted) tree,
→ High bias, Low variance

Many many (boosted) trees…
→ Lower bias, Higher variance
(than a single tree)

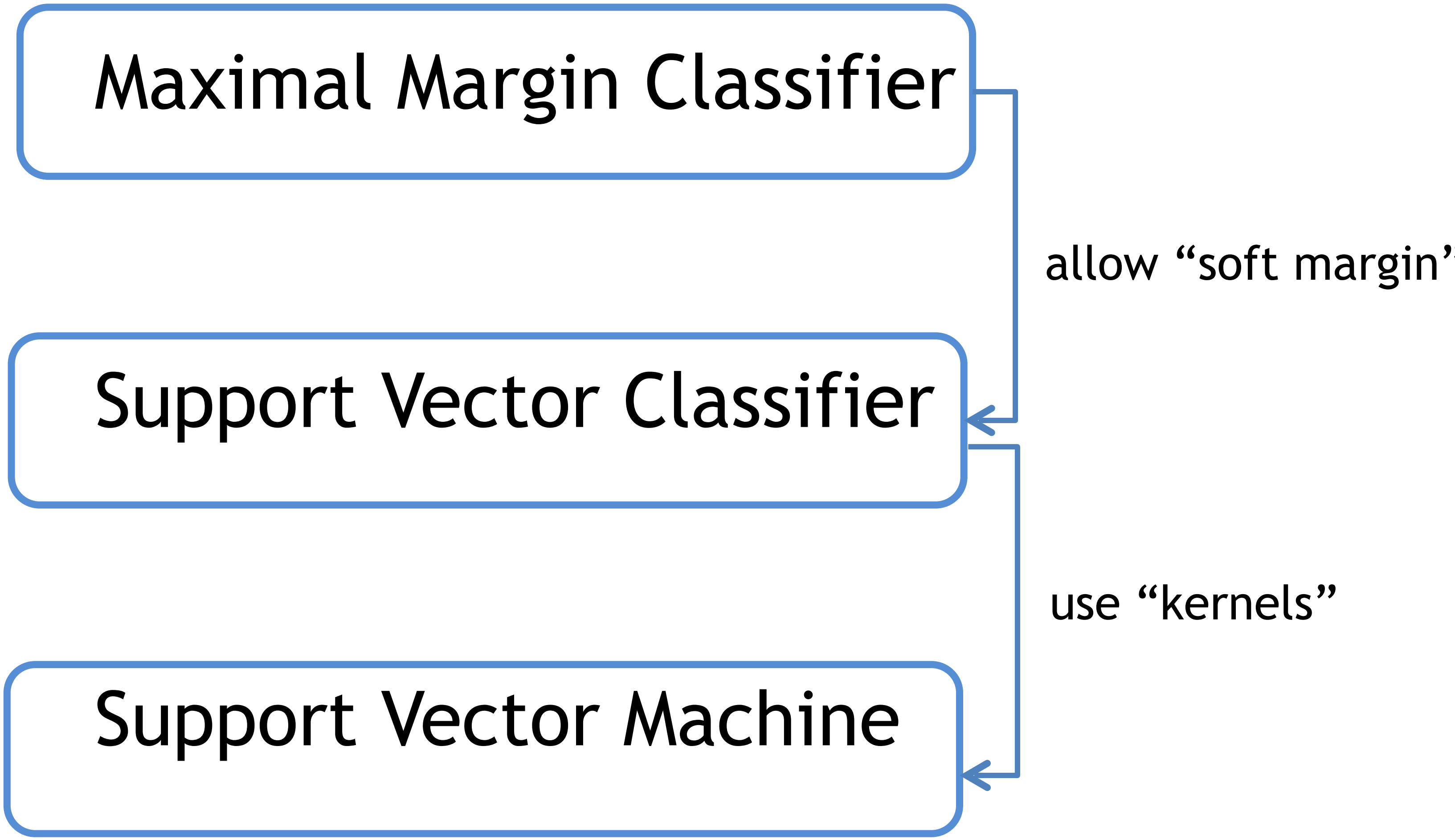
Boosting

As number of trees grows….



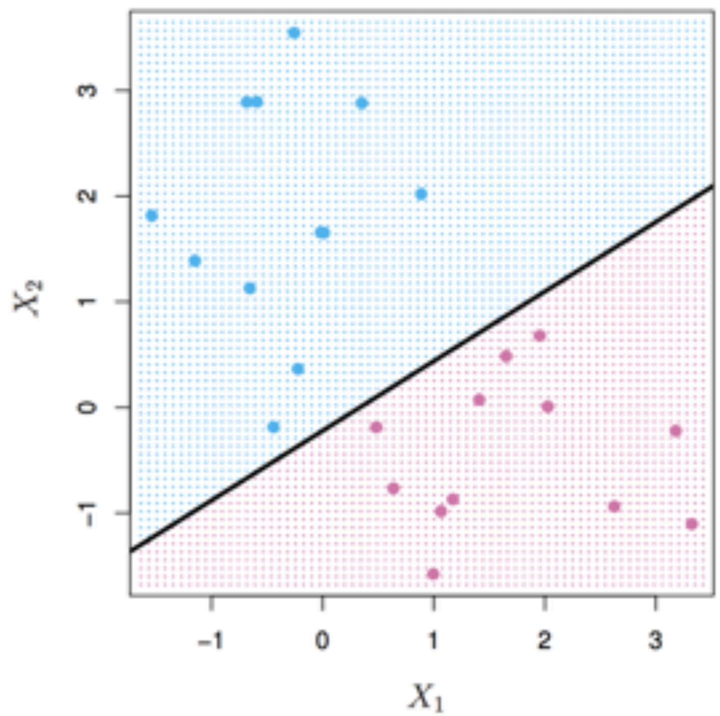


Building up to the SVM…



We have a *separating hyperplane*,
if for *all points*, we have…

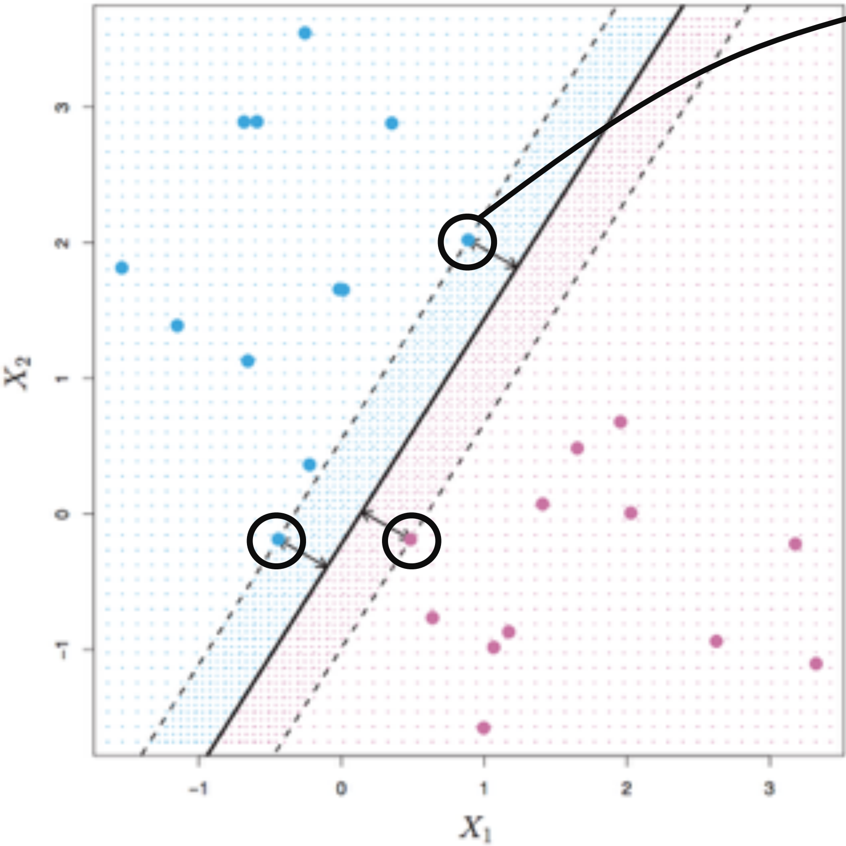
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} > 0 \;\; when \;\; y_i = +1$$



$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} < 0 \;\; when \;\; y_i = -1$$

In particular, we fit…

“Support Vectors”

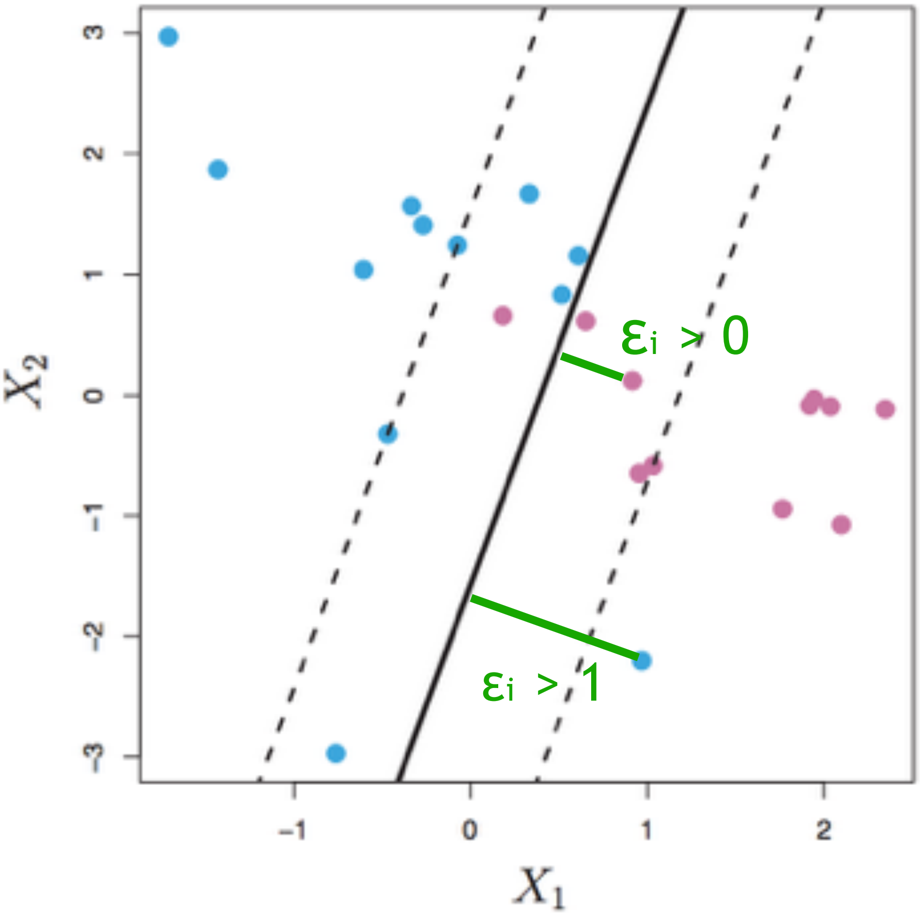


$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \quad \forall\, i = 1, \ldots, n$$

need some sort of *budget*



$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

Slack from each point

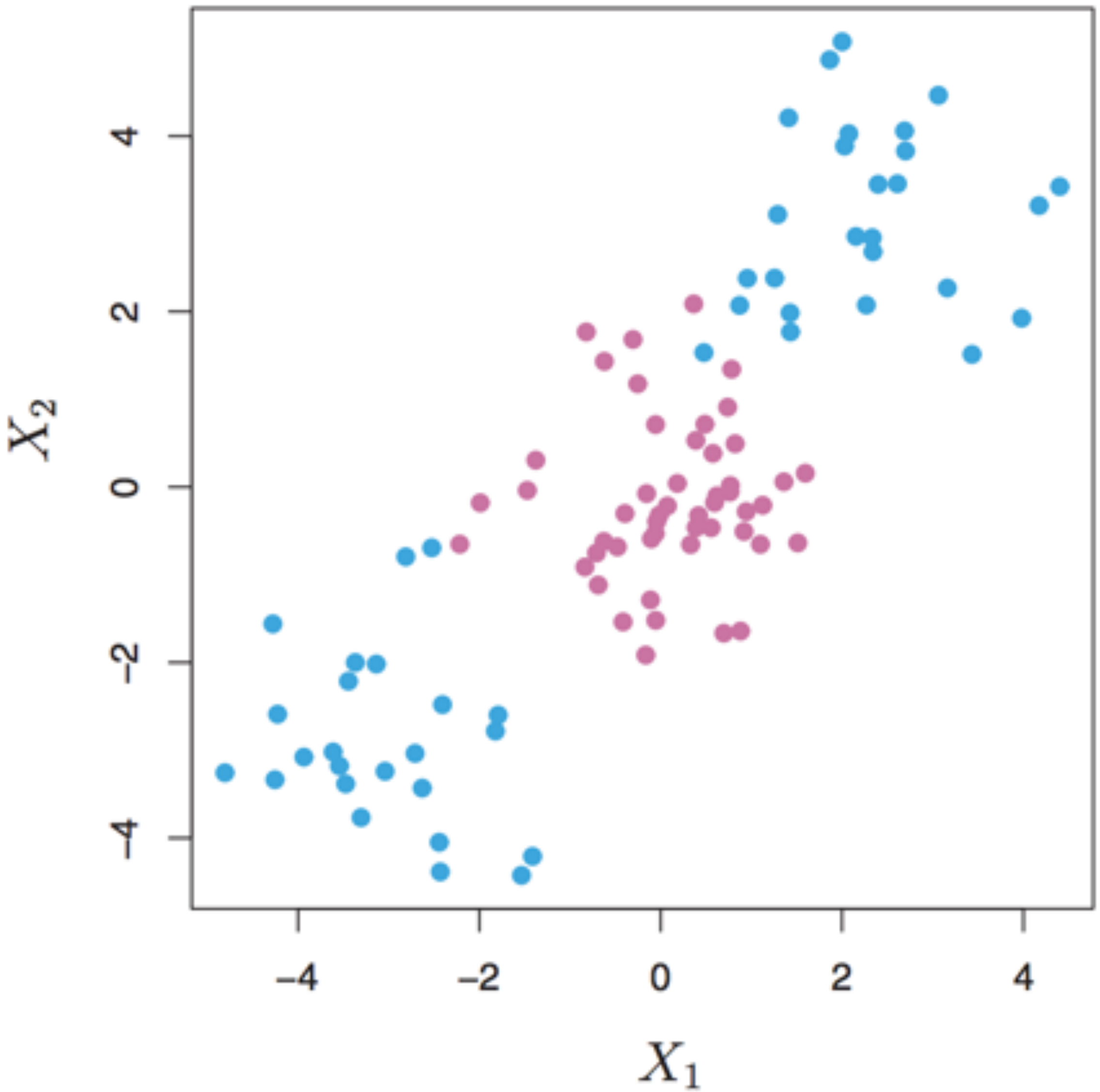
Budget that we can tune

Bias Variance Tradeoff

- C small ⇔ Low bias, High Variance
- C large ⇔ High bias, Low Variance (not quite as clear cut)

$\varepsilon_i = 0$ for being on correct side of margin
$\varepsilon_i > 0$ for violating the margin
$\varepsilon_i > 1$ for being on wrong side of hyperplane

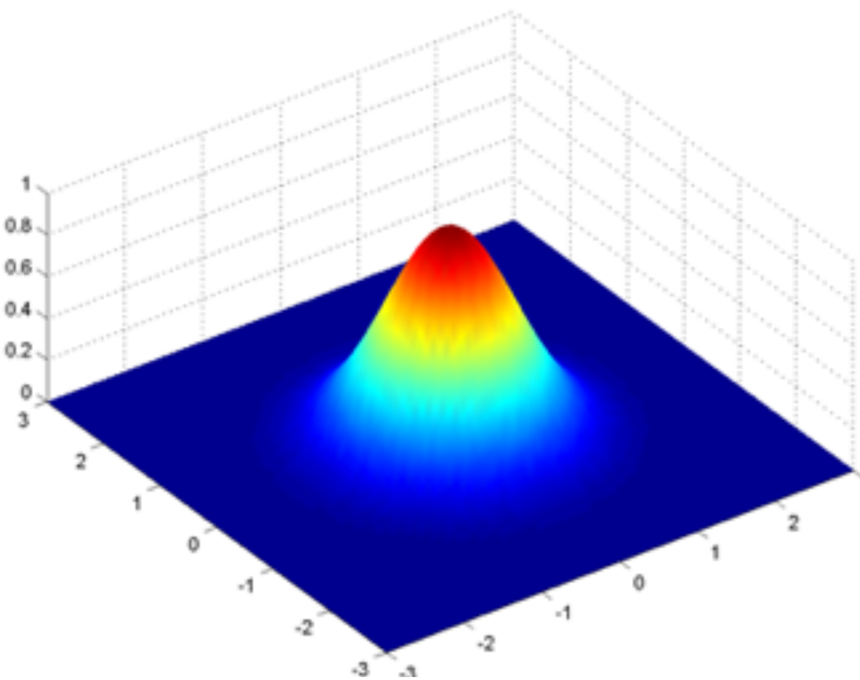
hmm....

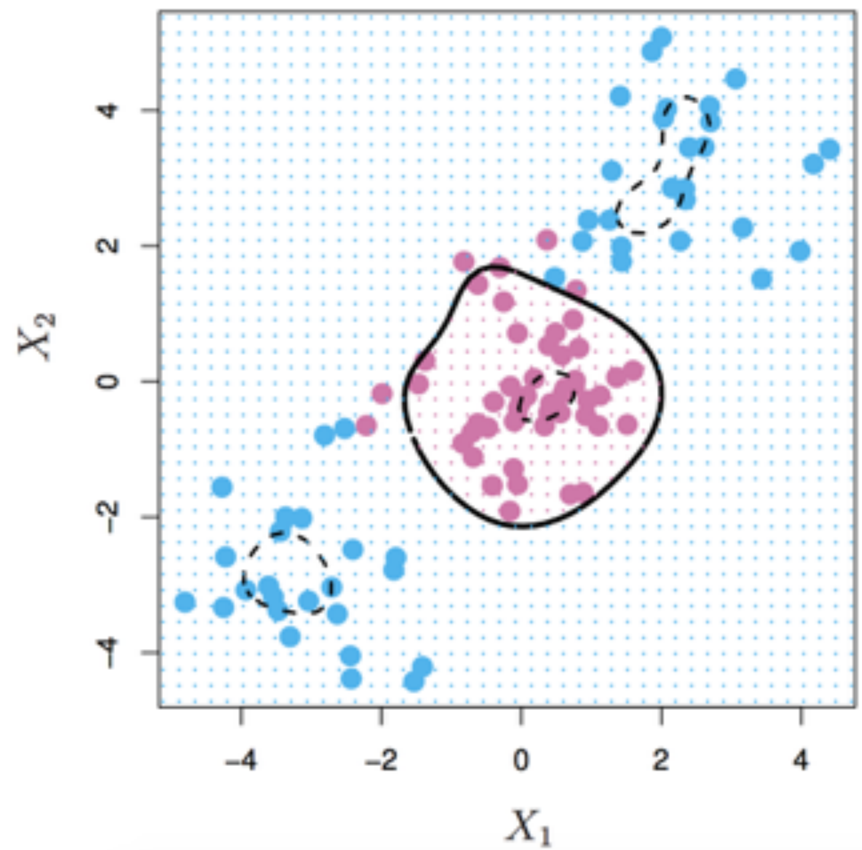


Radial Basis Kernel (Gaussian)

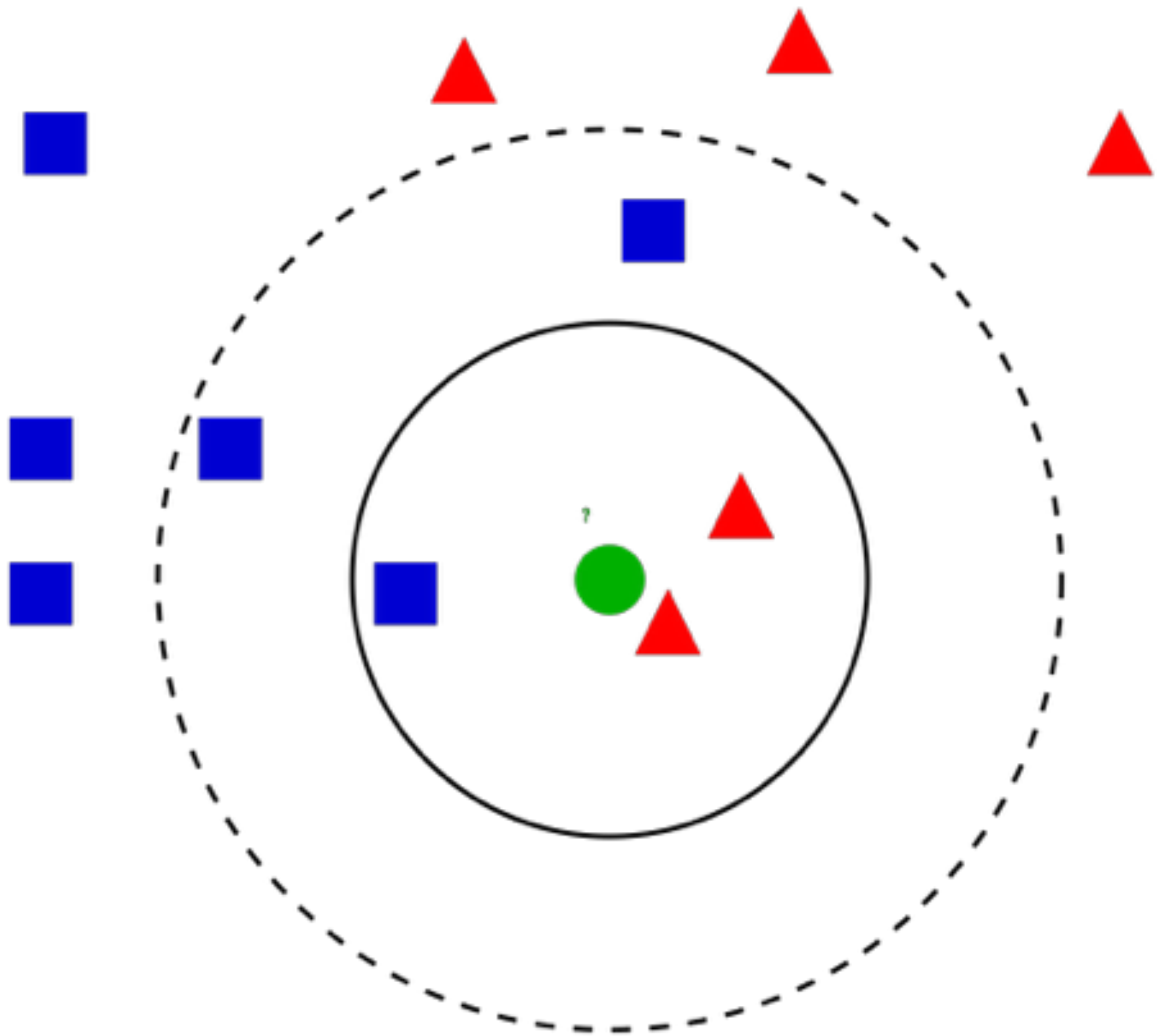
$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(x, x_i)$$





kth Nearest Neighbor



End Session