



# Bias/Variance Tradeoff and Ensemble Methods

Rishabh Iyer

University of Texas at Dallas

# Last Time

---

- PAC learning
- Bias/variance tradeoff
  - small hypothesis spaces (not enough flexibility) can have high bias
  - rich hypothesis spaces (too much flexibility) can have high variance
- Today: more on this phenomenon and how to get around it

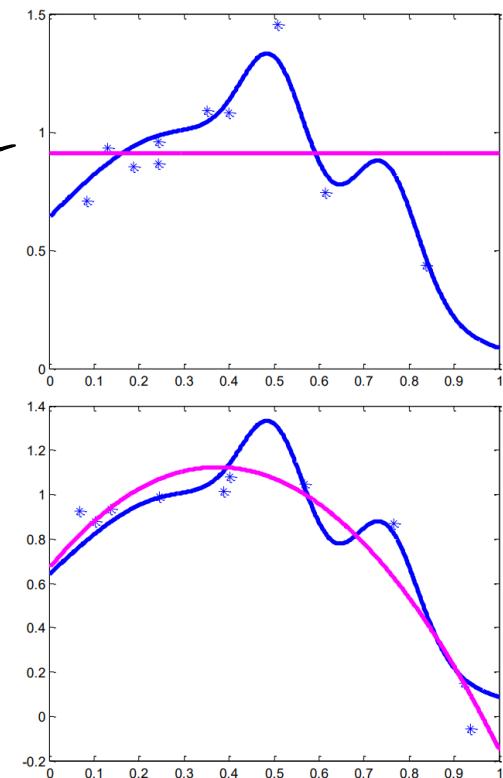
# High Variance or Overfitting

If we allow very complicated predictors, we could overfit the training data.

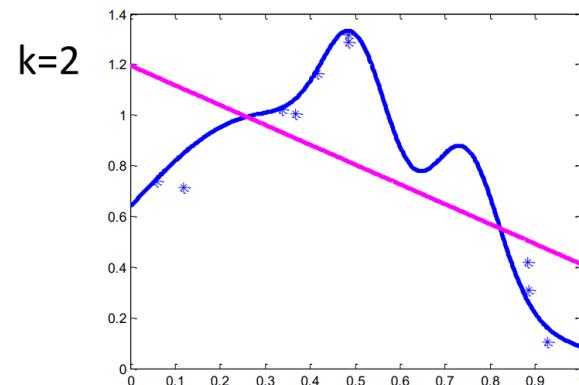
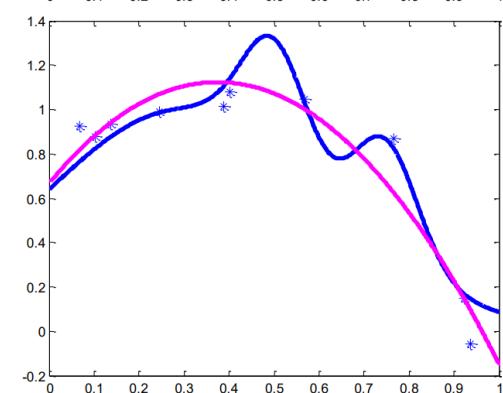
Examples: Regression (Polynomial of order  $k$  – degree up to  $k-1$ )

$$\sum_{i=1}^M (y^{(i)} - c)^2$$

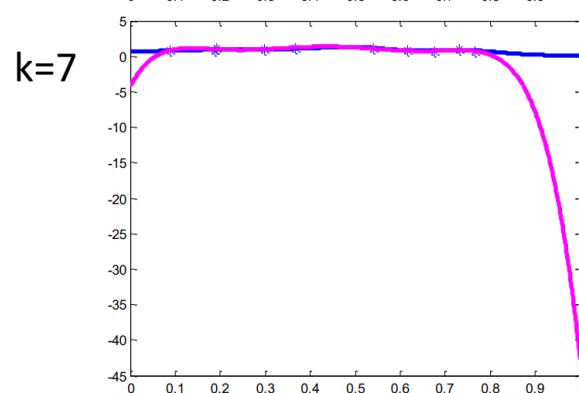
$$\min_c f(c)$$



$k=3$



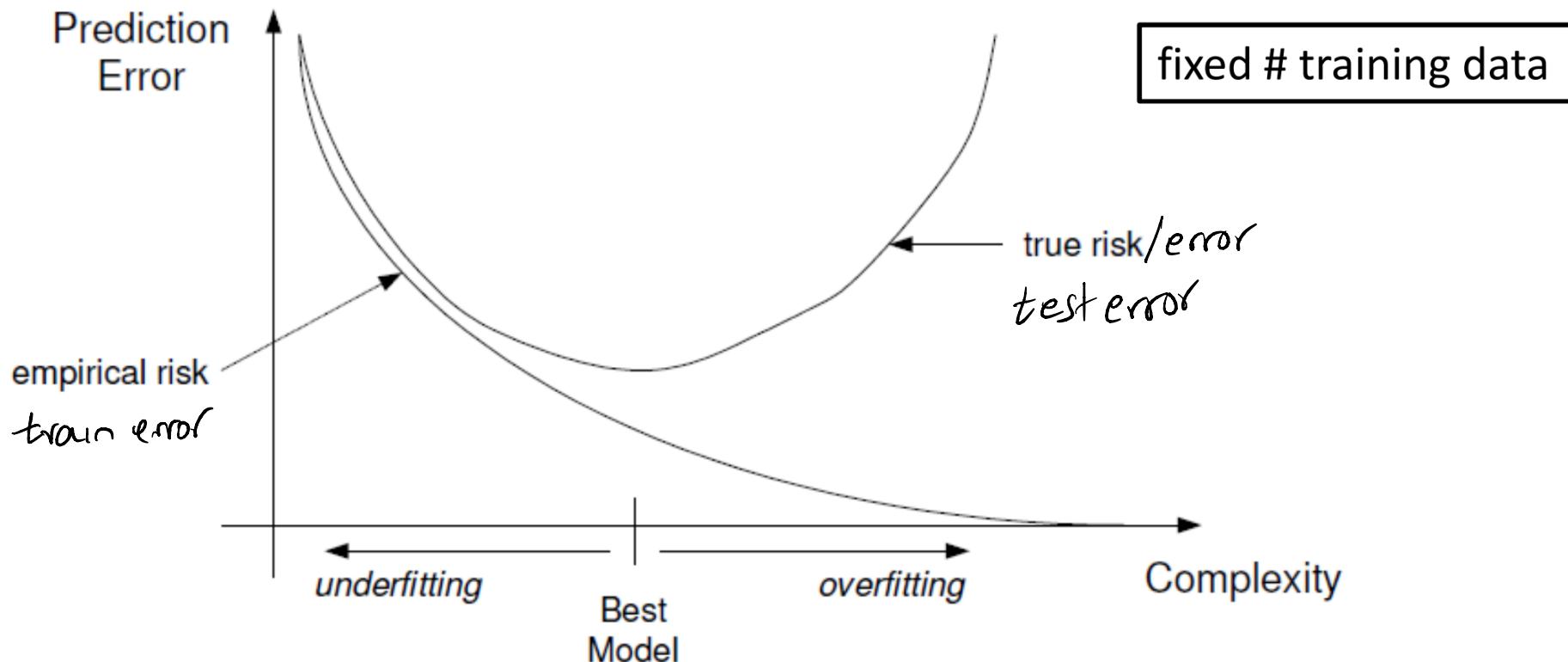
$k=7$



# Effect of Model Complexity



If we allow very complicated predictors, we could overfit the training data.



- Bias → Underfitting.
  - Measures the accuracy or quality of the algorithm
  - High bias means a poor match
- Variance → Overfitting.
  - Measures the precision or specificity of the match
  - High variance means a weak match
- We would like to minimize each of these
- Unfortunately, we can't do this independently, there is a trade-off

- Dataset:  $D = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$
- True function is  $y = f(x) + \epsilon$ 
  - Where noise,  $\epsilon$ , is normally distributed with zero mean and standard deviation  $\sigma$
- Given a set of training examples,  $(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})$ , we fit a hypothesis  $g(x) = w^T x + b$  to the data to minimize the squared error

$$\sum_i [y^{(i)} - g(x^{(i)})]^2$$

# Some Terminology



**Expected Label** (given  $\mathbf{x} \in \mathbb{R}^d$ ):

$$\bar{y}(\mathbf{x}) = E_{y|\mathbf{x}}[Y] = \int y \Pr(y|\mathbf{x}) \partial y.$$

## Expected Test Error (given $h_D$ ):

$$E_{(\mathbf{x},y) \sim P} \left[ (h_D(\mathbf{x}) - y)^2 \right] = \iint_{\mathcal{X} \times \mathcal{Y}} (h_D(\mathbf{x}) - y)^2 \Pr(\mathbf{x}, y) \partial y \partial \mathbf{x}.$$

### Expected Classifier (given $\mathcal{A}$ ):

$$\bar{h} = E_{D \sim P^n} [h_D] = \int h_D \Pr(D) \partial D$$

**Expected Test Error** (given  $\mathcal{A}$ ):

$$E_{\substack{(\mathbf{x}, y) \sim P \\ D \sim P^n}} \left[ (h_D(\mathbf{x}) - y)^2 \right] = \int_D \int_{\mathbf{x}} \int_y (h_D(\mathbf{x}) - y)^2 P(\mathbf{x}, y) P(D) \partial \mathbf{x} \partial y \partial D$$

# Probability Reminder

- Variance of a random variable,  $Z$

$$\begin{aligned}Var(Z) &= E[(Z - E[Z])^2] \\&= E[Z^2 - 2ZE[Z] + E[Z]^2] \\&= E[Z^2] - E[Z]^2\end{aligned}$$

- Properties of  $Var(Z)$

$$Var(aZ) = E[a^2Z^2] - E[aZ]^2 = a^2Var(Z)$$

$D_1, D_2, \dots, D_{1000} \leftarrow \text{Train}$

$$y = \underline{f(m)} + \varepsilon \quad , \quad \bar{y} = f(m)$$

Bias:  $E_{D,m} [h_D^{(m)} - \bar{y}]^2 = [\underset{\uparrow}{\text{Avg Error}}(D_i)) - \bar{y}]^2$

↑  
Avg Model perf  
↓  
True Label.

High bias = Model is far from true Label.

$\Rightarrow$  Underfitting-

## Variance

$D_1, \dots, D_{1000}$

$$E[h_0 - \bar{h}]^2$$

High variance: Models are far away from each other

$\Rightarrow$  overfitting

# Bias-Variance-Noise Decomposition



$$\begin{aligned} E_{\mathbf{x},y,D} \left[ [h_D(\mathbf{x}) - y]^2 \right] &= E_{\mathbf{x},y,D} \left[ [(h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) + (\bar{h}(\mathbf{x}) - y)]^2 \right] \\ &= E_{\mathbf{x},D} \left[ (\bar{h}_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] + 2 E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) (\bar{h}(\mathbf{x}) - y) \right] \\ &\quad + E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right] \end{aligned}$$

↑  
Expected Test error  
for an Algo.

# Bias-Variance-Noise Decomposition

$$\begin{aligned}
 E_{\mathbf{x},y,D} \left[ [h_D(\mathbf{x}) - y]^2 \right] &= E_{\mathbf{x},y,D} \left[ \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) + (\bar{h}(\mathbf{x}) - y) \right]^2 \right] \\
 &= E_{\mathbf{x},D} \left[ (\bar{h}_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] + 2 E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) (\bar{h}(\mathbf{x}) - y) \right] \\
 &\quad + E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right]
 \end{aligned}$$



The middle term of the above equation is 0 as we show below

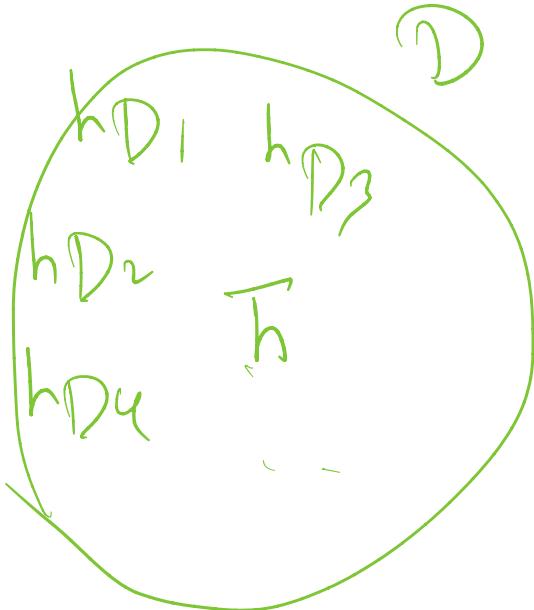
$$\begin{aligned}
 E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) (\bar{h}(\mathbf{x}) - y) \right] &= E_{\mathbf{x},y} \left[ E_D \left[ h_D(\mathbf{x}) - \bar{h}(\mathbf{x}) \right] (\bar{h}(\mathbf{x}) - y) \right] \\
 &= E_{\mathbf{x},y} \left[ (E_D [h_D(\mathbf{x})] - \bar{h}(\mathbf{x})) (\bar{h}(\mathbf{x}) - y) \right] \\
 &= E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - \bar{h}(\mathbf{x})) (\bar{h}(\mathbf{x}) - y) \right] \\
 &= E_{\mathbf{x},y} [0] \\
 &= 0
 \end{aligned}$$

# Bias-Variance-Noise Decomposition



Returning to the earlier expression, we're left with the variance and another term

$$E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - y)^2 \right] = \underbrace{E_{\mathbf{x},D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{Variance}} + E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right]$$



# Bias-Variance-Noise Decomposition

Returning to the earlier expression, we're left with the variance and another term

$$E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - y)^2 \right] = \underbrace{E_{\mathbf{x},D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{Variance}} + E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right]$$

We can break down the second term in the above equation as follows:

$$\begin{aligned} E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right] &= E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})) + (\bar{y}(\mathbf{x}) - y)^2 \right] \\ &= \underbrace{E_{\mathbf{x},y} \left[ (\bar{y}(\mathbf{x}) - y)^2 \right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))^2 \right]}_{\text{Bias}^2} + 2 E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})) (\bar{y}(\mathbf{x}) - y) \right] \end{aligned}$$

# Bias-Variance-Noise Decomposition

Returning to the earlier expression, we're left with the variance and another term

$$E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - y)^2 \right] = \underbrace{E_{\mathbf{x},D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{Variance}} + E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right]$$

We can break down the second term in the above equation as follows:

$$\begin{aligned} E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - y)^2 \right] &= E_{\mathbf{x},y} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})) + (\bar{y}(\mathbf{x}) - y)^2 \right] \\ &= \underbrace{E_{\mathbf{x},y} \left[ (\bar{y}(\mathbf{x}) - y)^2 \right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))^2 \right]}_{\text{Bias}^2} + 2 E_{\mathbf{x},y} [ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})) (\bar{y}(\mathbf{x}) - y) ] \end{aligned}$$

The third term in the equation above is 0

# Bias-Variance-Noise Decomposition

The third term in the equation above is 0, as we show below

$$\begin{aligned}
 E_{\mathbf{x},y} [(\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})) (\bar{y}(\mathbf{x}) - y)] &= E_{\mathbf{x}} [E_{y|\mathbf{x}} [\bar{y}(\mathbf{x}) - y] (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))] \\
 &= E_{\mathbf{x}} [E_{y|\mathbf{x}} [\bar{y}(\mathbf{x}) - y] (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))] \\
 &= E_{\mathbf{x}} [(\bar{y}(\mathbf{x}) - E_{y|\mathbf{x}} [y]) (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))] \\
 &= E_{\mathbf{x}} [(\bar{y}(\mathbf{x}) - \bar{y}(\mathbf{x})) (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))] \\
 &= E_{\mathbf{x}} [0] \\
 &= 0
 \end{aligned}$$

This gives us the decomposition of expected test error as follows

$$\underbrace{E_{\mathbf{x},y,D} [(h_D(\mathbf{x}) - y)^2]}_{\text{Expected Test Error}} = \underbrace{E_{\mathbf{x},D} [(h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2]}_{\text{Variance}} + \underbrace{E_{\mathbf{x},y} [(\bar{y}(\mathbf{x}) - y)^2]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}} [(\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))^2]}_{\text{Bias}^2}$$

# Bias, Variance, and Noise

This gives us the decomposition of expected test error as follows

$$\underbrace{E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - y)^2 \right]}_{\text{Expected Test Error}} = \underbrace{E_{\mathbf{x},D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{Variance}} + \underbrace{E_{\mathbf{x},y} \left[ (\bar{y}(\mathbf{x}) - y)^2 \right]}_{\text{Noise}}$$

$$+ \underbrace{E_{\mathbf{x}} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))^2 \right]}_{\text{Bias}^2}$$

**Variance:** Captures how much your classifier changes if you train on a different training set. How "overspecialized" is your classifier to a particular training set (overfitting)? If we have the best possible model for our training data, how far off are we from the average classifier?

**Bias:** What is the inherent error that you obtain from your classifier even with infinite training data? This is due to your classifier being "biased" to a particular kind of solution (e.g. linear classifier). In other words, bias is inherent to your model. ← underlining

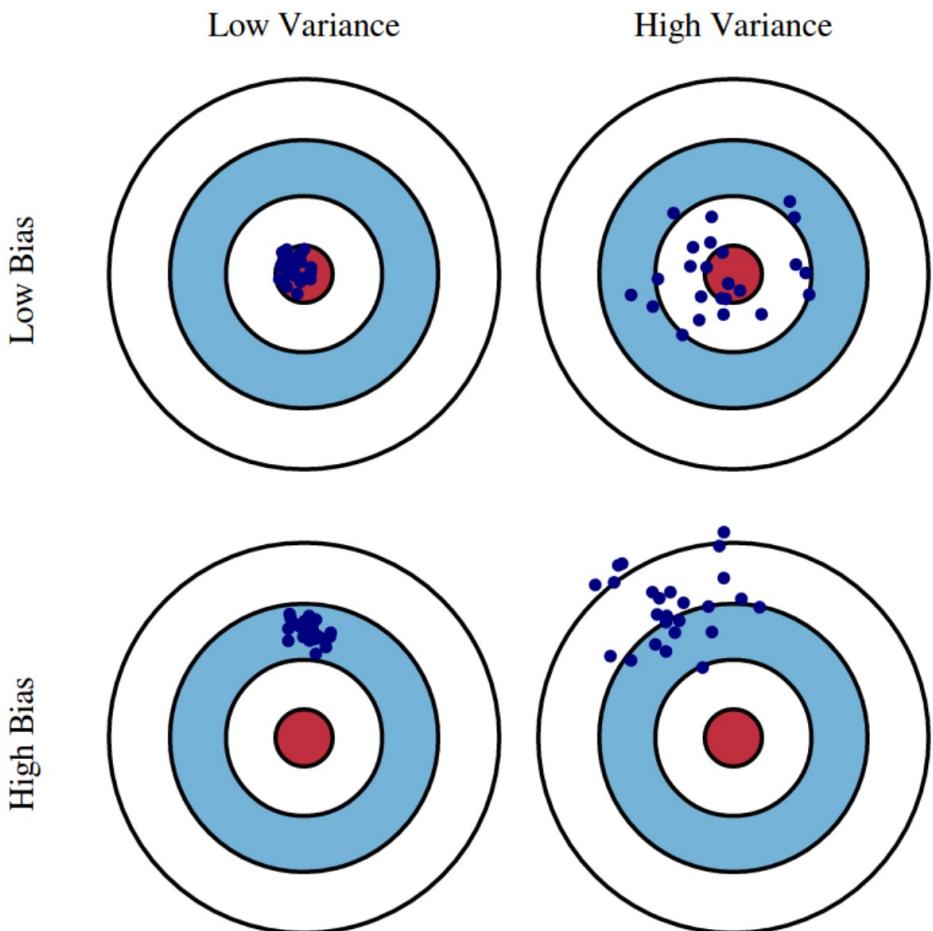
**Noise:** How big is the data-intrinsic noise? This error measures ambiguity due to your data distribution and feature representation. You can never beat this, it is an aspect of the data.

# Bias, Variance, and Noise

$$\underbrace{E_{\mathbf{x},y,D} \left[ (h_D(\mathbf{x}) - y)^2 \right]}_{\text{Expected Test Error}} = \underbrace{E_{\mathbf{x},D} \left[ (h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{Variance}}$$

$$+ \underbrace{E_{\mathbf{x},y} \left[ (\bar{y}(\mathbf{x}) - y)^2 \right]}_{\text{Noise}}$$

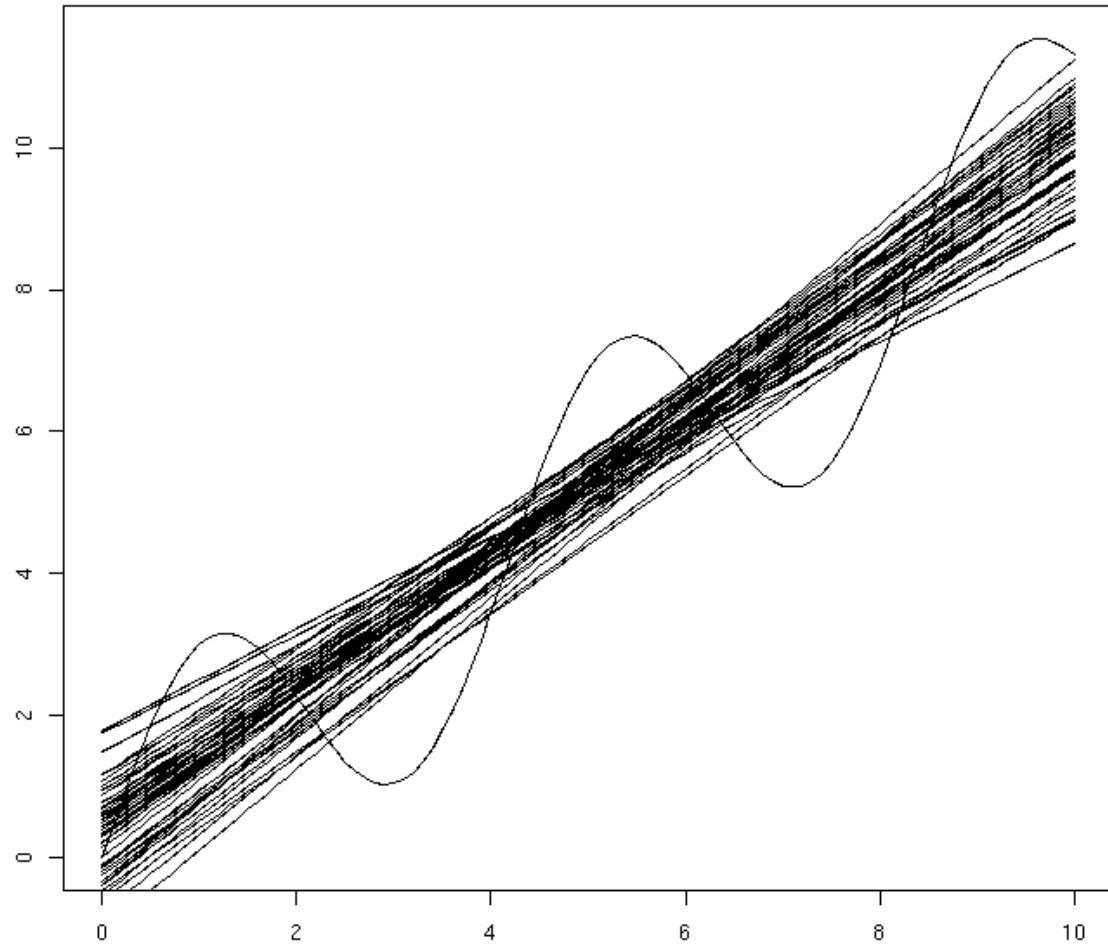
$$+ \underbrace{E_{\mathbf{x}} \left[ (\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x}))^2 \right]}_{\text{Bias}^2}$$



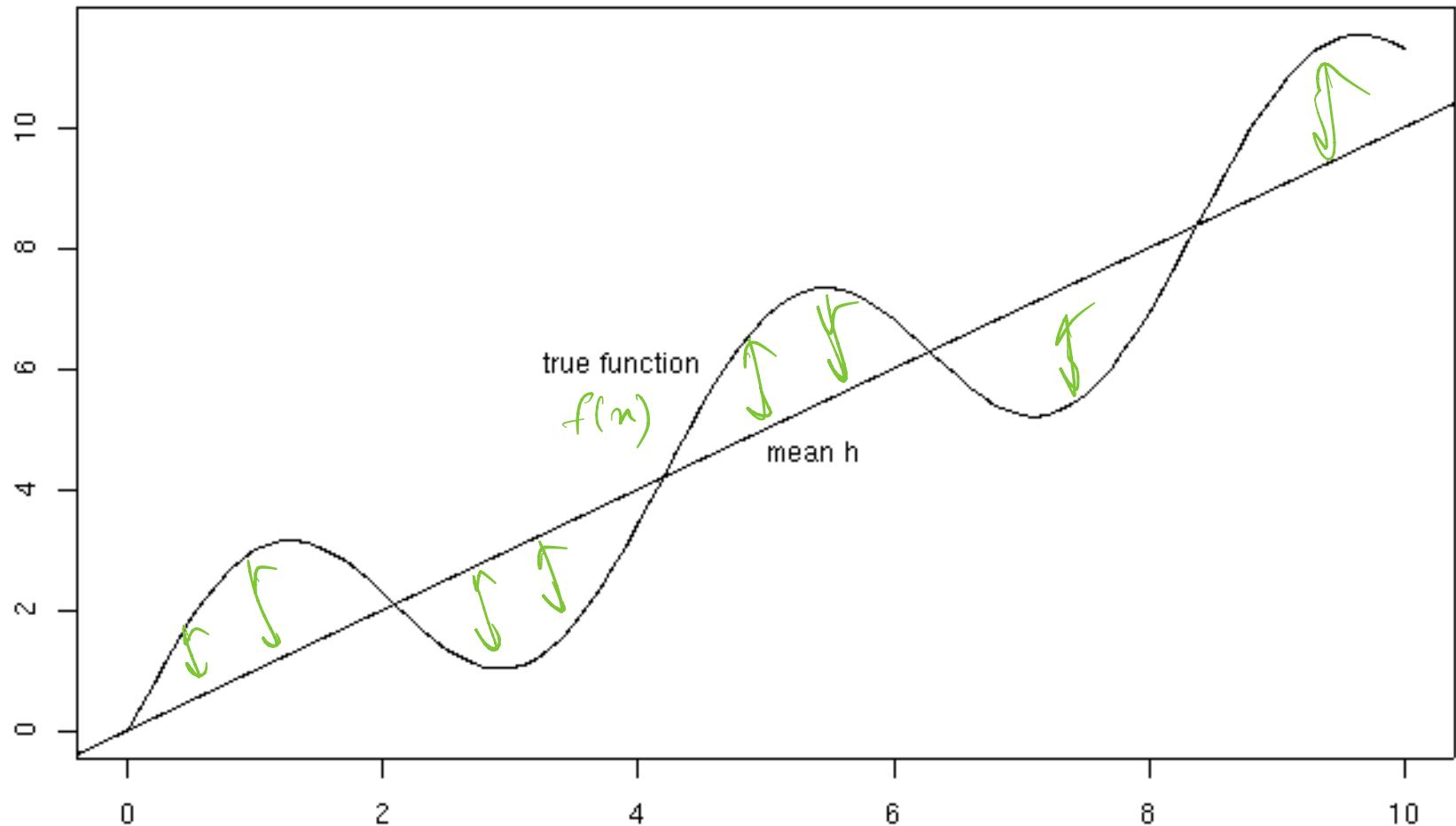
## 2-D Example



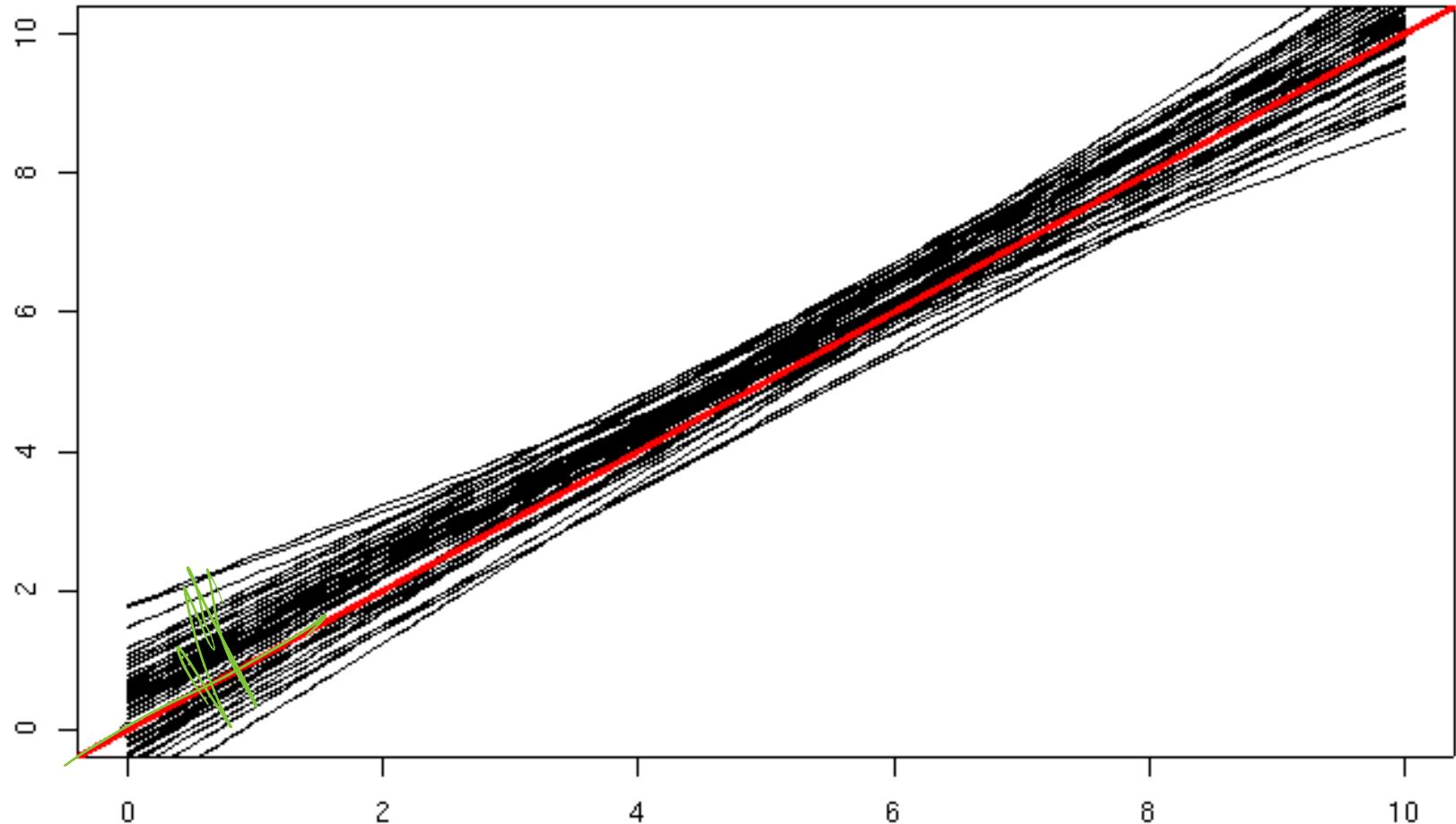
50 fits (20 examples each)



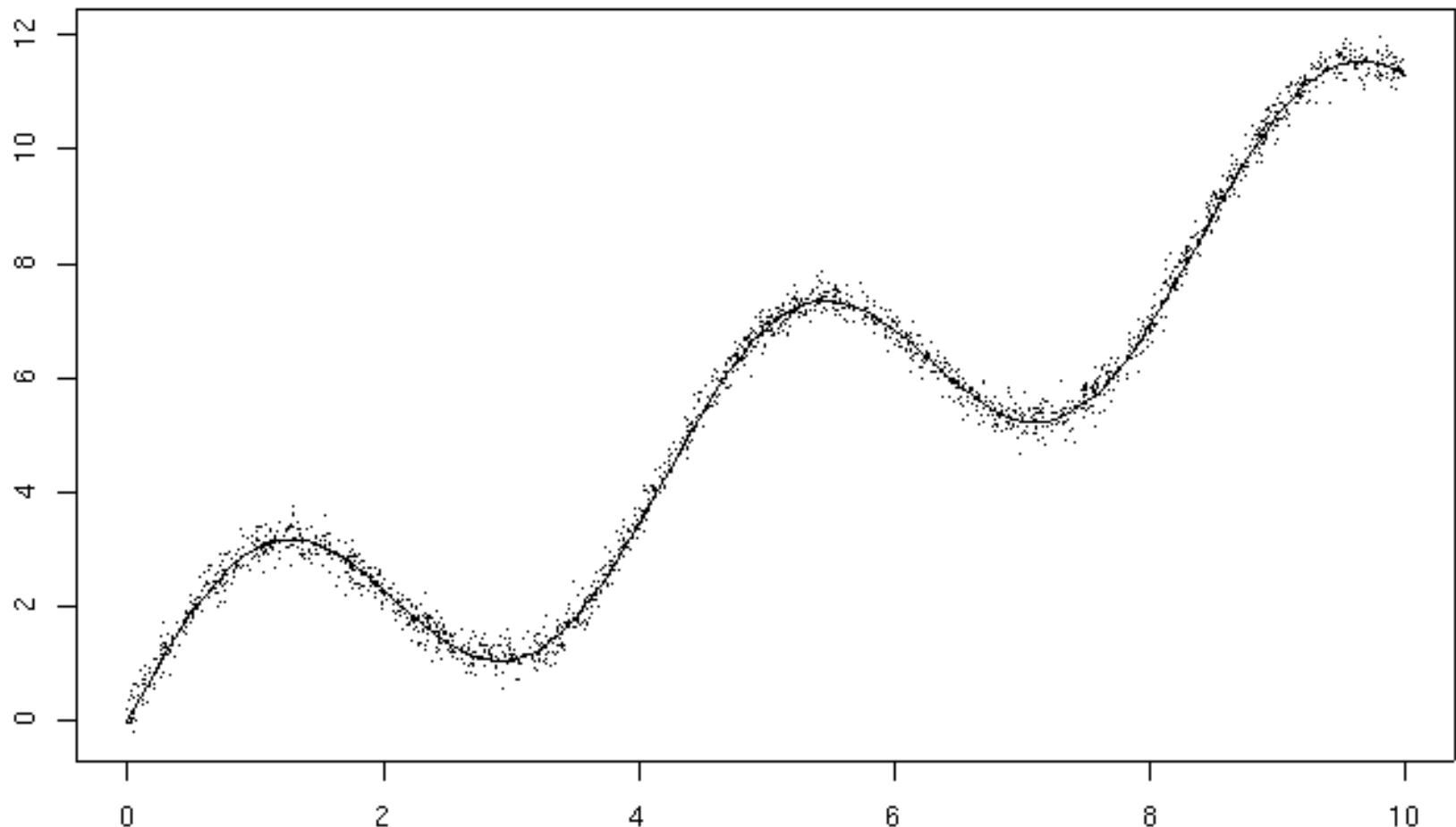
# Bias



# Variance



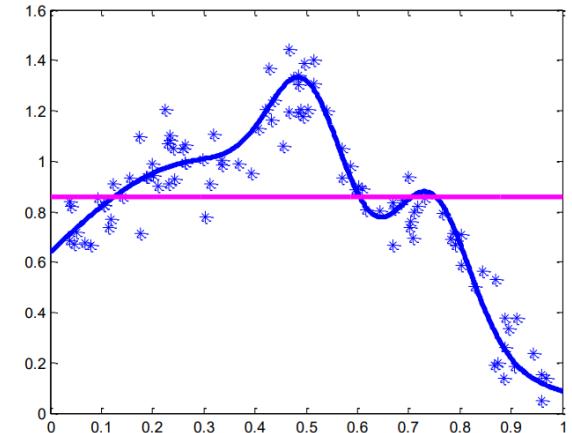
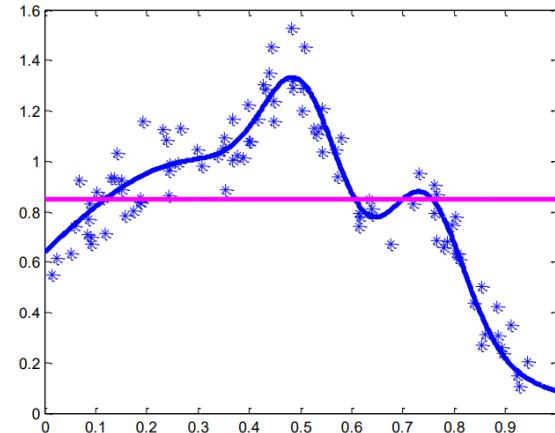
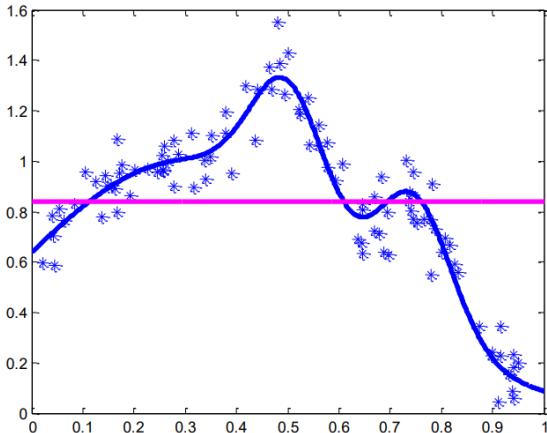
# Noise



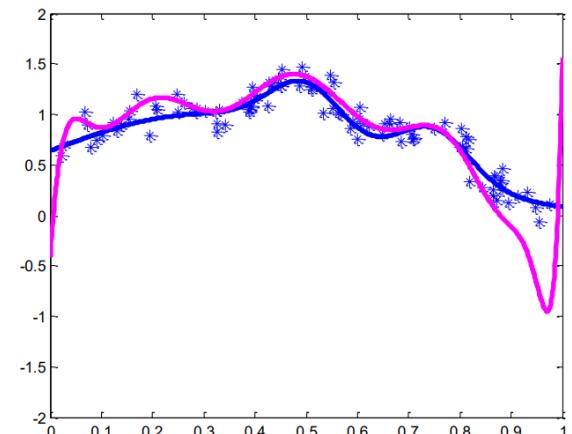
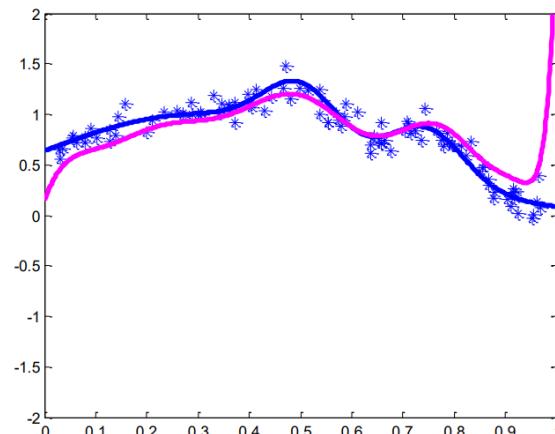
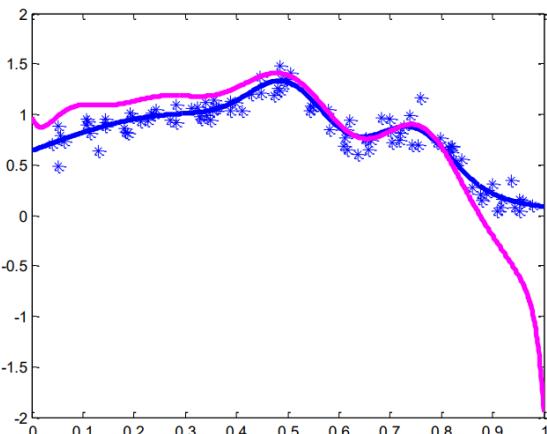
# Bias-Variance Tradeoff



Large bias, Small variance – poor approximation but robust/stable



Small bias, Large variance – good approximation but unstable



# Bias

- Low bias
  - ?
- High bias
  - ?

# Bias

---



- Low bias
  - Linear regression applied to linear data
  - 2nd degree polynomial applied to quadratic data
- High bias
  - Constant function applied to non-constant data
  - Linear regression applied to highly non-linear data

# Variance

---

- Low variance
  - ?
- High variance
  - ?

# Variance

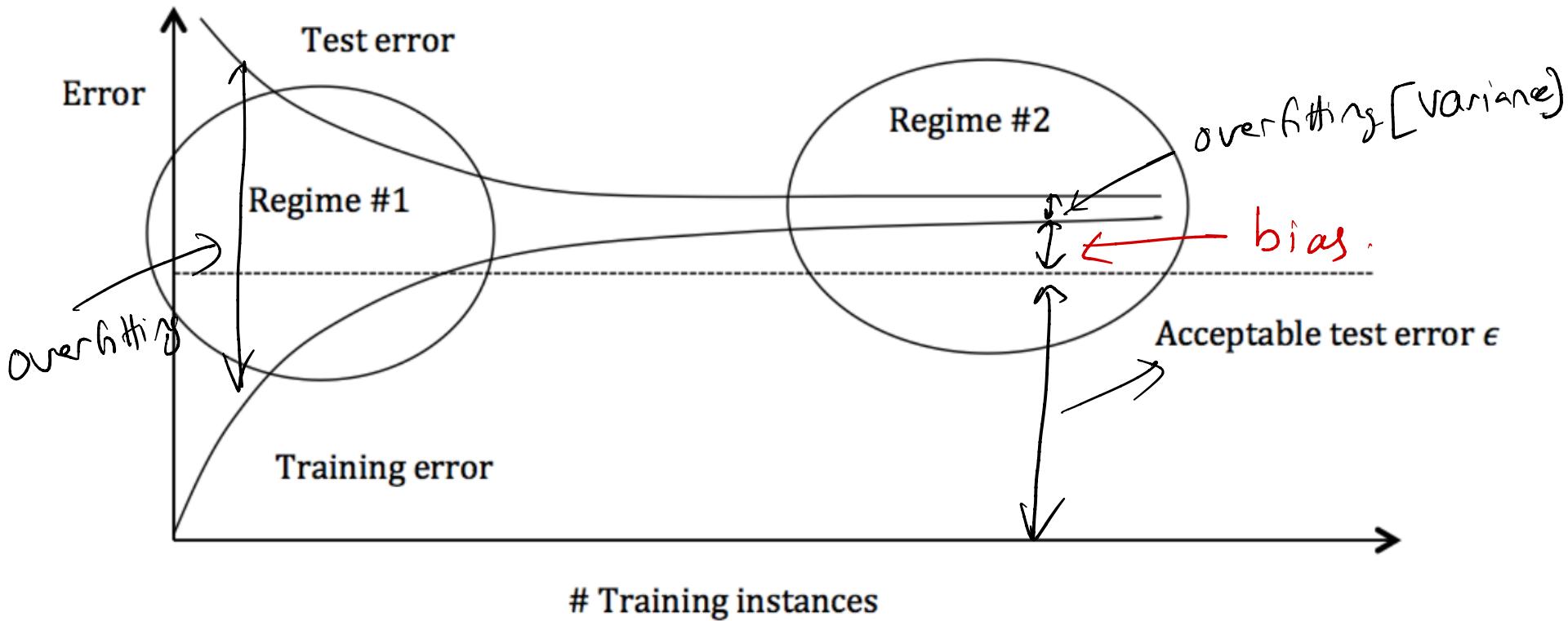
---

- Low variance
  - Constant function
  - Model independent of training data
- High variance
  - High degree polynomial

# Bias/Variance Tradeoff

- $(\text{bias}^2 + \text{variance})$  is what counts for prediction
- As we saw in PAC learning, we often have
  - Low bias  $\Rightarrow$  high variance
  - Low variance  $\Rightarrow$  high bias
  - How can we deal with this in practice?

# Detecting High Variance/Bias



# Detecting High Variance

↑  
Overfitting.

## Regime 1 (High Variance)

In the first regime, the cause of the poor performance is high variance.

### Symptoms:

1. Training error is much lower than test error
2. Training error is lower than  $\epsilon$
3. Test error is above  $\epsilon$

### Remedies:

- Add more training data
- Reduce model complexity -- complex models are prone to high variance
- Bagging (will be covered later in the course)

# Detecting High Bias

## Regime 2 (High Bias)

Unlike the first regime, the second regime indicates high bias: the model being used is not robust enough to produce an accurate prediction.

### Symptoms:

1. Training error is higher than  $\epsilon$

### Remedies:

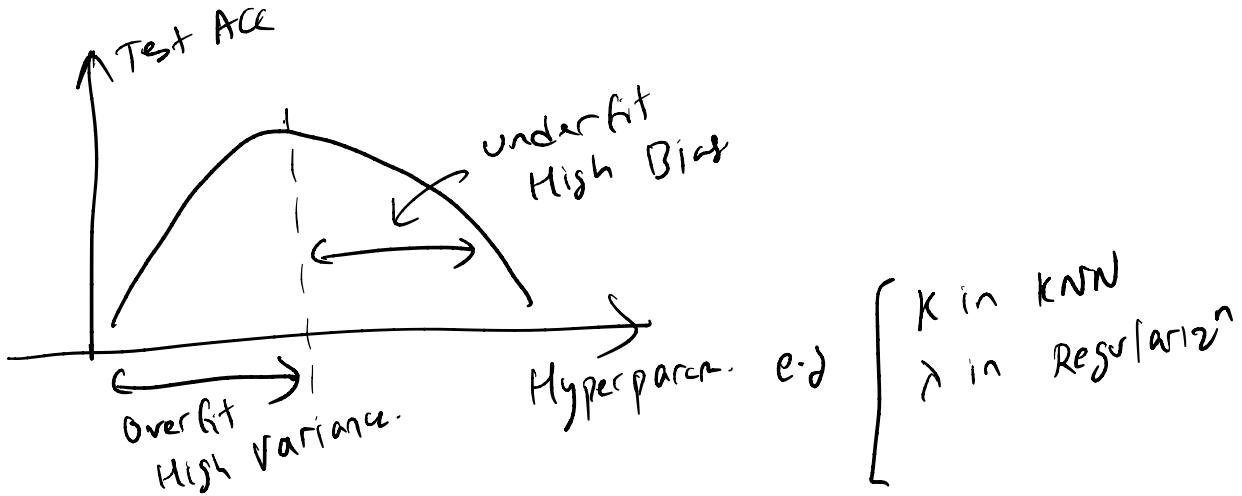
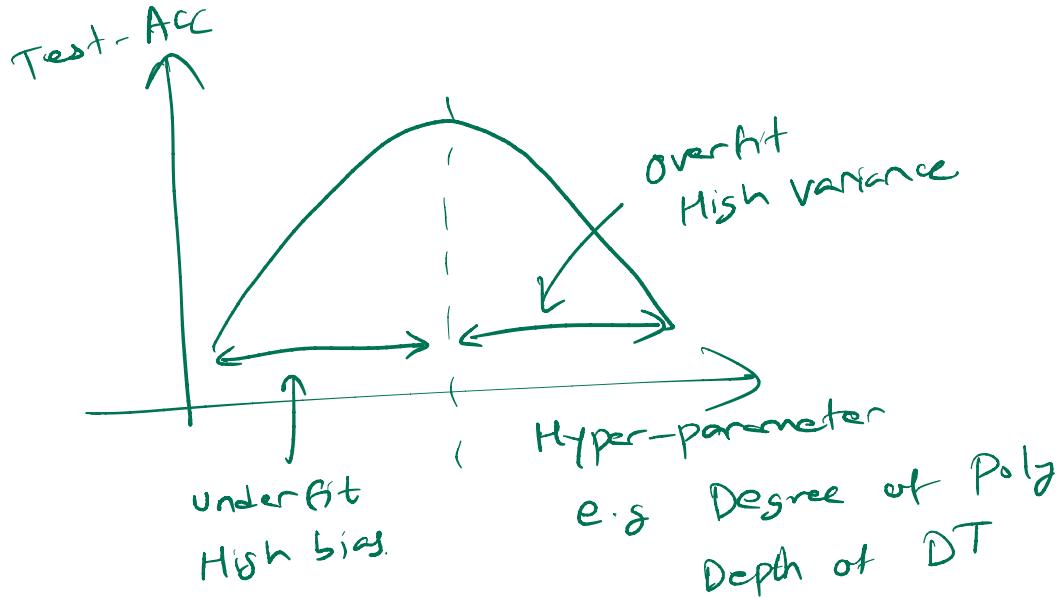
- Use more complex model (e.g. kernelize, use non-linear models)
- Add features
- Boosting (will be covered later in the course)

# How to select the right model?

Model Spaces with increasing complexity:

- Nearest-Neighbor classifiers with varying neighborhood sizes  $k = 1, 2, 3, \dots$   
Small neighborhood => Higher complexity
- Decision Trees with depth  $k$  or with  $k$  leaves  
Higher depth/ More # leaves => Higher complexity
- Regression with polynomials of order  $k = 0, 1, 2, \dots$   
Higher degree => Higher complexity
- Kernel Regression with bandwidth  $h$   
Small bandwidth => Higher complexity

**How can we select the right complexity model ?**



# Held out Validation Set

We would like to pick the model that has smallest generalization error.

Can judge generalization error by using an independent sample of data.

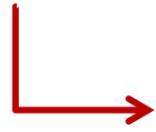
## Hold - out procedure:

n data points available  $D \equiv \{X_i, Y_i\}_{i=1}^n$

- 1) Split into two sets:      Training dataset      Validation dataset      NOT test Data !!  
 $D_T = \{X_i, Y_i\}_{i=1}^m$        $D_V = \{X_i, Y_i\}_{i=m+1}^n$

- 2) Use  $D_T$  for training a predictor from each model class:

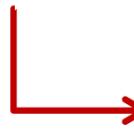
$$\hat{f}_\lambda = \arg \min_{f \in \mathcal{F}_\lambda} \hat{R}_T(f)$$

 Evaluated on training dataset  $D_T$

# Held out Validation Set

3) Use  $D_V$  to select the model class which has smallest empirical error on  $D_V$

$$\hat{\lambda} = \arg \min_{\lambda \in \Lambda} \hat{R}_V(\hat{f}_\lambda)$$

 Evaluated on validation dataset  $D_V$

4) Hold-out predictor

$$\hat{f} = \hat{f}_{\hat{\lambda}}$$

**Intuition:** Small error on one set of data will not imply small error on a randomly sub-sampled second set of data

Ensures method is “stable”

# Cross Validation

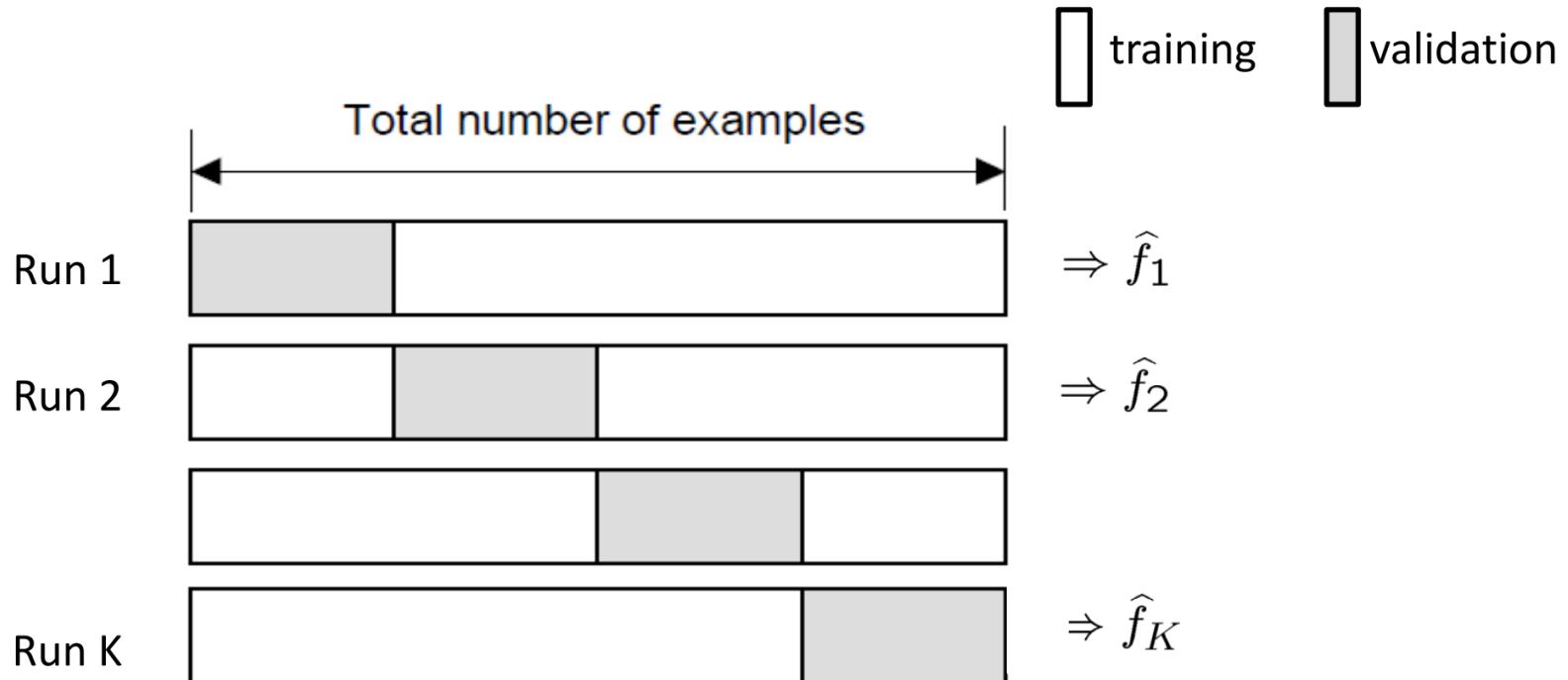


## K-fold cross-validation

Create K-fold partition of the dataset.

Form K hold-out predictors, each time using one partition as validation and rest K-1 as training datasets.

Final predictor is average/majority vote over the K hold-out estimates.



# Ensemble Methods

$$y = f(x)$$

$$y = \sum_{i=1}^T f_i(x) : f_1, f_2, \dots, f_T$$

# Reduce Variance Without Increasing Bias

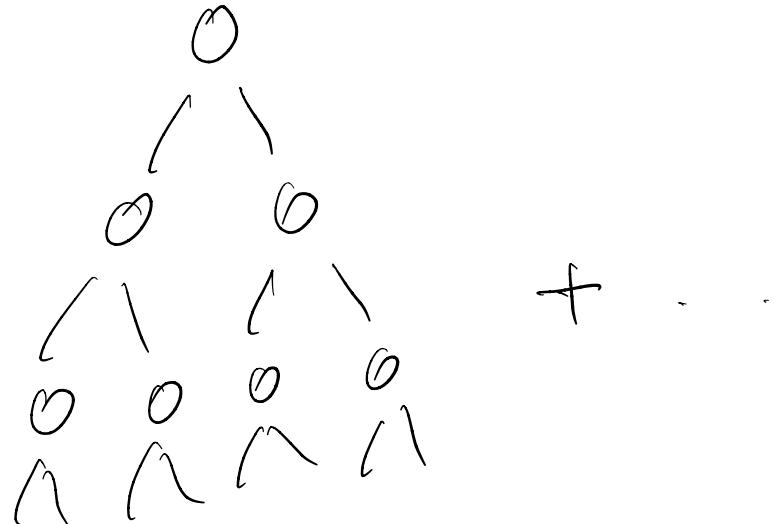


- **Averaging** reduces variance: let  $Z_1, \dots, Z_N$  be i.i.d random variables

$$Var\left(\overbrace{\frac{1}{N} \sum_i Z_i}^{Z_{\text{avg.}}}\right) = \frac{1}{N} Var(Z_i)$$
$$E[Z_{\text{avg.}}] = E[Z_i]$$

- Idea: average models to reduce model variance
- How to apply it to our setting?
  - Only one training set
  - Where do multiple models come from?

$$\overbrace{\sum_{i=1}^{10} DT_i(n)}^{10}$$



DT1

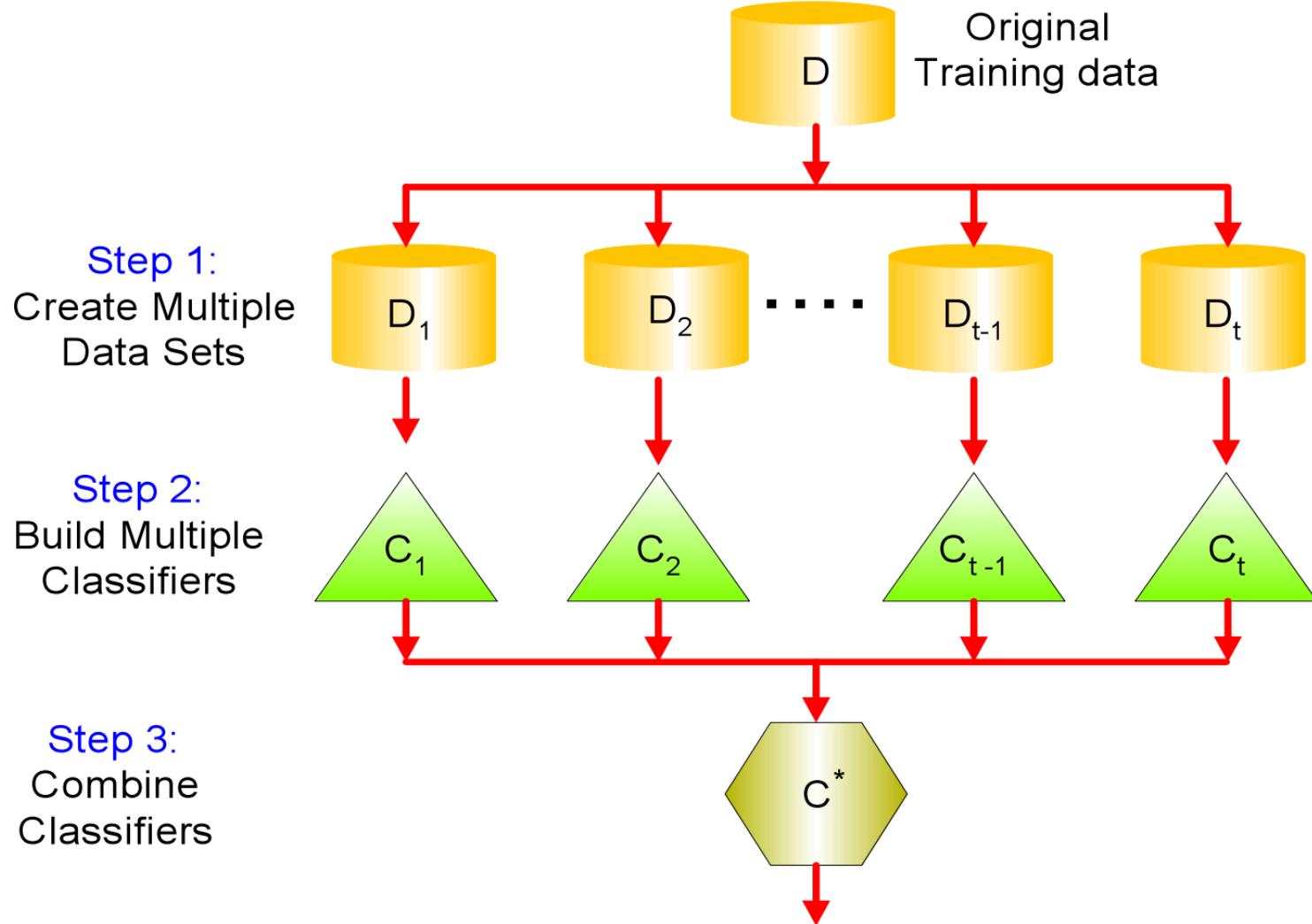
DT2

..

# Bagging: Bootstrap Aggregation

- Take repeated bootstrap samples from training set  $D$  (Breiman, 1994)
- **Bootstrap sampling:** Given set  $D$  containing  $N$  training examples, create  $D'$  by drawing  $N$  examples at random **with replacement** from  $D$
- **Bagging:**
  - Create  $k$  bootstrap samples  $D_1, \dots, D_k$
  - Train distinct classifier on each  $D_i$
  - Classify new instance by majority vote / average

# Bagging: Bootstrap Aggregation



# Bagging

| Data | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9  | 10 |
|------|---|---|---|----|---|---|---|---|----|----|
| BS 1 | 7 | 1 | 9 | 10 | 7 | 8 | 8 | 4 | 7  | 2  |
| BS 2 | 8 | 1 | 3 | 1  | 1 | 9 | 7 | 4 | 10 | 1  |
| BS 3 | 5 | 4 | 8 | 8  | 2 | 5 | 5 | 7 | 8  | 8  |

- Build a classifier from each bootstrap sample
- In each bootstrap sample, each data point has probability  $\left(1 - \frac{1}{N}\right)^N$  of not being selected
  - Expected number of distinct data points in each sample is then

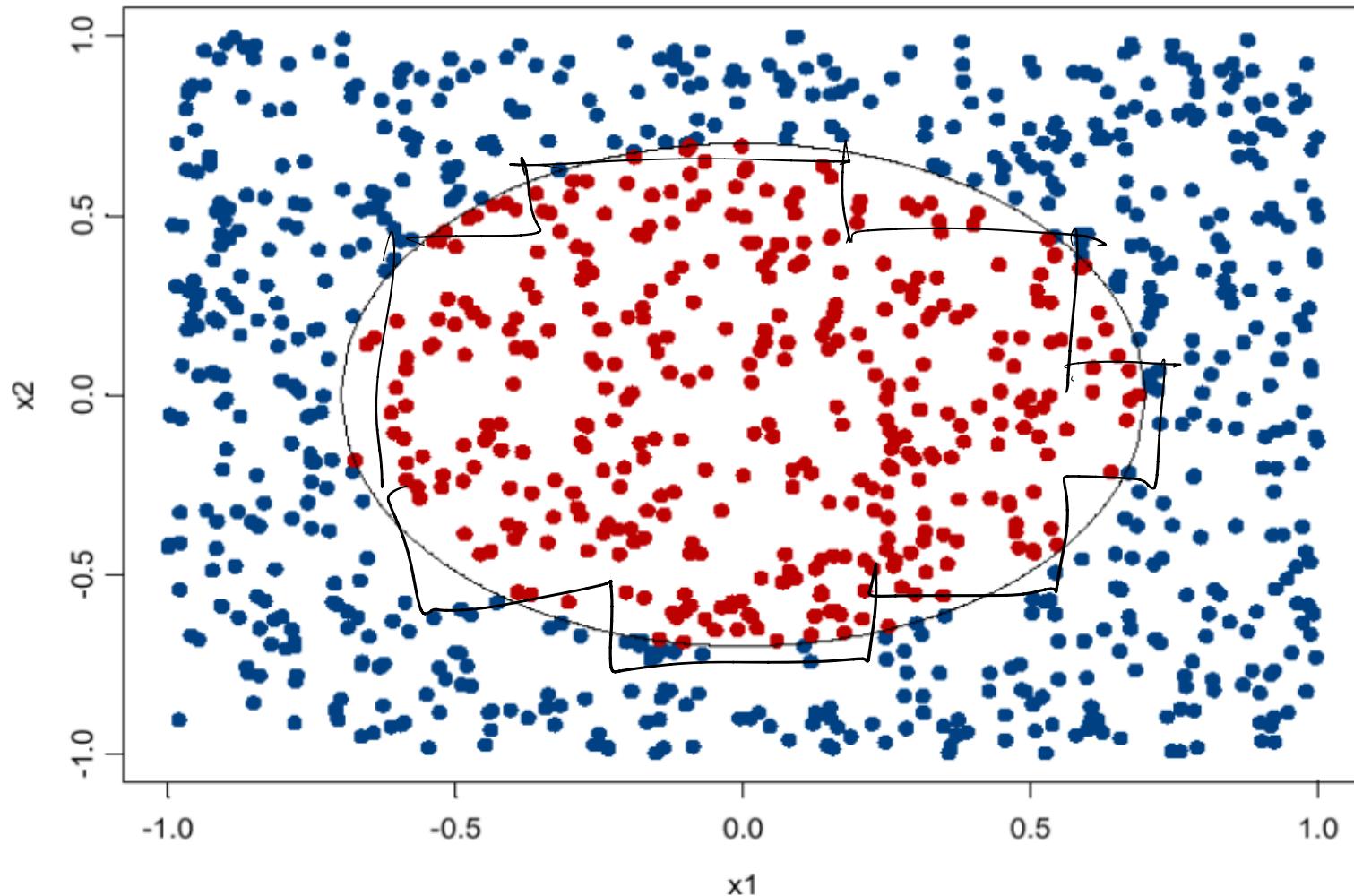
$$N \cdot \left(1 - \left(1 - \frac{1}{N}\right)^N\right) \approx N \cdot (1 - \exp(-1)) = .632 \cdot N$$

# Bagging

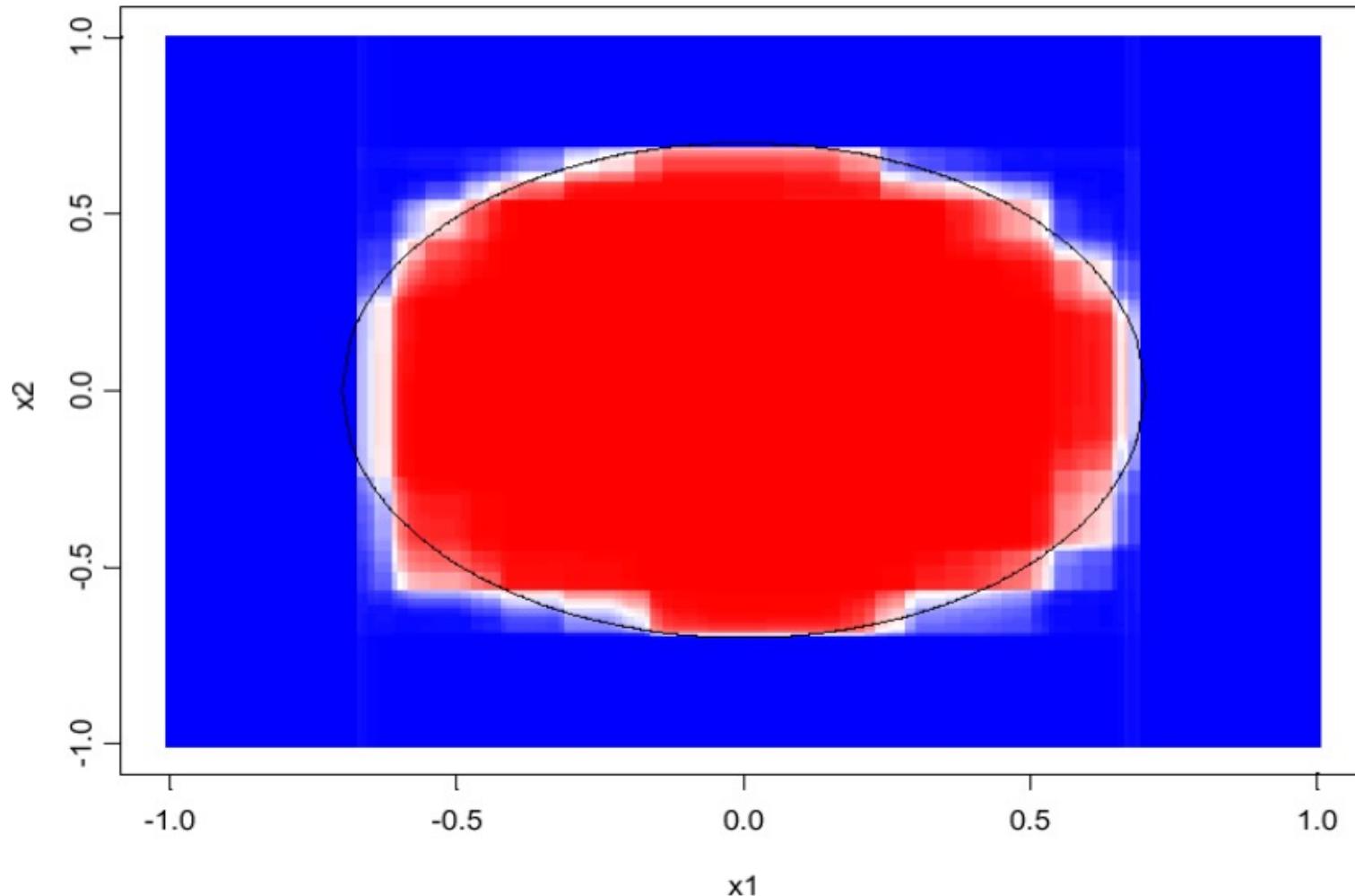
| Data | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9  | 10 |
|------|---|---|---|----|---|---|---|---|----|----|
| BS 1 | 7 | 1 | 9 | 10 | 7 | 8 | 8 | 4 | 7  | 2  |
| BS 2 | 8 | 1 | 3 | 1  | 1 | 9 | 7 | 4 | 10 | 1  |
| BS 3 | 5 | 4 | 8 | 8  | 2 | 5 | 5 | 7 | 8  | 8  |

- Build a classifier from each bootstrap sample
- In each bootstrap sample, each data point has probability  $\left(1 - \frac{1}{N}\right)^N$  of not being selected
  - If we have 1 TB of data, each bootstrap sample will be ~ 632GB (this can present computational challenges, e.g., you shouldn't replicate the data)

# Decision Tree Bagging



# Decision Tree Bagging (100 Bagged Trees)

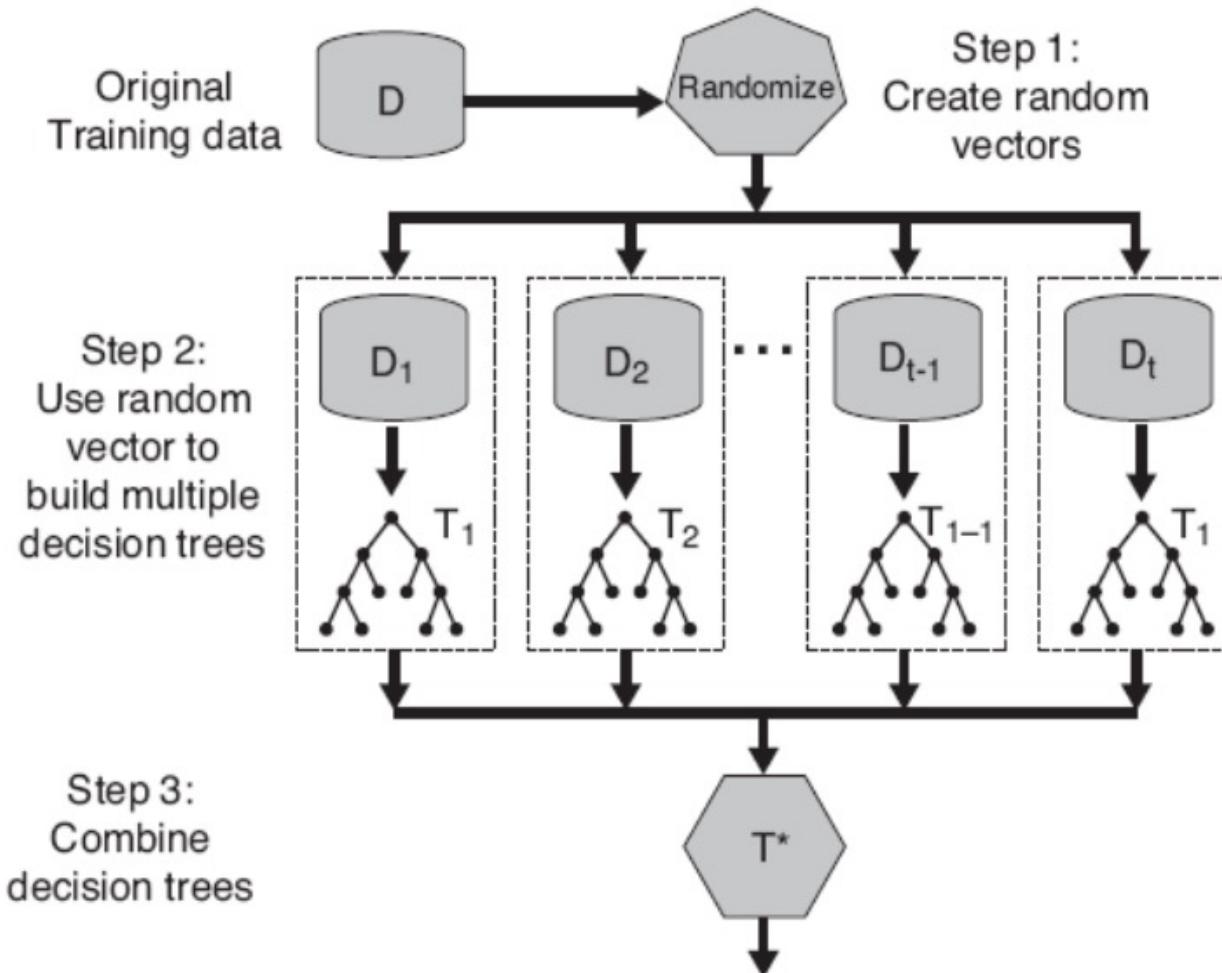


# Bagging Results

| Data Set      | Without<br>Bagging | With<br>Bagging | Decrease |
|---------------|--------------------|-----------------|----------|
| waveform      | 29.1               | 19.3            | 34%      |
| heart         | 4.9                | 2.8             | 43%      |
| breast cancer | 5.9                | 3.7             | 37%      |
| ionosphere    | 11.2               | 7.9             | 29%      |
| diabetes      | 25.3               | 23.9            | 6%       |
| glass         | 30.4               | 23.6            | 22%      |
| soybean       | 8.6                | 6.8             | 21%      |

Breiman “Bagging Predictors” Berkeley Statistics Department TR#421, 1994

# Random Forests



# Random Forests



- Ensemble method specifically designed for decision tree classifiers
- Introduce two sources of randomness: “bagging” and “random input vectors”
  - Bagging method: each tree is grown using a bootstrap sample of training data ← Fixes overfitting on data space.
  - **Random vector method:** best split at each node is chosen from a random sample of  $m$  attributes instead of all attributes → Fixes overfitting in feature space.

# Random Forest Algorithm

- For  $b = 1$  to  $B$ 
  - Draw a bootstrap sample of size  $N$  from the data
  - Grow a tree  $T_b$  using the bootstrap sample as follows
    - Choose  $\underline{m}$  attributes uniformly at random from the data
    - Choose the best attribute among the  $m$  to split on
    - Split on the best attribute and recurse (until partitions have fewer than  $s_{min}$  number of nodes)
- Prediction for a new data point  $x$ 
  - Regression:  $\frac{1}{B} \sum_b T_b(x)$
  - Classification: choose the majority class label among  $T_1(x), \dots, T_B(x)$

# Random Forest Demo

---



A [demo](#) of random forests implemented in JavaScript

# When Will Bagging Improve Accuracy?



- Depends on the stability of the base-level classifiers
- A learner is **unstable** if a small change to the training set causes a large change in the output hypothesis ← High Variance.
  - If small changes in  $D$  cause large changes in the output, then there will likely be an improvement in performance with bagging
- Bagging can help unstable procedures, but could hurt the performance of stable procedures
  - Decision trees are unstable
  - $k$ -nearest neighbor is stable

