



Lecture 16

Bias/Variance Tradeoff and Ensemble Methods

Rishabh Iyer

University of Texas at Dallas

Last Time

- PAC learning

$$\epsilon^{\text{test}} \leq \epsilon^{\text{train}} + \underbrace{\text{Var}(\text{VC}(h), M, \epsilon, \delta)}_{\text{Variance.}}$$

$\propto \frac{1}{\text{VC}(h)M}$

- Bias/variance tradeoff

- small hypothesis spaces (not enough flexibility) can have high bias

↑
Less Complex (Linear set) High bias / Low variance.

- rich hypothesis spaces (too much flexibility) can have high variance

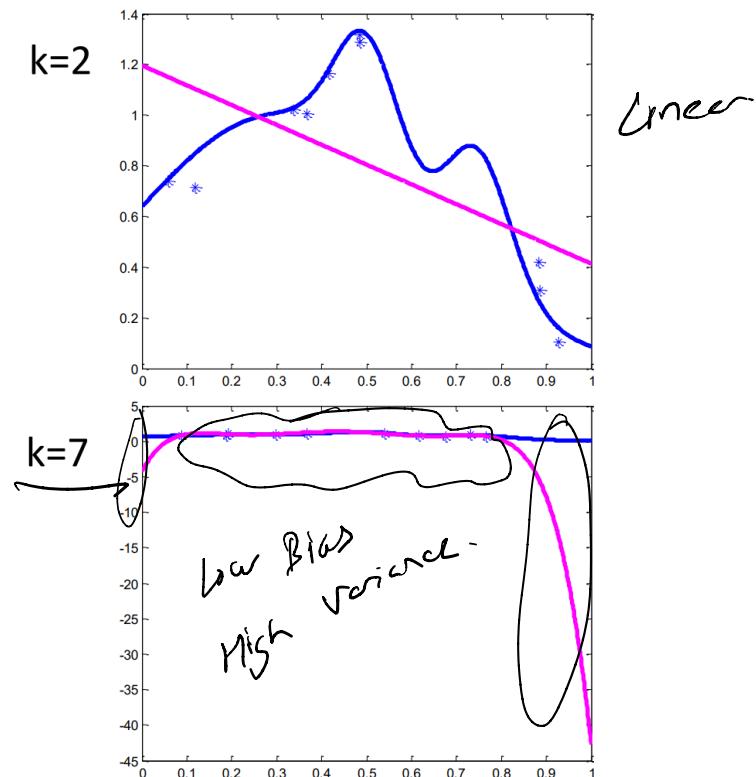
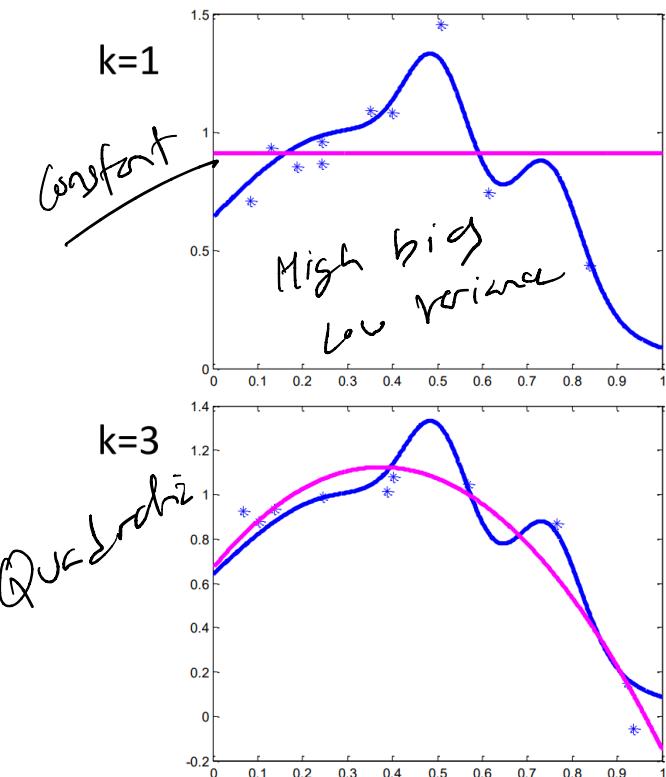
↑
More Complex Lower Bias / High Variance.

- Today: more on this phenomenon and how to get around it

High Variance or Overfitting

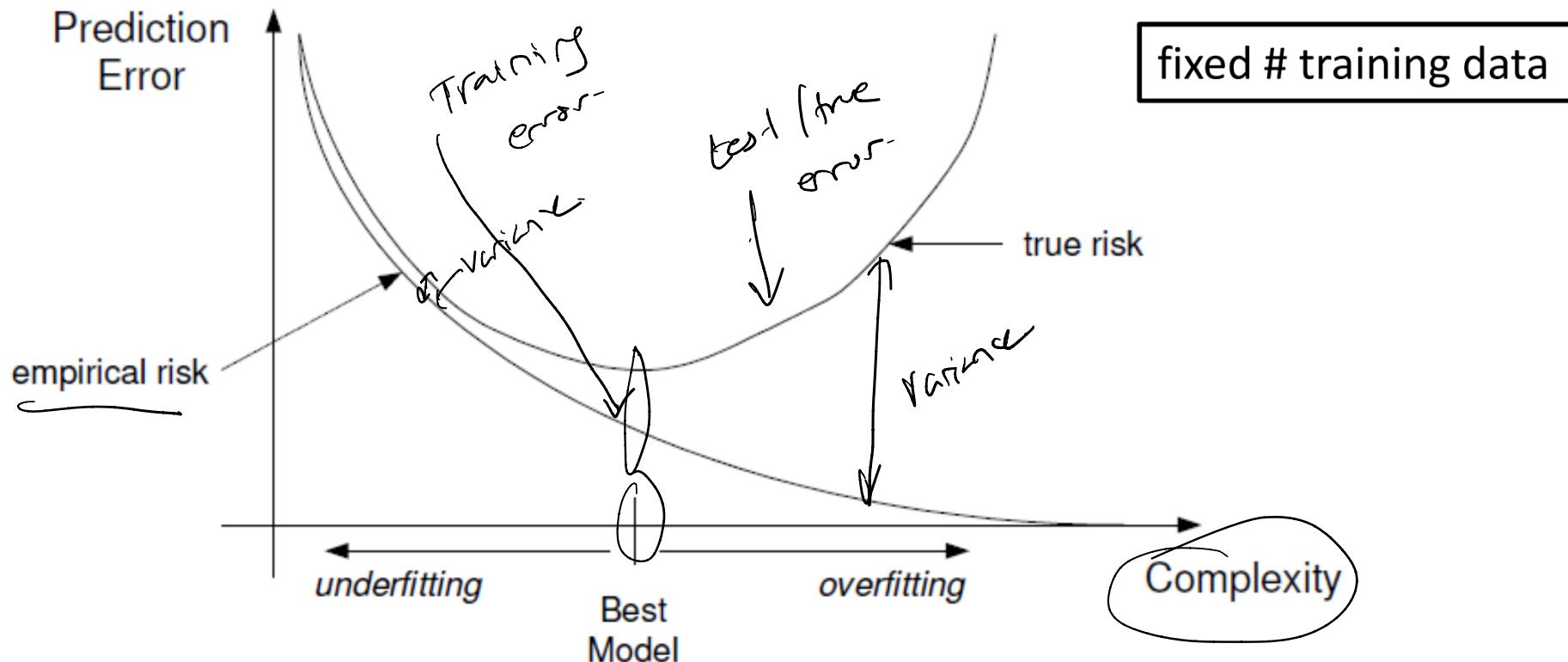
If we allow very complicated predictors, we could overfit the training data.

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



Effect of Model Complexity

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



Intuition



- Bias
 - Measures the accuracy or quality of the algorithm
 - High bias means a poor match
- Variance
 - 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

How well does the model minimize emp / train error

How close is the train error & test error

5

Bias-Variance Analysis in Regression



- Dataset: $D = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$
- True function is $y = f(x) + \epsilon$
 - Where noise, ϵ , is normally distributed with zero mean and standard deviation σ
- 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

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

Some Terminology [Focus on Regression]



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

$D \rightarrow \text{Dataset}$
 $h_D \rightarrow \text{Hyp trained on } D$
 $P = \text{true v prob dist}$
 joint

$$\bar{y}(\mathbf{x}) = E_{y|\mathbf{x}}[Y] = \int_y y \underbrace{\Pr(y|\mathbf{x})}_{\substack{\text{true} \\ \text{prob}}} dy.$$

\uparrow
 true label

Expected Test Error (given $\underline{h_D}$):

$$\underbrace{E_{(\mathbf{x},y) \sim P} \left[(h_D(\mathbf{x}) - y)^2 \right]}_{E(ny)} = \iint_{x,y} (h_D(\mathbf{x}) - y)^2 \Pr(\mathbf{x}, y) \partial y \partial \mathbf{x}.$$

Expected Classifier (given \mathcal{A}):

$$\overline{h} = \underbrace{E_{D \sim P^n}[h_D]}_{\text{Dataset}} = \int_D h_D \Pr(D) \partial D$$

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

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

$P^n = \text{Joint dist of } \{(x_1, y_1), \dots, (x_n, y_n)\}$

Probability Reminder

- Variance of a random variable, Z

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

- Properties of $Var(Z)$

$$\underline{Var(aZ)} = E[a^2Z^2] - E[aZ]^2 = \underline{a^2Var(Z)}$$

Bias-Variance-Noise Decomposition

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

Bias-Variance-Noise Decomposition

$$\begin{aligned}
 E_{\mathbf{x},y,D} \left[[h_D(\mathbf{x}) - y]^2 \right] &= E_{\mathbf{x},y,D} \left[[(h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) + (\bar{h}(\mathbf{x}) - y)]^2 \right] \\
 &= \underbrace{E_{\mathbf{x},D} \left[(\bar{h}_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{true error}} + \underbrace{2 E_{\mathbf{x},y,D} \left[(h_D(\mathbf{x}) - \bar{h}(\mathbf{x})) (\bar{h}(\mathbf{x}) - y) \right]}_{0} \\
 &\quad + \underbrace{E_{\mathbf{x},y} \left[(\bar{h}(\mathbf{x}) - y)^2 \right]}_{0}
 \end{aligned}$$

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

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

$\widehat{h}(n) \leq E_D[h_D(n)]$

Bias-Variance-Noise Decomposition

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

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

$h_D(\mathbf{x}) \rightarrow$ Classifier specific to a dataset D
 $\bar{h}(\mathbf{x}) \rightarrow$ Mean classifier

High variance comes from the fact that given different instantiations of a dataset D , we can obtain very different classifiers (h_D)

Bias-Variance-Noise Decomposition

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

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

D_1, D_2, \dots, D_K

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

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

$\bar{y}(\mathbf{x})$
 $\bar{h}(\mathbf{x})$
 \downarrow
 0
 $y = f(\mathbf{x}) + \epsilon$
 $f(\mathbf{x})$ true function which generates y

avg pred

avg classfns - $f(\mathbf{x})$

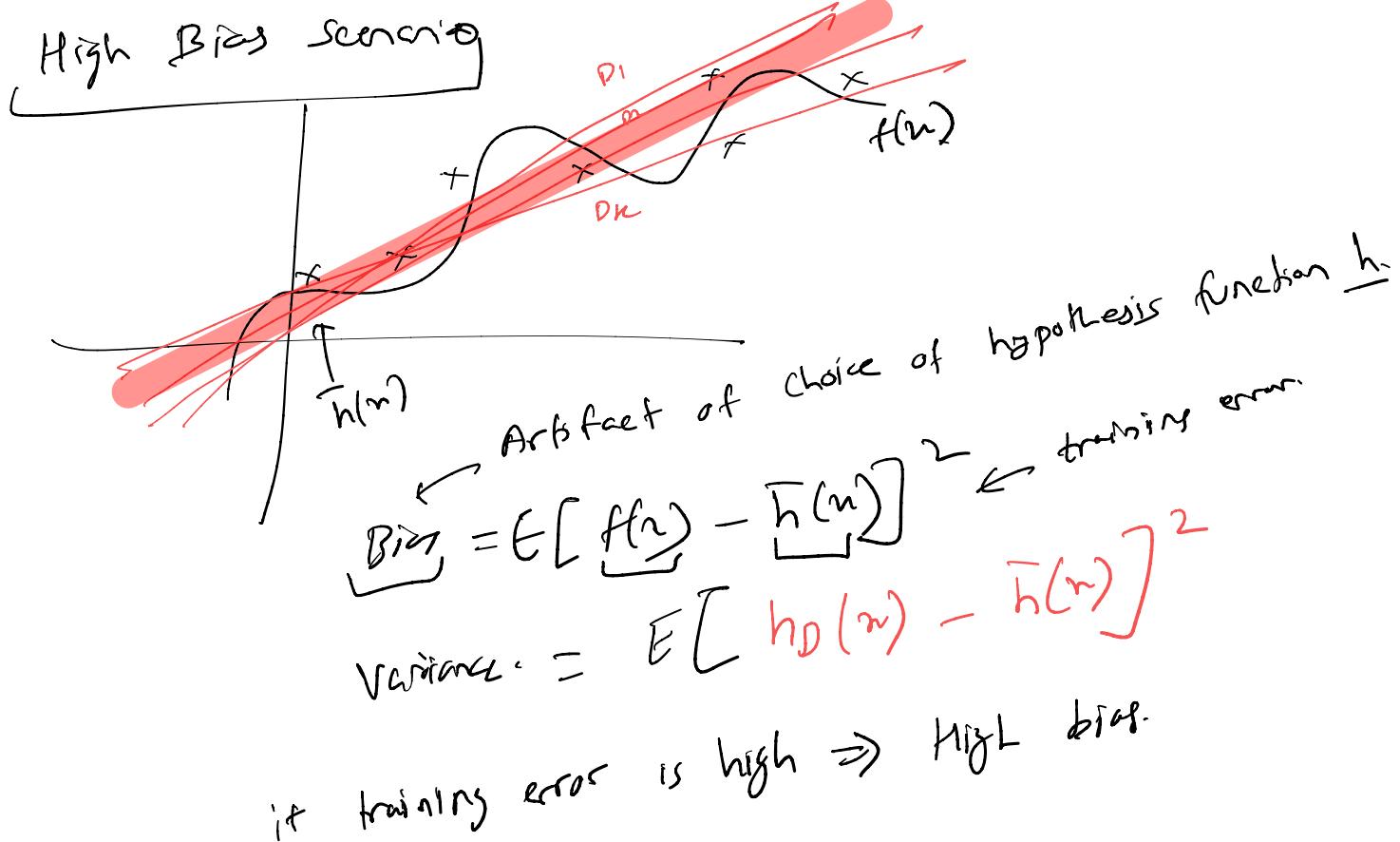
σ^2

Bias^2

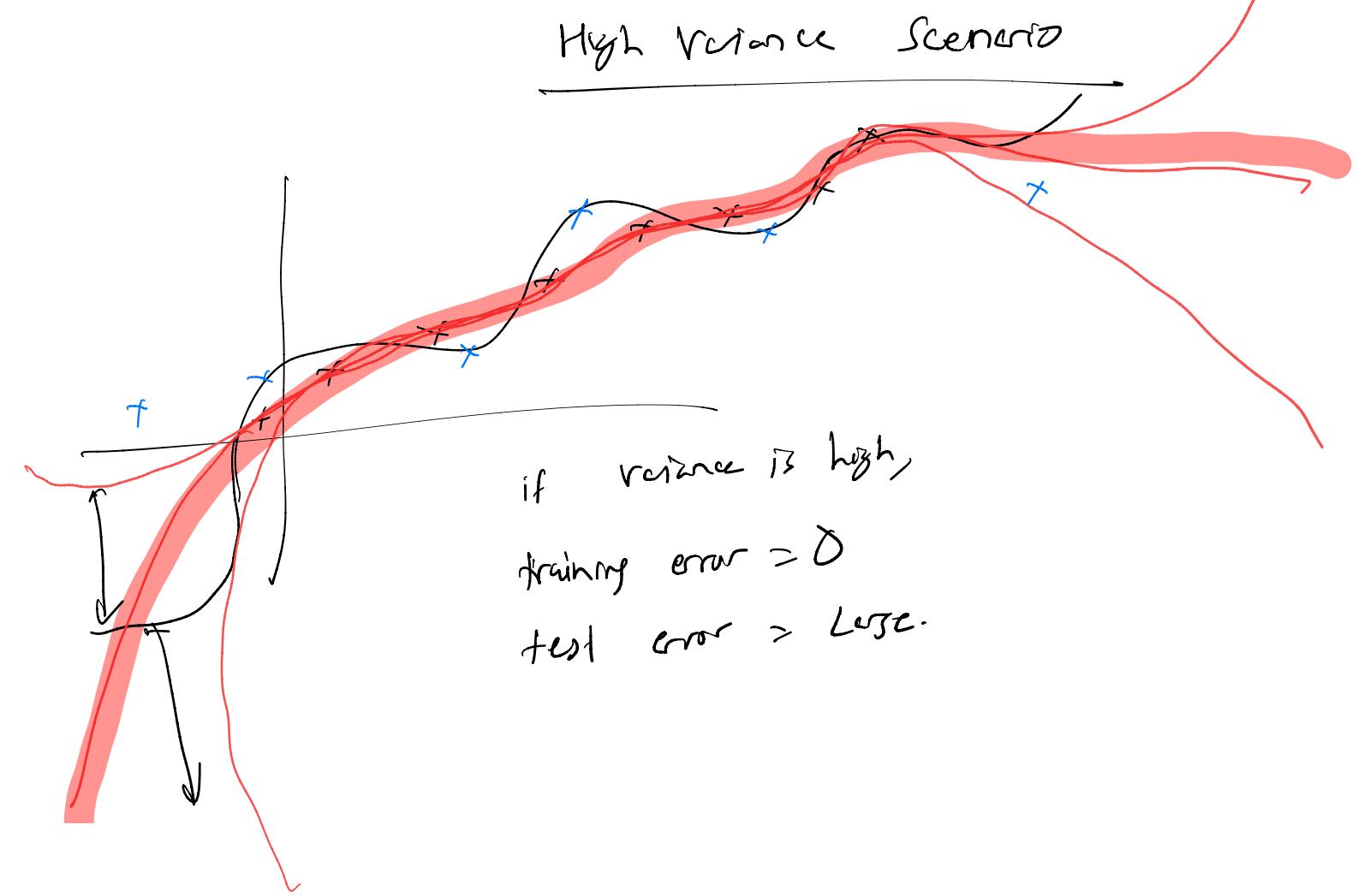
ϵ

$\bar{y}(\mathbf{x}) = f(\mathbf{x})$

$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon$



High Variance Scenario



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:

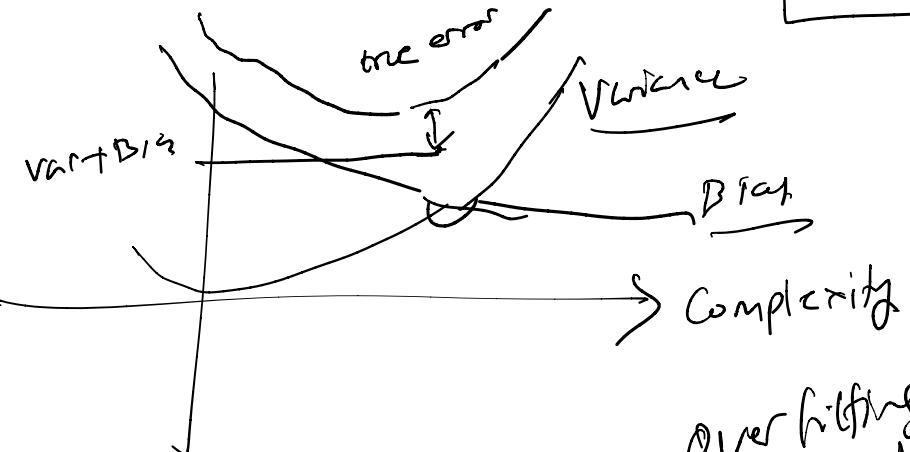
$$\begin{aligned} E_{\mathbf{x},y} \left[(\bar{h}(\mathbf{x}) - y)^2 \right] &= E_{\mathbf{x},y} \left[(\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})) + (\bar{y}(\mathbf{x}) - y)^2 \right] \\ &= \underbrace{E_{\mathbf{x},y} \left[(\bar{y}(\mathbf{x}) - y)^2 \right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}}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


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

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



Over fitting
High complexity

Cannot be helped

Underfitting
Low complexity

This gives us the decomposition of expected test error as follows

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

True.

σ^2

Bias, Variance, and Noise



This gives us the decomposition of expected test error as follows

$$\begin{aligned}
E_{\mathbf{x},y,D} \left[(h_D(\mathbf{x}) - y)^2 \right] &= \underbrace{E_{\mathbf{x},D} \left[(h_D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right]}_{\text{Expected Test Error}} + \underbrace{E_{\mathbf{x},y} \left[(\bar{y}(\mathbf{x}) - y)^2 \right]}_{\text{Noise}} \\
&\quad + \underbrace{E_{\mathbf{x}} \left[(\underbrace{\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})}_{\text{Bias}^2})^2 \right]}_{\text{Variance as } m \rightarrow \infty, \text{ var} \rightarrow 0} \\
&\quad \text{as } m \rightarrow \infty, h_D(n) \approx \hat{h}(n)
\end{aligned}$$

Variance: Captures how much your classifier changes if you train on a different training set. How "over-specialized" 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?

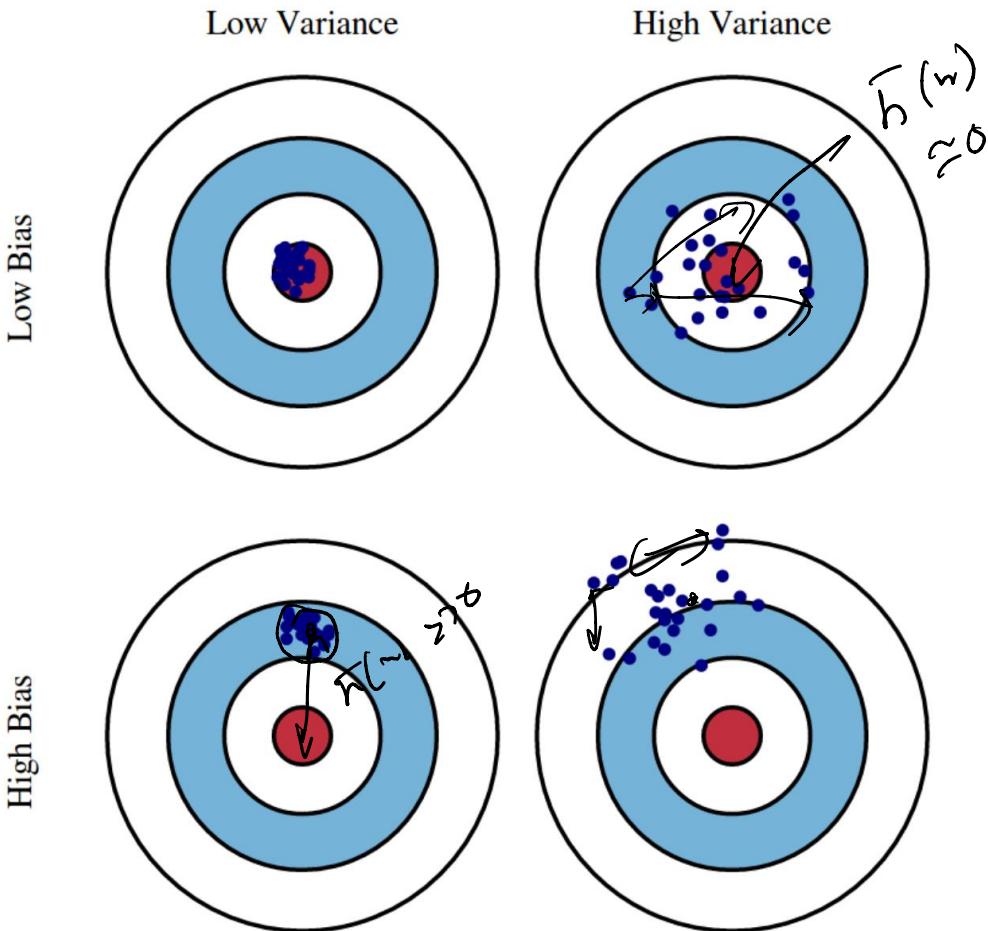
Bias: What is the inherent error that you obtain from your classifier even with infinite training data? This is due to your classifier being "biased" to a particular kind of solution (e.g. linear classifier). In other words, bias is inherent to your model.

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

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

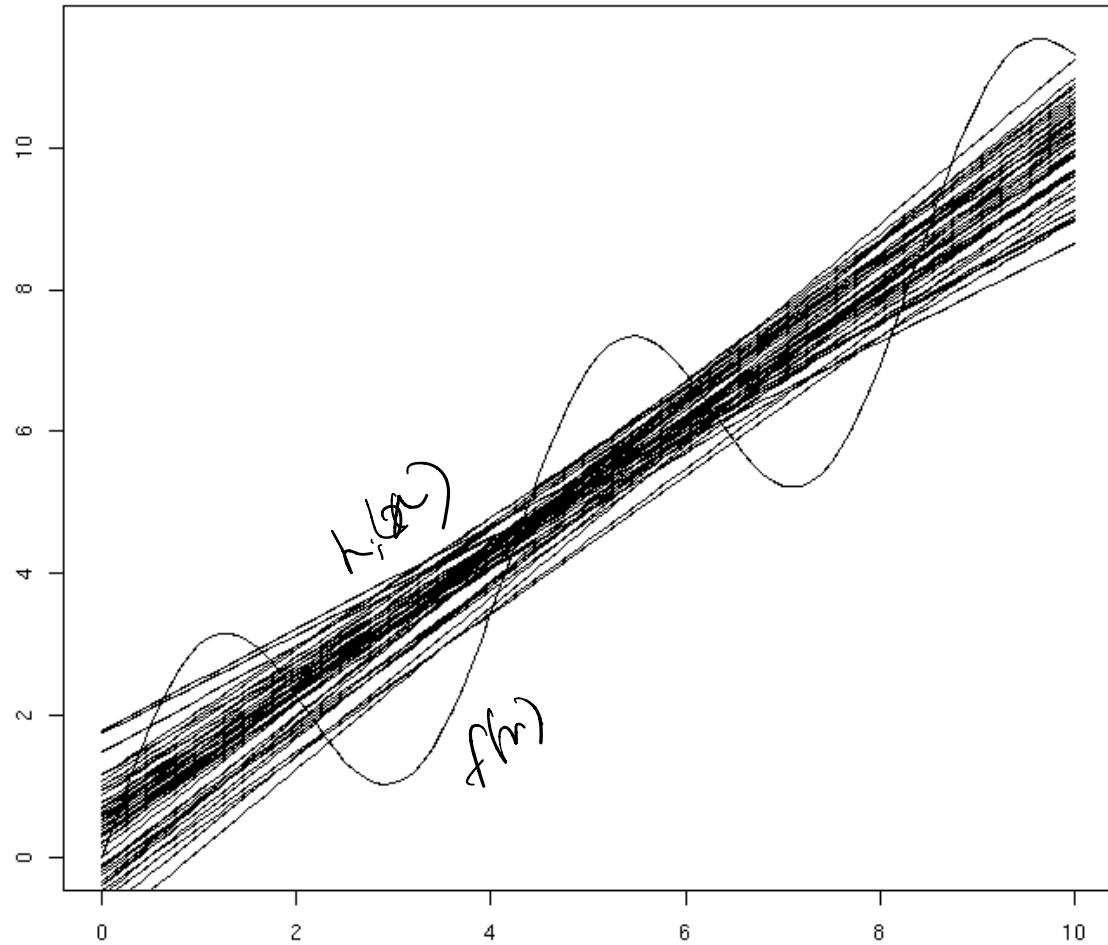
$E_{\mathbf{x}} [h(\mathbf{x})] = \int p(\mathbf{x}) h(\mathbf{x})$



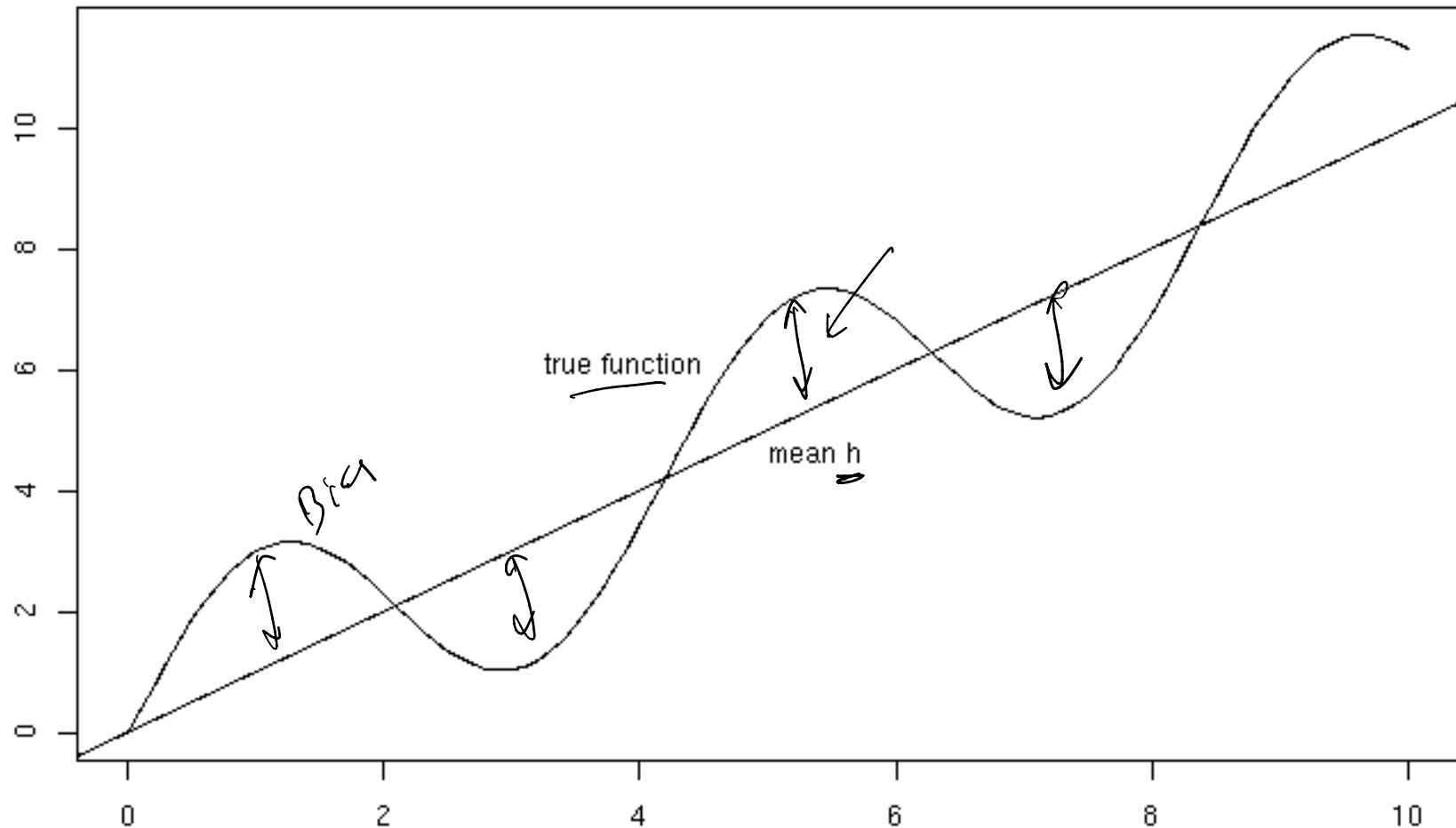
2-D Example



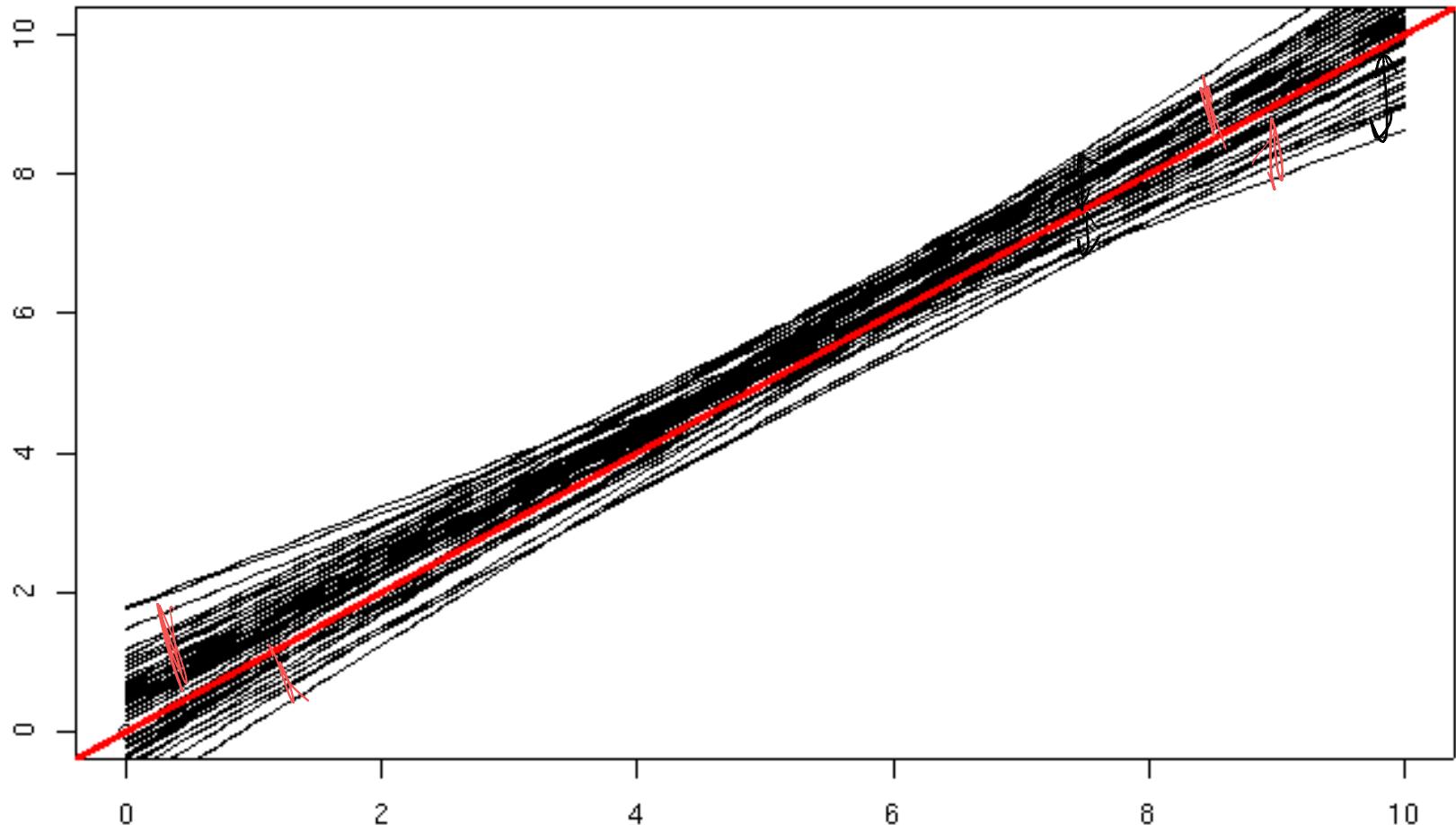
50 fits (20 examples each)



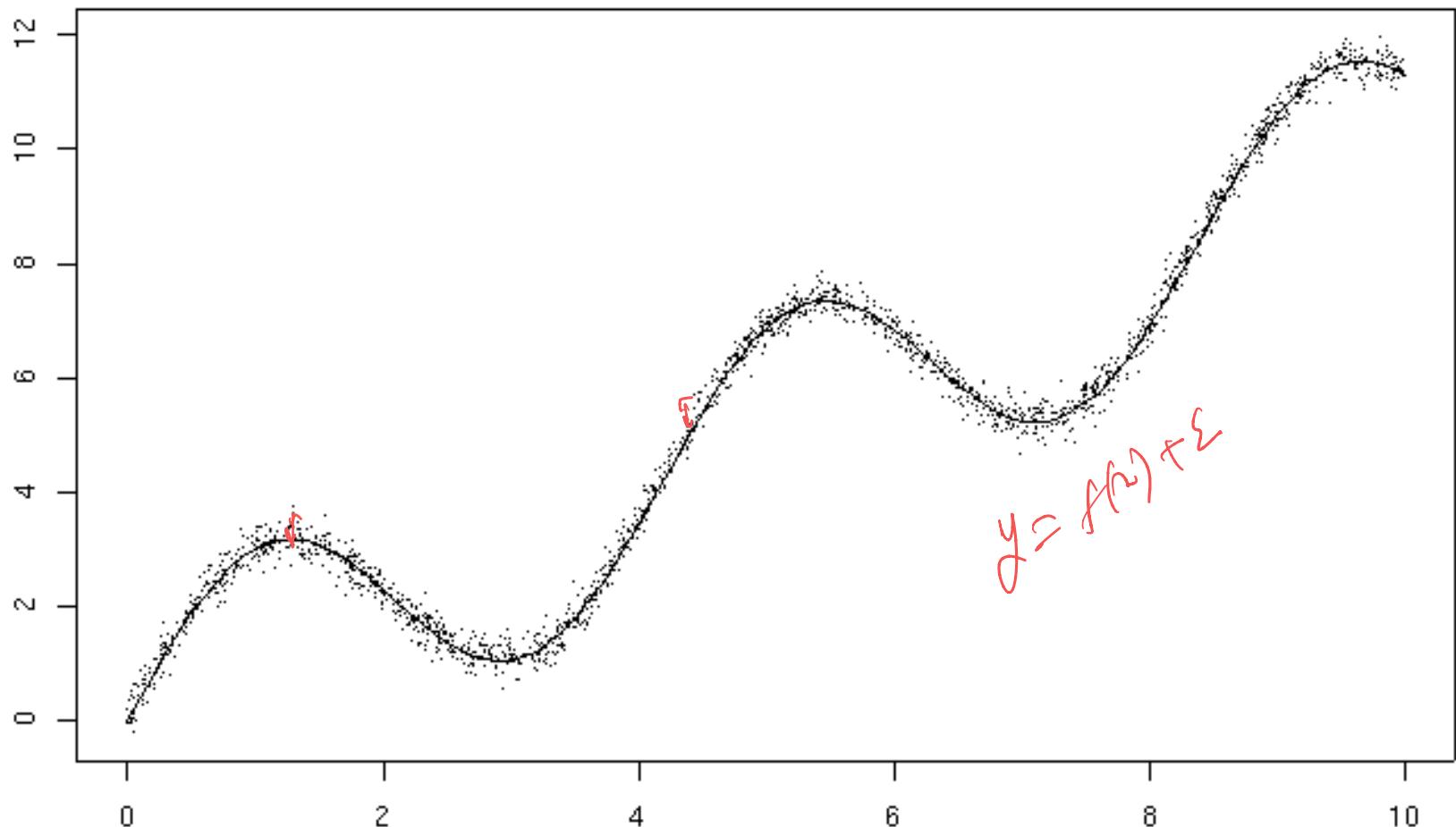
Bias



Variance



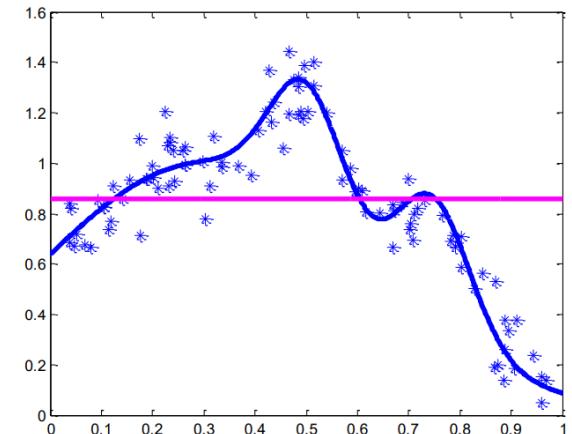
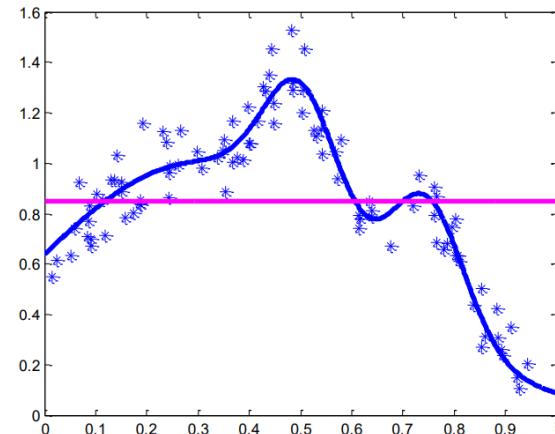
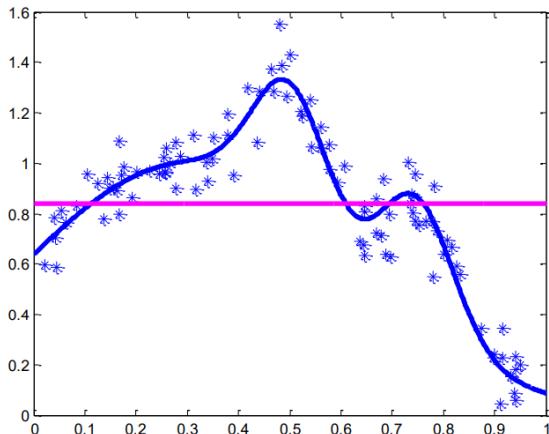
Noise



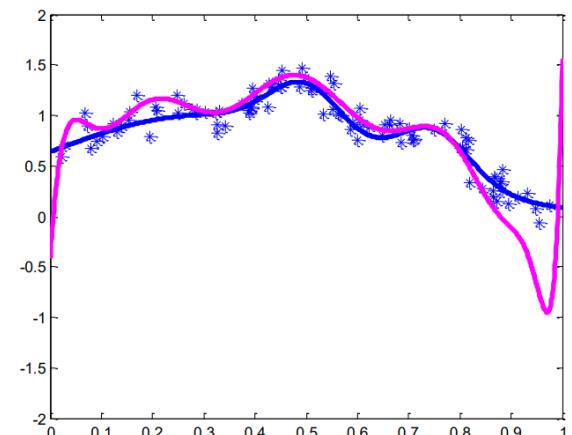
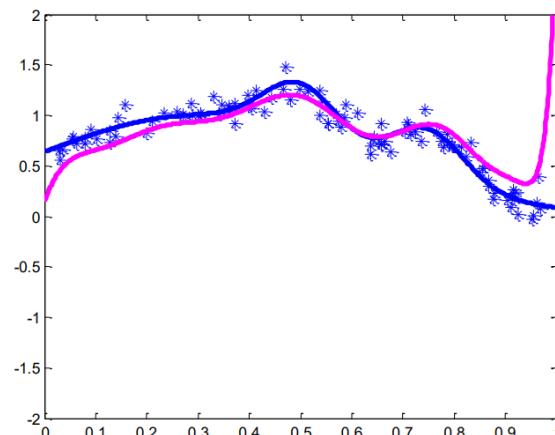
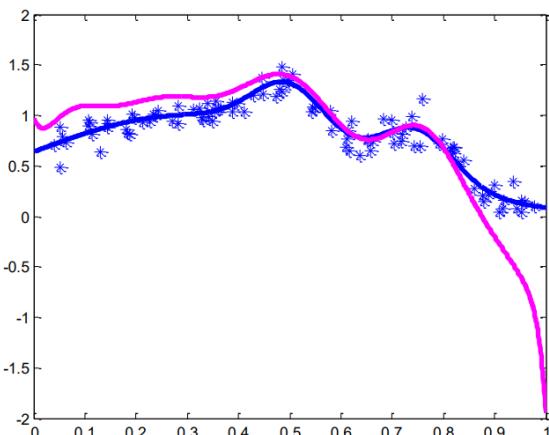
Bias-Variance Tradeoff



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

Match bet
 f & \hat{f}_n

Mis match
between
 f & \hat{f}_n

Variance

- Low variance
 - ?
- High variance
 - ?

Variance

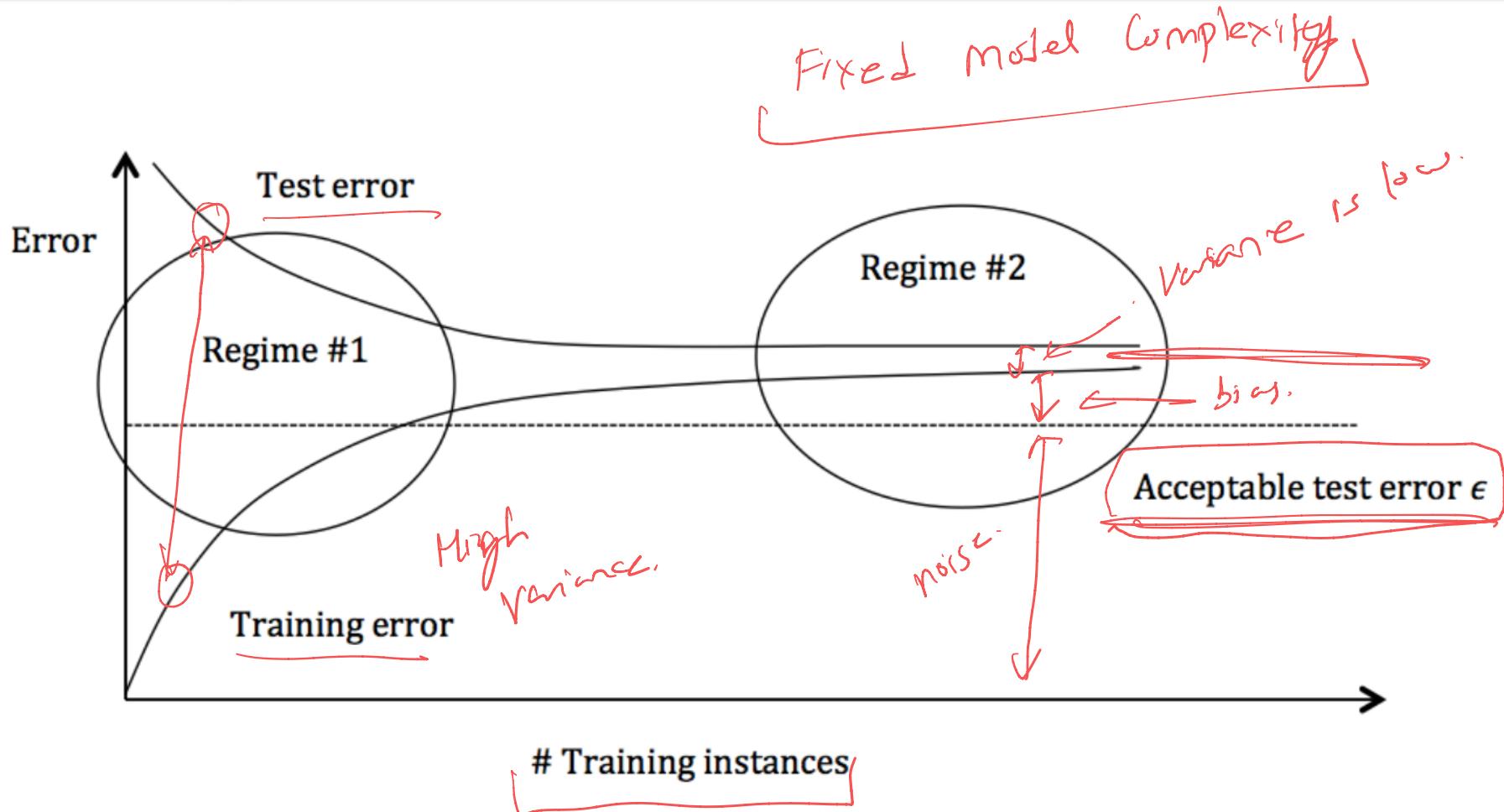
- Low variance
 - Constant function
 - Model independent of training data
- High variance
 - High degree polynomial

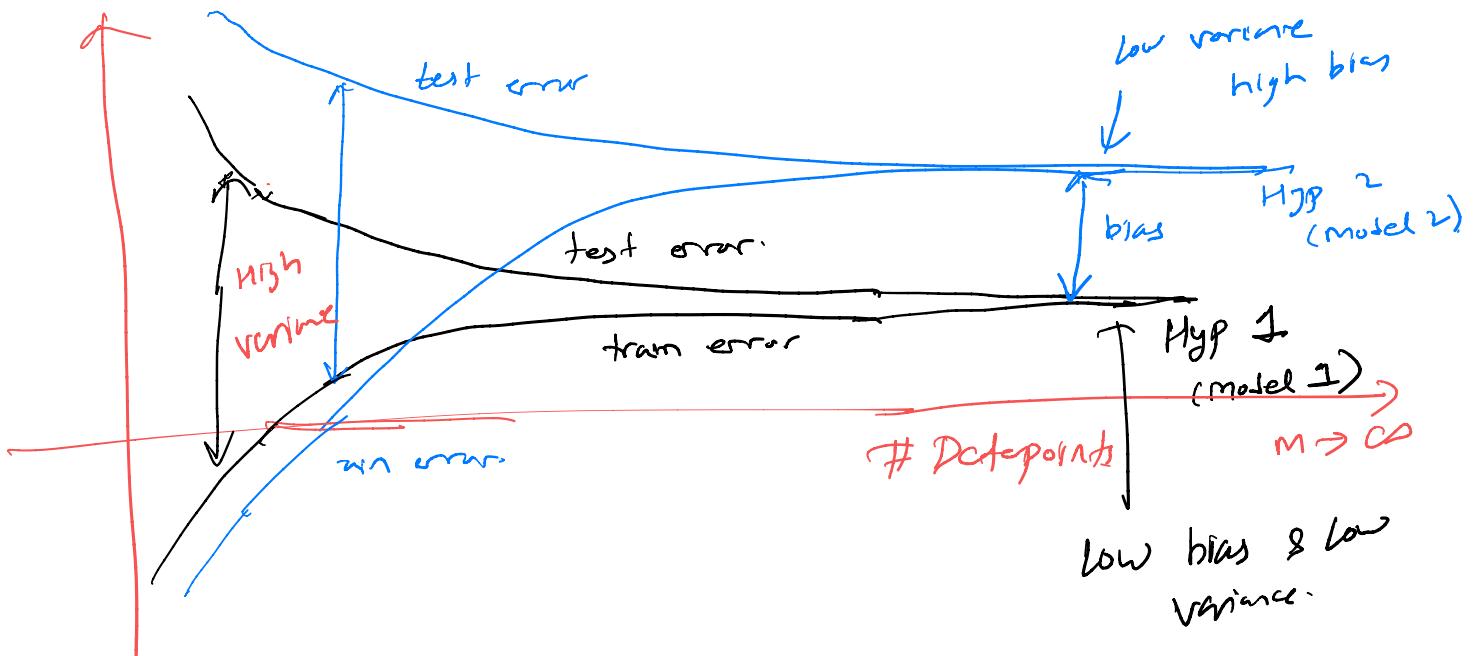
(Degree of h is \ggg Degree of s)

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 (*overfitting*)
 - Low variance \Rightarrow high bias (*underfitting*)
 - How can we deal with this in practice?

Detecting High Variance/Bias





Bias/Variance Trade-off with multiple
models

Detecting High Variance

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 ϵ
3. Test error is above ϵ

Remedies:

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

lecture.

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 ϵ

Low variance \Rightarrow train error \approx test-error.

Remedies:

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

*↑
next lecture.*

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

Complexity
 $\approx \# \text{parameters}$

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:

$$\mathcal{F}_P : f_{P,1} \geq f_{P,2} \geq \dots \geq f_{P,n}$$

\leftarrow Model Complexity

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

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

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

$$\hat{f}_\lambda = \arg \min_{f \in \mathcal{F}_\lambda} \hat{R}_T(f) \quad \leftarrow \text{Train Loss}$$

→ 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

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

$\Lambda = \{\lambda_1, \dots, \lambda_K\}$

← val set error / loss
 → 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

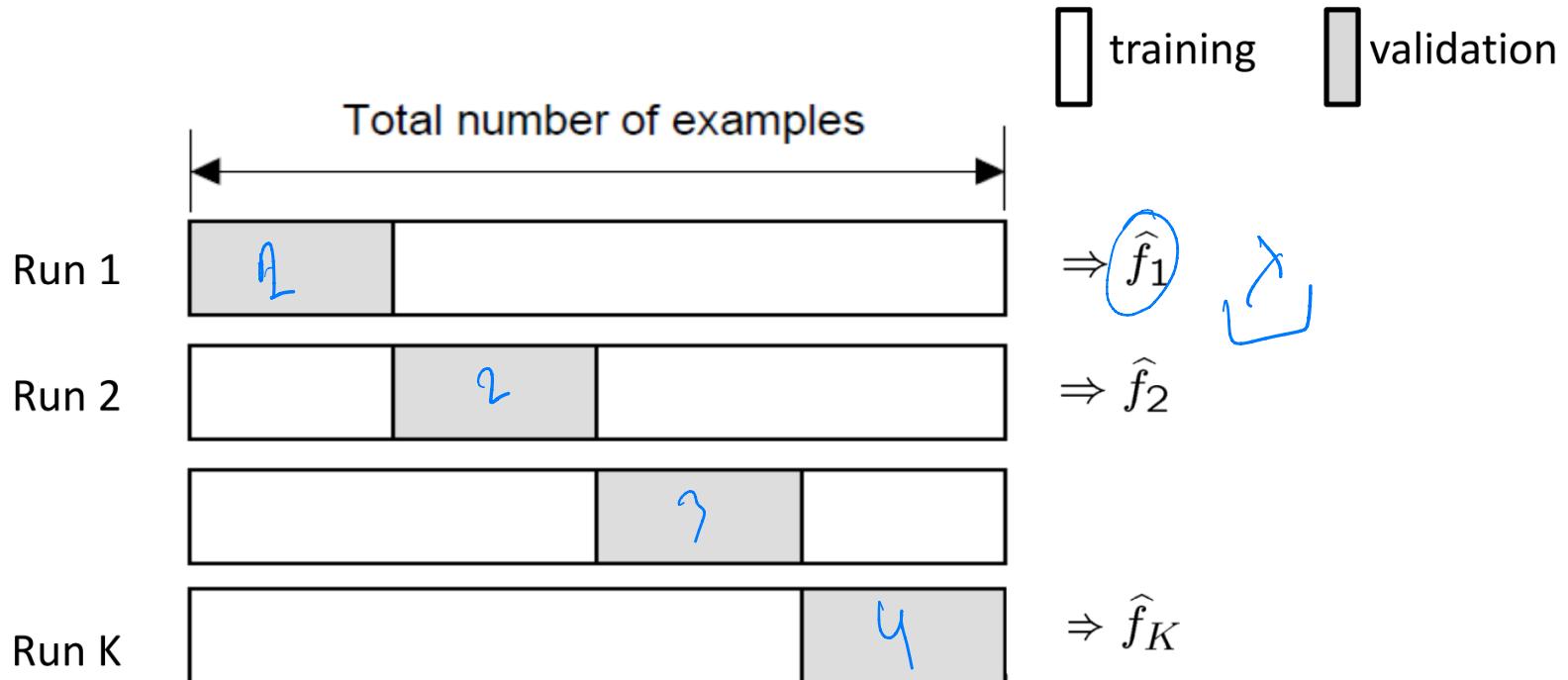


K-fold cross-validation

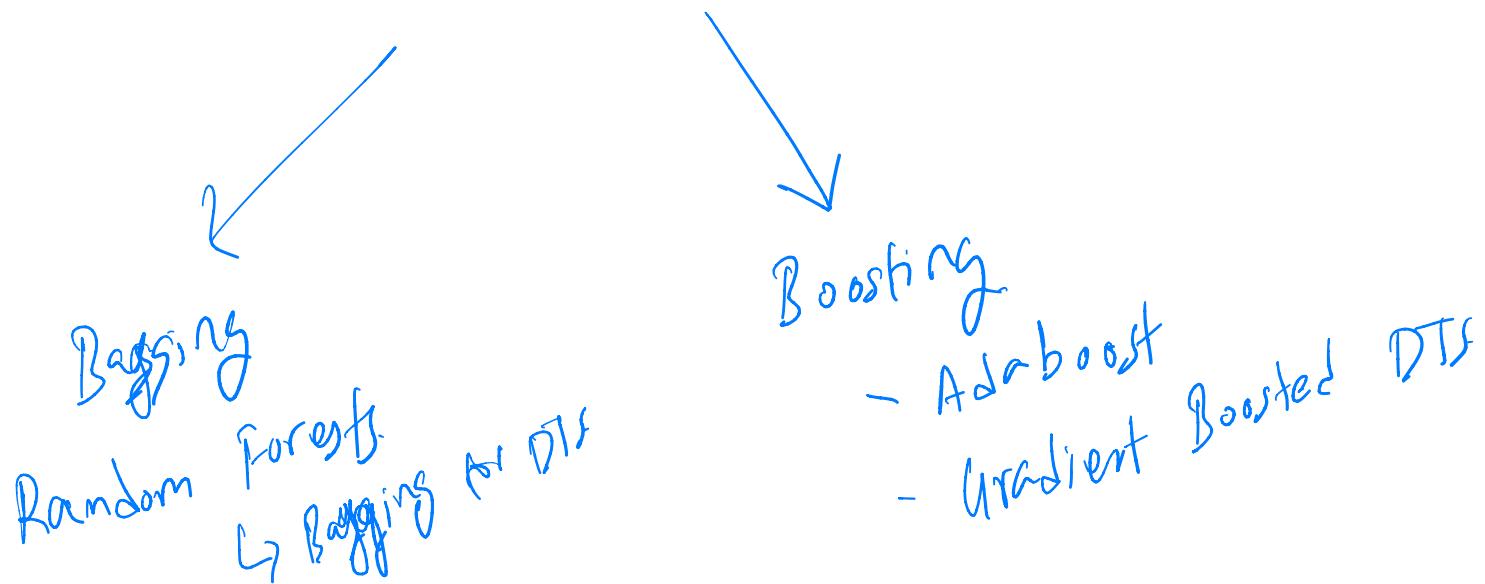
Create K-fold partition of the dataset.

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



Reduce Variance Without Increasing Bias



Assume "low Bias" Model, e.g. DTs

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

$$Var\left(\frac{1}{N} \sum_i Z_i\right) = \frac{1}{N} Var(Z_i)$$

} large number

- Idea: average models to reduce model variance
- The problem
 - Only one training set
 - Where do multiple models come from?

$$\begin{aligned} \tilde{z}_i &= \frac{z_1 + z_2 + \dots + z_N}{N} \\ \text{Var}(z_i) &= \sigma^2, \forall i \\ \text{Var}(\tilde{z}_n) &= \frac{\sigma^2}{N} \end{aligned}$$

Bagging: Bootstrap Aggregation



- Take repeated bootstrap samples from training set D (Breiman, 1994)
$$D = \{(x_1, y_1), \dots, (x_N, y_N)\}$$
- **Bootstrap sampling:** Given set \underline{D} containing N training examples, create $\underline{D'}$ by drawing N examples at random **with replacement** from D
Sample - with replacement
Many repetitions
- **Bagging:**
 - Create k bootstrap samples D_1, \dots, D_k
 - Train distinct classifier on each D_i
 - Classify new instance by majority vote / average

$$D = \{1, 2, \dots, 10\}$$

$$D_1 = \{5, 7, 1, 3, 7, 8, 10, 2, 6, 8\}$$

2 distinct items, 3 rep]

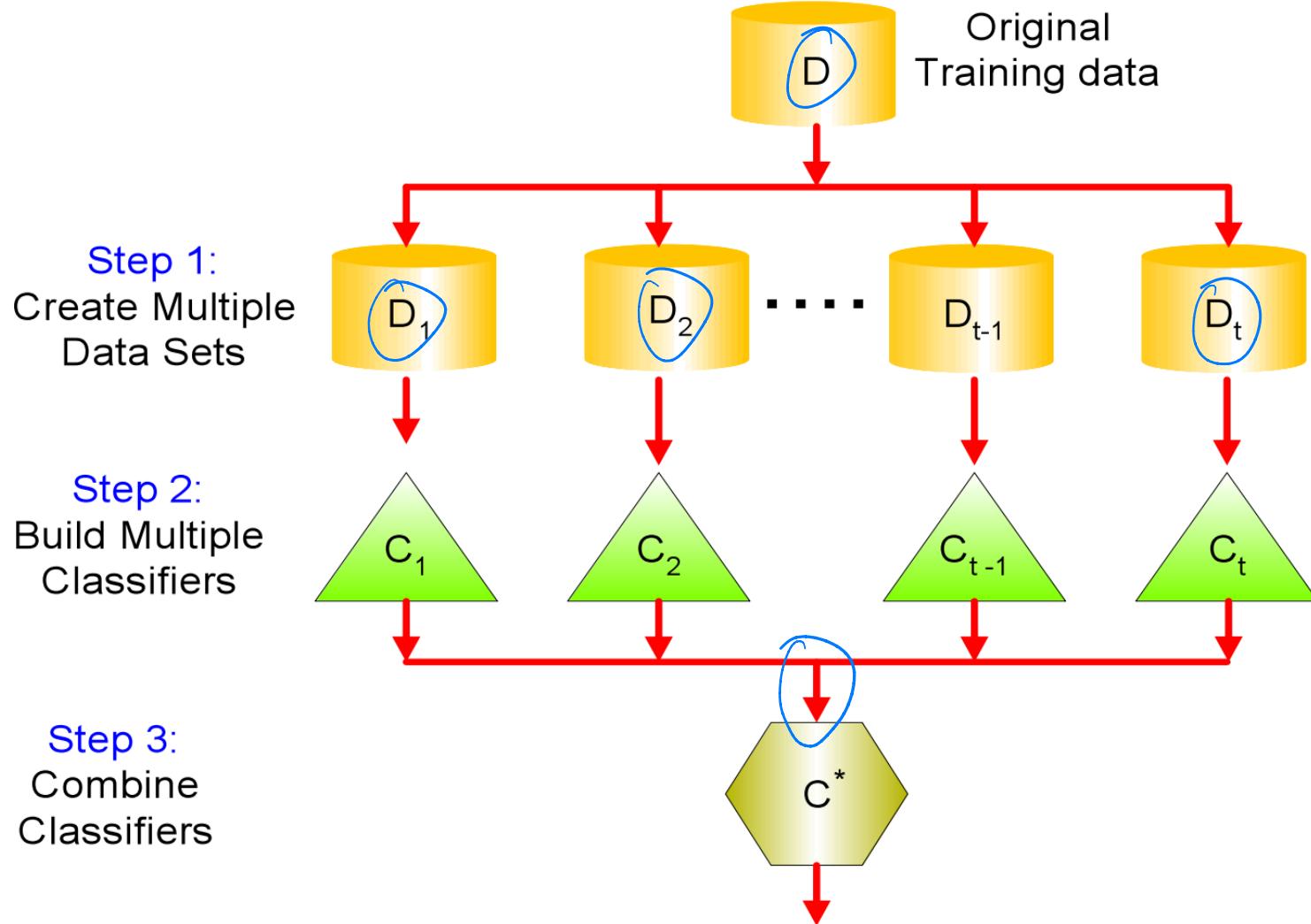
$$D_2 = \{ \dots \}$$

6 distinct items, 3 rep

$$1 - \left(1 - \frac{1}{N}\right)^N \approx 0.63$$

Distinct numb: $\sqrt{0.63^N}$

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

8
7

- 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 e^{-1}

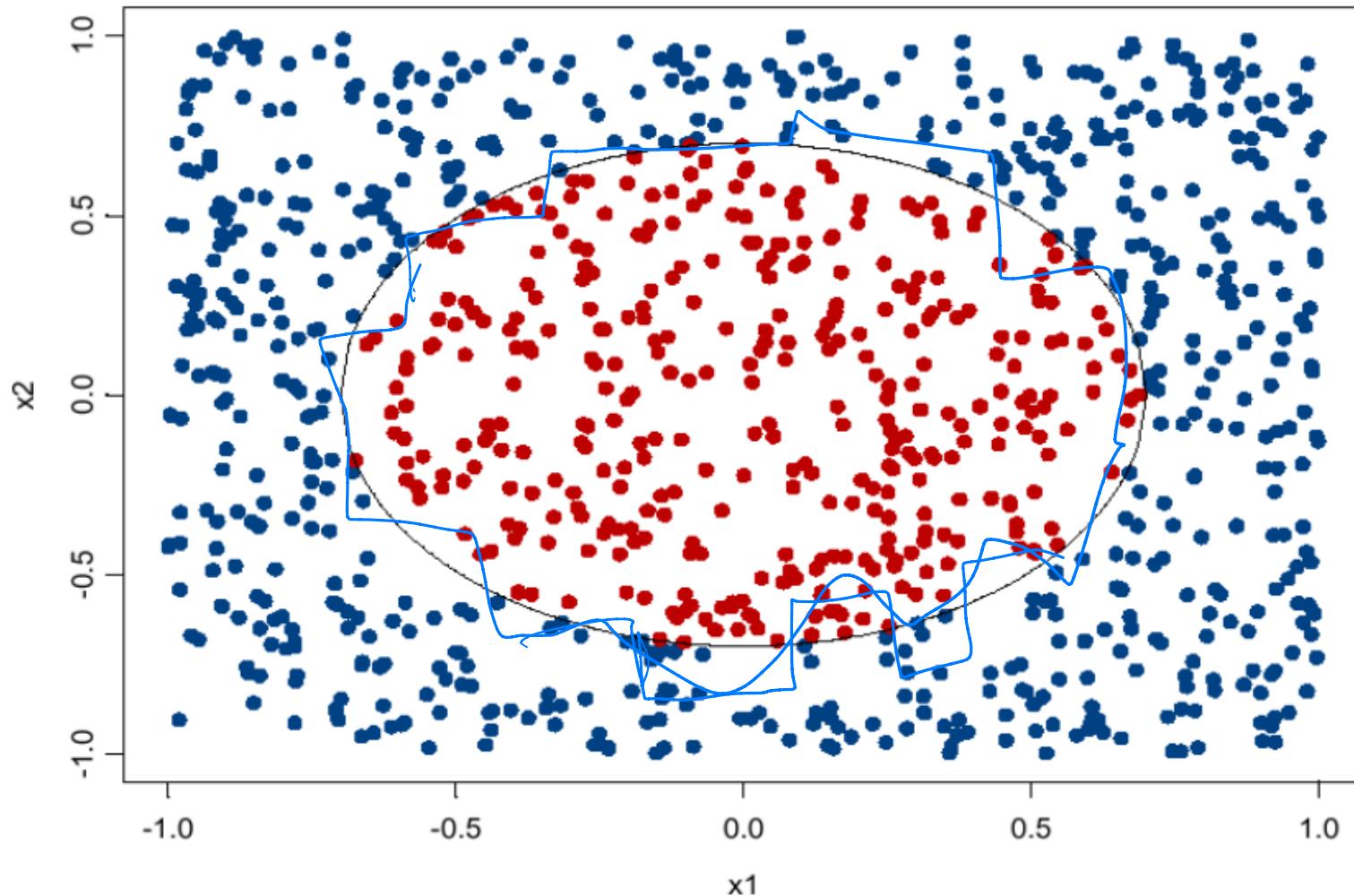
$$N \cdot \left(1 - \left(1 - \frac{1}{N}\right)^N\right) \approx N \cdot (1 - \exp(-1)) = .632 \cdot N$$

Bagging

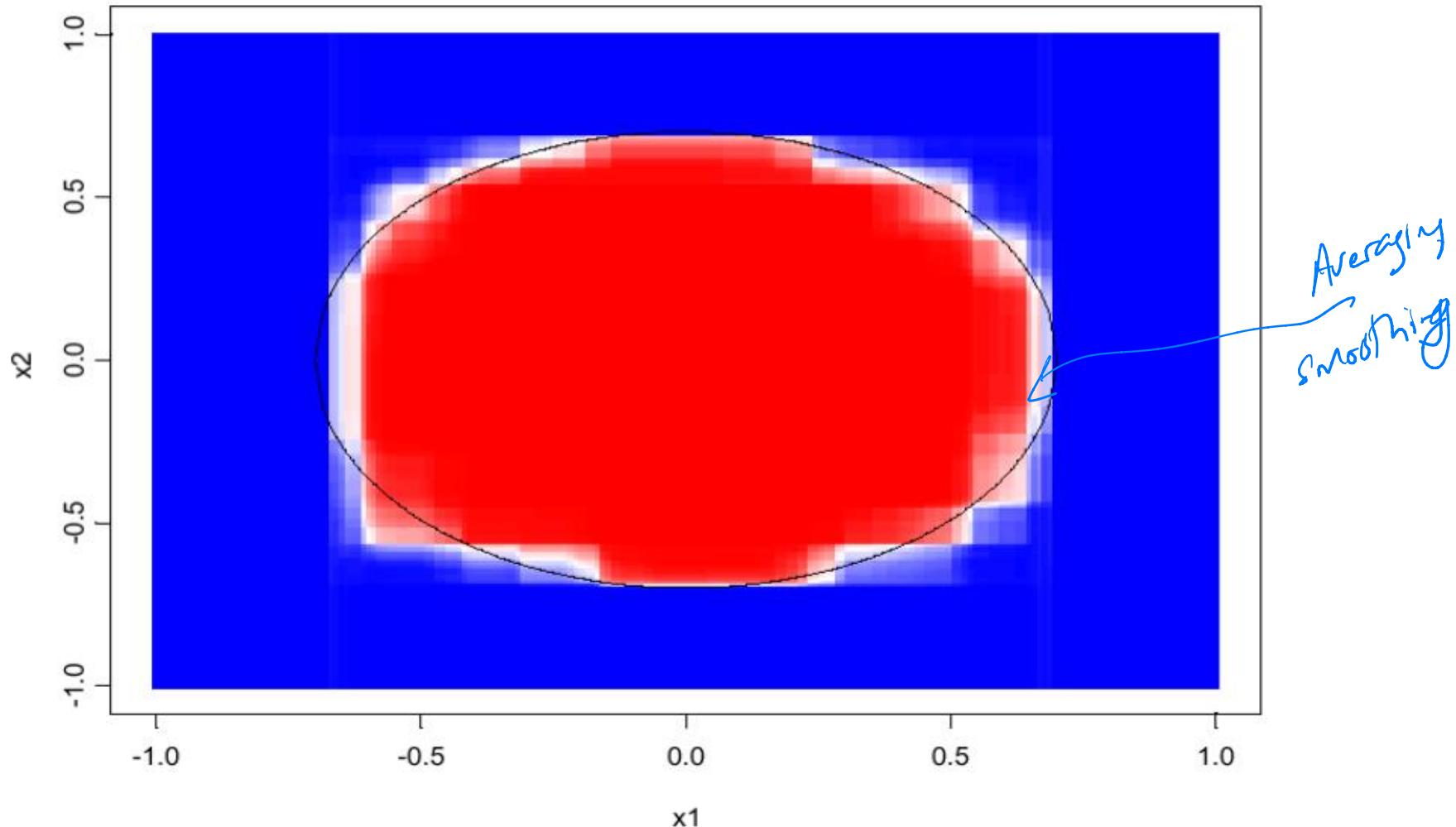
Data	1	2	3	4	5	6	7	8	9	10
BS 1	7	1	9	10	7	8	8	4	7	2
BS 2	8	1	3	1	1	9	7	4	10	1
BS 3	5	4	8	8	2	5	5	7	8	8

- 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 $\sim 632\text{GB}$ (this can present computational challenges, e.g., you shouldn't replicate the data)

Decision Tree Bagging



Decision Tree Bagging (100 Bagged Trees)



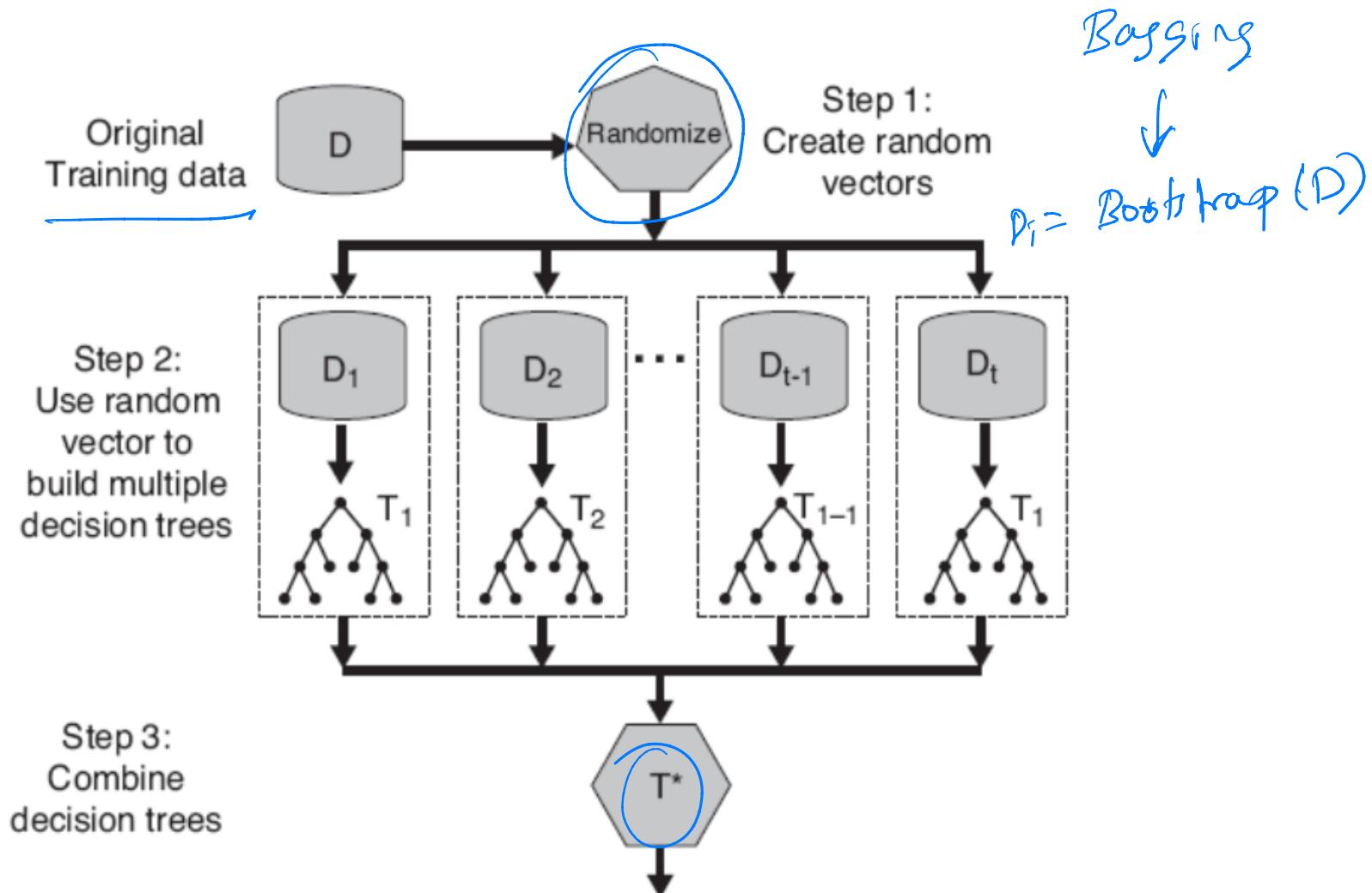
Bagging Results

VC

Data Set	\bar{e}_S	\bar{e}_B	Decrease
	Without Bagging	With Bagging	
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
 - Random vector method: best split at each node is chosen from a random sample of m attributes instead of all attributes

Random Forest Algorithm

Random Forest (m)

$O(B \cdot T)$

- 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 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 ϵ_D
- A learner is **unstable** if a small change to the training set causes a large change in the output hypothesis
 - 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 (*Low bias, High variance*)
 - k -nearest neighbor is stable

Bagging / Bootstrap

$$h' = \text{Agg}(h_1, \dots, h_B)$$

$$h'(x) = \overbrace{\sum_{i=1}^B h_i(x)}^B$$

$$e_{h'} = \overbrace{\sum_{i=1}^B e_{h_i}}^B = \overbrace{\sum_{i=1}^B e_{D_i}^h}^B$$

true error = bias + var + noise

$$e_h = b_h + \text{Var}(h) + \sigma^2 \quad \xrightarrow{\text{BLVN } h}$$

$$e_{h'} = b_{h'} + \overbrace{\text{Var}(h)}^B + \sigma^2$$

$$e_{h'} = b_{h'} + \overbrace{\text{Var}(h)}^B + \sigma^2$$

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
 - 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