Performance Evaluation and an Introduction to Overfitting

Ben Galili

Ariel Shamir

Zohar Yakhini

IDC

Outline

- Performance evaluation – a first intro
- Generalization error
- Overfitting:
  - An abstract definition
  - Polynomial regression
  - Decision trees

Approximation vs. Generalization

- Approximation/training error: measures how good your hypothesis fits the training data (ERM principle)
- Generalization error: measures how good your hypothesis is expected to fit new data

- In machine learning we are interested in generalization
- We want to be able to assess generalization performance from the sample data

Error of a classifier (ERM)

Consider a classification task over a space of features X.
Assume that some probability measure, π, is defined over X.
We are trying to predict a concept $c \in 2^X$.
Represent c as a function $c: X \to \{0,1\}$
For a hypothesis $h: X \to \{0,1\}$, we want to define the error that it would commit, as a prediction rule (a model) for c.
The error that we are really interested in is:

$$Error_\pi(h) = \pi(x: h(x) \neq c(x))$$

However, we have no access to π nor to the entire space X. We only have access to the training data and we use it to define the training error:

$$Error_{train}(h) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{I}\left(h(x^{(i)}) \neq c(x^{(i)})\right)$$

Overfitting

A hypothesis $h \in H$ is said to overfit the training data $\left\{\left(x^{(i)}, c\left(x^{(i)}\right)\right)\right\}_{i=1}^{m}$
if there exists an alternative hypothesis, $\bar{h} \in H$, so that

$$Error_{train}(h) < Error_{train}(\bar{h})$$

But

$$Error_{\pi}(h) > Error_{\pi}(\bar{h})$$

Balance > 100K	Owns a house	Monthly salary	Clean history	Owns a car	Credit trust	h
+	+	-	+	+	+	+
+	-	-	-	+	-	
-	-	+	-	+	+	+
-	-	+	+	+	+	+
+	+	+	+	+	+	+
-	+	+	-	+	+	+
-	+	-	+	-	-	
-	-	+	+	-	+	+
+	+	-	-	-	-	

$X = \{0,1\}^5$, with a uniform distribution π
c $=$ all configurations that did not default credit
H $=$ all Boolean functions on X (how many?)
h $=$ + as indicated, in the training data, and – for all other configurations

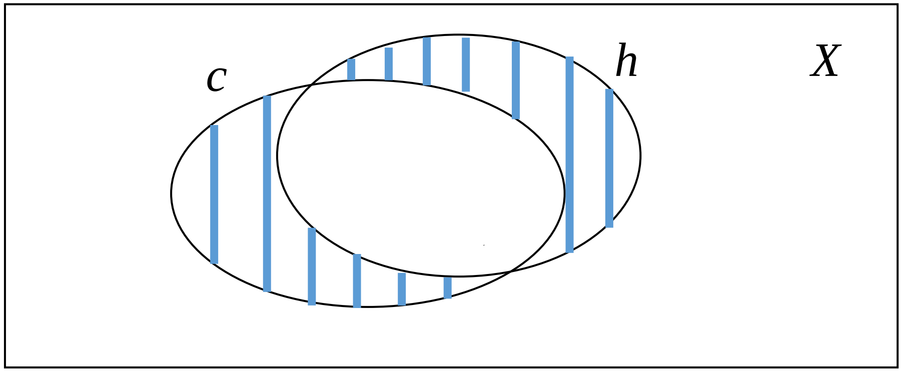
Can you find concepts c for which h would be an overfitting model?

True Error (Classifiers)

- The true error of a hypothesis h with respect to a target concept c is the probability that h will misclassify an instance drawn at random according to the distribution π of the data:

$$\mathit{error}_{\pi}(h) = \Pr_{x \sim \pi}[c(x) \neq h(x)]$$

- The error is highly dependent on the distribution π!



Statistical Estimation

- We can use a test set to estimate the <u>true error</u> of a candidate hypothesis/model.
- If the test set is all the rest of X then we will know the true error!
- Of course this is unrealistic – we must depend on sampling. We therefore define the sample error, for a set S:

  error$_S$(h) = the fraction of misclassified samples in S
- The larger the test set, the better the estimate.
  We want to understand the quality of the estimate.
- Our true error estimation task is equivalent to the following question in statistics:

  Estimate, from a sample, the proportion of a population possessing some property.

- In our case the property of $x \in X$ is that our hypothesis h misclassifies x.

The distribution of the sample error

- For a specific instance x, the probability of misclassification is by definition $error_\pi(h) := p$
- We assume that our (test) sample consists of n random instances drawn independently from X.
- Let $\boldsymbol{R}$ be a random variable defined as the number of misclassifications produced by h when applied to the sample dataset.
- Then:

$$\text{Prob}(R = k) = \frac{n!}{k!(n-k)!} \cdot p^k (1-p)^{n-k}$$

This does NOT, however, provide an estimate of p

Statistical Estimation Procedure

$x^{(i)}$ such that $h(x^{(i)}) \neq c(x^{(i)})$

- Using a "test set" of size $|S| = n$. Assume the number of errors is r.
- It can be shown that r/n is an (MLE) estimator for the generalization error.
- Depending on the size of our test set we can produce statistical guarantees like:

With 95% confidence we estimate that the true error is smaller than $\frac{r}{n} + \varepsilon(n)$

Confidence interval for proportions

$$\boxed{Prob\left(p \in \left[\hat{p} \pm \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}\right]\right) \approx 1-\alpha}$$

Error of a regression model

Consider a regression task over a space of features X.
Assume that some probability measure, π, is defined over X.
We are trying to predict a function $f: X \rightarrow \mathbb{R}$
For a hypothesis $h: X \rightarrow \mathbb{R}$, we want to define the error that it would commit, as a prediction rule for f.
The error that we are really interested in is:

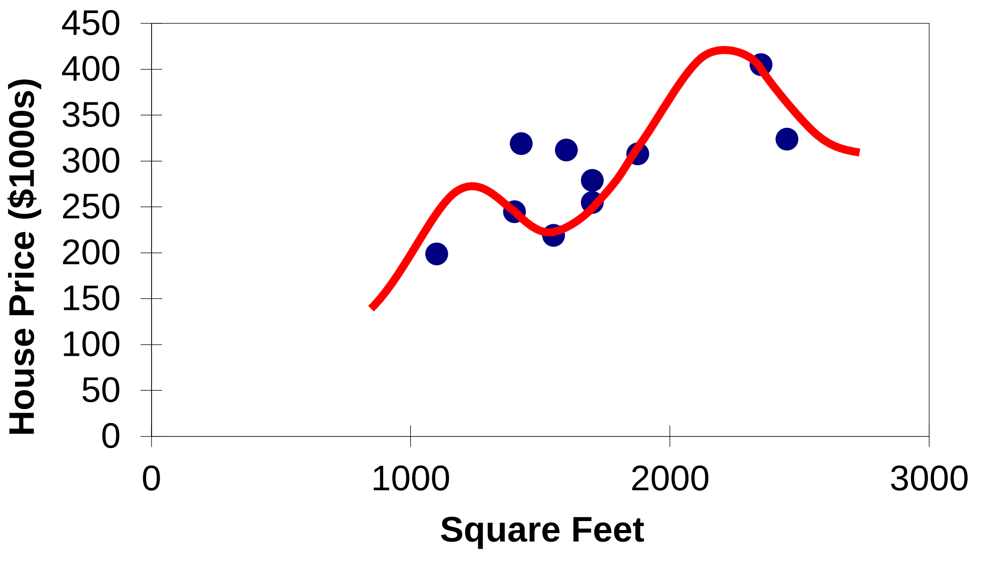
$$Error_{\pi}(h) = E_{\pi}(\, d(h(x), c(x)))$$

However, we have no access to π nor to the entire space X. We only have access to the training data and we use it to define the training error:

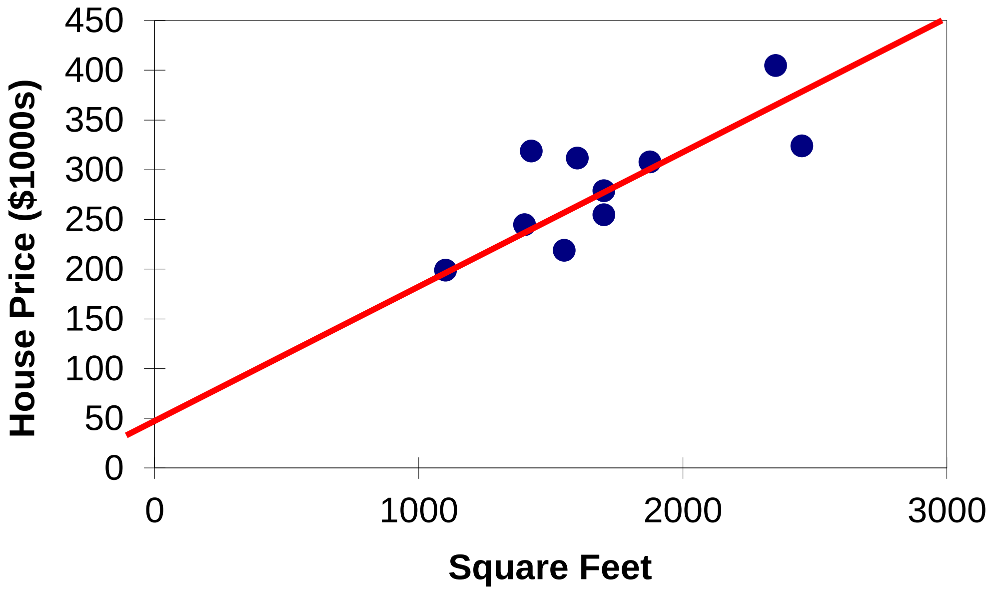
$$Error_{train}(h) = \frac{1}{m}\sum_{i=1}^{m} d\left(h\left(x^{(i)}\right), c(x^{(i)})\right)$$

Polynomial Regression

- Assume that training is $\left\{\left(x^{(i)}, y^{(i)}\right)\right\}$.
- To obtain a polynomial model by running a linear regression learning process augment the vectors $x^{(i)}$ by adding features like $\left(x_s^{(i)}\right)^2$ or $\left(x_s^{(i)}\right)^7$ or $x_s^{(i)} \cdot x_t^{(i)}$ etc ...
- We can thus use polynomials of any degree to try to fit a given function to the training data.
- But do we really want to use higher degrees to better fit the training?



450
400
350
300
250
200
150
100
50
0
House Price ($1000s)
0
1000
2000
3000
Square Feet



450
400
350
300
250
200
150
100
50
0
House Price ($1000s)
0
1000
2000
3000
Square Feet

Overfitting in polynomial regression, from Bishop

$$y = f(x) = \sin(\pi x)$$

Data generated by adding some small Gaussian noise

The figures show polynomial fits of different degrees (M)

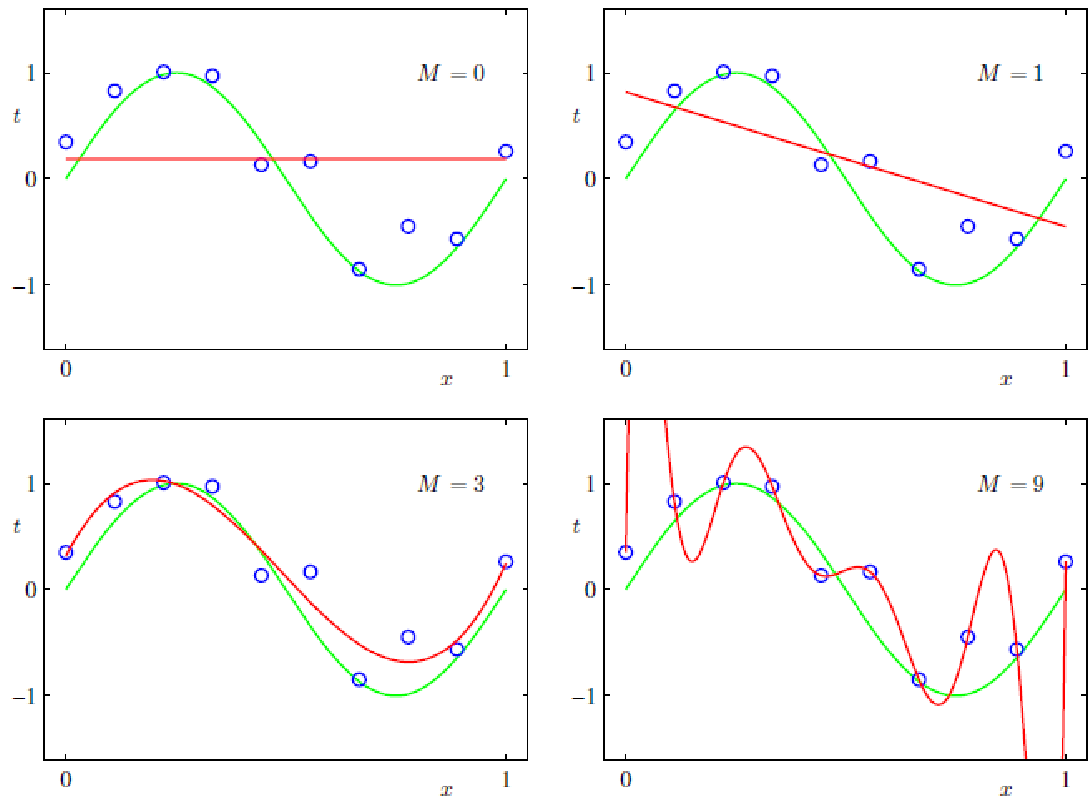


Image source: CM Bishop, Pattern Recognition and Machine Learning, Springer 2006

Overfitting in polynomial regression

$$y = f(x) = \sin(\pi x)$$

Data generated by adding some small Gaussian noise

The figures show polynomial fits of different degrees (M)

For an independent dataset the error obtained for M=9 is larger than that obtained for M=3

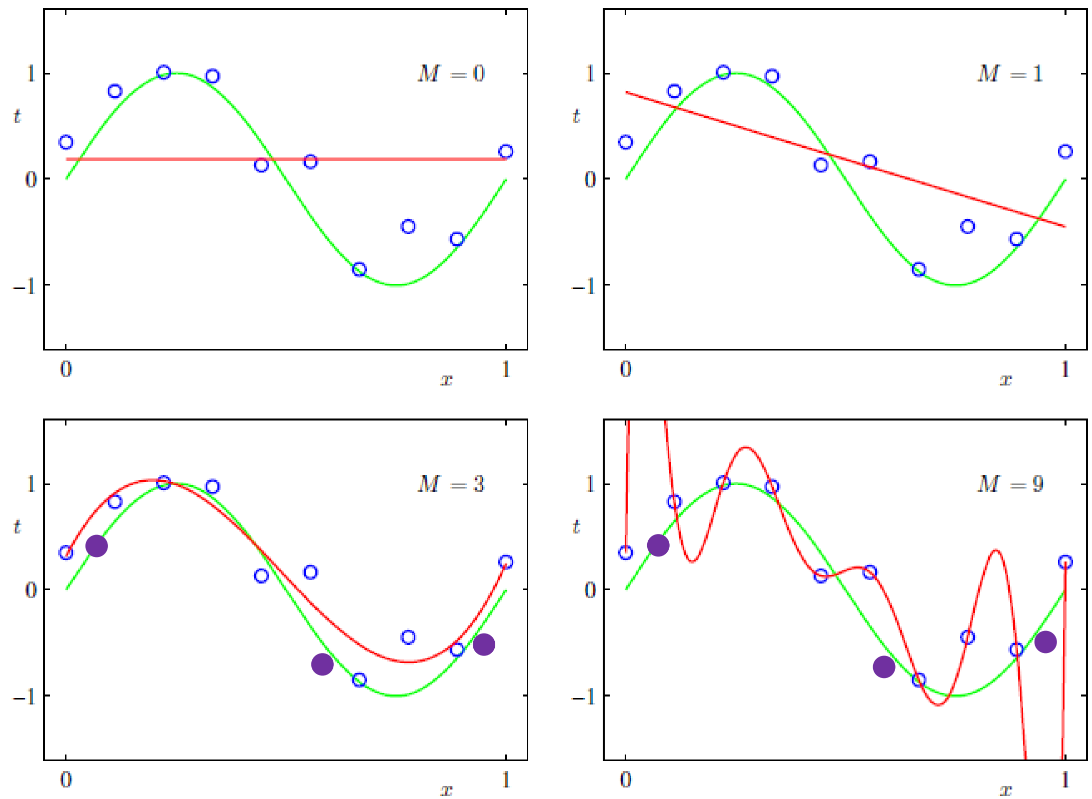
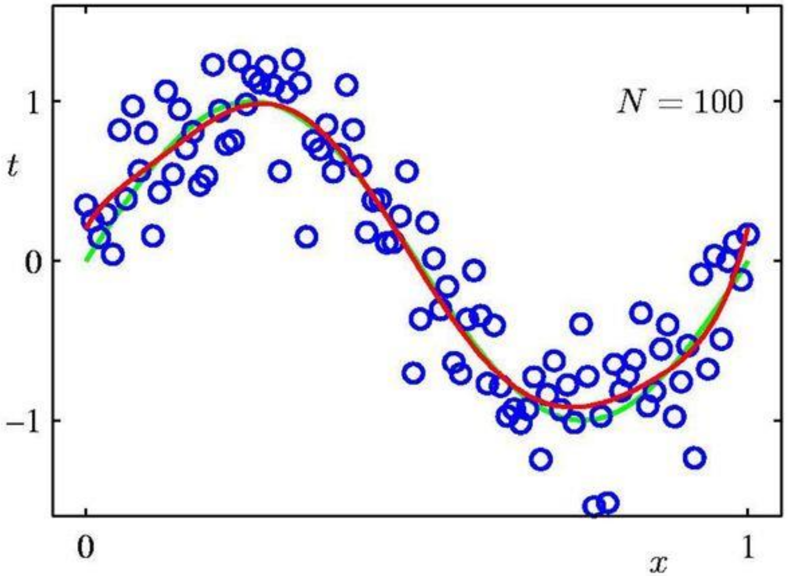


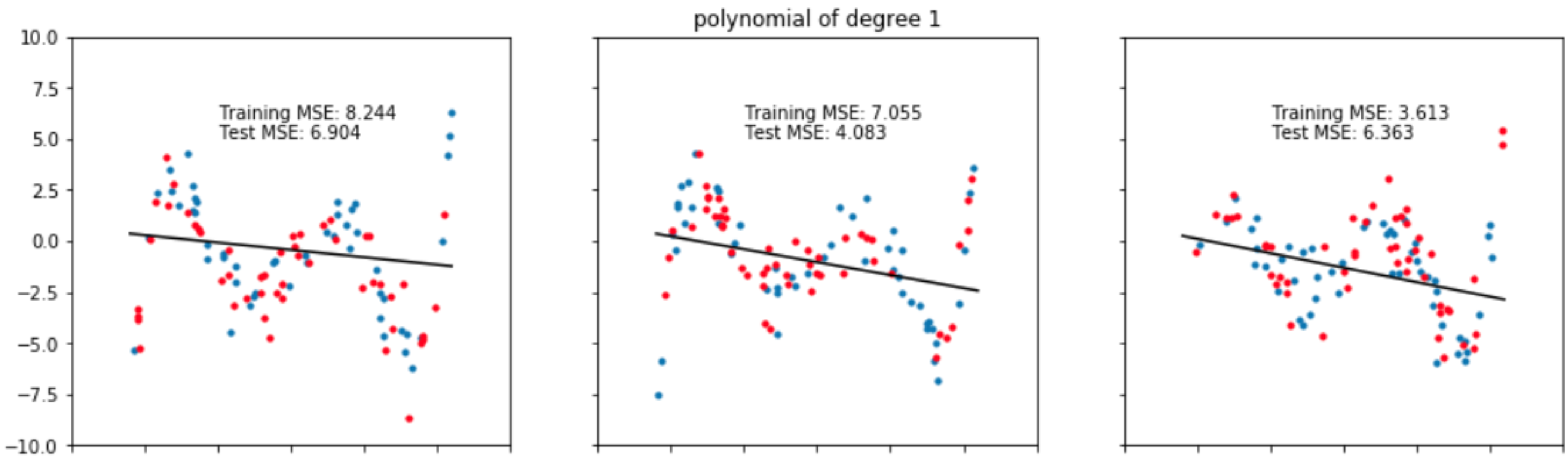
Image source: CM Bishop, Pattern Recognition and Machine Learning, Springer 2006

Data Set Size: $N = 100$

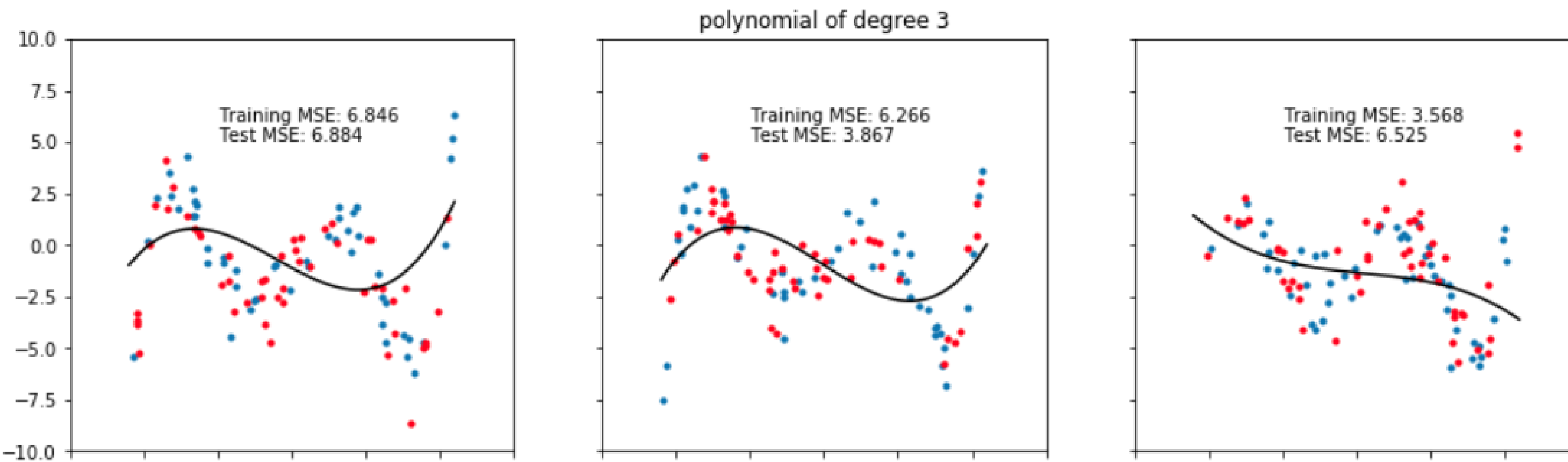
9^{th} Order Polynomial

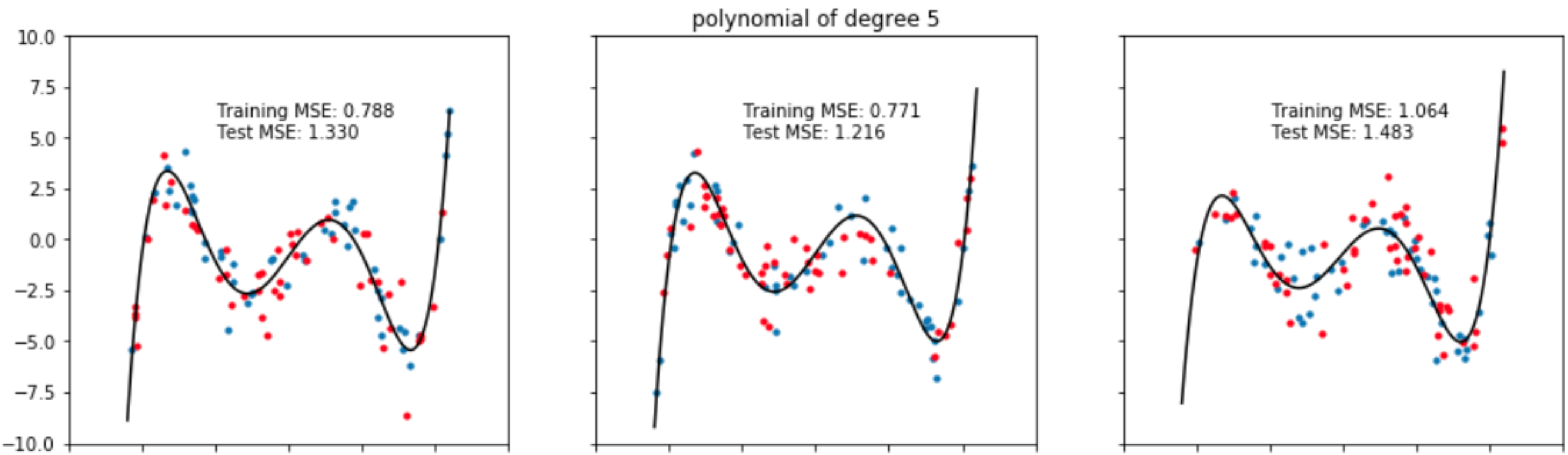


Slide credit: Chris Bishop



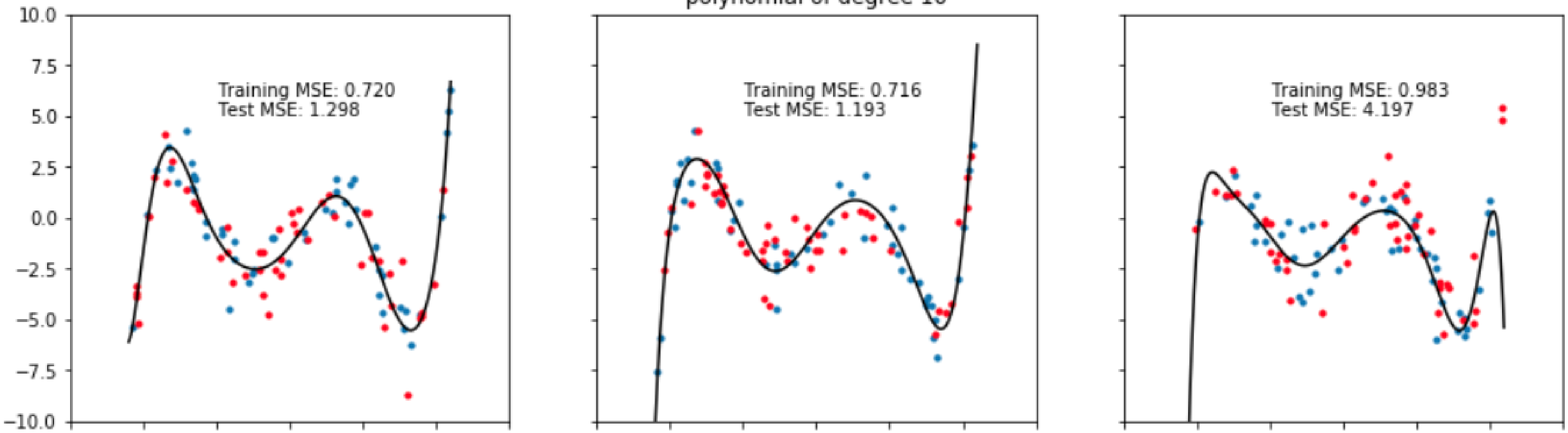
Galili, Yakhini, IDC


polynomial of degree 3
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
Training MSE: 6.846
Test MSE: 6.884
Training MSE: 6.266
Test MSE: 3.867
Training MSE: 3.568
Test MSE: 6.525

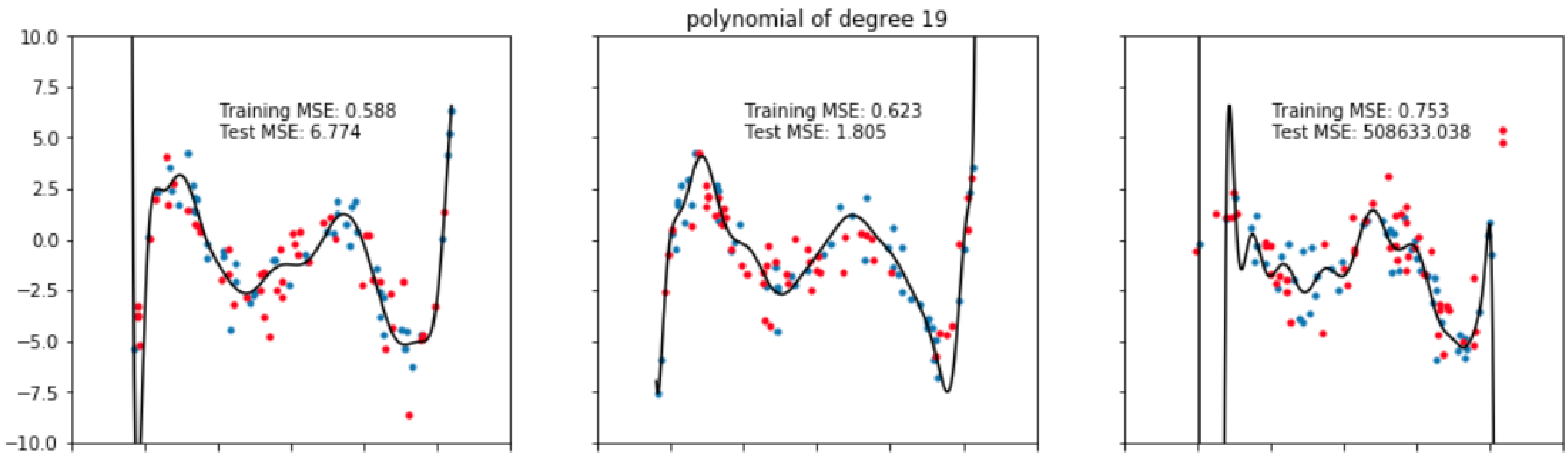


Galili, Yakhini, IDC

polynomial of degree 10



Galili, Yakhini, IDC

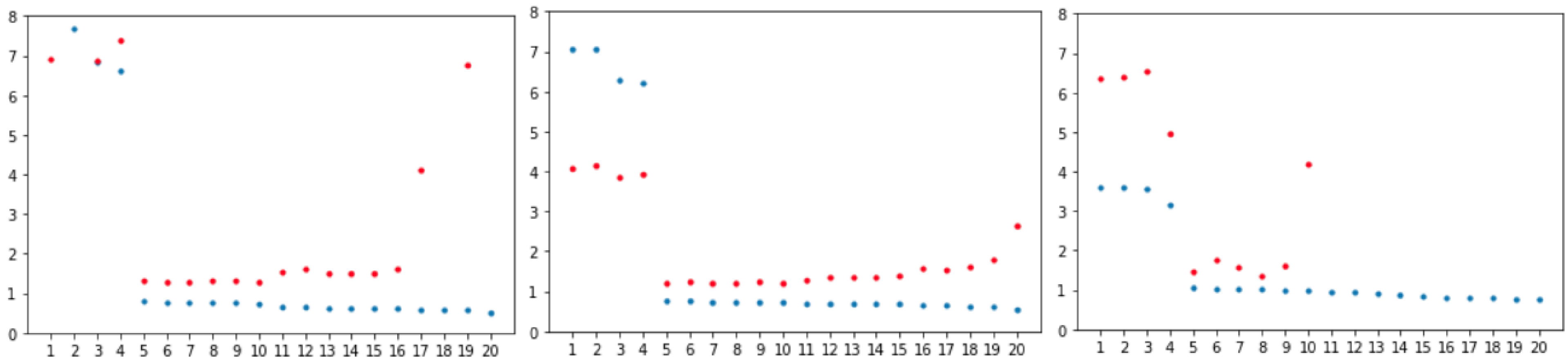


Galili, Yakhini, IDC

MSE as a function of Δ

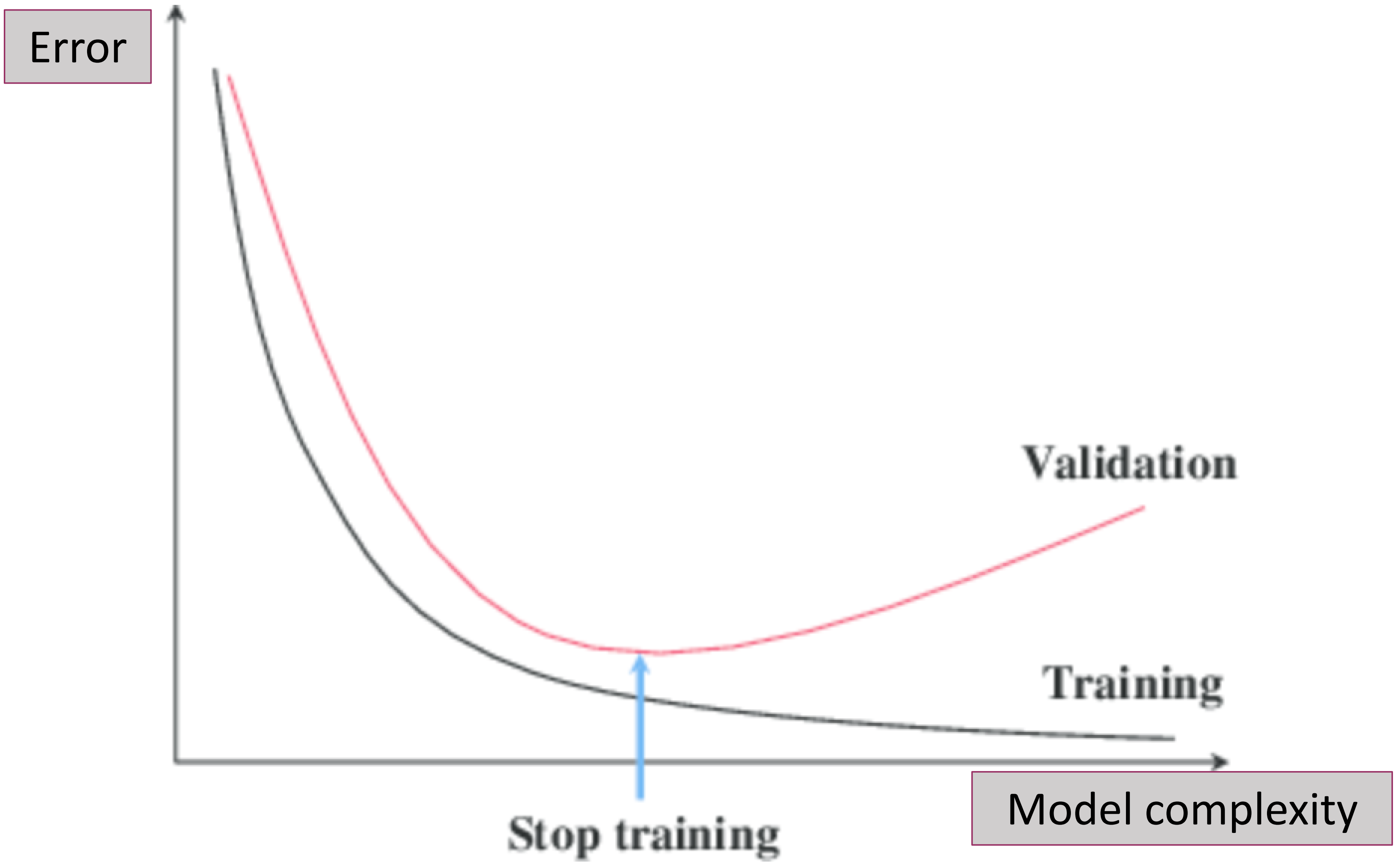
MSE

MSE values for
Δ>10 are out of range



Δ

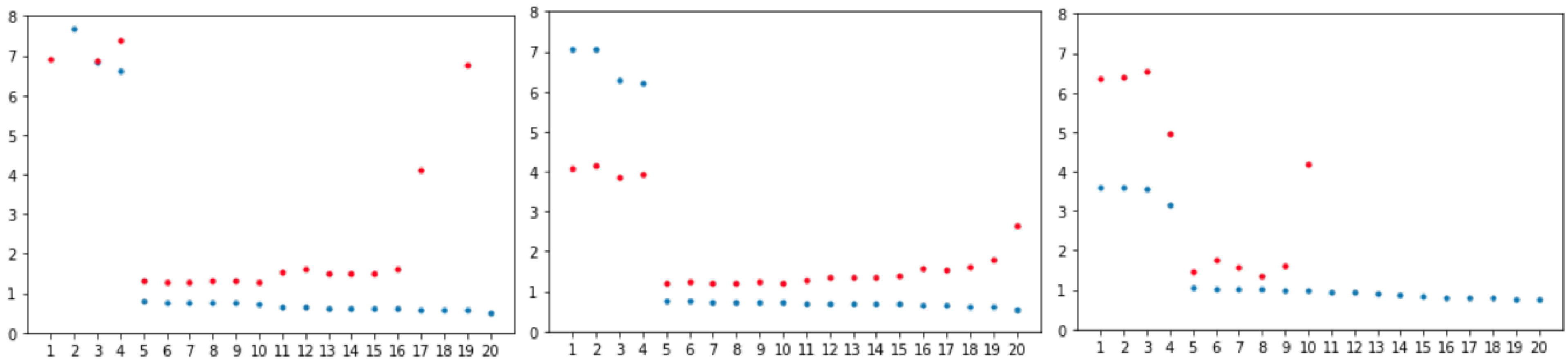
Galili, Yakhini, IDC


Error
Validation
Training
Stop training
Model complexity

MSE as a function of Δ

MSE

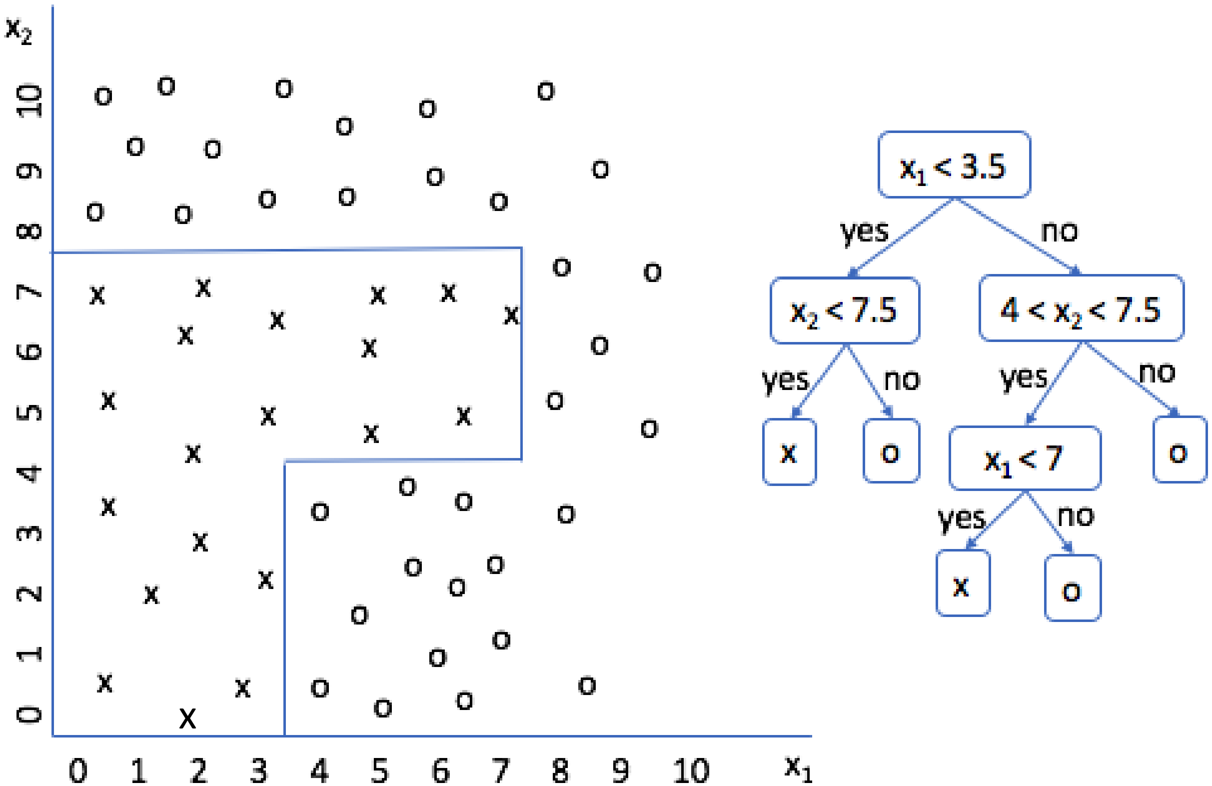
MSE values for
Δ>10 are out of range



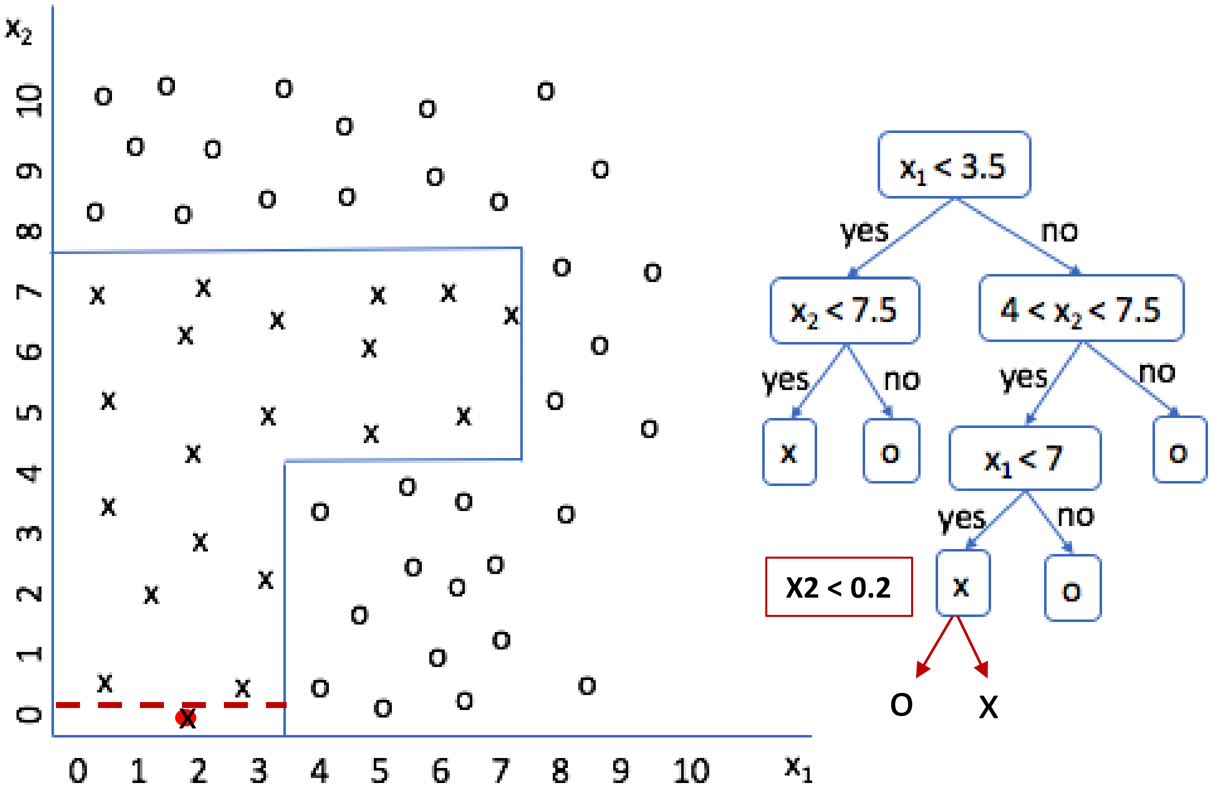
Δ

Galili, Yakhini, IDC

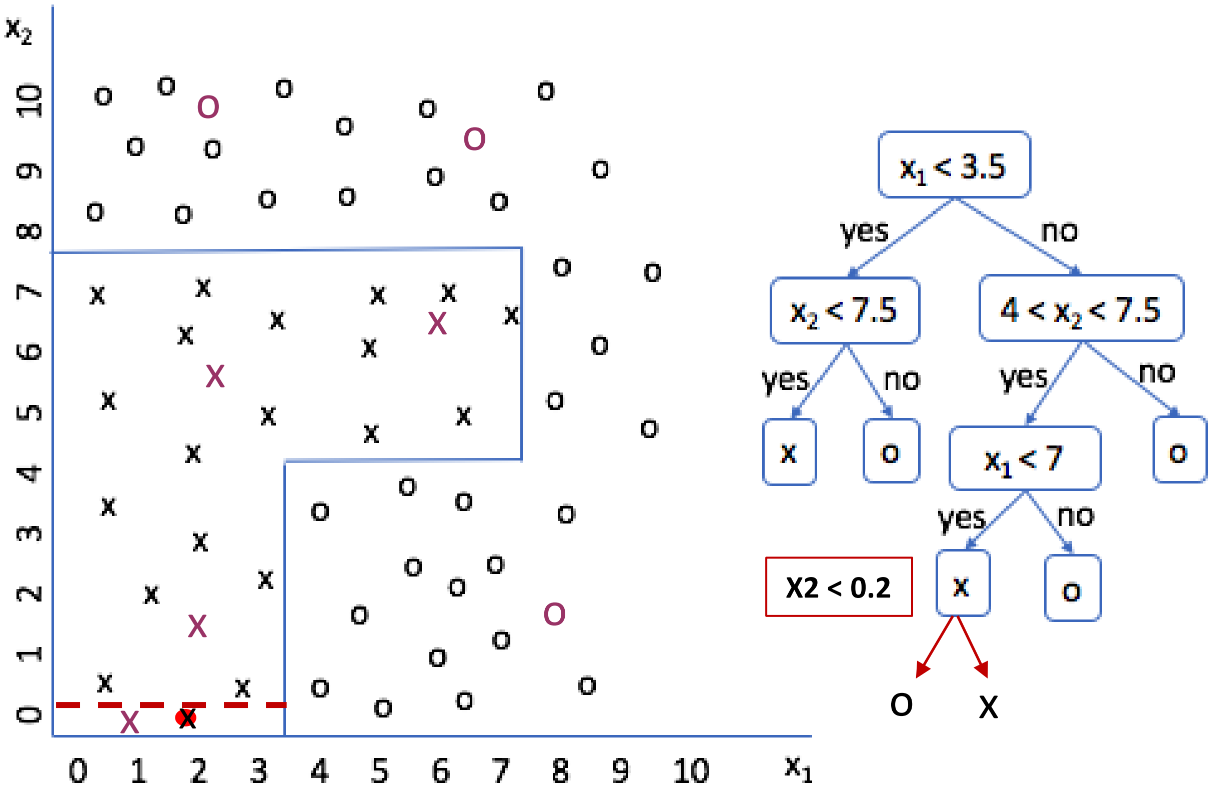
Decision trees - revisited



Decision trees - revisited



Decision trees - revisited



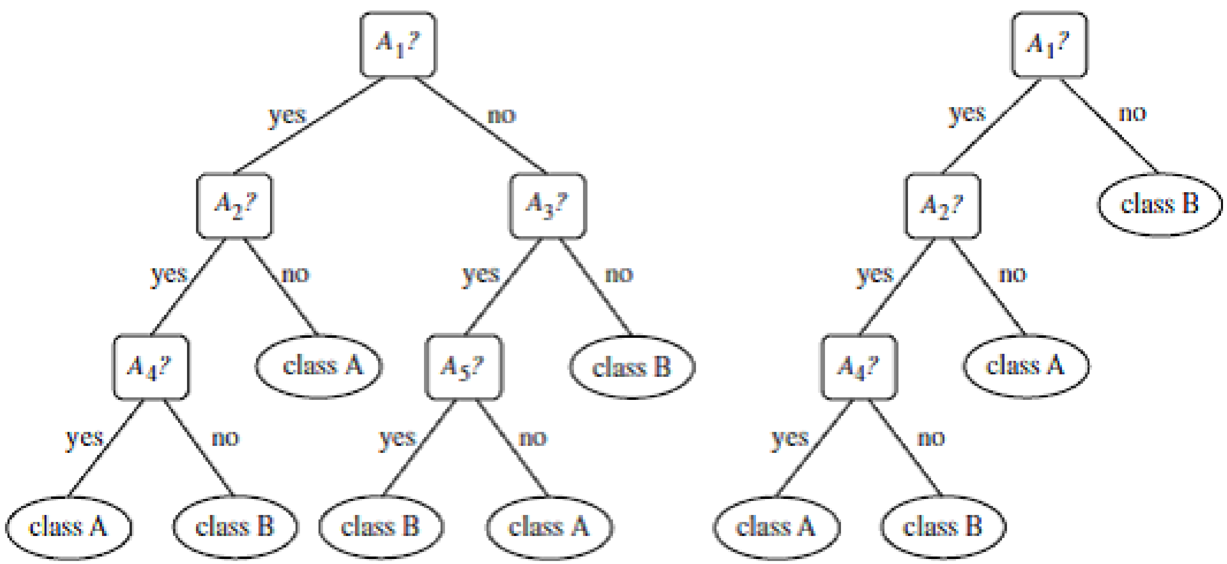
Post-Pruning using a Validation Set

- Split data into training and validation sets
- Construct tree on the training data
  - Doesnt have to be all the way to completion.
  - May use stop conditions
  - May use pre-pruning. For example - based on chi-square.
- Eliminate (or collapse) splits of nodes when upon doing so the performance on the validation test improves

Explained in the next slide

Post-Pruning using a Validation Set

1. Split data into training and validation sets
2. Construct tree on the training data
3. For all nodes in the tree compare validation accuracy before and after pruning at that node.
4. Prune the node for which you observe greatest improvement in validation accuracy. (IF any?)
5. Repeat Step 3 until no nodes with non negative improvement (can be loosely defined) are observed



2-class Confusion Matrix

True class	Predicted class: positive	Predicted class: negative
positive (#P)	#TP	#P - #TP (FN)
negative (#N)	#FP	#N - #FP (TN)

Summary

- Overfitting in regression and in decision trees
- Statistical estimation of the true error
- Using a validation set to determine the adequate complexity (inc pruning a tree)
- The confusion matrix
(we will come back to it …)