AI 330: Machine Learning

Fall 2023

Dr. Wessam EL-Behaidy

Associate Professor, Computer Science Department,

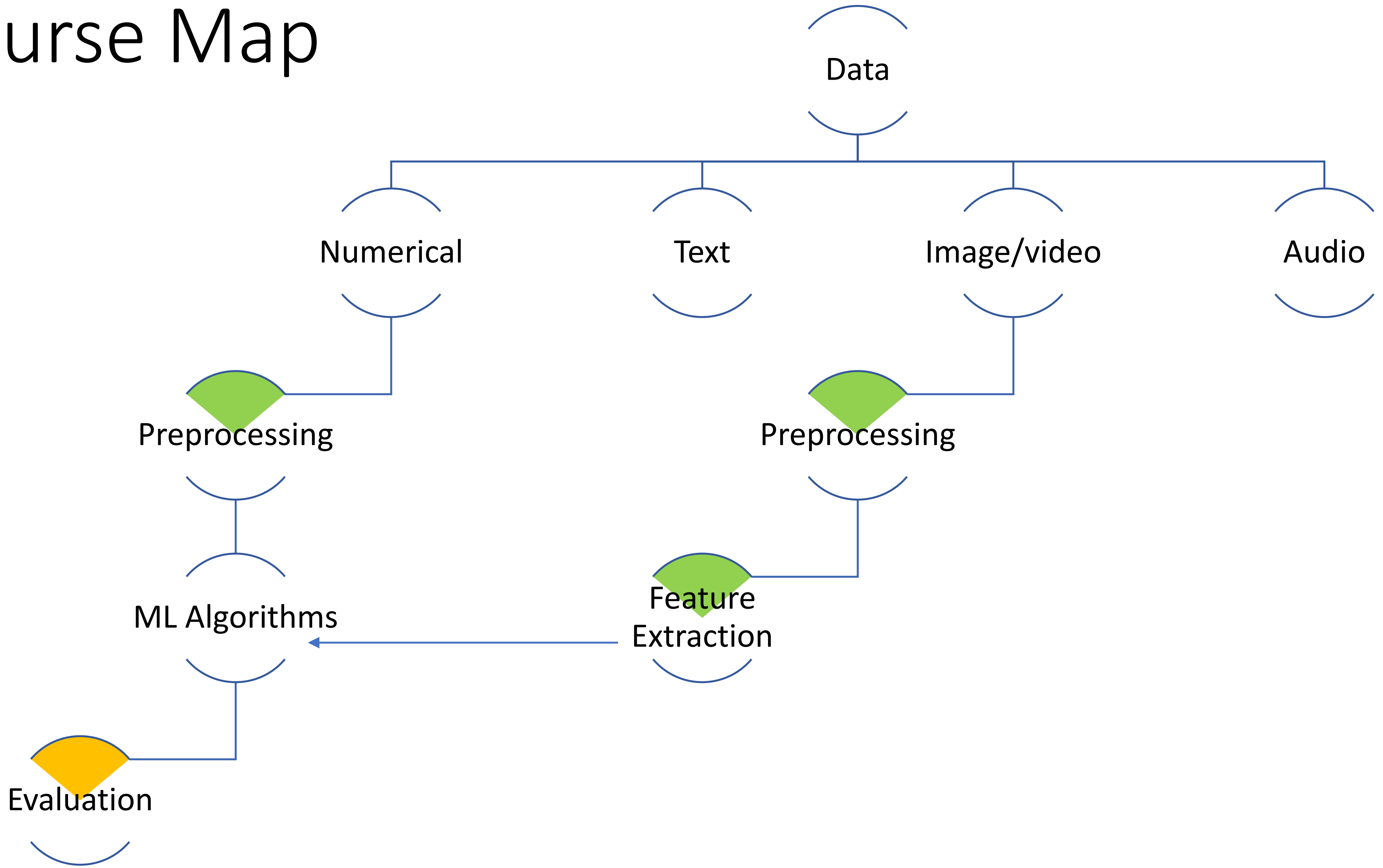
Faculty of Computers and Artificial Intelligence,

Helwan University.

Dr. Ensaf Hussein

Associate Professor, Computer Science Department,
Faculty of Computers and Artificial Intelligence,
Helwan University.

Course Map



Data
Numerical
Text
Image/video
Audio
Preprocessing
Preprocessing
ML Algorithms
Feature Extraction
Evaluation

Lecture 8

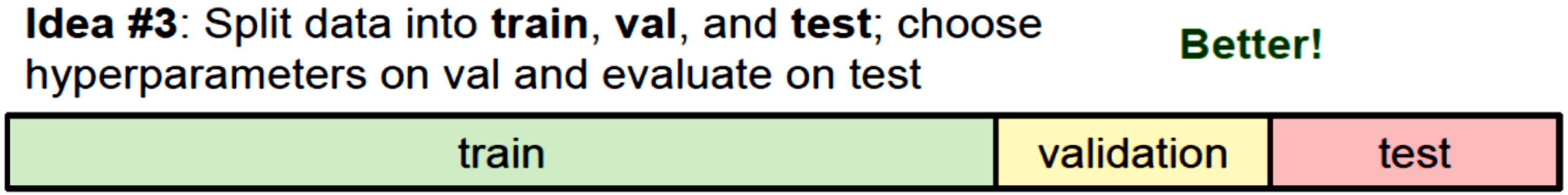
Model Evaluation and Diagnosis

Slides of:
https://www.coursera.org/learn/machine-learning at Stanford University (Prof. Andrew Ng)

Hypothesis Evaluation

To Evaluate hypothesis

- One way to break down our dataset into the three sets is:
- **Training set**: 60%
- **Cross validation set**: 20% (This validation set is essentially used as a **fake test set** to **tune the hyper-parameters)**
- **Test set**: 20%



60% 20% 20%

Why validation set is important ?

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$

Choose $\theta_0 + \ldots \theta_5 x^5$

How well does the model generalize? Report test set error $J_{test}(\theta^{(5)})$

Problem: $J_{test}(\theta^{(5)})$ is likely to be an optimistic estimate of generalization error. i.e. our extra parameter (d = degree of polynomial) is fit to test set.

Train/ validation/ test error for Linear Regression

Training error:

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Cross Validation error:

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

Test error:

$$J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_\theta(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

Model Selection

- We can now calculate **three separate error** values for the three different sets using the following method:

1. Optimize the parameters in Θ using the training set for each polynomial degree.
2. Find the polynomial degree d with the <u>least</u> error using the cross validation set.
3. Estimate the generalization error using the test set with $J_{test}(\Theta^{(d)})$ (***d*** = theta from polynomial with lower error)

- This way, the degree of the polynomial ***d*** has not been trained using the test set.

Train/ validation/ test error for Logistic Regression

- Learn parameter θ from training data.
- Tune hyperparameters using validation data.
- Compute test set error:

$$J_{test}(\theta) = -\frac{1}{m_{test}} \sum_{i=1}^{m_{test}} y_{test}^{(i)} \log h_\theta(x_{test}^{(i)}) + (1 - y_{test}^{(i)}) \log h_\theta(x_{test}^{(i)})$$

- Misclassification error (0/1 misclassification error):

$$err\ (h_\theta(x), y) = \begin{cases} 1, & if\, h_\theta(x) \geq 0.5\ , y = 0, or\ , \quad if\ h_\theta(x) \leq 0.5\ , y = 1\ (error) \\ 0, & otherwise \end{cases}$$

➔ Example: Error = 5% , so accuracy= 95%

Cross-Validation

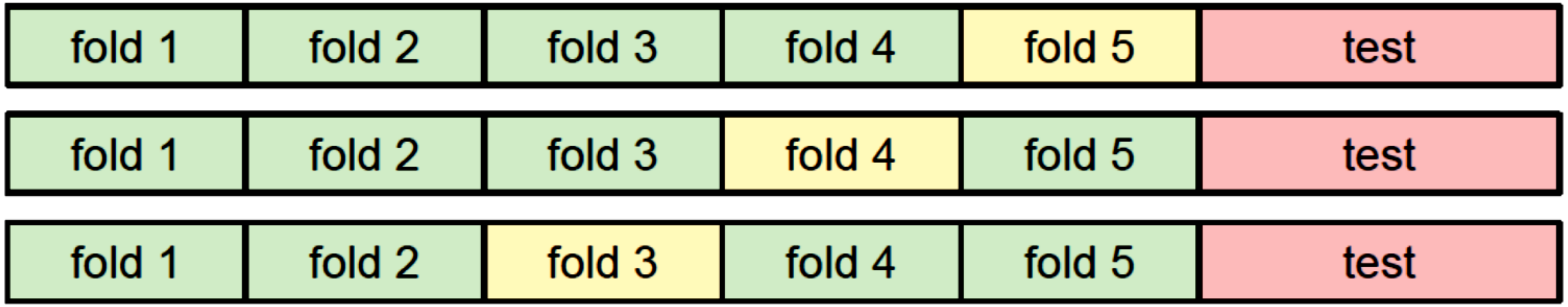
- For small datasets, sometimes we use a more sophisticated technique for hyperparameter tuning called **cross-validation**.
  - ◭ **Instead of arbitrarily picking** the first data points to be the validation set and rest training set,
  - ◭ **Get a better and less noisy estimate** of how well hyperparameters work by iterating over different validation sets and averaging the performance across these.

Cross-Validation

- For example: 5-fold cross-validation

1. Split the training data into 5 equal folds (parts),
2. Use 4 of them for training, and 1 for validation.
3. Iterate over which fold is the validation fold, and evaluate the performance,
4. Finally average the performance across the different folds.

Idea #4: **Cross-Validation**: Split data into **folds**,
try each fold as validation and average the results



Useful for small datasets, but not used too frequently in deep learning

Performance Metrics

Accuracy in a Classification Model

- **Accuracy** is measured as the percentage of predicted results that match the expected results.

- Ex: if there are 1000 results and 850 predicted results match the expected results , then the accuracy is 85%

Problem with accuracy metric (measure): Skewed classes

- Skewed classes basically refer to a dataset, wherein the number of training example belonging to one class out-numbers heavily the number of training examples belonging to the other.

- Consider a binary classification (cancer is labelled 1 and not cancer labelled 0), where a cancerous patient is to be detected based on some features.

  - only 1 % of the data provided has cancer positive.

- If a system naively gives the prediction as all 0's, still the prediction accuracy will be 99%.

Problem with accuracy metric (measure): Irrelevant features

- The data may be fitted against a feature that is not relevant.

Ex:

- In image classification, if all images of one class have small/similar background, the model may match based on the background, not the object in the image.

Commonly used Metrics

- Accuracy is only one metric.

- **Other metrics commonly used are:**

- Precision
- Recall (Sensitivity)
- Specificity
- F1-score
- ROC AUC

Confusion Matrix

- The confusion matrix is a performance measurement technique that visualizes the accuracy of a classifier by comparing the actual and predicted classes.
- It is called a confusion matrix because it shows how confused the model is between the classes.
- The class of interest is commonly called **the positive class**, and the rest **negative class**

Binary Confusion Matrix

Confusion matrix		Predicted class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

- True Positive (TP) : The outcome is correctly classified as positive.
- False Negative (FN) : The outcome is incorrectly classified as negative when it is positive.
- False positive (FP): The outcome is incorrectly classified as positive when it is negative.
- True Negative (TN): The outcome is correctly classified as negative.

Example of Confusion Matrix

- **If class "Daisy" is the positive class (y=1), so:**
- TP=9 FN=1
- FP=2 TN=8

Predicted Label

True Label

	Daisy	Tulip
Daisy	9	1
Tulip	2	8

Accuracy

- Accuracy is calculated as the number of all correct predictions divided by the total number of the dataset
- The best ACC is 1.0, whereas the worst is 0.0

Accuracy: (TP + TN) / (P + N)

Predicted Label

True Label

	1	0
1	9	1
0	2	8

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

- Accuracy = (9+8) / (9+2+1+8) = 0.85

Precision

- ➢ **Precision is calculated as the number of correct positive predictions divided by the total number of positive predictions** (predicted as positives)
- ➢ **The best precision is 1.0, whereas the worst is 0.0**

Precision: TP / (TP + FP)

Predicted Label

True Label

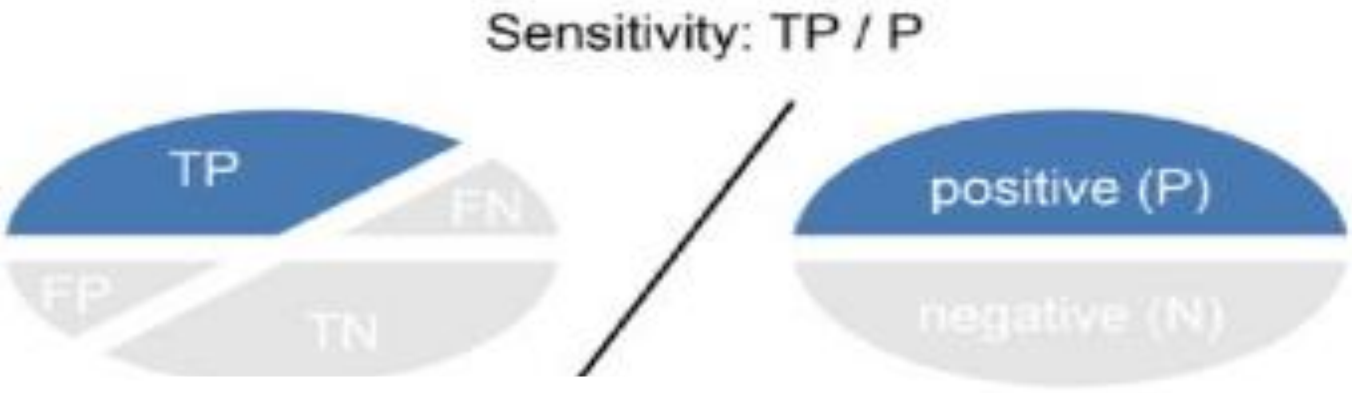
	1	0
1	9	1
0	2	8

$$\text{Precision} = \frac{tp}{tp + fp}$$

- Precision = 9 / (9+2) = 0.81

Recall

- Sensitivity = Recall = True Positive Rate
- Recall is calculated as the number of correct positive predictions divided by the total number of positives (true positives)
- The best recall is 1.0, whereas the worst is 0.0



Predicted Label

True Label

	1	0
1	9	1
0	2	8

$$\text{Recall} = \frac{tp}{tp + fn}$$

- Recall = 9 / (9+1) = 0.9

Specificity

- **Specificity= True Negative Rate**

Specificity is calculated as the number of correct negative predictions divided by the total number of negatives (true negatives)

Predicted Label

True Label		1	0
	1	9	1
	0	2	8

$$\text{Specificity} = \frac{tn}{tn+fp}$$

- Specificity= 8/(8+2)= 0.8

Precision/Recall for skewed data

$y = 1$ in presence of **rare class** (i.e. has cancer) that we want to detect

	Actual Class 1	Actual Class 0
Predicted Class 1	True positive	False Positive
Predicted Class 0	False Negative	True Negative

Precision

(Of all patients where we predicted $y = 1$, what fraction actually has cancer?)

$$\frac{True\ positive}{\#\ predicted\ positive} = \frac{True\ positive}{True\ positive + False\ positive}$$

Recall

(Of all patients that actually have cancer, what fraction did we correctly detect as having cancer?)

$$\frac{True\ positive}{\#\ actual\ positive} = \frac{True\ positive}{True\ positive + False\ negative}$$

Now, if we evaluate a scenario where the classifier predicts all 0's then TP=0, and the recall of the model will be 0, which then points out the inability of the system.

Trading off precision and recall

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

Logistic regression: $0 \leq h_\theta(x) \leq 1$

Predict 1 if $h_\theta(x) \geq 0.5$, 0.7, 0.9, 0.3

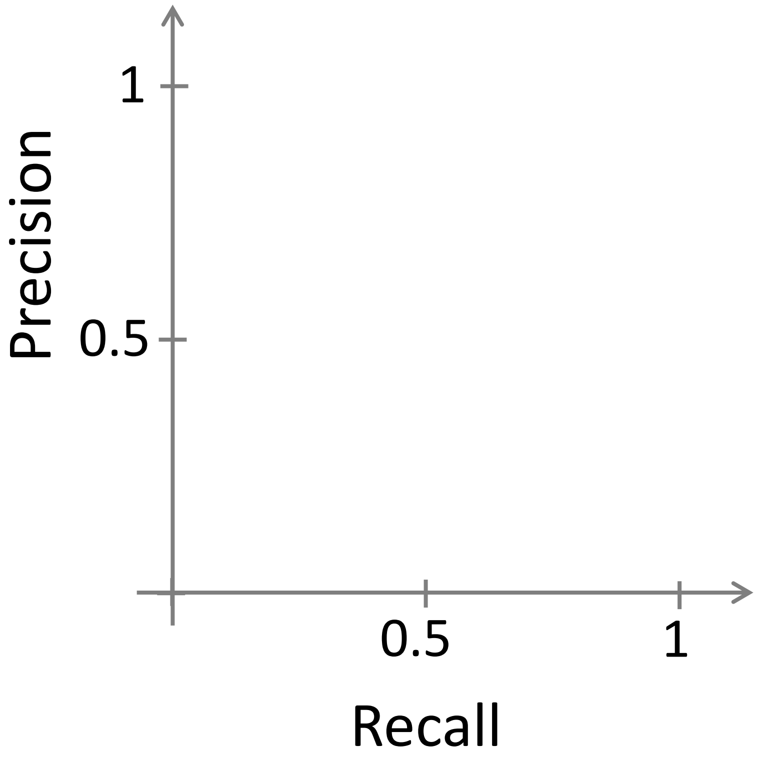
Predict 0 if $h_\theta(x) < 0.5$, 0.7, 0.9, 0.3

Suppose we want to predict $y = 1$ (cancer)only if very confident.

→ Higher precision, lower recall

Suppose we want to avoid missing too many cases of cancer (avoid false negatives).

→ Higher recall, lower precision

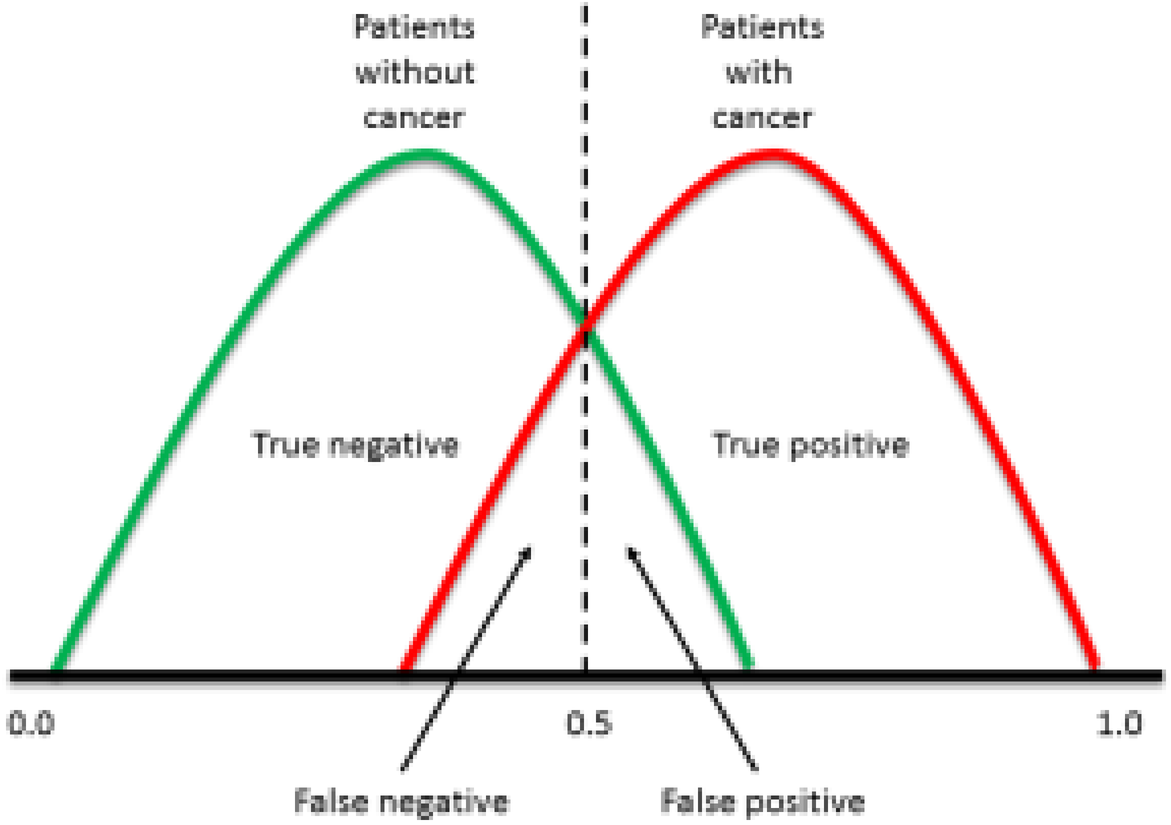


Threshold

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

- More generally: Predict 1 if $h_\theta(x) \geq$ threshold.



F_1 Score (F score)

How to compare precision/recall numbers?

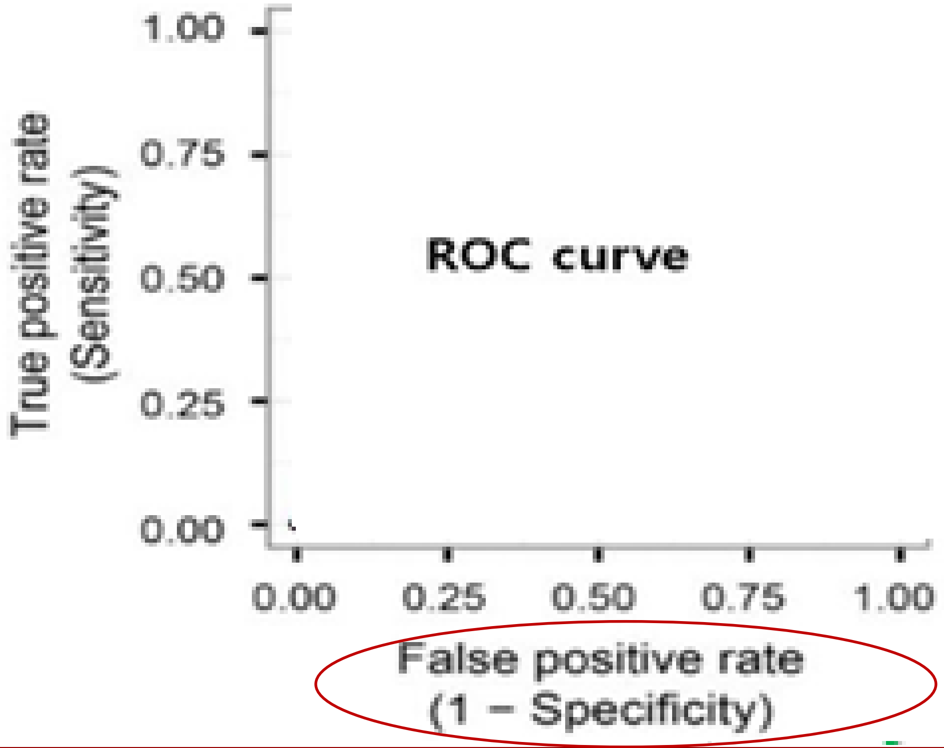
	Precision(P)	Recall (R)	Average	F_1 Score
Algorithm 1	0.5	0.4	0.45	0.444
Algorithm 2	0.7	0.1	0.4	0.175
Algorithm 3	0.02	1.0	0.51	0.0392

Average: $\frac{P+R}{2}$

F_1 Score: $2\frac{PR}{P+R}$

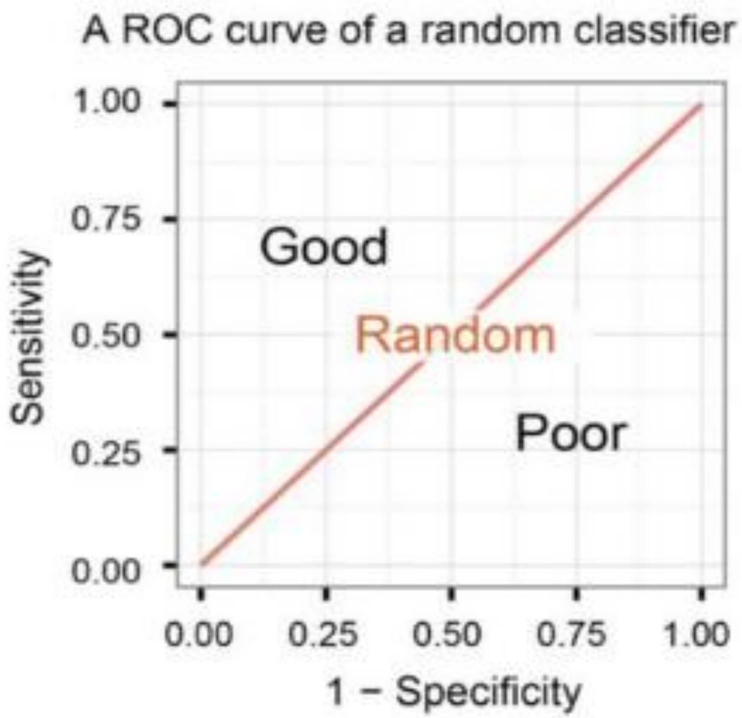
ROC Curve

- The Receiver Operating Characteristics(ROC) curve
- The ROC curve is a evaluation measure
  that is based on two basic evaluation measures
  - Specificity = True Negative Rate
  - Sensitivity = Recall = True Positive Rate



ROC Curve

- **A classifier with the random performance level always shows a straight line**
- **Two areas separated by this ROC curve**
  - ROC curves in the area with the top left corner indicate good performance levels
  - ROC curves in the other area with the bottom right corner indicate poor performance levels



How to Plot ROC Curve?

- **Dynamic cut-off thresholds**

Instance	Yes	No	Actual
1	0.008	0.992	N
2	0.011	0.989	N
3	0.021	0.979	Y
4	0.009	0.991	N
5	0.014	0.986	N
6	0.015	0.985	N
7	0.012	0.988	N
8	0.015	0.985	Y

Cut-off = 0.020

Instance	Predict	Type
1	N	TN
2	N	TN
3	Y	TP
4	N	TN
5	N	TN
6	N	TN
7	N	TN
8	N	FN

TP=1	FN=1
FP=0	TN=6

Cut-off = 0.015

Instance	Predict	Type
1	N	TN
2	N	TN
3	Y	TP
4	N	TN
5	N	TN
6	Y	FP
7	N	TN
8	Y	TP

TP=2	FN=0
FP=1	TN=5

Cut-off = 0.010

Instance	Predict	Type
1	N	TN
2	Y	FP
3	Y	TP
4	N	TN
5	Y	FP
6	Y	FP
7	Y	FP
8	Y	TP

TP=2	FN=0
FP=4	TN=2

How to Plot ROC Curve?

- True positive rate (TPR) = $TP/(TP+FN)$
  and False positive rate (FPR) = $FP/(FP+TN)$
- Use different cut-off thresholds (0.00, 0.01, 0.02,..., 1.00), calculate the TPR and FPR, and plot them into graph. That is receiver operating characteristic (ROC) curve.
- Example

TP=1	FN=1
FP=0	TN=6

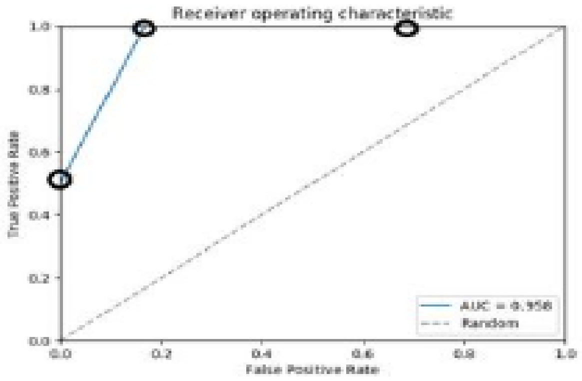
TPR = 0.5
FPR = 0

TP=2	FN=0
FP=1	TN=5

TPR = 1
FPR = 0.167

TP=2	FN=0
FP=4	TN=2

TPR = 1
FPR = 0.667

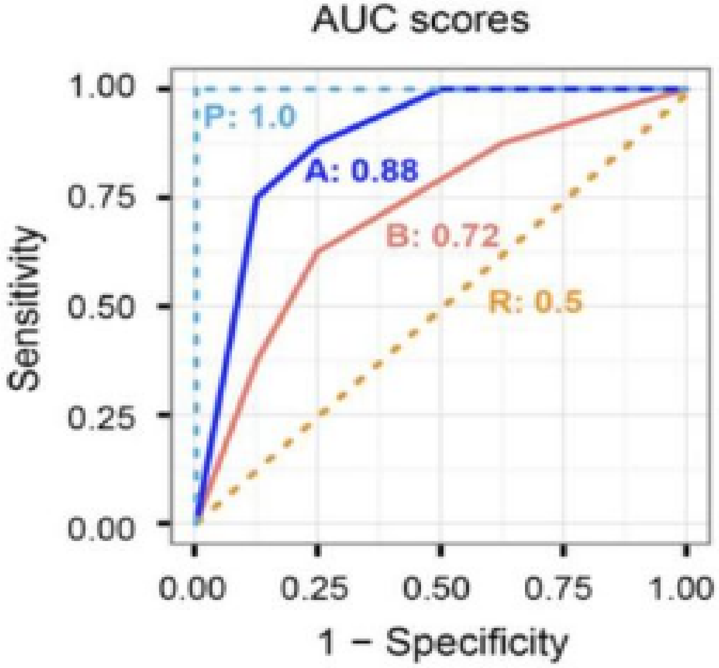


AUC

➢ **AUC(Area under the ROC curve) score**

- An advantage of using ROC curve is a single measure called AUC score
- As the name indicates, it is an area under the curve calculated in the ROC space
- Although the theoretical range of AUC score is between 0 and 1,
  the actual scores of meaningful classifiers are greater than 0.5,
  which is the AUC score of a random classifier

- ROC curves clearly shows classifiers A
  outperforms classifier B



The ROC curve is a useful tool for a few reasons:

•The curves of different models can be compared directly in general or for different thresholds.

•The area under the curve (AUC) can be used as a summary of the model skill.

Confusion Matrix with 3 Classes
True Positives

- The True positive value is where the actual value and predicted value are the same. They exists at diagonal.

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model correctly classified 50 apples and 50 oranges.

True Negatives for Class Apple

- The True Negative value for a class will be the sum of values of all columns and rows except the values of that class that we are calculating the values for.

		Predicted class		
		Apple	Orange	Pear
	Apple	50	5	50
Actual class	Orange	10	50	20
	Pear	5	5	0

The model correctly classified 75 cases as not belonging to class apple.

True Negatives for Class Orange

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model correctly classified 105 cases as not belonging to class orange.

True Negatives for Class Pear

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model correctly classified 115 cases as not belonging to class pear.

False Positives of Class Apple

- The False-positive value for a class will be the sum of values of the corresponding column except for the TP value.

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model incorrectly classified 15 cases as apples.

False Positives of Class Orange

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model incorrectly classified 10 cases as oranges.

False Positives of Class Pear

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model incorrectly classified 70 cases as pears.

False Negatives of Class Apple

- The False-negative value for a class will be the sum of values of corresponding rows except for the TP value.

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model incorrectly classified 55 cases as not belonging to class apple.

False Negatives of Class Orange

		Predicted class		
		Apple	Orange	Pear
	Apple	50	5	50
Actual class	Orange	10	50	20
	Pear	5	5	0

The model incorrectly classified 30 cases as not belonging to class orange.

False Negatives of Class Pear

		Predicted class		
		Apple	Orange	Pear
Actual class	Apple	50	5	50
	Orange	10	50	20
	Pear	5	5	0

The model incorrectly classified 10 cases as not belonging to class pears.

Example 2

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{30+60+80}{300} = 170/300 = .556$$

$$Recall_{class=0} = \frac{TP_{class=0}}{TP_{class=0}+FN_{class=0}} = \frac{30}{30+20+10} = .5$$

$$Recall_{class=1} = \frac{TP_{class=1}}{TP_{class=1}+FN_{class=1}} = \frac{60}{60+50+10} = .5$$

$$Recall_{class=2} = \frac{TP_{class=2}}{TP_{class=2}+FN_{class=2}} = \frac{80}{80+20+20} = .667$$

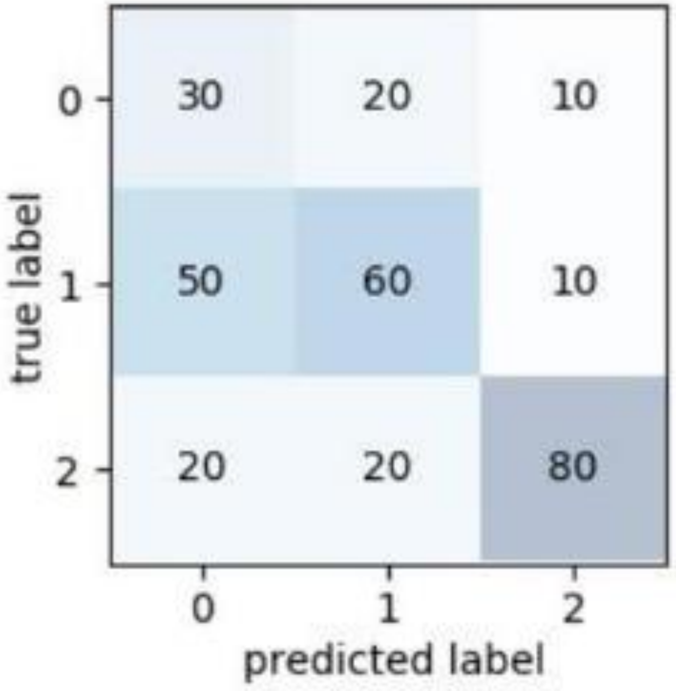
$$Recall = \frac{.5+.5+.667}{3} = 0.556$$

$$Precision_{class=0} = \frac{TP_{class=0}}{TP_{class=0}+FP_{class=0}} = \frac{30}{30+50+20} = .3$$

$$Precision_{class=1} = \frac{TP_{class=1}}{TP_{class=1}+FP_{class=1}} = \frac{60}{60+20+20} = .6$$

$$Precision_{class=2} = \frac{TP_{class=2}}{TP_{class=2}+FP_{class=2}} = \frac{80}{80+10+10} = 0.8$$

$$Precision = \frac{.3+.6+.8}{3} = 0.556$$



Model Diagnosis

Debugging a learning algorithm

- However, when you test your hypothesis on a new test set, you find that it makes unacceptably large errors in its predictions. What should you try next?

  - Get more training examples
  - Try smaller sets of features
  - Try getting additional features
  - Try adding polynomial features $(x_1^2, x_2^2, x_1 x_2, \text{etc.})$
  - Try decreasing λ
  - Try increasing λ

Machine Learning Diagnostic

- Diagnostic: A test that you can run to gain insight into what is/isn’t working with a learning algorithm, and gain guidance as to how best to improve its performance.

- Diagnostics can take time to implement, but doing so can be a very good use of your time.

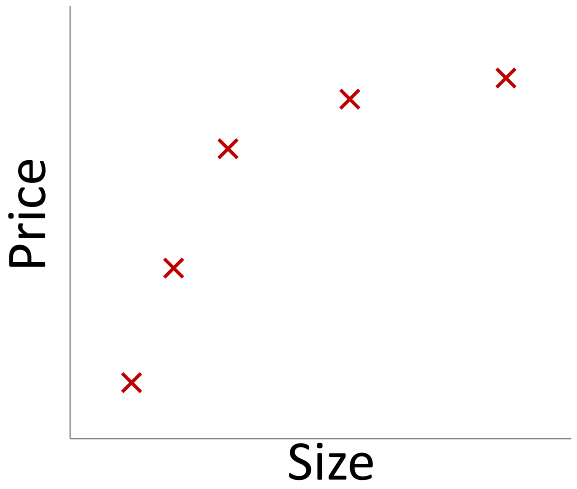
Model selection (Degree of the polynomial)

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$

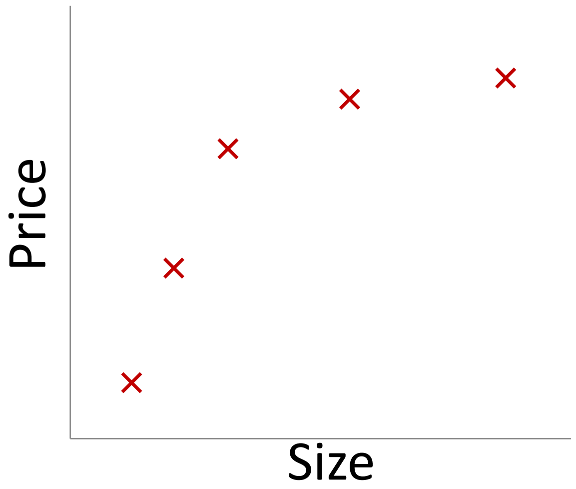
Bias/variance



$$\theta_0 + \theta_1 x$$

High bias
(underfit)

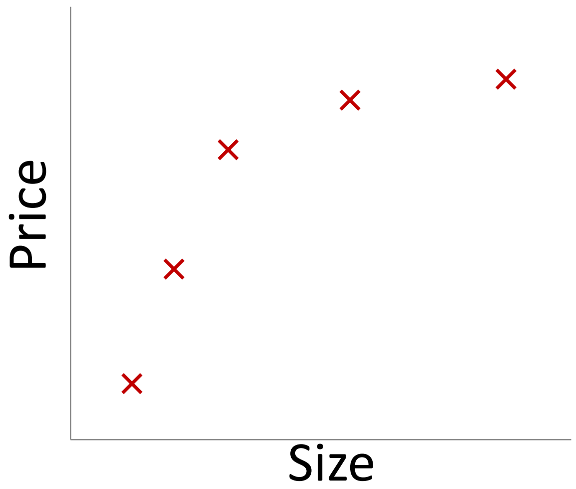
d=1



$$\theta_0 + \theta_1 x + \theta_2 x^2$$

“Just right”

d=2



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

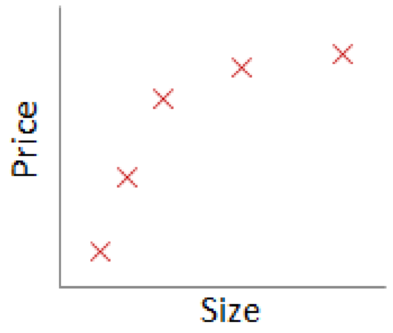
High variance
(overfit)

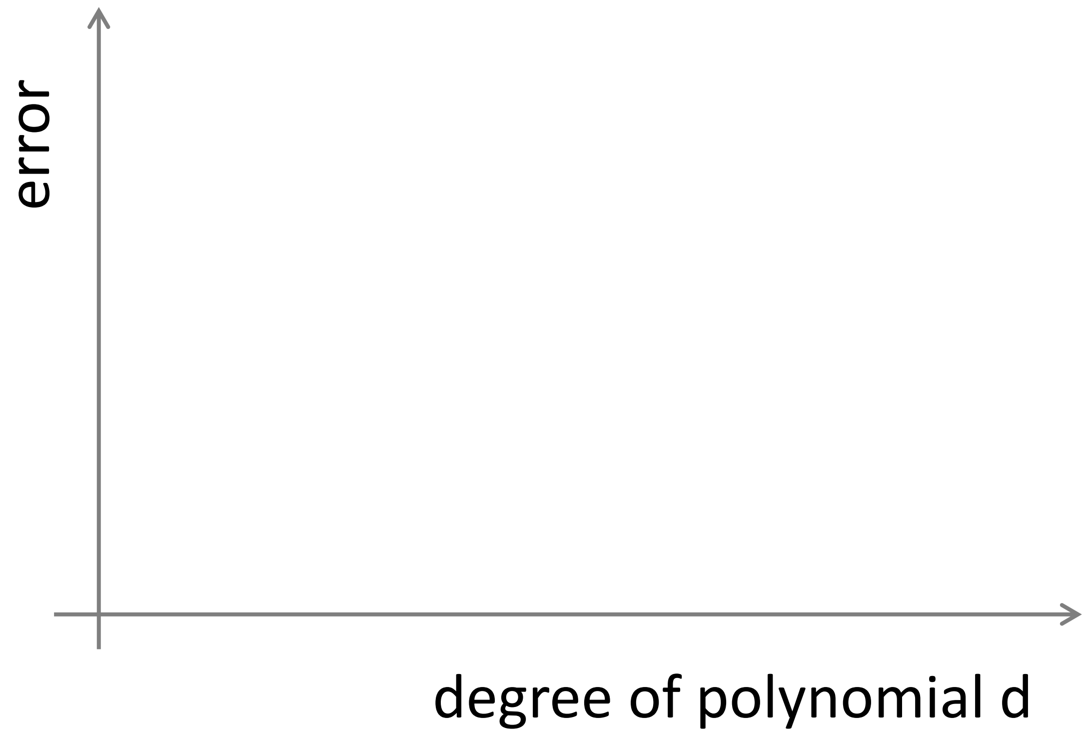
d=4

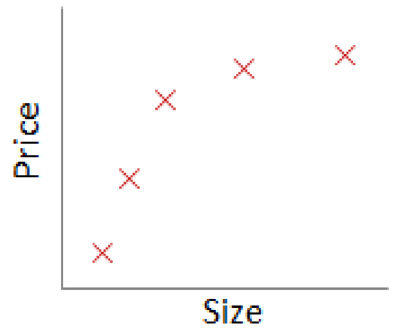
Bias/variance

Training error: $J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

Cross validation error: $J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$

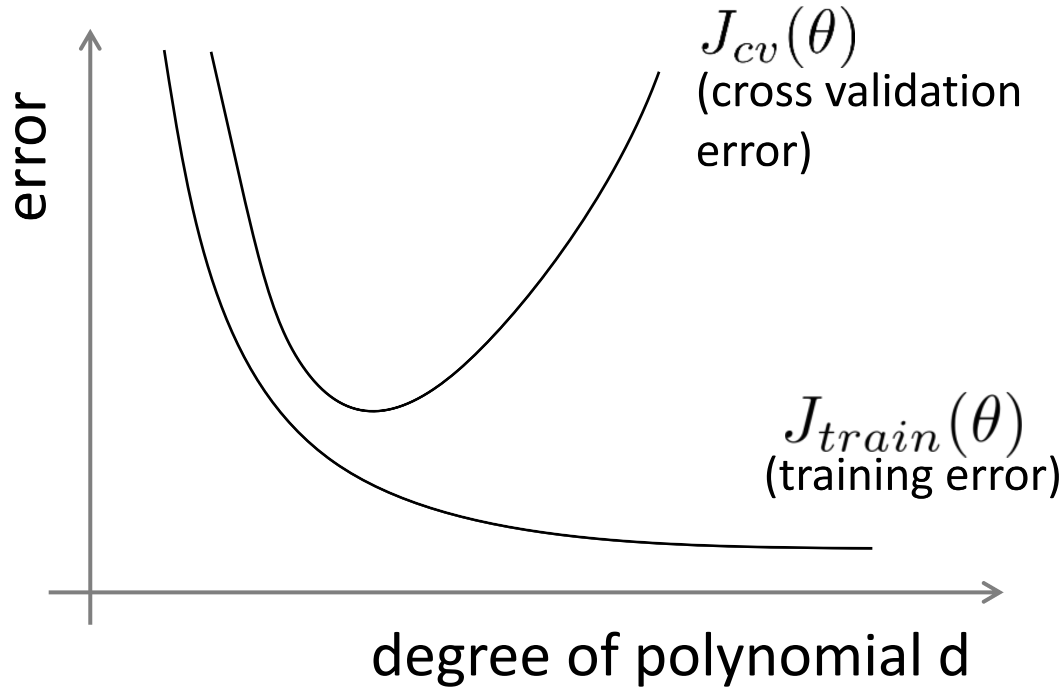






Diagnosing bias vs. variance

Suppose your learning algorithm is performing less well than you were hoping. ($J_{cv}(\theta)$ or $J_{test}(\theta)$ is high.) Is it a bias problem or a variance problem?



Bias (underfit):

Variance (overfit):

Choosing the regularization parameter λ

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

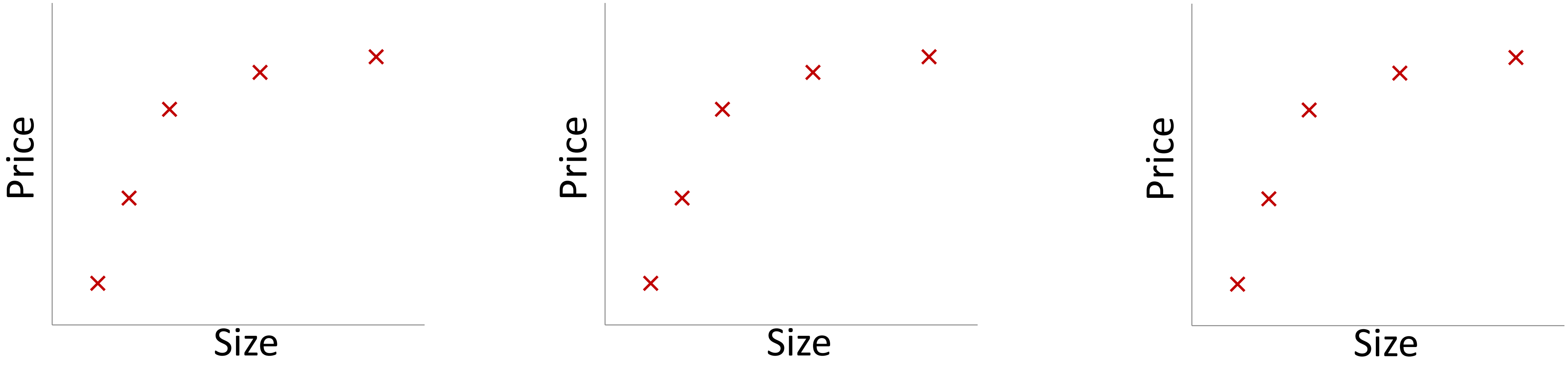
$$J_{test}(\theta) = \frac{1}{2m_{test}}\sum_{i=1}^{m_{test}}(h_\theta(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

These will be our cost function and we try to find the best regularization parameter

Linear regression with regularization

Model: $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$



Large λ — High bias (underfit)

Intermediate λ — "Just right"

Small λ — High variance (overfit)

$\lambda = 10000.\ \theta_1 \approx 0, \theta_2 \approx 0, \ldots$

$h_\theta(x) \approx \theta_0$

Choosing the regularization parameter λ

Model: $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

1. Try $\lambda = 0$
2. Try $\lambda = 0.01$
3. Try $\lambda = 0.02$
4. Try $\lambda = 0.04$
5. Try $\lambda = 0.08$

$\vdots$

12. Try $\lambda = 10$

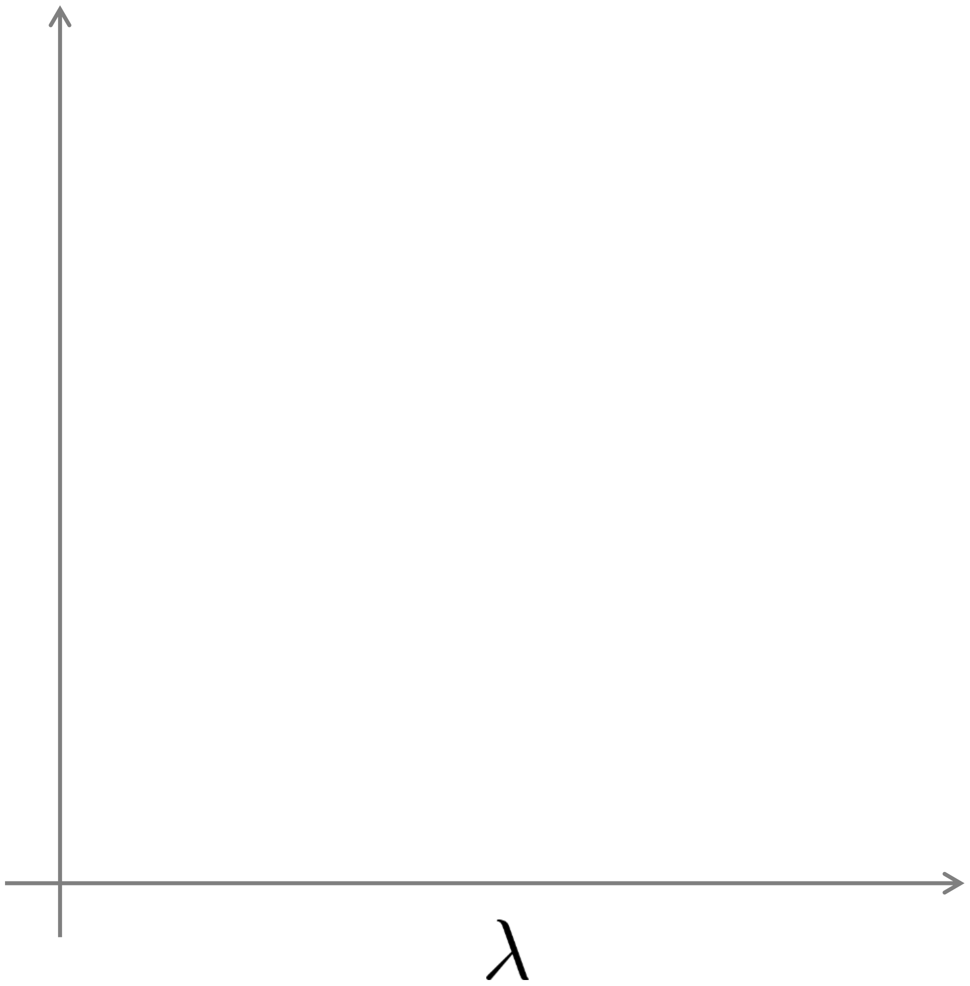
Pick (say) $\theta^{(5)}$. Test error:

Bias/variance as a function of the regularization parameter λ

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

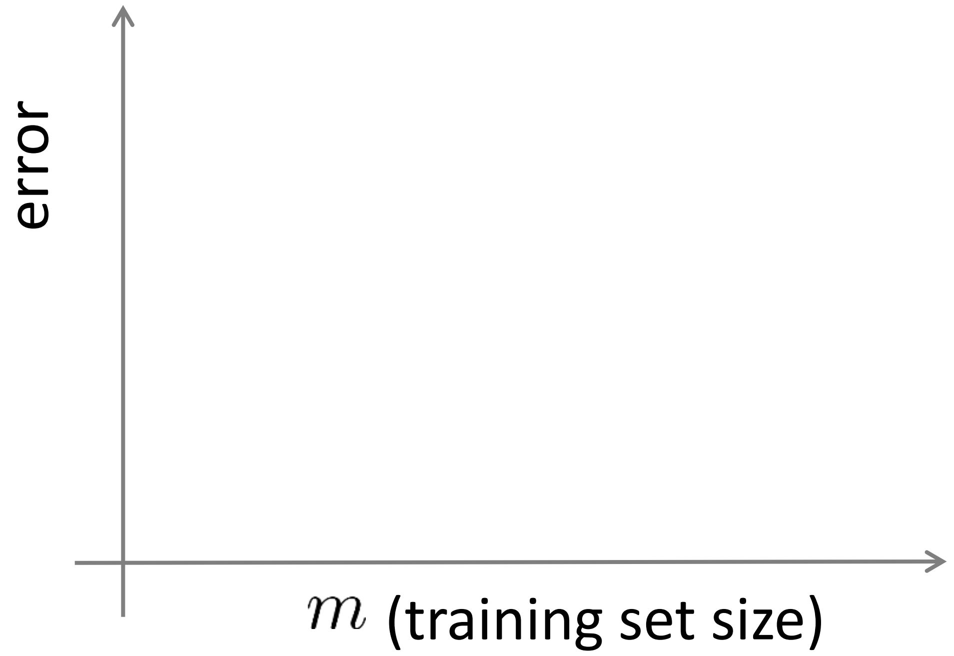
$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$



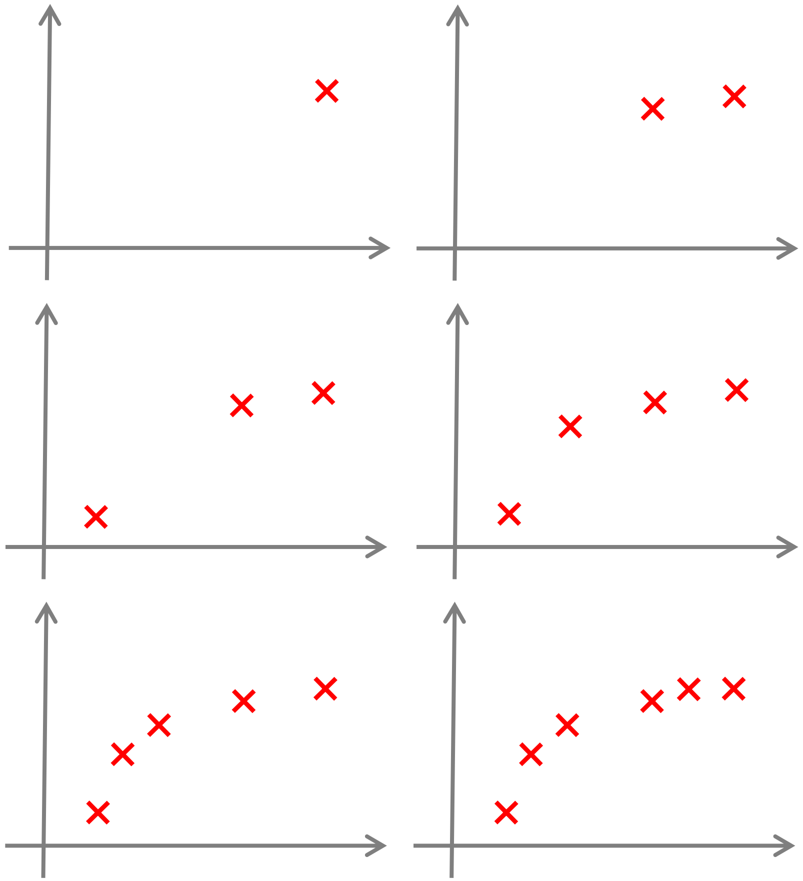
Learning curves

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

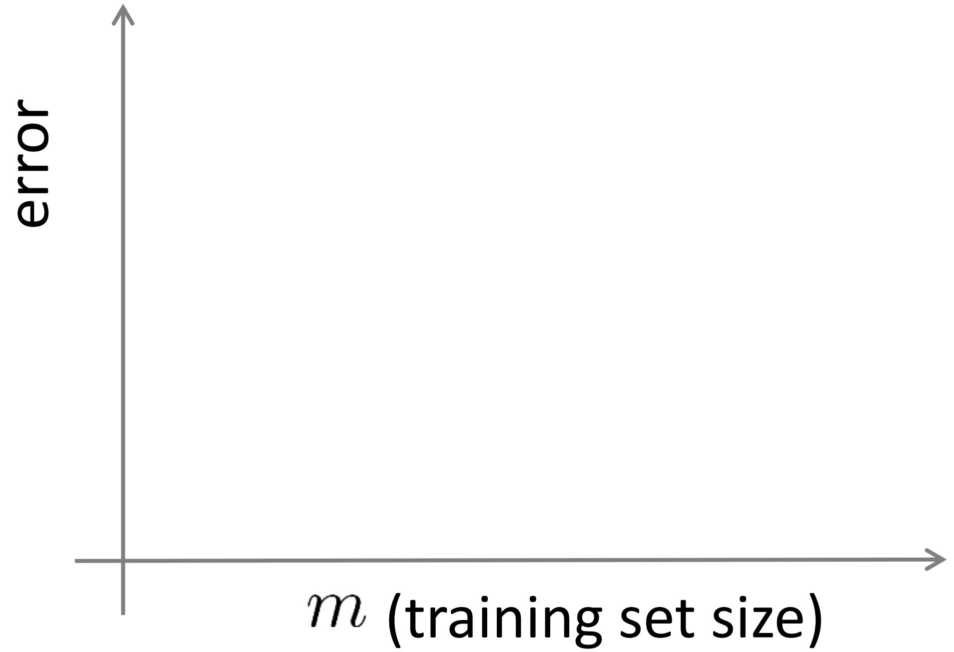


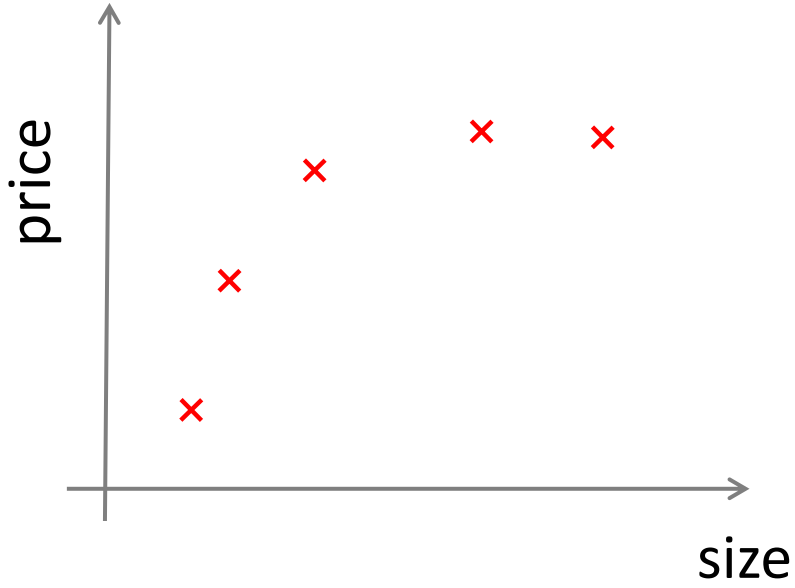
$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$

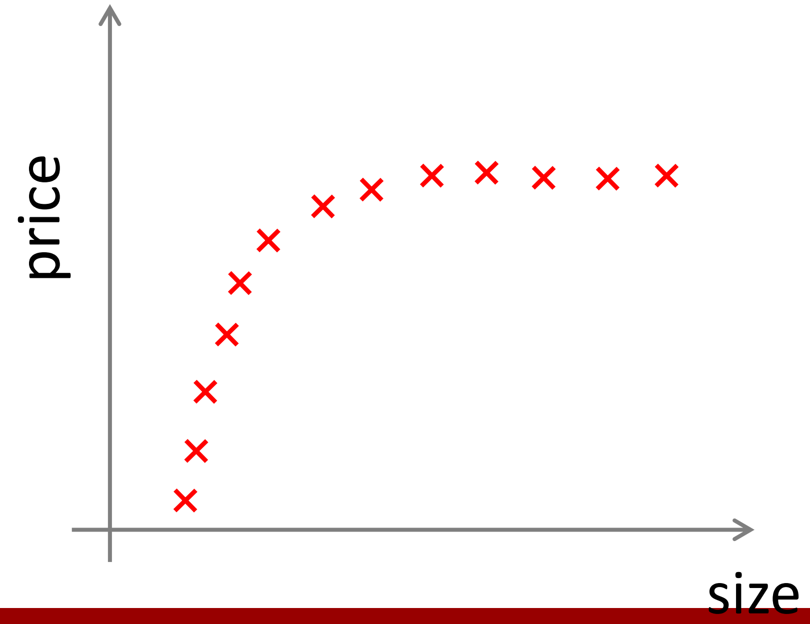


High bias

$$h_\theta(x) = \theta_0 + \theta_1 x$$





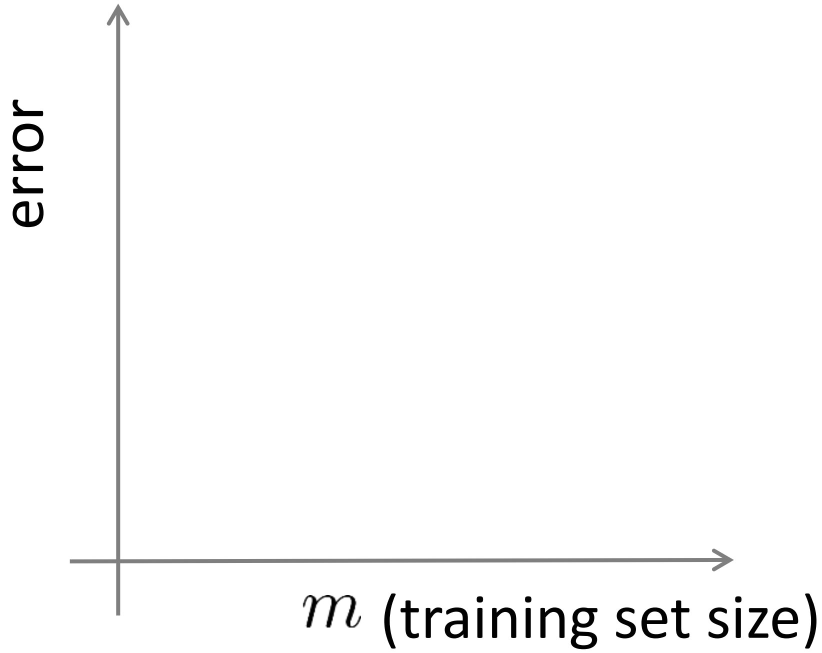


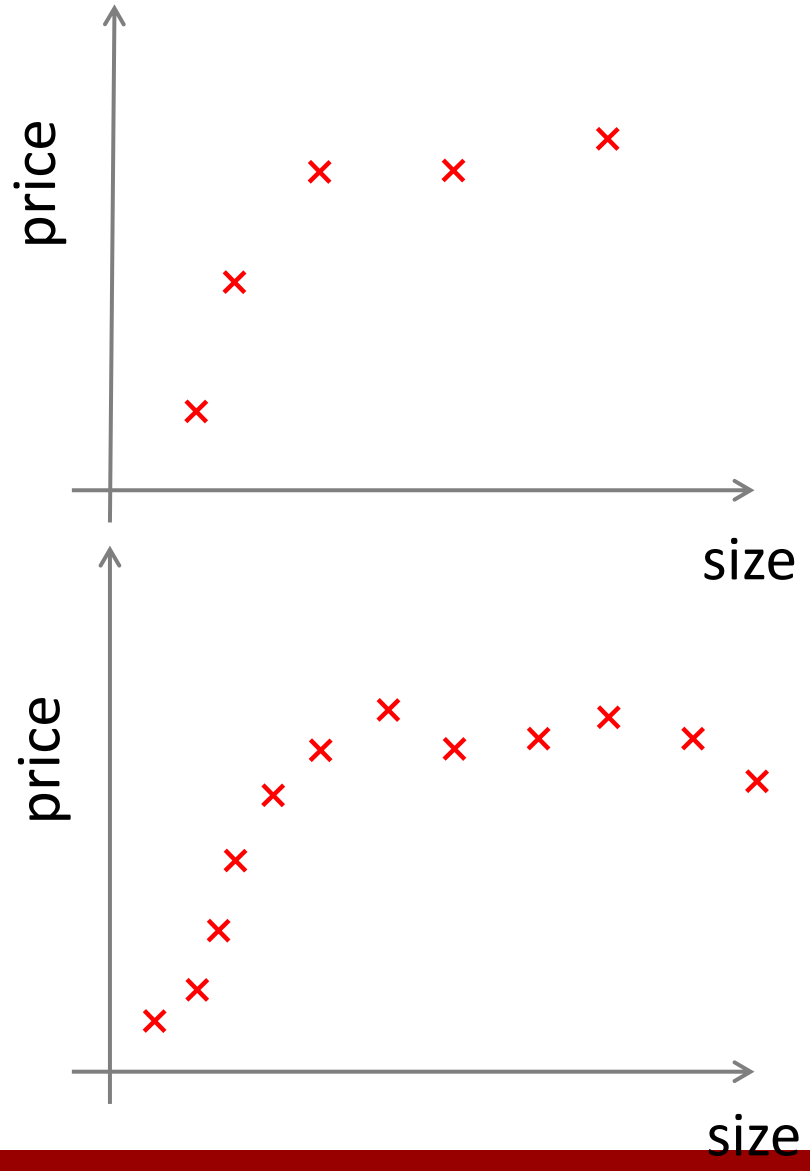
If a learning algorithm is suffering from high bias, **getting more training data will not (by itself) help much.**

High variance

$$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{100} x^{100}$$

(and small λ)





If a learning algorithm is suffering from high variance, **getting more training data is likely to help**.

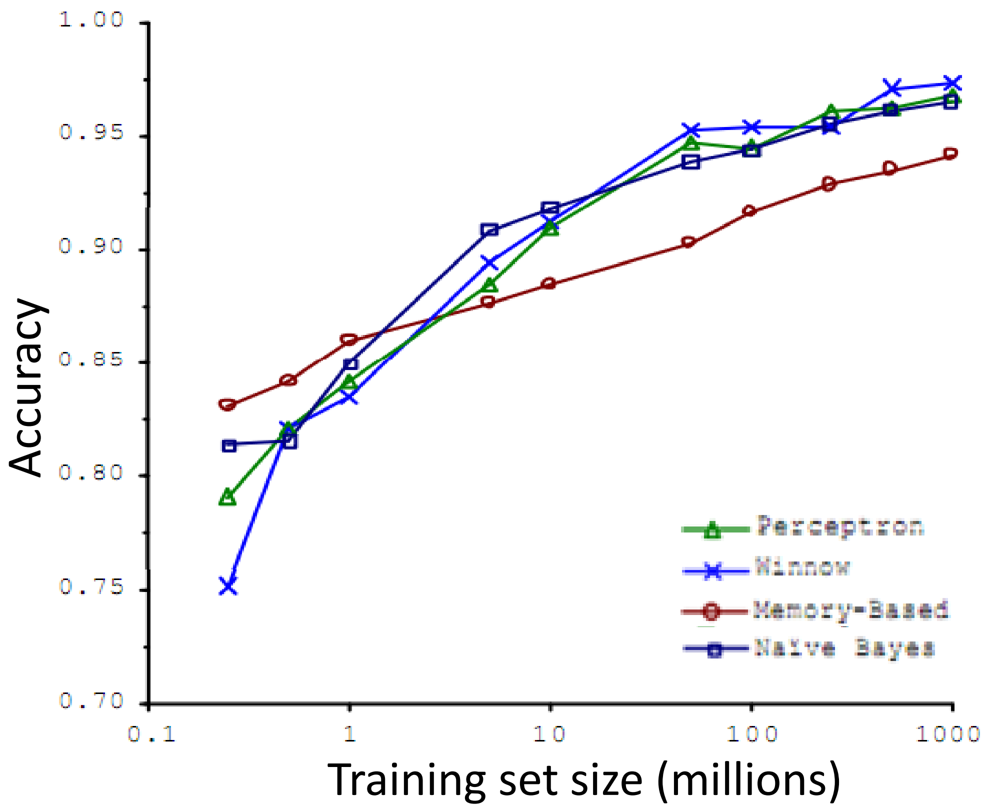
Designing a high accuracy learning system

E.g. Classify between confusable words.
{to, two, too}, {then, than}

For breakfast I ate ______ eggs.

Algorithms

- Perceptron (Logistic regression)
- Winnow
- Memory-based
- Naïve Bayes



"It's not who has the best algorithm that wins.
It's who has the most data."

[Banko and Brill, 2001]

Large data rationale

Use a learning algorithm with many parameters (e.g. logistic regression/linear regression with many features; neural network with many hidden units).

Use a very large training set (unlikely to overfit)

Debugging a learning algorithm:

Suppose you have implemented regularized linear regression to predict housing prices. However, when you test your hypothesis in a new set of houses, you find that it makes unacceptably large errors in its prediction. What should you try next?

- Get more training examples $fix\ high\ variance$
- Try smaller sets of features $fix\ high\ variance$
- Try getting additional features $fix\ high\ bias$
- Try adding polynomial features $(x_1^2, x_2^2, x_1x_2, \text{etc})$ $fix\ high\ bias$
- Try decreasing λ $fix\ high\ bias$
- Try increasing λ $fix\ high\ variance$

Andrew Ng

Thanks