Applied Machine Learning

Lecture: 7-1
Model Evaluation

Ekarat Rattagan, Ph.D.

Slides adapted from Andrew NG, Eric Eaton, Raquel Urtasun, and Patrick Winston

Model selection & evaluation

Model Selection

Model selection: estimating the performance of different models in order to choose the best one.

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

Which one to choose? how a model generalizes to unseen test data.

Credit: Andrew NG

Model Evaluation

- A part of the model development process to find the best model

- Two methods of evaluating models are

  1. Hold-out

  2. Cross validation

1. Hold-out method

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243

Train

$$(x^{(1)}, y^{(1)})$$
$$(x^{(2)}, y^{(2)})$$
$$\vdots$$
$$(x^{(m)}, y^{(m)})$$

Test

$$(x^{(1)}_{test}, y^{(1)}_{test})$$
$$(x^{(2)}_{test}, y^{(2)}_{test})$$
$$\vdots$$
$$(x^{(m_{test})}_{test}, y^{(m_{test})}_{test})$$

```
>>> import numpy as np
>>> from sklearn.model_selection import train_test_split
>>> X, y = np.arange(10).reshape((5, 2)), range(5)
>>> X
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7],
       [8, 9]])
>>> list(y)
[0, 1, 2, 3, 4]
>>> X_train, X_test, y_train, y_test = train_test_split(
...     X, y, test_size=0.30, random_state=42)
>>> X_train
array([[4, 5],
       [0, 1],
       [6, 7]])
>>> y_train
[2, 0, 3]
>>> X_test
array([[2, 3],
       [8, 9]])
>>> y_test
[1, 4]
```

1. Hold-out method

- Learn parameter θ from training data (70%)

- Compute test set (30%)

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

Credit: Andrew NG

1. Hold-out method

Method	Advantage	Disadvantage
1. Holdout	• Simple • Takes no longer to compute	• Its evaluation can have a high error

2. Cross-Validation

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243

50%

$$(x^{(1)}, y^{(1)})$$
$$(x^{(2)}, y^{(2)})$$
$$\vdots$$
$$(x^{(m)}, y^{(m)})$$

25%

$$(x_{cv}^{(1)}, y_{cv}^{(1)})$$
$$(x_{cv}^{(2)}, y_{cv}^{(2)})$$
$$\vdots$$
$$(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$$

25%

$$(x_{test}^{(1)}, y_{test}^{(1)})$$
$$(x_{test}^{(2)}, y_{test}^{(2)})$$
$$\vdots$$
$$(x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$$

Train/validation/test error

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:

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

Credit: Andrew NG

2. Cross Validation (K-fold)

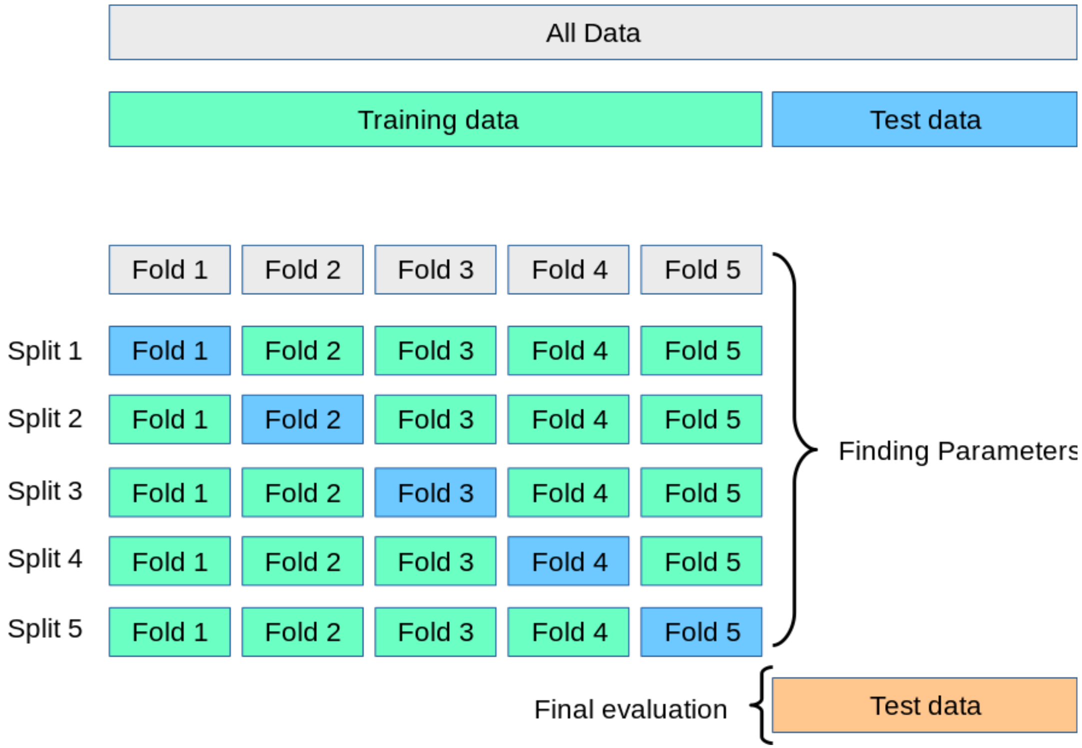


Figure from Hands-on
Machine Learning

The training set is divided into k subsets, and the **holdout method** is repeated k times.

Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set.

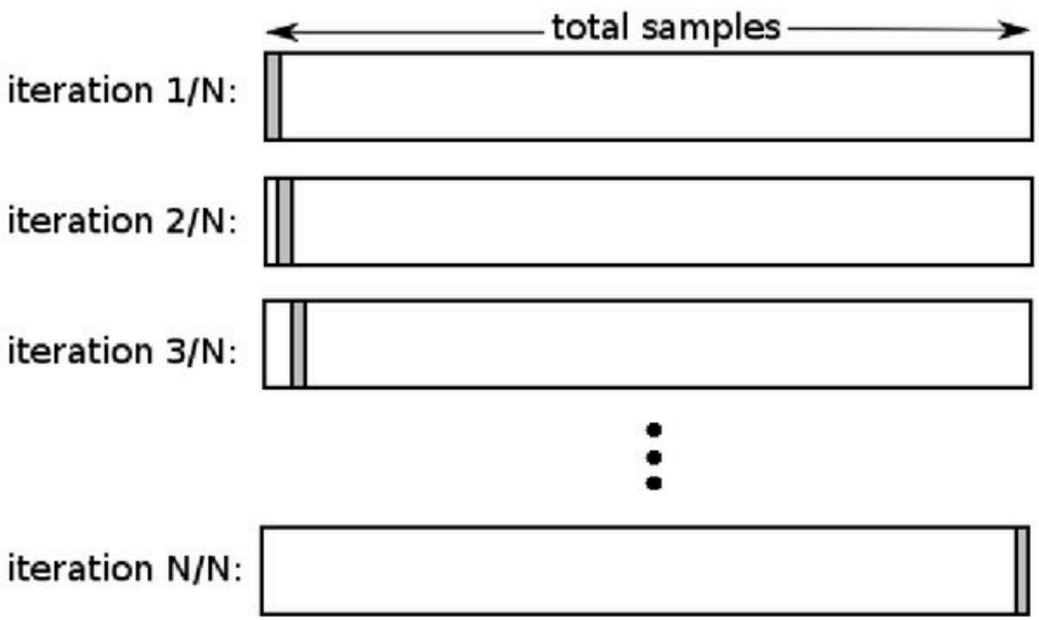
Then the average error across all k trials is computed.

```
>>> import numpy as np
>>> from sklearn.model_selection import KFold
>>> X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
>>> y = np.array([1, 2, 3, 4])
>>> kf = KFold(n_splits=2)
>>> kf.get_n_splits(X)
2
>>> print(kf)
KFold(n_splits=2, random_state=None, shuffle=False)
>>> for train_index, test_index in kf.split(X):
...     print("TRAIN:", train_index, "TEST:", test_index)
...     X_train, X_test = X[train_index], X[test_index]
...     y_train, y_test = y[train_index], y[test_index]
TRAIN: [2 3] TEST: [0 1]
TRAIN: [0 1] TEST: [2 3]
```

Evaluation method

Method	Advantage	Disadvantage
1. Holdout	• Simple • Low computation	• Its evaluation can have a high variance
2. Cross validation	• Every data point gets to be in a test set exactly once, and gets to be in a training set k-1 times. • The variance of the resulting estimate is reduced as *k* is increased	• High computation

LOOCV (Leave-one-out Cross Validation)



K-fold cross validation taken to its logical extreme, with K equal to N.

```
>>> from sklearn.model_selection import LeaveOneOut

>>> X = [1, 2, 3, 4]
>>> loo = LeaveOneOut()
>>> for train, test in loo.split(X):
...     print("%s %s" % (train, test))
[1 2 3] [0]
[0 2 3] [1]
[0 1 3] [2]
[0 1 2] [3]
```

Machine Learning Methods for Genomic Signature Extraction

Figure from Hands-on Machine Learning

Evaluation method

Method	Advantage	Disadvantage
1. Holdout	• Simple • Low computation	• Waste data (30% in this slide) • Its evaluation can have a high variance
2. Cross validation	• Every data point gets to be in a test set exactly once, and gets to be in a training set k-1 times. • The variance of the resulting estimate is reduced as *k* is increased	• High computation
3. Leave-One-Out	•	• High computation

- Zero randomness
- Lower bias as model is trained on the entire dataset

Diagnosing bias vs. variance

Machine learning diagnostic

Suppose you have implemented regularized linear regression to predict housing prices.

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

However, when you test your hypothesis on a new set of houses, 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

Credit: Andrew NG

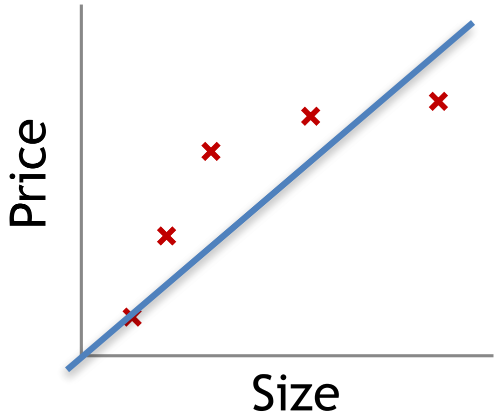
Machine learning diagnostic:

Diagnostic:
A test that you can run to gain insight 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.

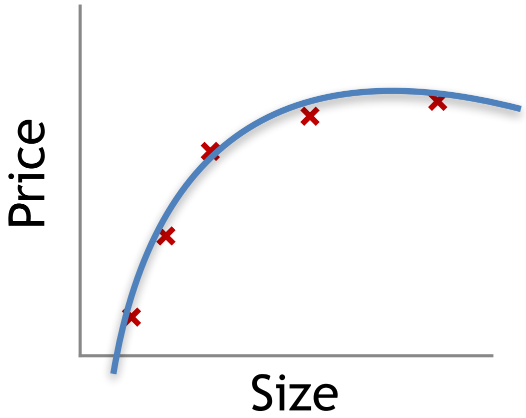
Credit: Andrew NG

Bias/variance



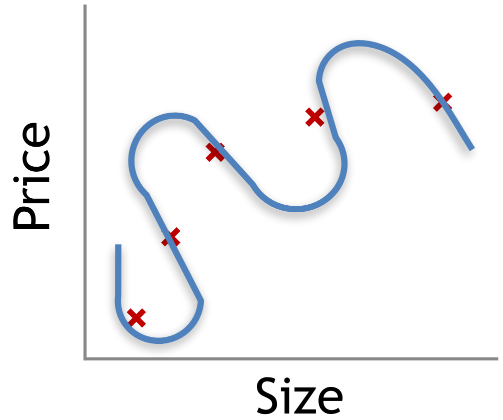
$$\theta_0 + \theta_1 x$$

High bias
(underfit)



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

"Just right"



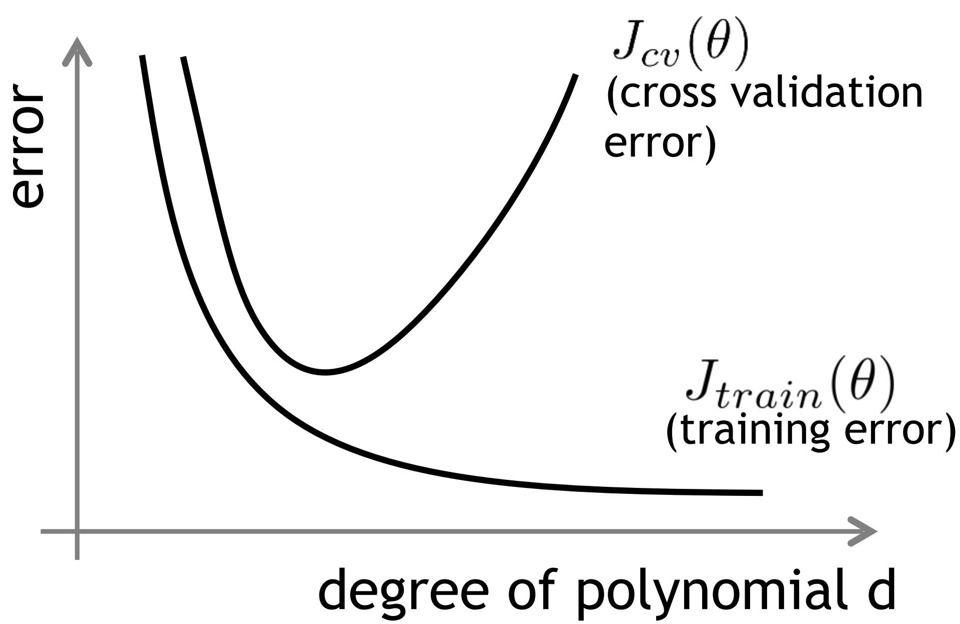
$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

High variance
(overfit)

Credit: Andrew NG

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):

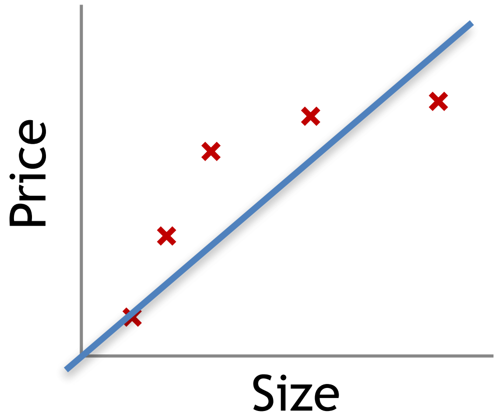
Credit: Andrew NG

Regularization and bias/ variance

Linear regression with regularization

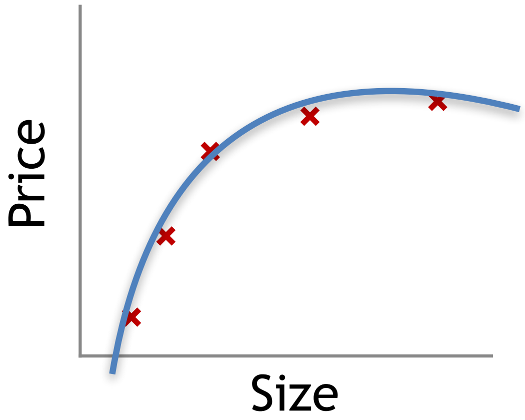
Model: $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

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

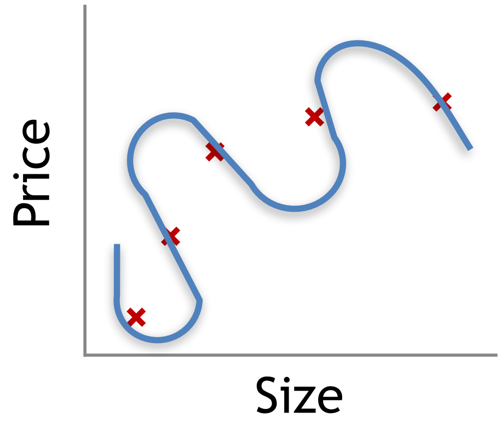


Large λ
High bias (underfit)

$\lambda = 10000.\ \theta_1 \approx 0, \theta_2 \approx 0, \ldots$
$h_\theta(x) \approx \theta_0$



Intermediate λ
"Just right"



Small λ
High variance (overfit)

Credit: Andrew NG

Choosing the regularization parameter λ

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

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

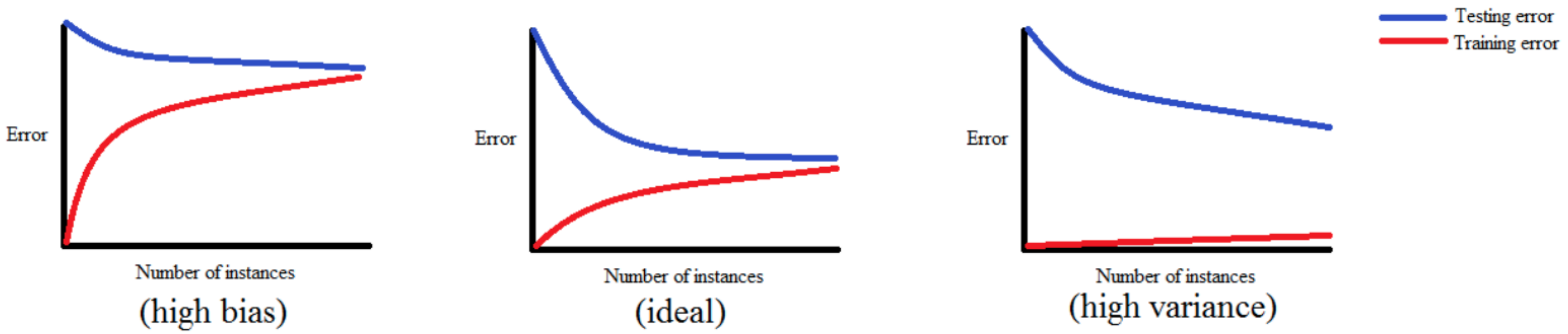
$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

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

Credit: Andrew NG

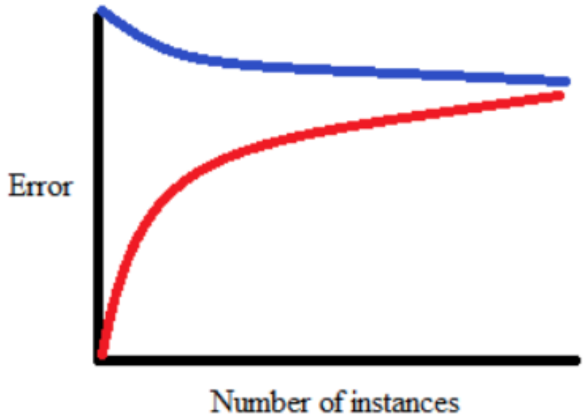
#Training data and bias/variance

Error VS #Training data



https://rmartinshort.jimdofree.com/2019/02/17/overfitting-bias-variance-and-leaning-curves/

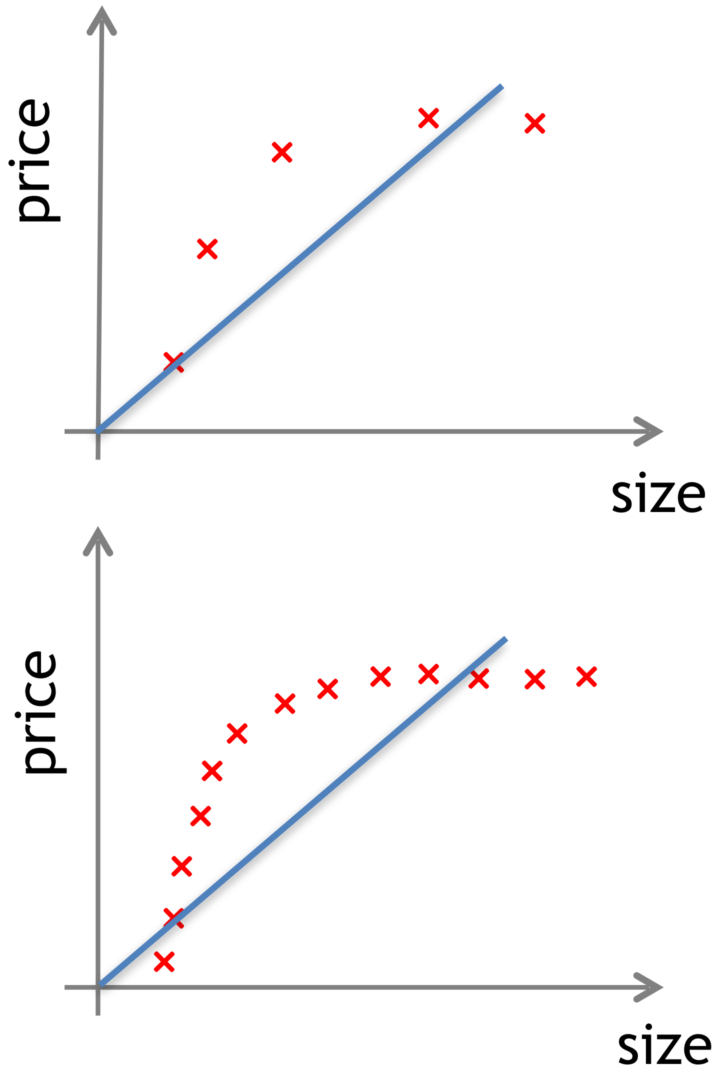
High bias



(high bias)

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

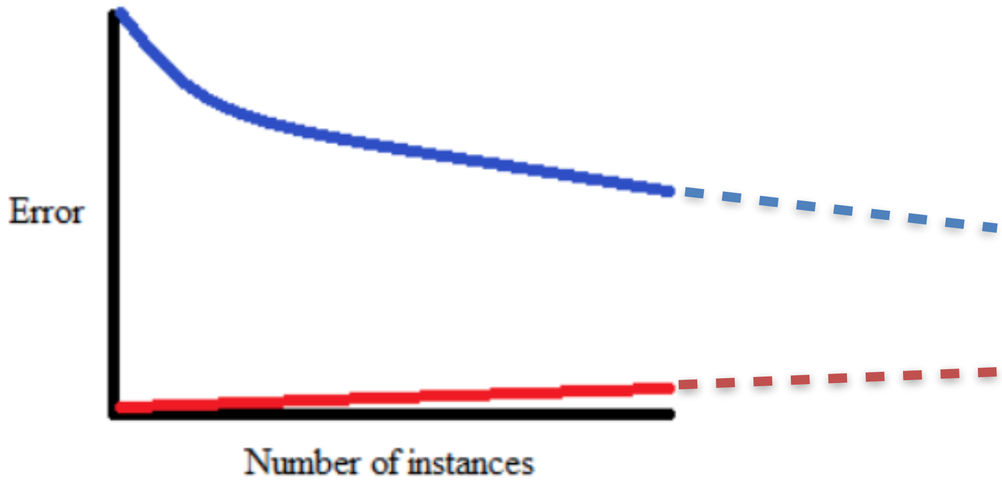
$$h_\theta(x) = \theta_0 + \theta_1 x$$



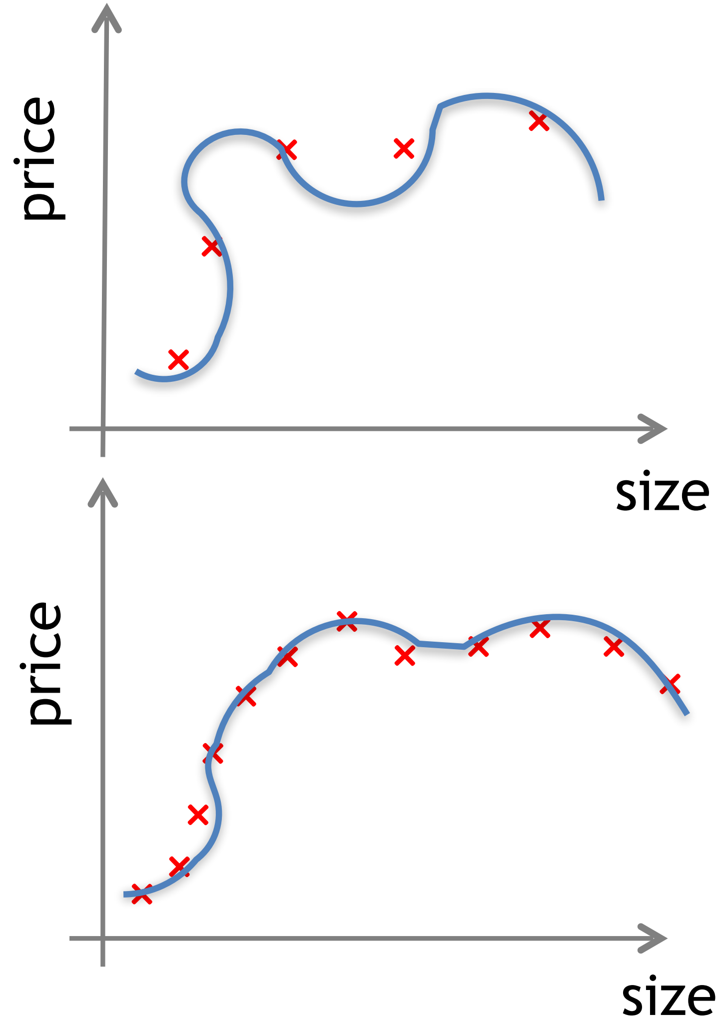
High variance

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

(and small λ)



(high variance)



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

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
- 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

Credit: Andrew NG

Error metrics

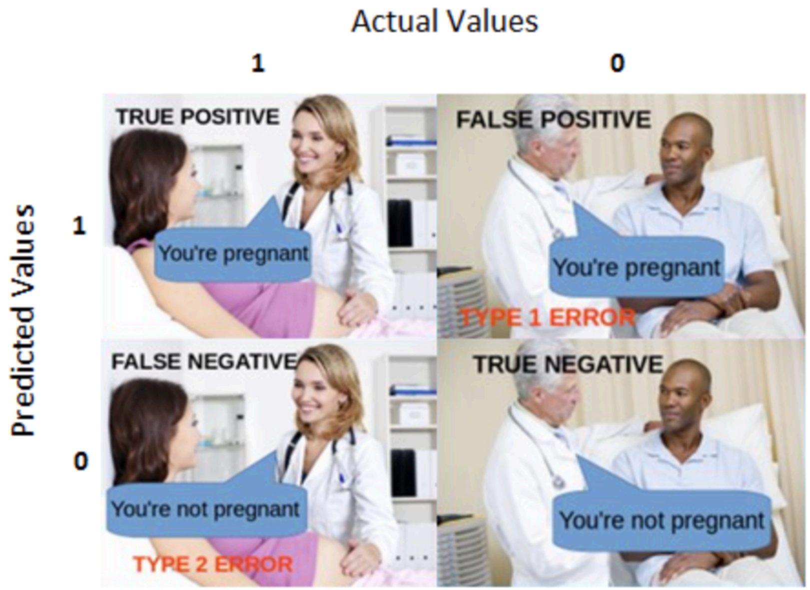
Error metrics

- Confusion Matrix
- Precision , Recall , and Accuracy
- F1-score
- ROC AUC Curve and score

Confusion Matrix

- A performance measurement for ML classification

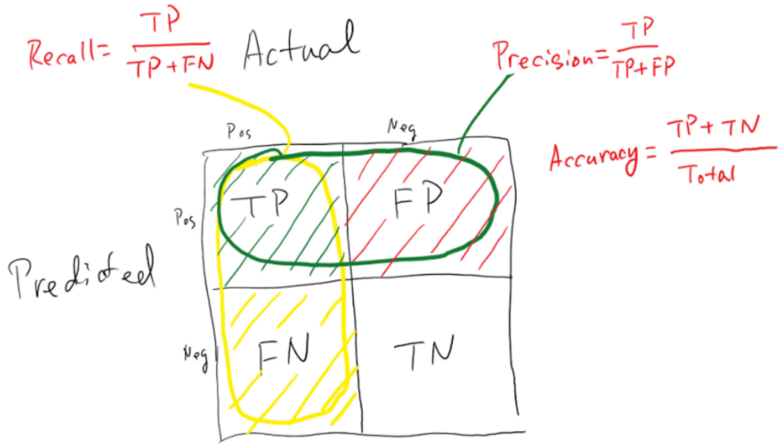
		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN



https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

Confusion Matrix

y	y pred	output for threshold 0.6	Recall	Precision	Accuracy
0	0.5	0	**1/2**	**2/3**	**4/7**
1	0.9	1			
0	0.7	1			
1	0.7	1			
1	0.3	0			
0	0.4	0			
1	0.5	0			



https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

Precision/Recall

$y = 1$ in presence of rare class that we want to detect

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

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

Credit: Andrew NG

Trading off precision and recall

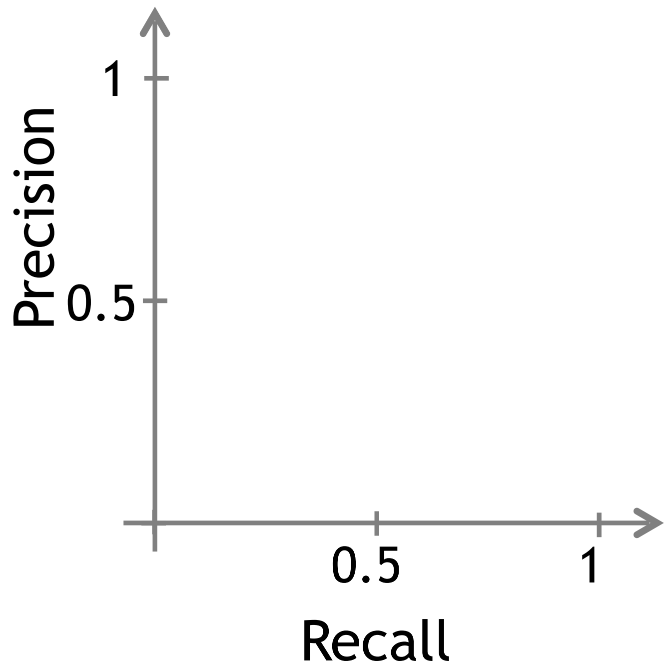
$$\text{precision} = \frac{\text{true positives}}{\text{no. of predicted positive}}$$

$$\text{recall} = \frac{\text{true positives}}{\text{no. of actual positive}}$$

Logistic regression: $0 \leq h_\theta(x) \leq 1$
Predict 1 if $h_\theta(x) \geq 0.5$
Predict 0 if $h_\theta(x) < 0.5$

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

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



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

Credit: Andrew NG

F_1 Score (F score)

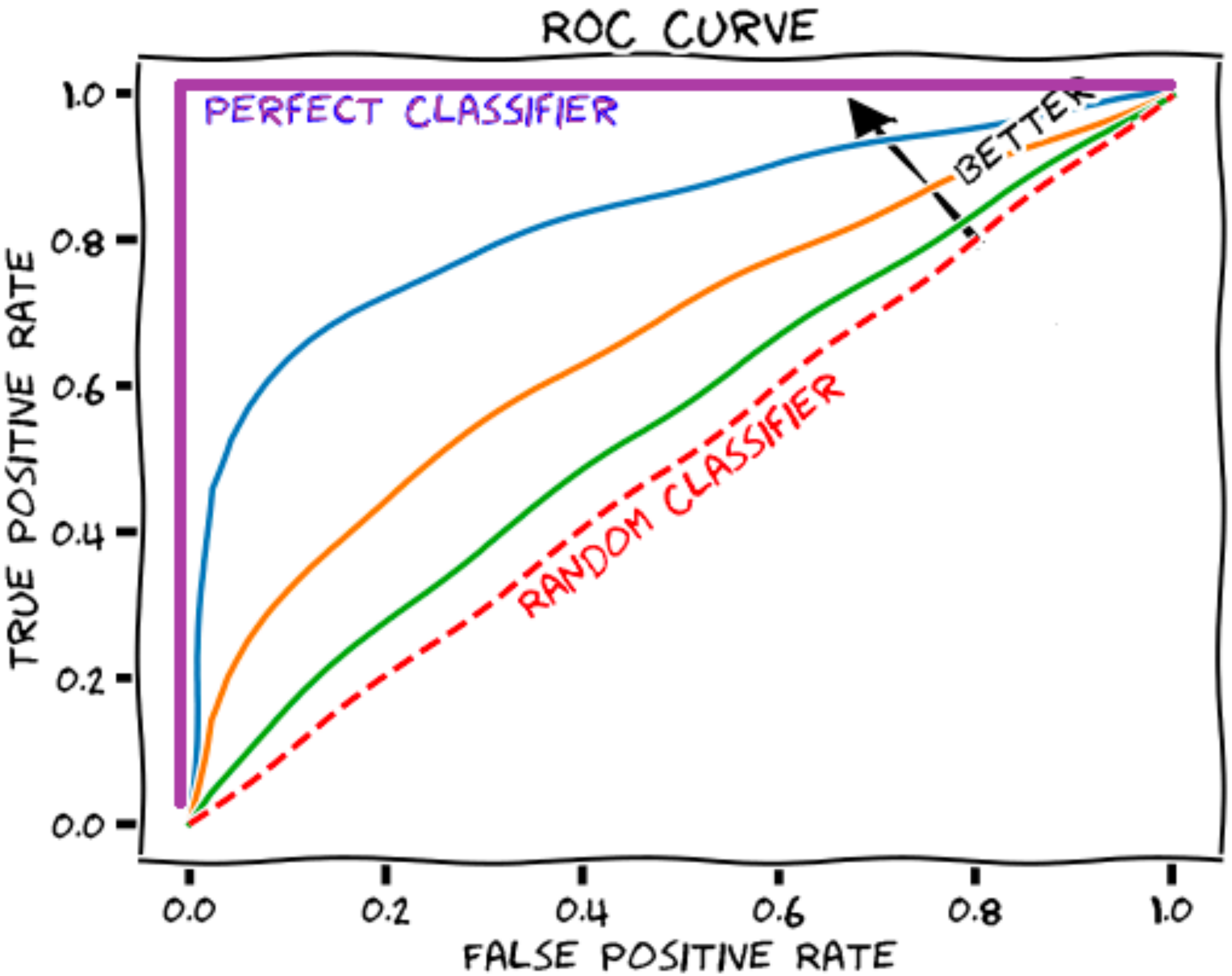
How to compare precision/recall numbers?

	Precision(P)	Recall (R)
Algorithm 1	0.5	0.4
Algorithm 2	0.7	0.1
Algorithm 3	0.02	1.0

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

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

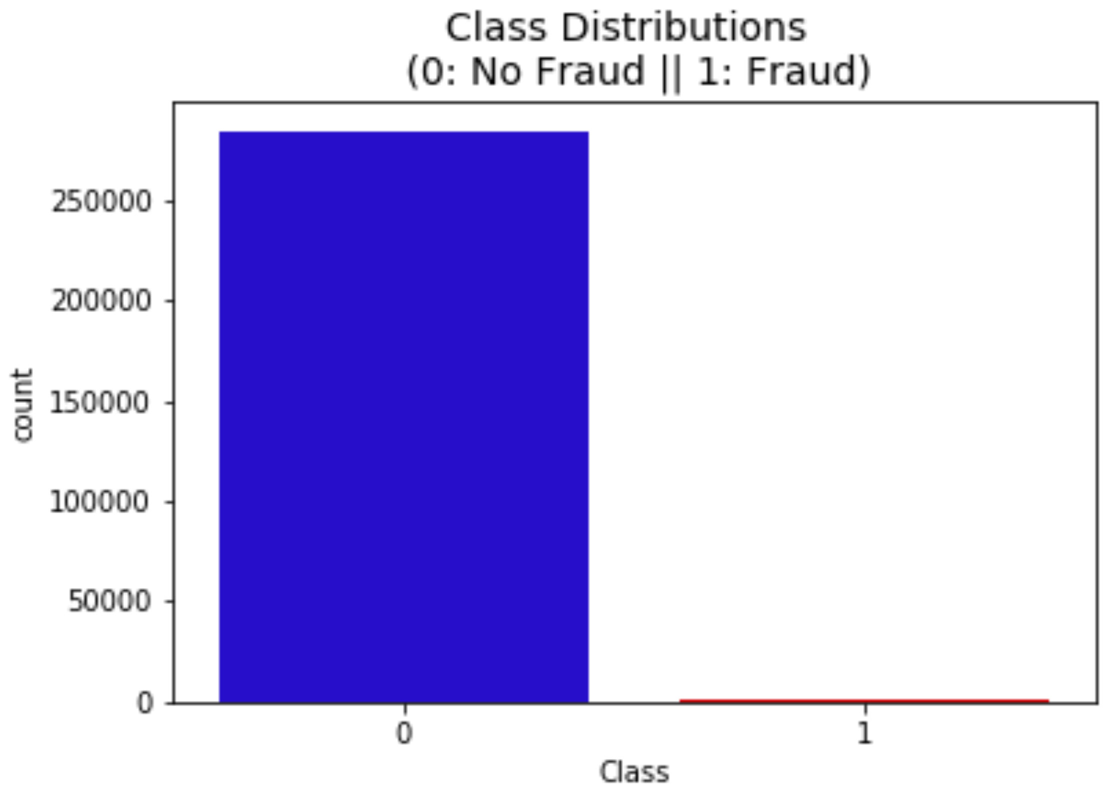
Credit: Andrew NG



https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/

Imbalanced data

What is imbalanced data



https://www.kaggle.com/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets