Performance Evaluation of Machine Learning Systems

Simone Filice
filice.simone@gmail.com
University of Roma Tor Vergata

Web Mining e Retrieval 2013/2014

Motivations

- Is a ML system performing properly?
- Among a set of different algorithms/models, which one is performing better on a given task?
- What can I do to improve my system?

Overview

- Performance Evaluation Metrics
  - Classifier Evaluation Metrics
  - Information Retrieval Systems Evaluation Metrics
- Tuning and Evaluation Methods
- Error Diagnostics

Overview

- **Performance Evaluation Metrics**
  - Classifier Evaluation Metrics
  - Information Retrieval Systems Evaluation Metrics
- Tuning and Evaluation Methods
- Error Diagnostics

Classifier Evaluation: Confusion Matrix

		PREDICTED VALUE		
ACTUAL VALUE		Class A	Class B	Class C
	Class A	38	12	0
	Class B	5	43	2
	Class C	6	0	44

$$accuracy = \frac{\#correct\ classifications}{\#classifications} = \frac{38 + 43 + 44}{150} = 83.33\%$$

$$error\ rate = \frac{\#incorrect\ classifications}{\#classifications} = \frac{12 + 5 + 2 + 6}{150} = 16.67\%$$

Evaluation with skewed data

- Accuracy is not a suitable metric for task with imbalanced classes (for instance a spam detector)

		PREDICTED VALUE	
ACTUAL VALUE		Spam	Non-Spam
	Spam	0	10
	Non-Spam	0	9990

$$accuracy = \frac{\#correct\ classifications}{\#classifications} = \frac{9990}{10000} = 99.9\%$$

Single Class Metrics

		PREDICTED VALUE	
ACTUAL VALUE		**Class C**	**Not Class C**
	Class C	**TP** True Positive	**FN** False Negative
	Not Class C	**FP** False Positive	**TN** True Negative

$$precision = \frac{TP}{TP + FP}$$

what percentage of instances the classifier labeled as positive are actually positive?

$$recall = \frac{TP}{TP + FN}$$

what percentage of positive instances did the classifier label as positive?

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

F-measure is the harmonic mean of precision and recall

Overview

- **Performance Evaluation Metrics**
  - Classifier Evaluation Metrics
  - Information Retrieval System Evaluation Metrics
- Tuning and Evaluation Methods
- Error Diagnostics

Challenging in Evaluating IR Models

- The output provided by an Information Retrieval System is not simply correct or wrong
- Ideally we need to estimate *user happiness*
- Happiness is elusive to measure
  - Most common proxy: *relevance* of search results

Challenging in Evaluating IR Models

- Effectiveness depends on the relevance of retrieved documents
- Relevance is hard to model. It should be a continuous function and not a binary value
- Relevance is:
  - Subjective: depends on the user's point of view
  - Contextual: depends on the current user's needs
  - Cognitive: is perceived and experienced by the user
  - Dynamic: changes over the time

Challenging in Evaluating IR Models

- A search engine is effective if it is able to provide documents that addresses user **information need**

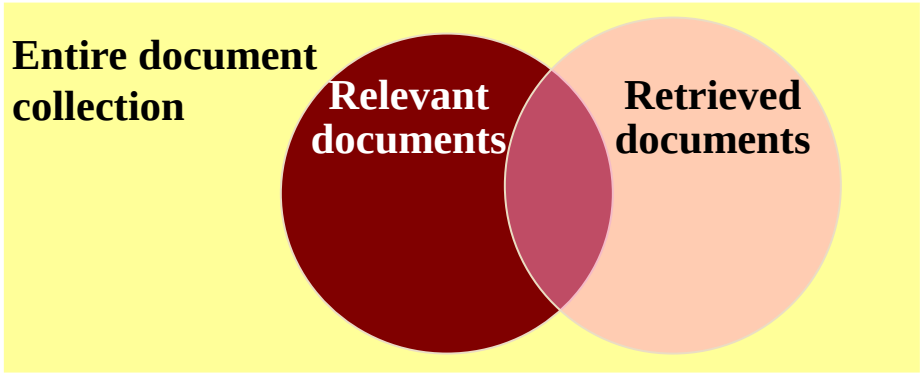
- The **information need** is translated into a **query**
- Relevance is assessed relative to the **information need** *not* the **query**
- E.g., Information need: *I'm looking for information on whether drinking red wine is more effective at reducing your risk of heart attacks than white wine.*
- Query: ***wine red white heart attack effective***
- Evaluate whether the doc addresses the information need, not whether it has these words

Evaluating IR Systems

- Tests directly involving users are the most reliable way to evaluate an IR system
  - A/B testing
  - Surveys…

- Offline tests are necessay to minimize the cost of the evaluation. Human Labeled Corpora (*Gold Standard*):
  - A benchmark document collection
  - A benchmark suite of queries
  - A usually binary assessment of either Relevant or Nonrelevant for each query and each document

Evaluating IR Systems



	retrieved	not retrieved
irrelevant	retrieved & irrelevant	Not retrieved & irrelevant
relevant	retrieved & relevant	not retrieved but relevant

$$recall = \frac{Number\ of\ relevant documents\ retrieved}{Total\ number\ of\ relevant documents}$$

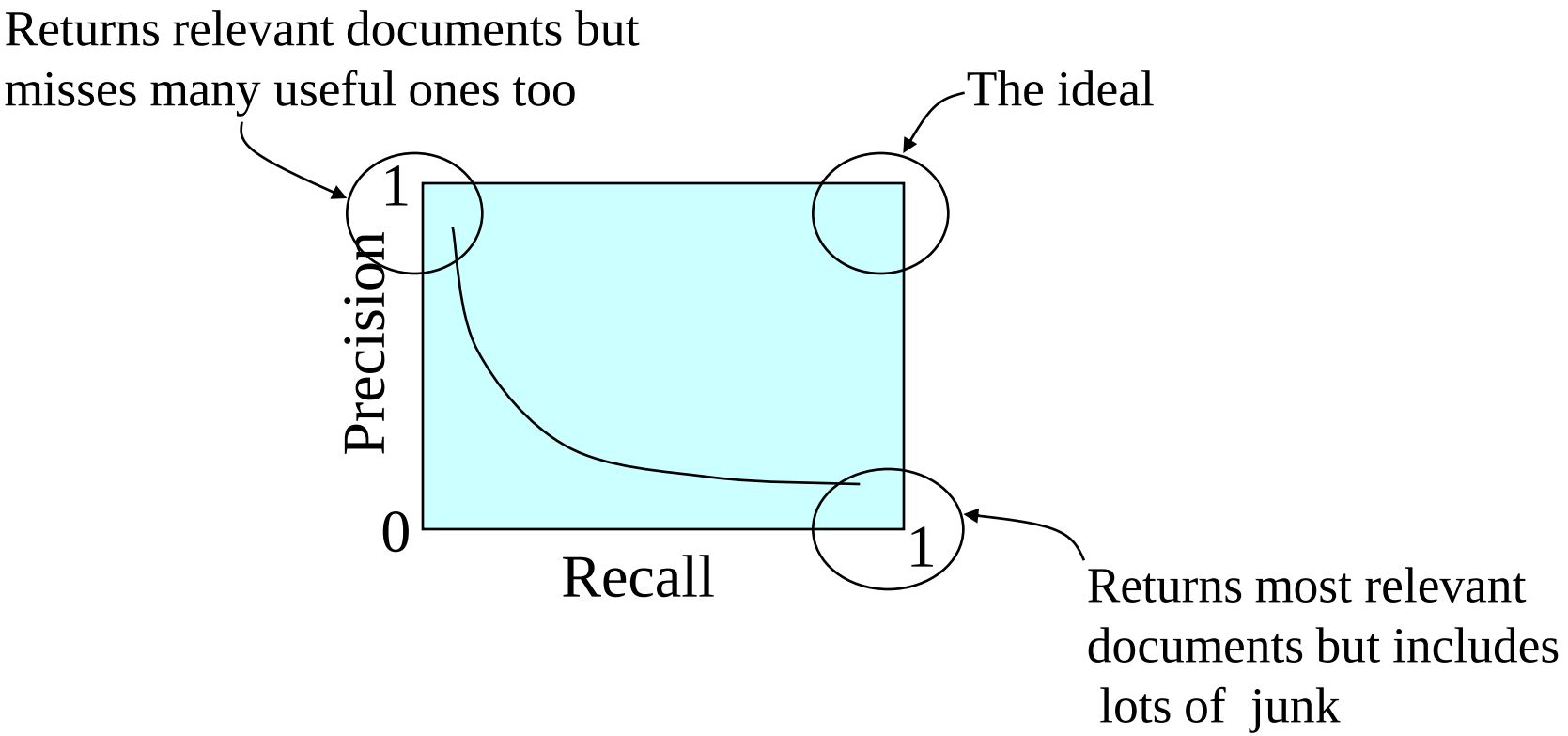
$$precision = \frac{Number\ of\ relevant documents\ retrieved}{Total\ number\ of\ documents\ retrieved}$$

What about accuracy???

Trade-off between Precision and Recall

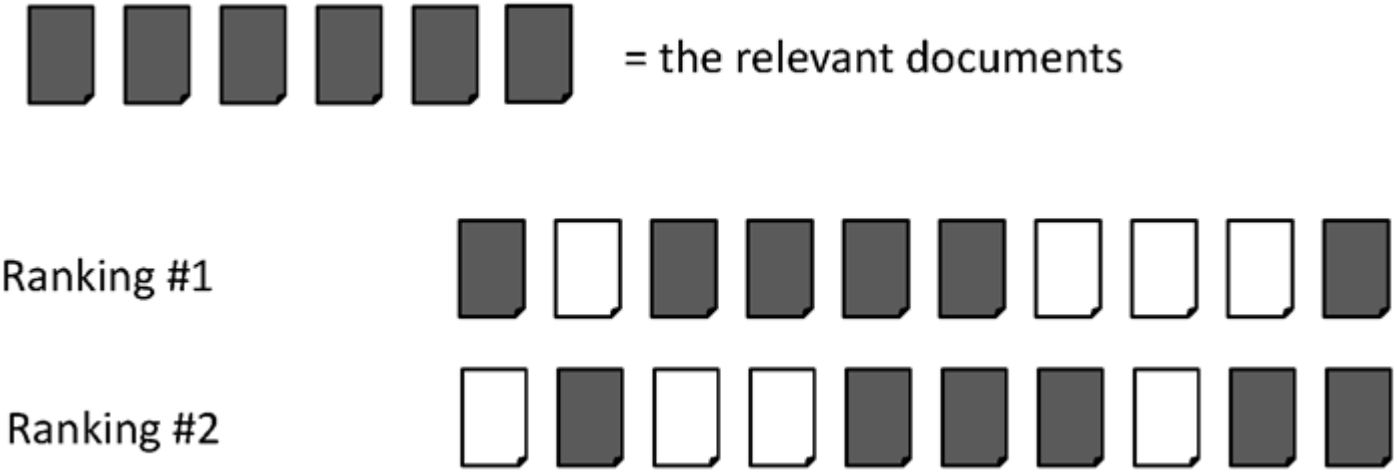
- You can get high recall (but low precision) by retrieving all docs for all queries!
- Recall is a non-decreasing function of the number of docs retrieved
- In a good system, precision decreases as either the number of docs retrieved or recall increases
  - This is not a theorem, but a result with strong empirical confirmation

Trade-off between Precision and Recall



Evaluating ranked results

- IR systems usually outputs the retrieved documents in a ranked list
  - A proper evaluating should mainly consider elements in the top of the list



Recall/Precision Points

- Compute a recall/precision pair for each position in the ranked list that contains a relevant document.

n	doc #	relevant
1	588	x
2	589	x
3	576	
4	590	x
5	986	
6	592	x
7	984	
8	988	
9	578	
10	985	
11	103	
12	591	
13	772	x
14	990	

Total number of relevant docs = 6
Check each new recall point:

R=1/6=0.167; P=1/1=1

R=2/6=0.333; P=2/2=1

R=3/6=0.5; P=3/4=0.75

R=4/6=0.667; P=4/6=0.667

R=5/6=0.833; P=5/13=0.38

Missing one relevant document
Never reach 100% recall

Averaging over Queries

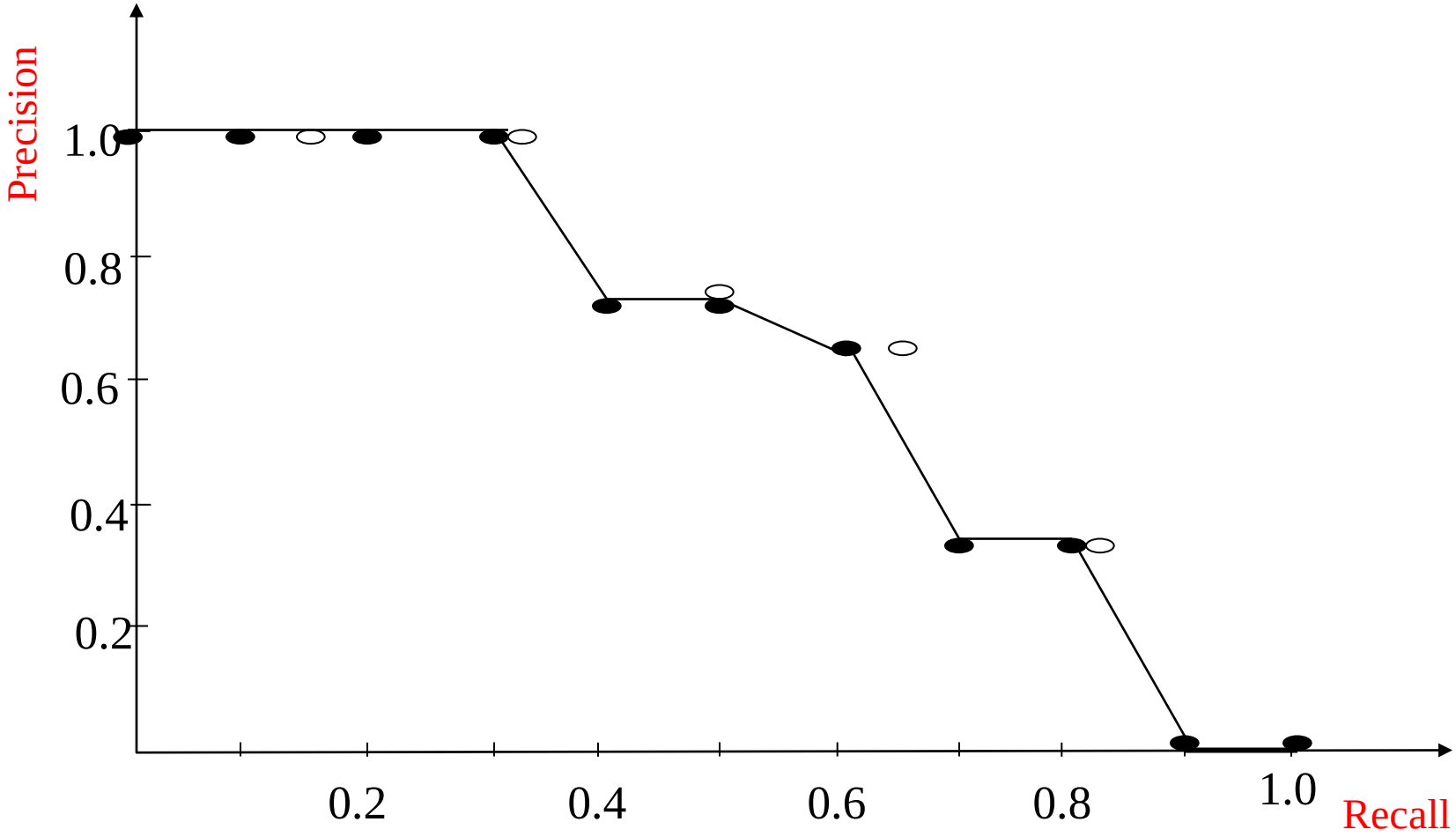
- A precision-recall graph for one query isn't a very sensible thing to look at
  - You need to average performance over a whole bunch of queries
- Some standard recall levels r_j are set. Typically:

  $r_0 = 0.0, r_1 = 0.1, \ldots, r_{10}=1.0$ (11-point interpolated average precision)
- For each query the precision corresponding to each standard recall levels are estimated via interpolation:

$$P_{interp}(r_j) = \max_{r \geq r_j} P(r)$$

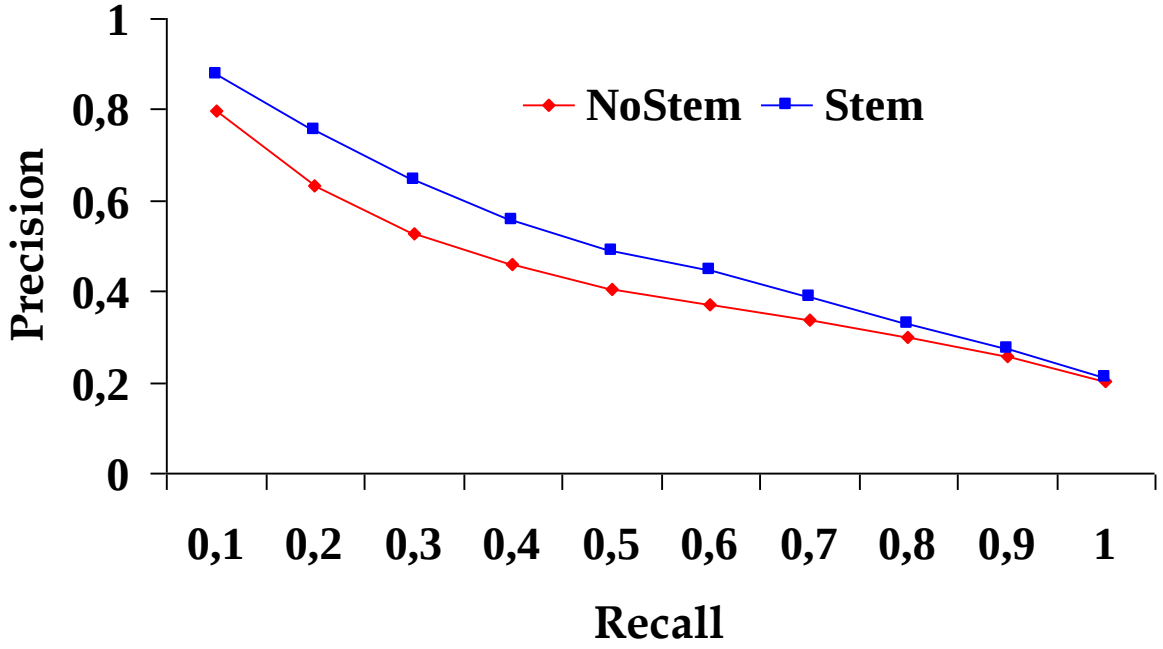
- Plot average precision/recall curves to evaluate overall system performance on a document/query corpus.

Interpolating a Recall/Precision Curve



Compare two or more Systems

- The curve closest to the upper right-hand corner of the graph indicates the best performance



- Graphs are good, but people want a summary measure….

Ranking metrics

- Precision at fixed retrieval level
  - Precision-at-*k* *(P@k)*: Precision of top *k* results
  - Perhaps appropriate for most of web search: all people want are good matches on the first one or two result pages

- Mean Average Precision (MAP)

$$MAP(Q) = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{|R_q|} \sum_{d \in R_q} P@k_{q,d}$$

Q = set of queries

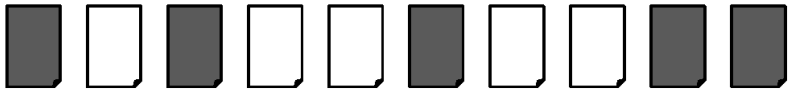
R_q=set of relevant documents for the query *q*

$K_{q,d}$=ranking of the document *d* retrieved throught the query *q*

Mean Average Precision


= relevant documents for query 1

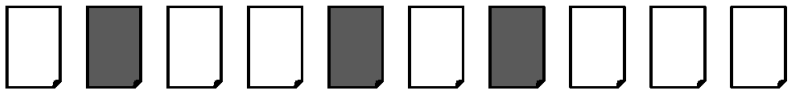
Ranking #1



Recall	0.2	0.2	0.4	0.4	0.4	0.6	0.6	0.6	0.8	1.0
Precision	1.0	0.5	0.67	0.5	0.4	0.5	0.43	0.38	0.44	0.5


= relevant documents for query 2

Ranking #2



Recall	0.0	0.33	0.33	0.33	0.67	0.67	1.0	1.0	1.0	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.33	0.43	0.38	0.33	0.3

$$average\ precision\ query\ 1 = (1.0 + 0.67 + 0.5 + 0.44 + 0.5)/5 = 0.62$$
$$average\ precision\ query\ 2 = (0.5 + 0.4 + 0.43)/3 = 0.44$$

$$mean\ average\ precision = (0.62 + 0.44)/2 = 0.53$$

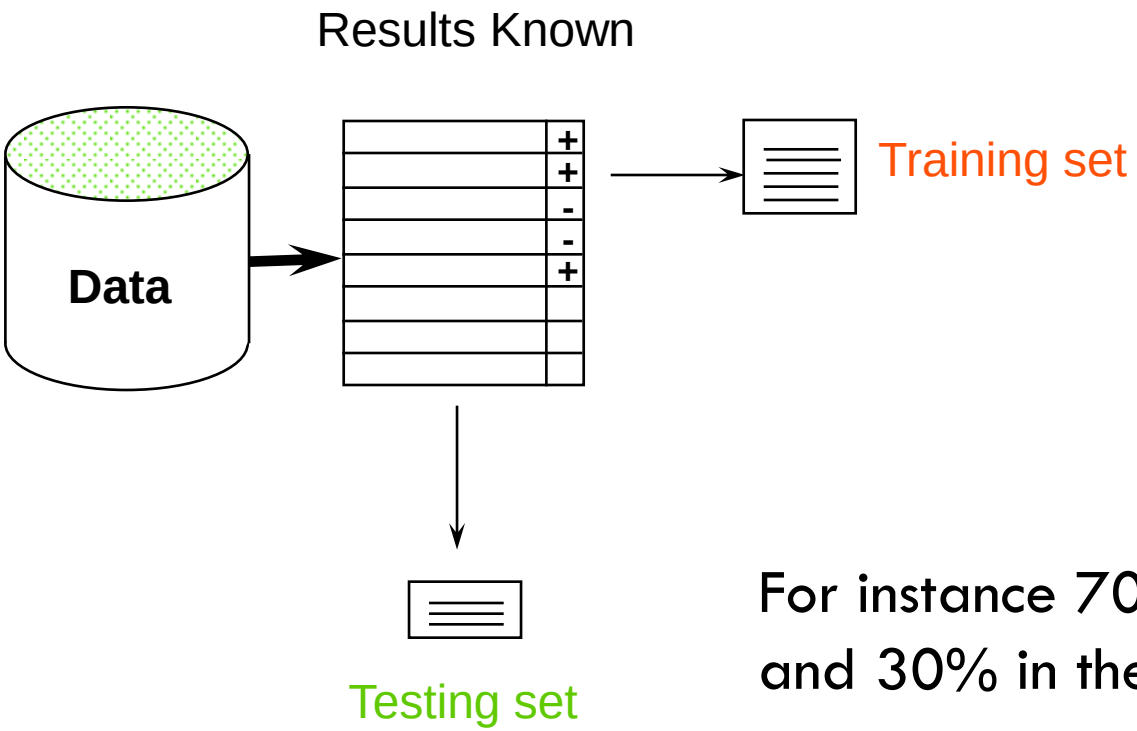
Overview

- Performance Evaluation Metrics
  - Classifier Evaluation Metrics
  - Information Retrieval Systems Evaluation Metrics
- **Tuning and Evaluation Methods**
- Error Diagnostics

Testing Data

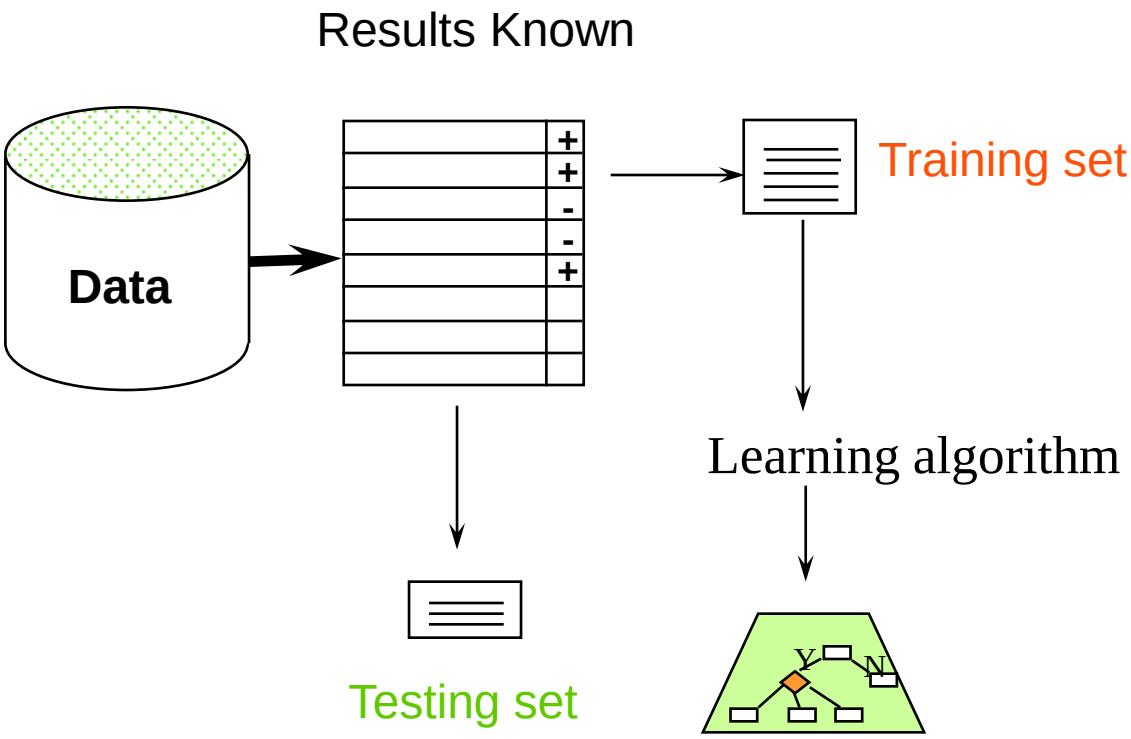
- To obtain a reliable estimation, test data must be instances not used during the training step
  - Error on the training data is *not* a good indicator of performance on future data, because new data will probably not be **exactly** the same as the training data!
  - Overfitting – fitting the training data too precisely - usually leads to poor results on new data
  - We want to evaluate how predictive the model we learned is, and not its memorization capability

Step 1: dataset splitting

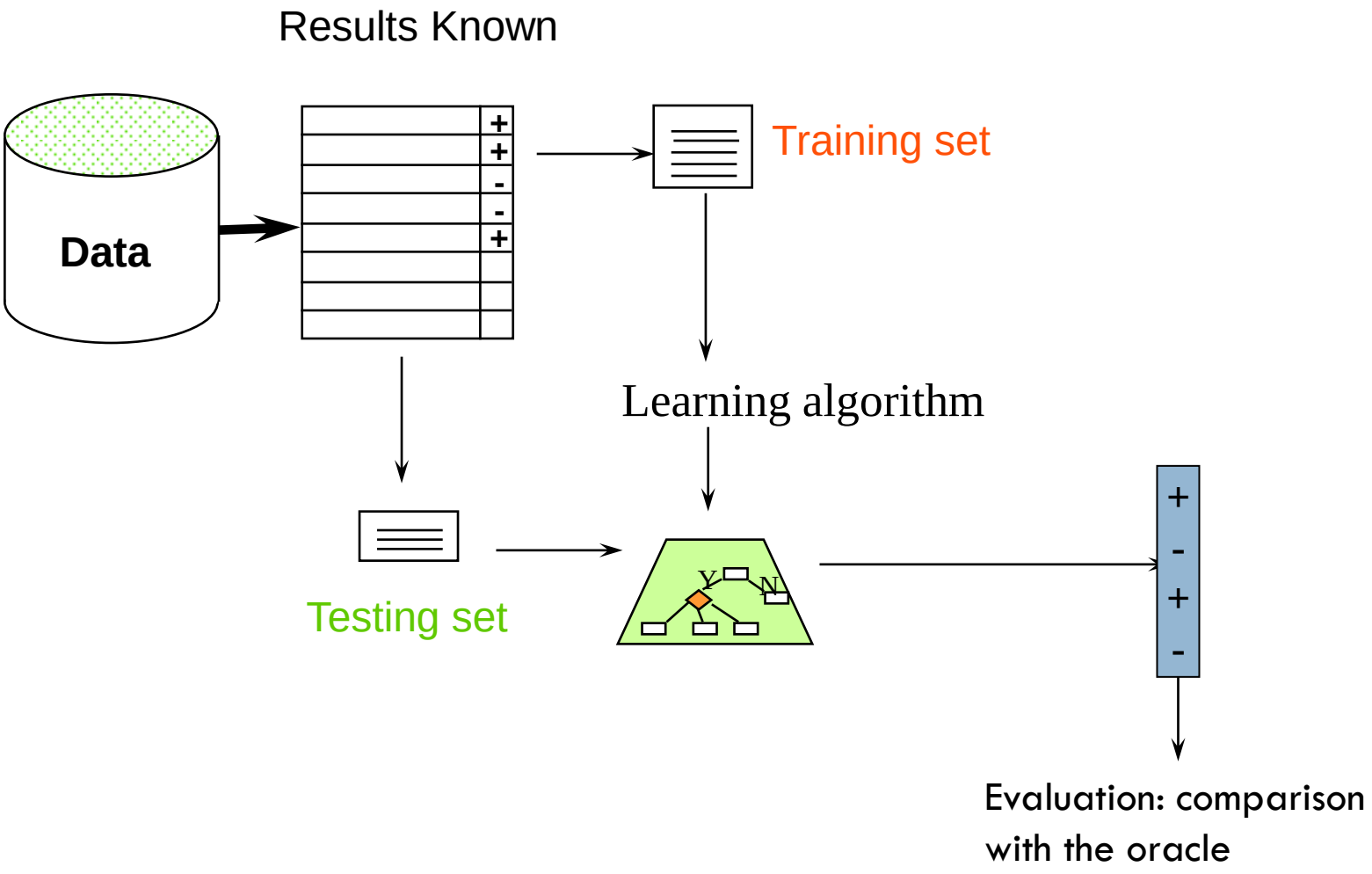


For instance 70% in the training set and 30% in the test set

Step 2: learning phase



Step 3: testing the model



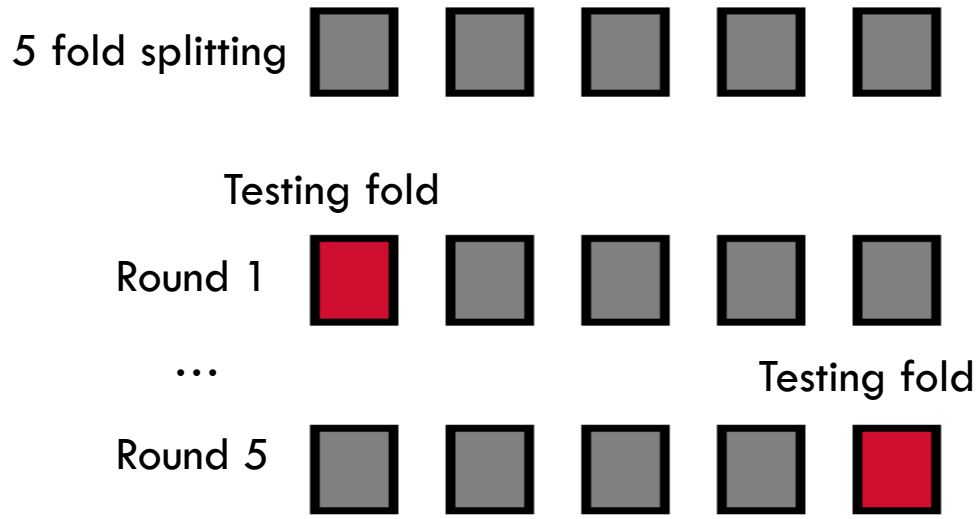
Evaluation on Few Data

- When data is scarce (totally or for a single class), a single evaluation process could not be enough representative
  - The testing set could contain too few instances to produce a reliable result

- The evaluation process must be repeated with different splitting

N-Fold Cross Validation

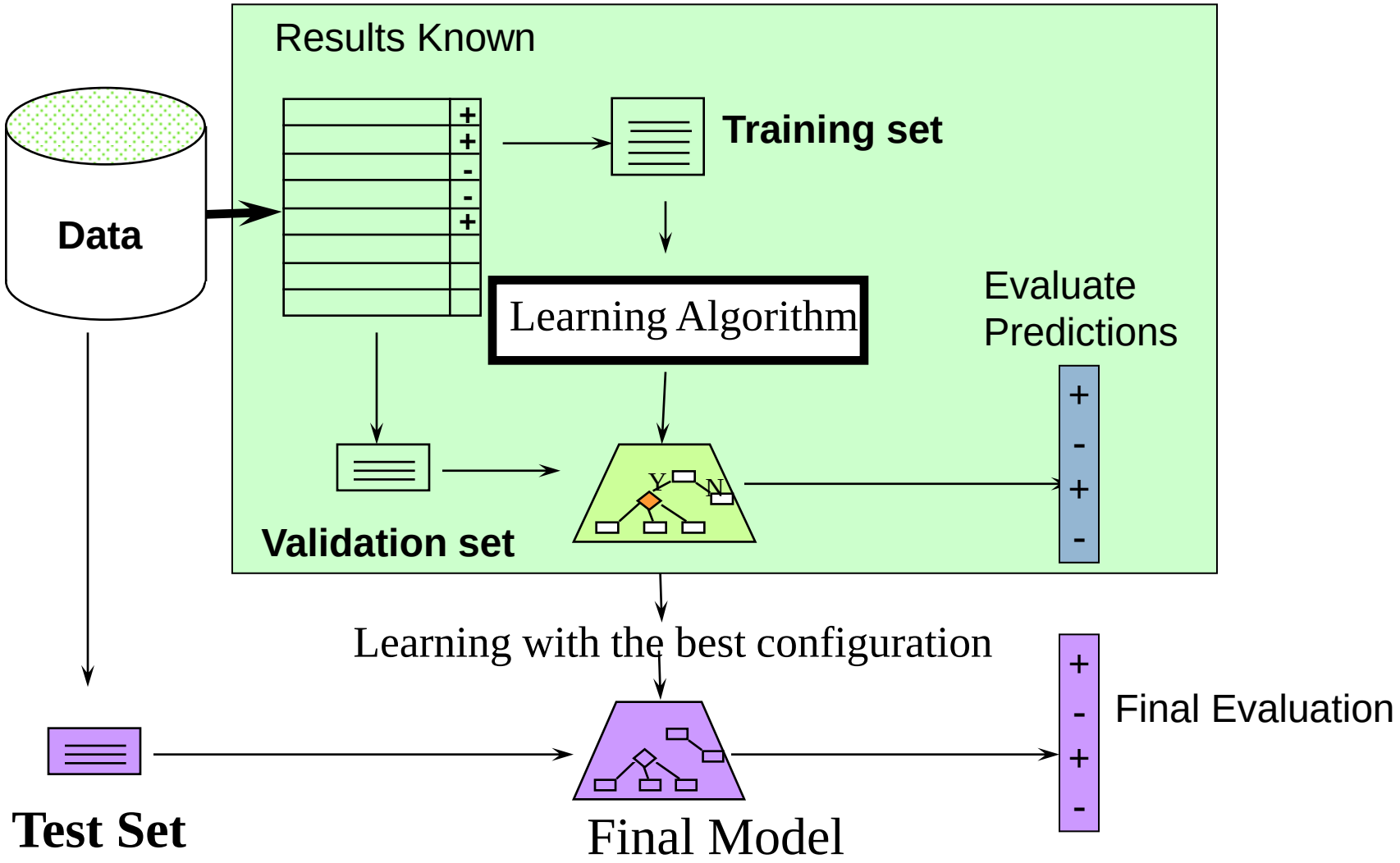
- Data is split into *n* subsets of equal size
- Each subset in turn is used for testing and the remainders *n-1* for training
- The metrics estimated in each round are averaged



Tuning a Classifier

- Most of ML algorithms depends on some parameters (example *k* in KNN)
- The best configuration must be choosen after a proper tuning stage:
  - A set of configurations must be established (for instance k=1,2,5,10,15,20,30,50)
  - Each configuration must be evaluated on a validation (or tuning) set

Complete ML Process



Overview

- Performance Evaluation Metrics
  - Classifier Evaluation Metrics
  - Information Retrieval Systems Evaluation Metrics
- Tuning and Evaluation Methods
- **Error Diagnostics**

Error Diagnostics

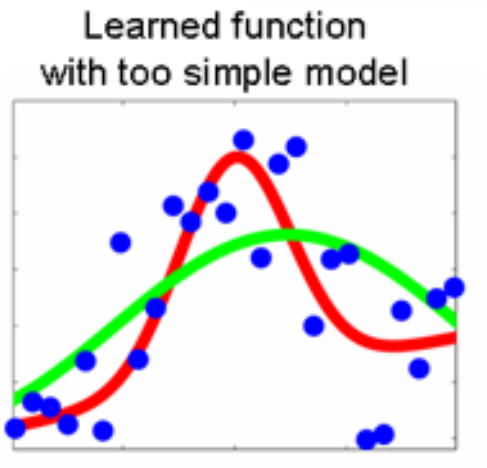
- Error Diagnostics helps in identifying what problem is affecting an ML systems that performs poorly
- Understanding the problem is useful in coming up with promising solutions for improving the system

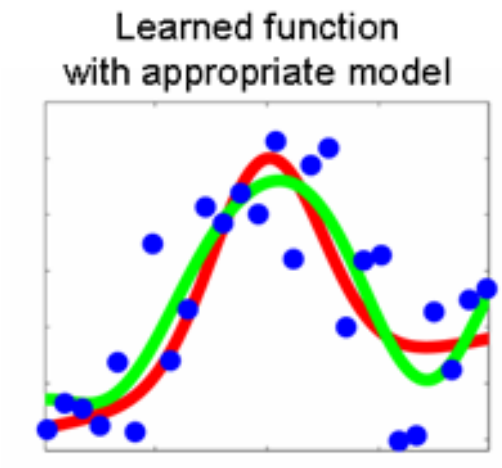
- Two opposite issues:
  - **Bias Problem**
  - **Variance Problem**

Bias Versus Variance

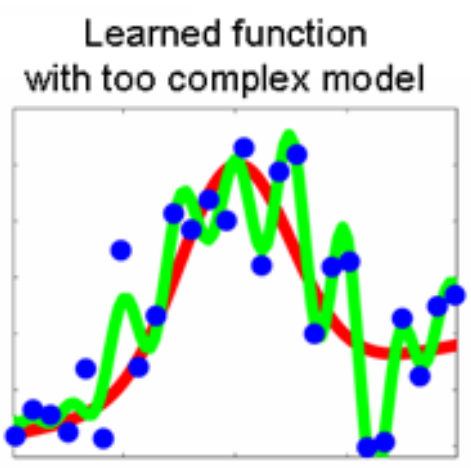
- Example in Regression

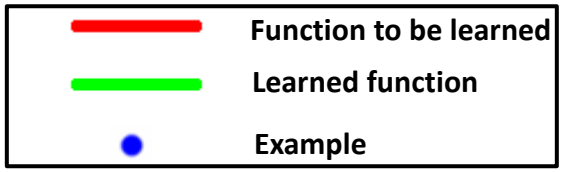
BIAS PROBLEM:





VARIANCE PROBLEM:





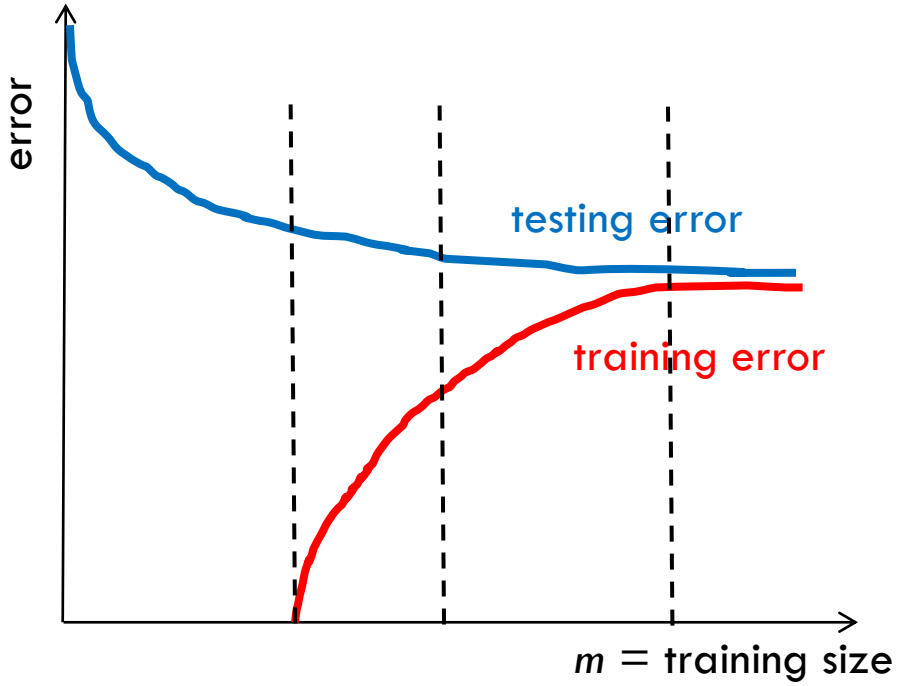
Diagnosing Bias vs Variance

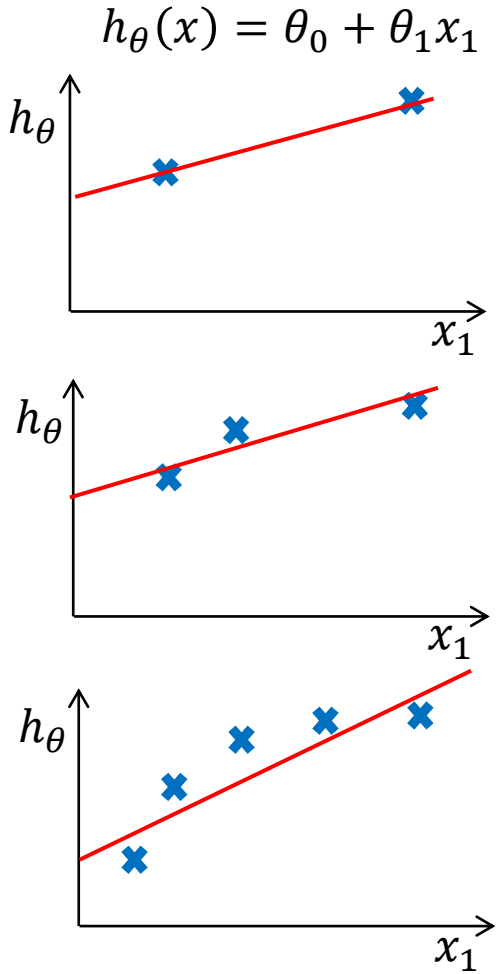
- Bias
  - *Underfitting*: the model is not enough expressive to fit the complexity of the underlying concept to be learned
  - A high error is observed both in training and testing

- Variance
  - O*verfitting*: the model perfectly fits training data but is too complex (example: an extremely deep decision tree) and does not generalize well on new data
  - A high difference between the training error and the testing error

Diagnosing High Bias via Learning Curve

Example in regression: we want to fit a 2D data distribution with a straight line



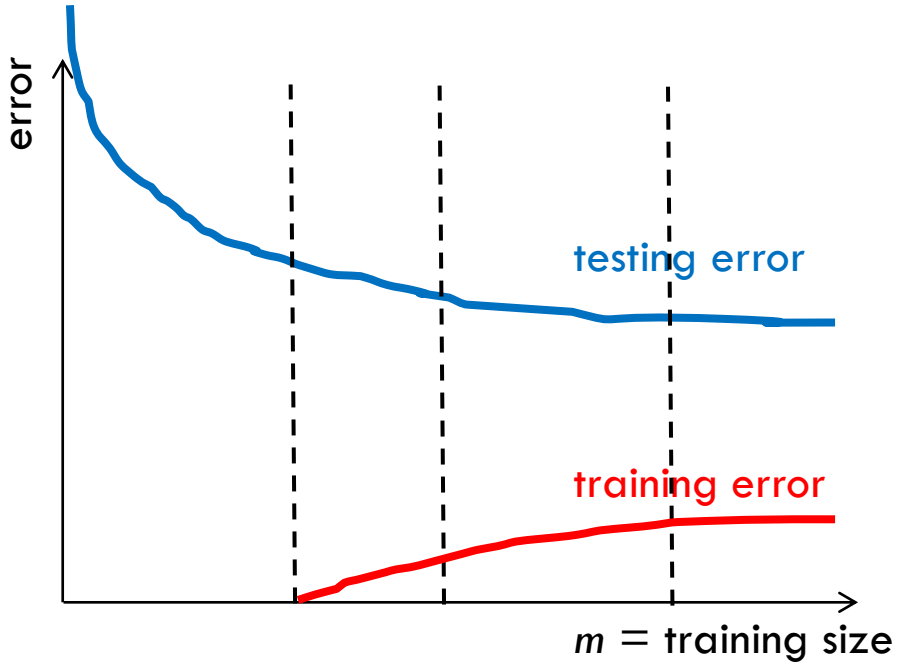


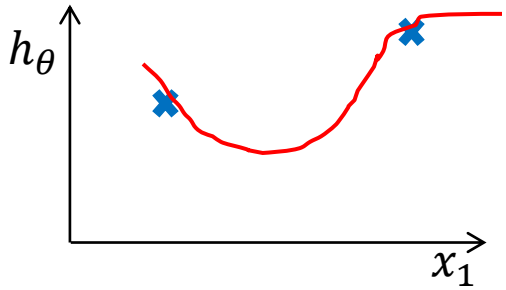
After a certain value of m, the learning process saturates and the testing error becomes similar to the training error → getting more example will not help too much

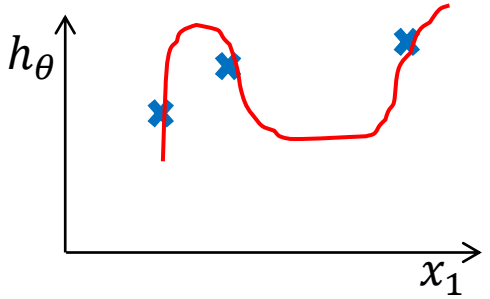
Diagnosing High Variance via Learning Curve

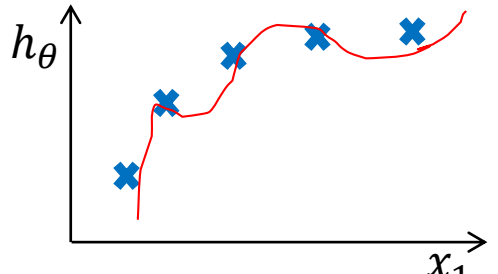
Example in regression: we want to fit a 2D data distribution with 10-th degree polynomial function

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \ldots + \theta_{10} {x_1}^{10}$$









A large gap between the training error and the testing error is observed. The saturation point is still not reached → new examples should help

Solutions for Bias and Variance

- Bias
  - Add new informative features
  - Use a more sophisticated algorithm (or the same algorithm with a more complex parameterization)
- Variance
  - Add new examples
  - Remove irrelevant and noisy features
  - Use a less complicated parameterization (example simpler polynomial function in regression)

Summary

- The effectiveness of ML or IR systems can be assessed with different evaluation metrics
  - we saw just the most popular, but a lot of other metrics exist!!!
- A reliable evaluation should follow some guideline
- An error diagnostics is useful for understanding how improving the system performance