Social Media Computing

Lecture 5: Source Fusion and Evaluation

Lecturer: Aleksandr Farseev

E-mail: farseev@u.nus.edu

Slides: http://farseev.com/ainlfruct.html

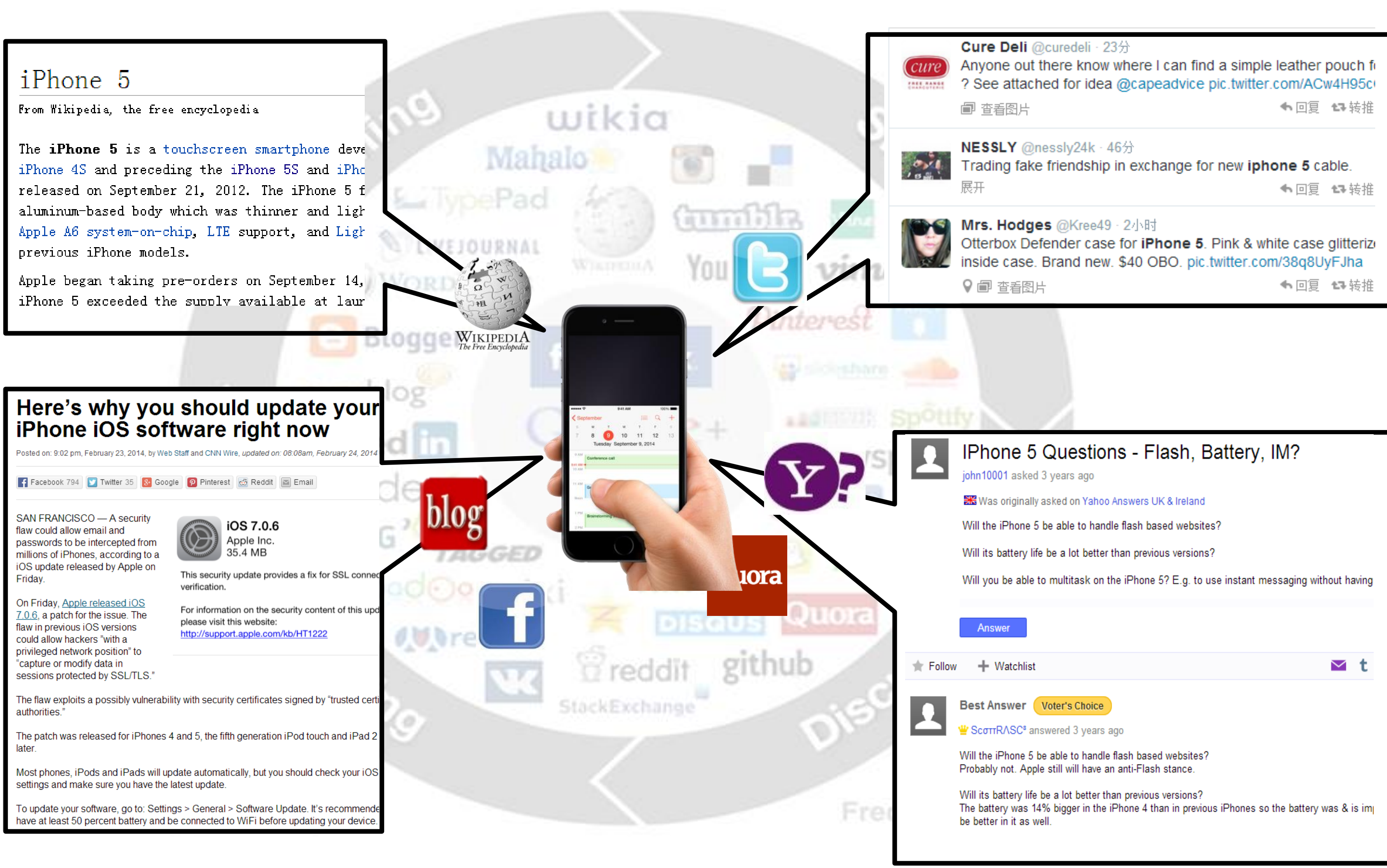
References

- Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *ICML* (Vol. 96, pp. 148-156).
- Kuncheva, L. I. (2004). Combining Pattern Classifiers. Wiley Interscience,.

Contents

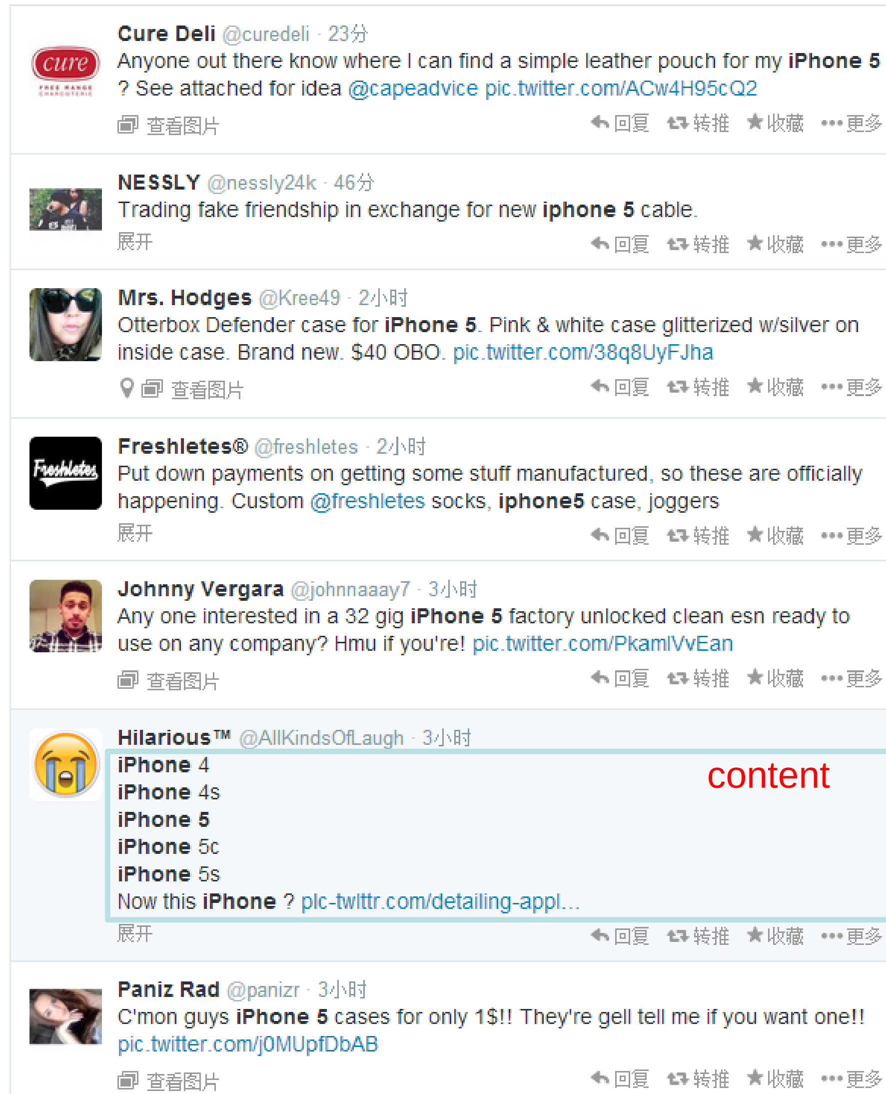
- Multi-source heterogeneous data
- Data Fusion Techniques
- Evaluation Measures
- Summary

Knowledge in Social Media Content



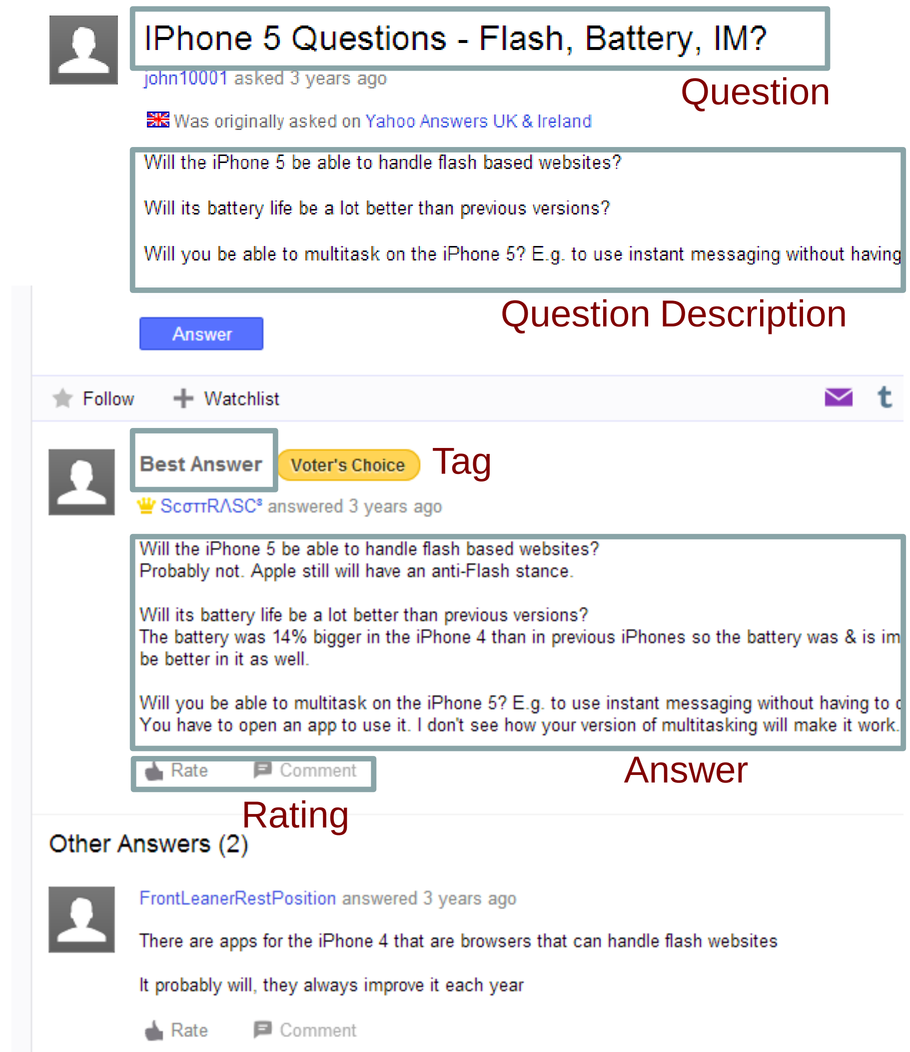
Tweet

- **Features**
  - Short contents (<140w)
  - Unstructured
    - Casually written
  - Social
    - re-tweet, @people
    - follower/followee
- **Sites**



Community QA

- **Features**
  - Focused contents
  - Semi-structured
    - Question & Answer
    - Rating, tag, category
  - Interactive
- **Sites**



Blog

- **Features**
  - Rich contents
  - Simple structure
    - Title & Content
  - Authoritative

- **Sites**

Title

Conten
t

Here's why you should update your iPhone iOS software right now

Posted on: 9:02 pm, February 23, 2014, by Web Staff and CNN Wire, *updated on: 08:08am, February 24, 2014*

Facebook 794 | Twitter 35 | Google | Pinterest | Reddit | Email

SAN FRANCISCO — A security flaw could allow email and passwords to be intercepted from millions of iPhones, according to a iOS update released by Apple on Friday.

On Friday, Apple released iOS 7.0.6, a patch for the issue. The flaw in previous iOS versions could allow hackers "with a privileged network position" to "capture or modify data in sessions protected by SSL/TLS."

iOS 7.0.6
Apple Inc.
35.4 MB

This security update provides a fix for SSL connection verification.

For information on the security content of this update, please visit this website: http://support.apple.com/kb/HT1222

The flaw exploits a possibly vulnerability with security certificates signed by "trusted certificate authorities."

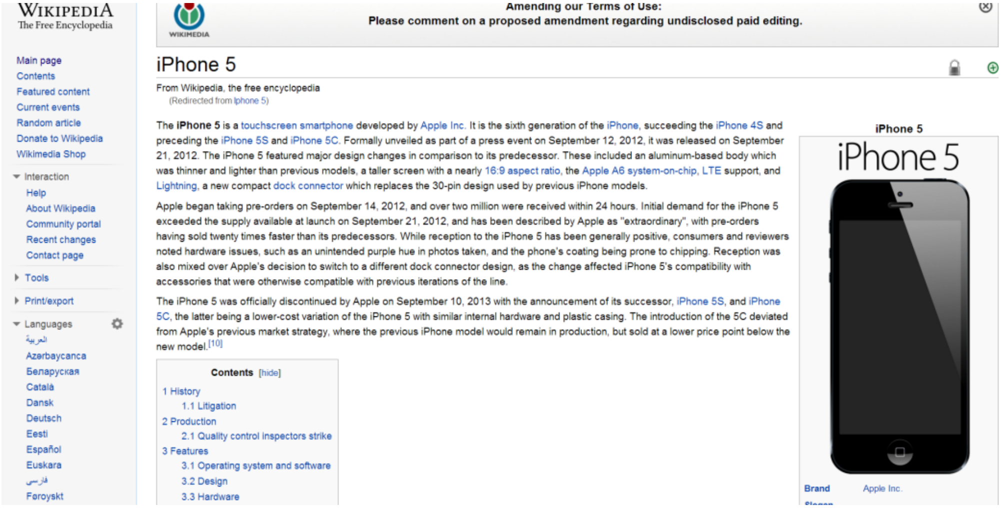
The patch was released for iPhones 4 and 5, the fifth generation iPod touch and iPad 2 and later.

Most phones, iPods and iPads will update automatically, but you should check your iOS 7 settings and make sure you have the latest update.

To update your software, go to: Settings > General > Software Update. It's recommended you have at least 50 percent battery and be connected to WiFi before updating your device.

Online Encyclopedia

- **Features**
  - High-quality contents
  - Established topics
  - Very limited data size
  - Structure
    - Infobox (Wikipedia)
    - Fact entry (Freebase)
- **Sites**



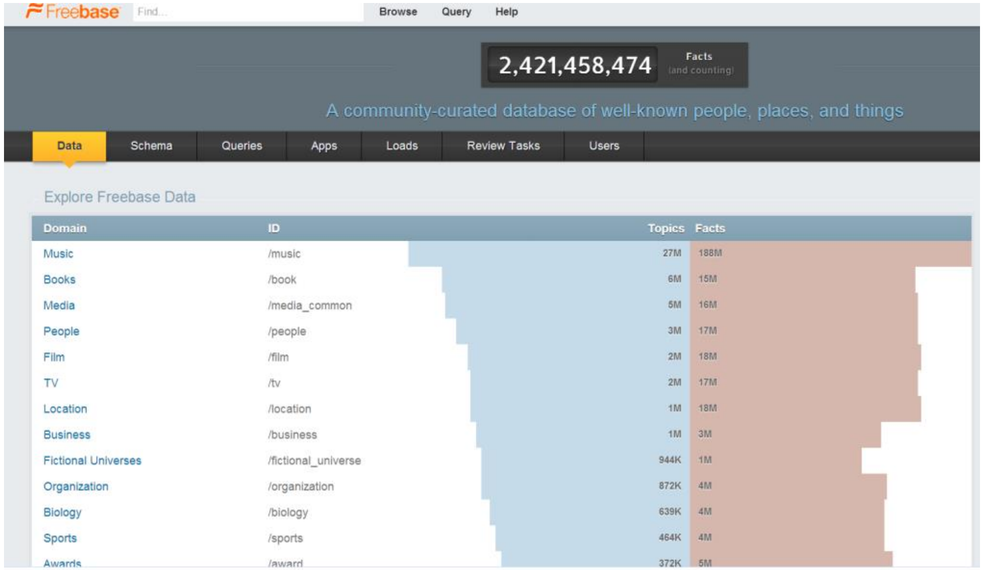


Image Sharing Services

- **Features**
  - Color-based features
  - SIFT
  - Visual concepts distribution
  - Color moments
  - Edge distribution
  - Deep features (DNN)
- **Sites**

Location-Based Social Networks

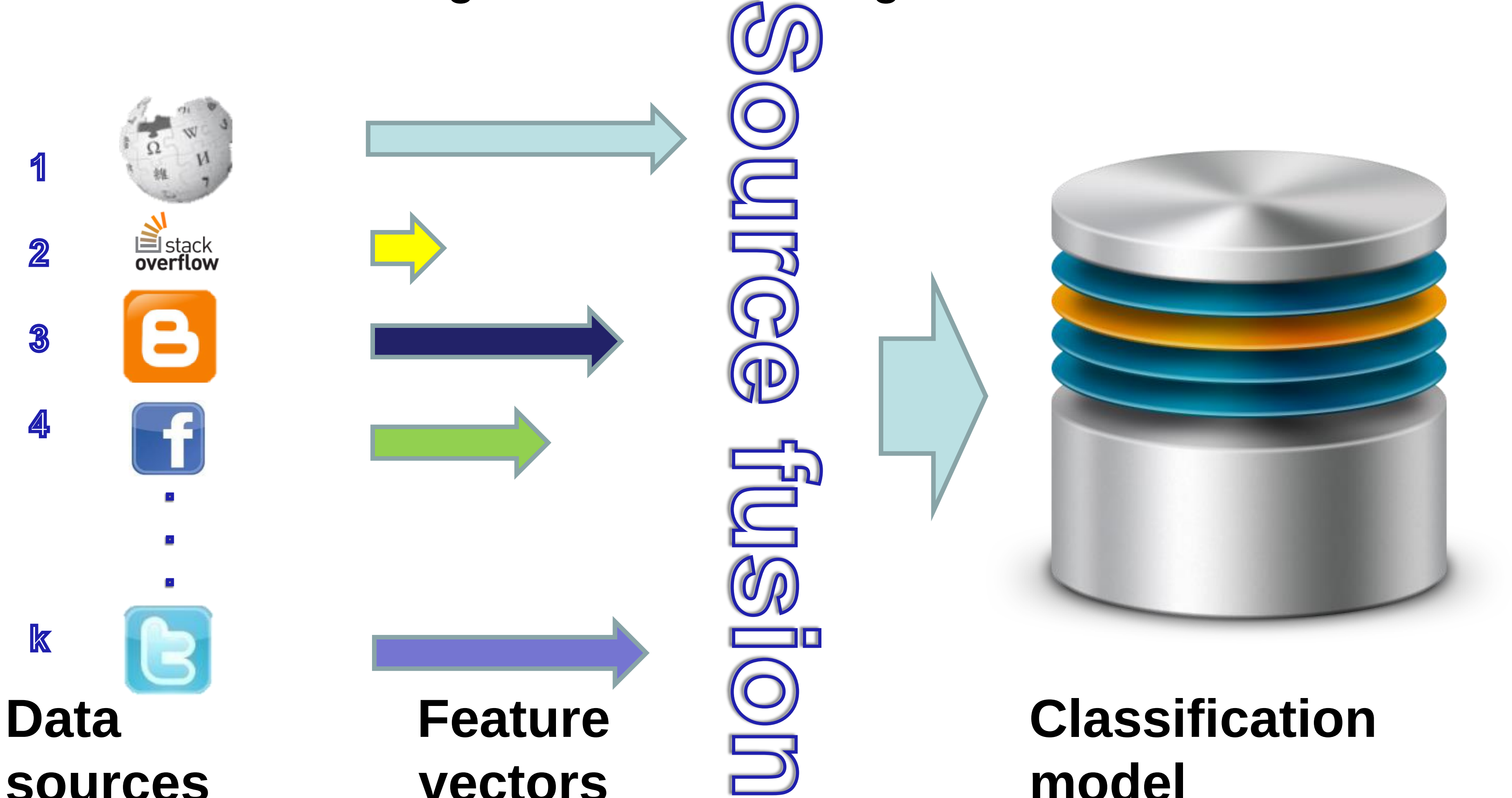
- **Features**
  - Venue Semantics
  - Mobility features (movement patterns, areas of interest)
  - Temporal features
- **Source**

Sensor Data

- **Features**
  - Frequency domain features
  - Statistics feature
  - Activity semantics
- **Source – Fitness Pal**

Source fusion

- Given a set of k data sources, the role of source fusion is to combine these sources in one model to solve a classification, regression or ranking task.



Contents

- Multi-source heterogeneous data
- Data Fusion Techniques
  - Early source fusion strategy
  - Late source fusion strategy
- Evaluation Measures
- Summary

Early source fusion strategy

- Feature vectors from each of k sources are concatenated into one feature vector; and used for model training

$\{S_{1,1}, S_{1,2}, \dots, S_{1,f}\}$

$\{S_{2,1}, S_{2,2}, \dots, S_{2,g}\}$

$\{S_{k,1}, S_{k,2}, \dots, S_{k,h}\}$

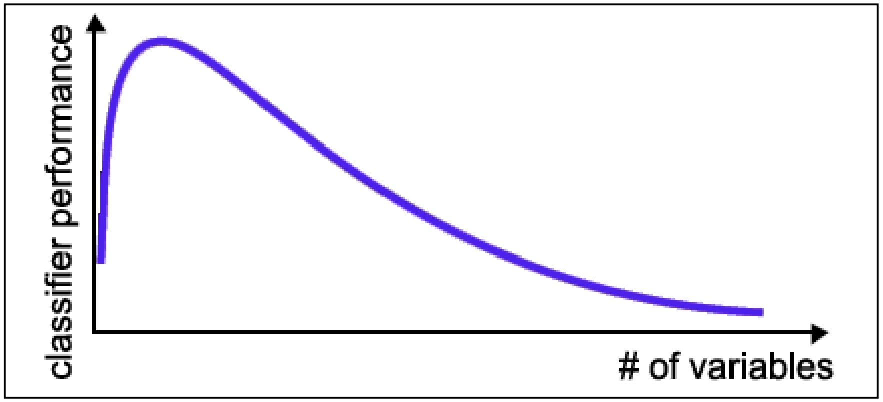
$\{S_{1,1}, S_{1,2}, \dots, S_{1,f}, \; S_{2,1}, S_{2,2}, \dots, S_{2,g}, \; S_{k,1}, S_{k,2}, \dots, S_{k,h}\}$

Number of features is too large!

Curse of dimensionality

- The required number of samples (to achieve the same accuracy) grows exponentially with the number of variables!
- In practice: number of training examples is fixed!

=> the classifier's performance usually will degrade with a large number of features!



In many cases the information that is lost by discarding variables is made up for by a more accurate mapping/ sampling in the lower-dimensional space !

Solution to: Curse of dimensionality problem

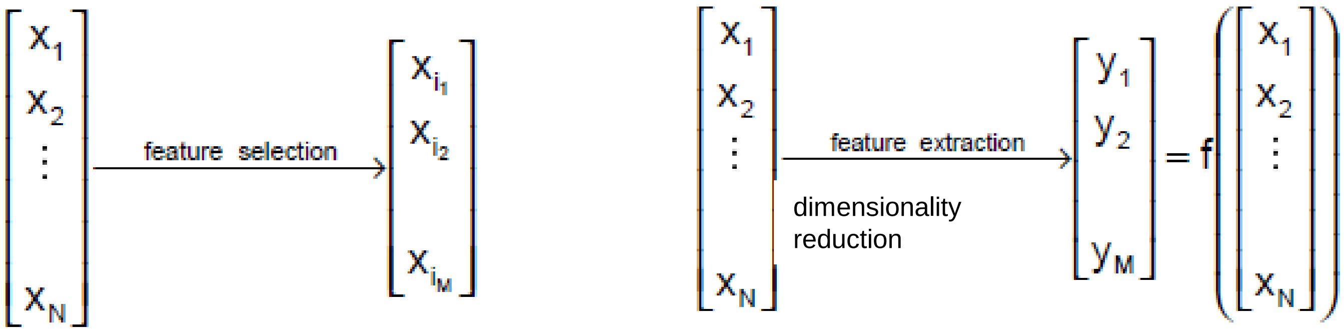
$$\{S_{1,1}, S_{1,2}, \ldots, S_{1,f}, S_{2,1}, S_{2,2}, \ldots, S_{2,g}, S_{k,1}, S_{k,2}, \ldots, S_{k,h}\}$$

Keep just useful for certain problems features

$$\{S_{1,1}, S_{1,2}, \ldots, \cancel{S_{1,f}}, S_{2,1}, \cancel{S_{2,2}}, \ldots, S_{2,g}, \cancel{S_{k,1}}, \cancel{S_{k,2}}, \ldots, S_{k,h}\}$$

Feature Selection

- Given a set of n features, the role of feature selection is to select a subset of d features ($d < n$) in order to minimize the classification error.



- Many techniques have been introduced, including:
  - Feature selection methods, such as *correlation based*
  - Dimensionality reduction methods (e.g., PCA or LDA) based on feature projection to new space
- Train Classifier based on feature set

Contents

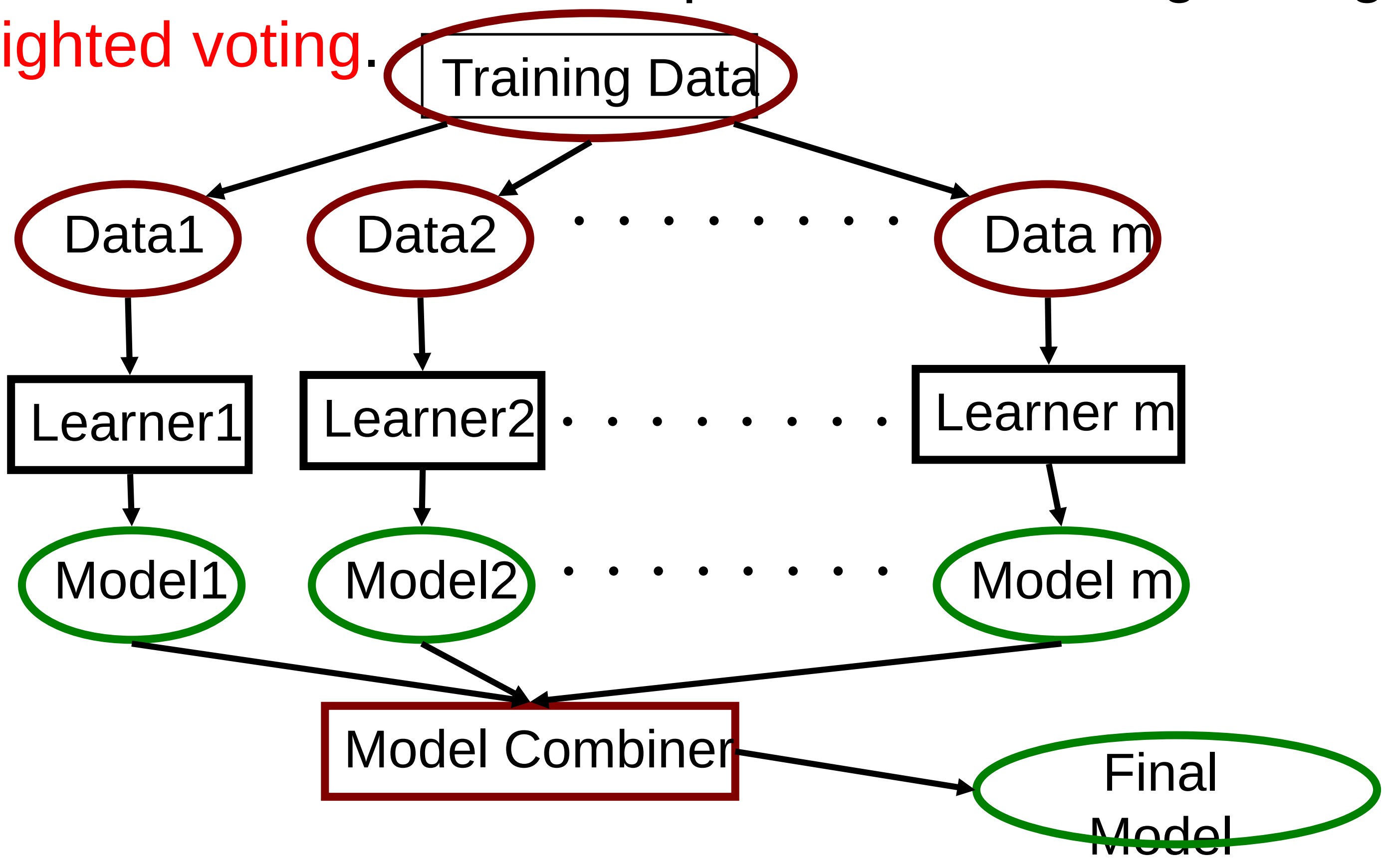
- Multi-source heterogeneous data
- Data Fusion Techniques
  - Early source fusion strategy
  - Late source fusion strategy
- Summary

Ensemble Learning

- So far, we introduce learning methods that learn a single hypothesis, chosen from a hypothesis space that is used to make predictions.
- **Ensemble learning** → select a collection (ensemble) of hypotheses and combine their predictions.
- Example: generate 100 different decision trees from the same or different training set and have them vote on the best classification for a new example.
- Key motivation: reduce error rate.
  Hope is that it will be much more unlikely that the ensemble of methods will misclassify an example.

General Learning Ensembles

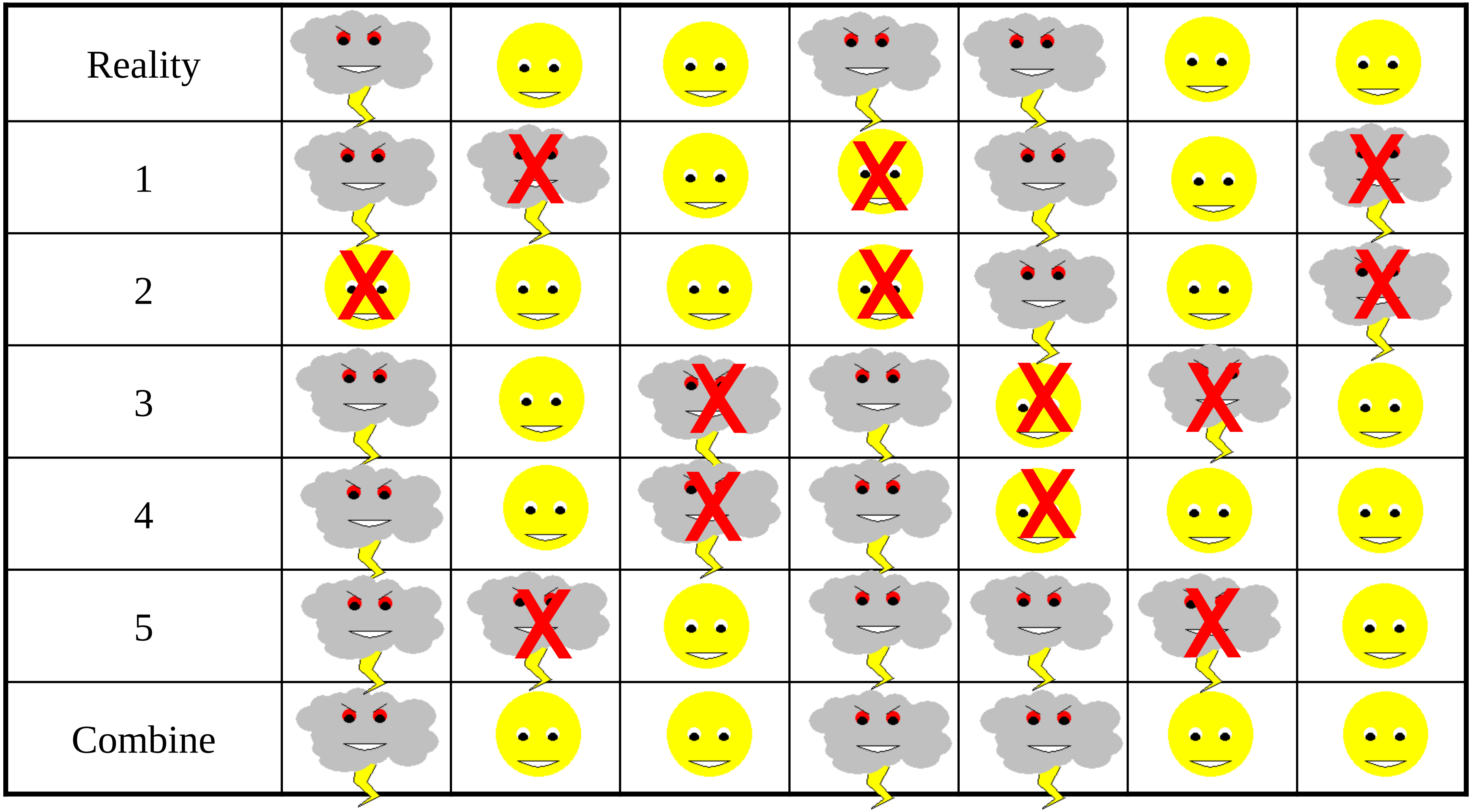
- Learn multiple alternative definitions of a concept using different training data or different learning algorithms.
- Combine decisions of multiple definitions, e.g. using weighted voting.



Value of Ensembles

- "No Free Lunch" Theorem
  - No single algorithm wins all the time!
- When combing multiple independent and diverse decisions each of which is at least more accurate than random guessing, then random errors may cancel each other out, reinforcing correct decisions

Example: Weather Forecast



Intuitions

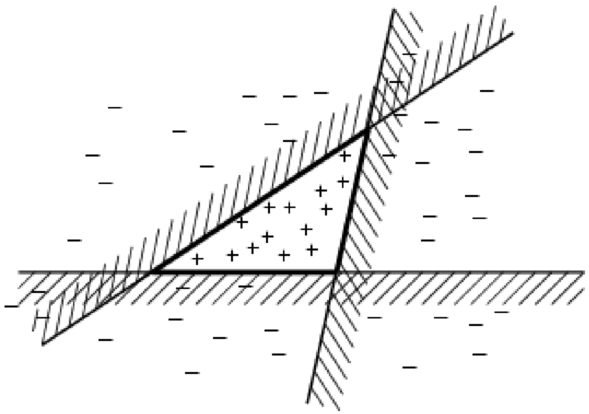
- Majority vote
- Suppose we have 5 completely independent classifiers, then based on binomial distribution theory, we have…
  - If accuracy is 70% for each classifier:
    - $(.7^5)+5(.7^4)(.3)+10\,(.7^3)(.3^2)$
    - 83.7% majority vote accuracy
  - 101 such classifiers:
    - 99.9% majority vote accuracy
  - But if the accuracy is less than 50% for each classifier, would the above still holds?

Note: Binomial Distribution: The probability of observing *x* heads in a sample of *n* independent coin tosses, where in each toss the probability of heads is *p*, is:

$$P(X=x|p,n)=\frac{n!}{r!(n-x)!}p^x(1-p)^{n-x}$$

Ensemble Learning

- Another way of thinking about ensemble learning:
- → way of enlarging the hypothesis space, i.e., the ensemble itself is a hypothesis and the new hypothesis space is the set of all possible ensembles constructible from hypotheses of the original space.



Increasing power of ensemble learning:

- Three linear threshold hypothesis (positive examples on the non-shaded side);
- Ensemble classifies as positive for any example that are classified positively for all three;
- The resulting triangular region hypothesis is not expressible in the original hypothesis space.

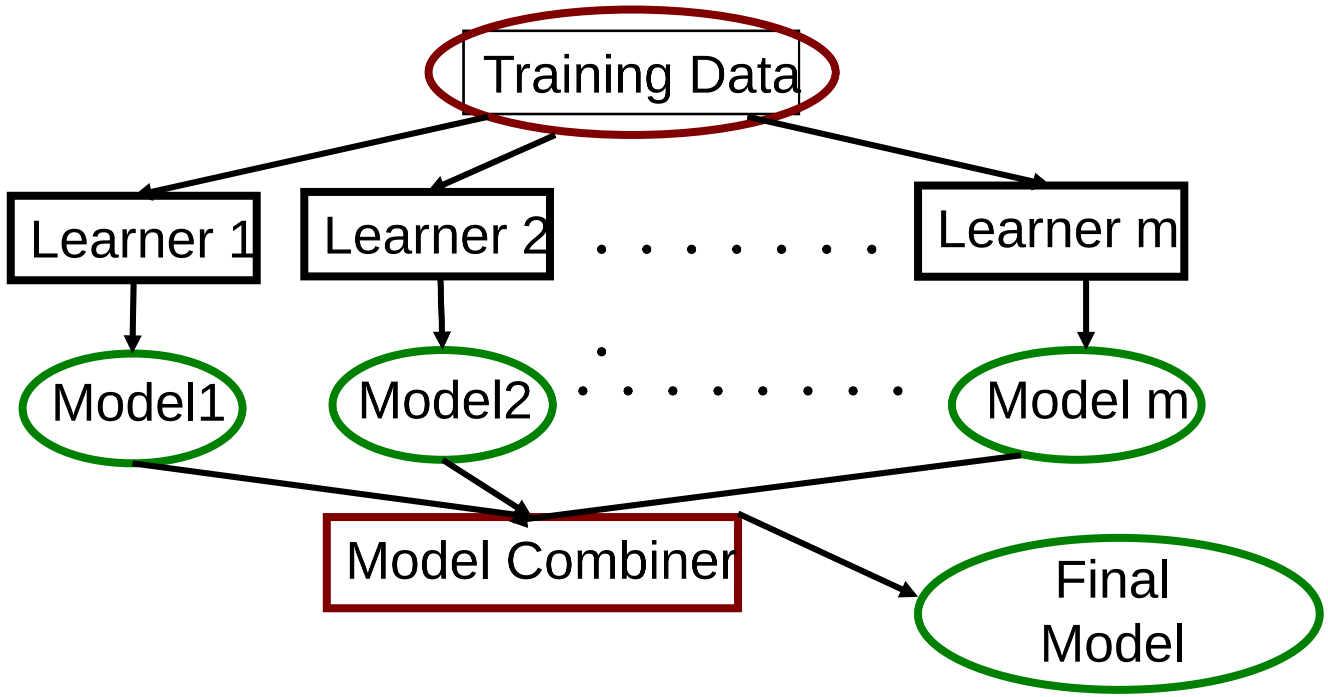
Different Learners

1) Different learning algorithms
2) Algorithms with different choice for parameters
3) Data set with different features
4) Data set = different subsets

1) Ensemble with Multiple Learning Algorithms

- Learn multiple classifiers using different learning algorithms

- Can combine decisions of multiple classifiers using:
  - Majority voting
  - Weighted voting



Model Combinations:
Majority Vote

- Assume label outputs of *m* classifiers are:
  - $[d_{i,1}, \ldots d_{i,m}]^T \in \{0,1\}_c$, l = 1, .., L; // L label classes
    where $d_{i,j}$ = 1 if Classifier $\boldsymbol{D_i}$ labels $\boldsymbol{x}$ in class $\boldsymbol{w_j}$; and 0 otherwise
- Plurality vote will result in an ensemble decision for class $\boldsymbol{w_k}$ if:

$$\sum_{i=1}^{L} d_{i,k} = \max_{j=1 \to c} \sum_{i=1}^{L} d_{i,j}$$

  - It is termed majority vote;
    coincide with the simple majority in the case of 2 classes (c=2)
  - The majority vote will give an accurate class label if at least [L/2] + 1 classifiers give correct answers.

Model Combinations: Weighted Majority Vote

- If the classifiers of ensemble are not of identical accuracy, then it is reasonable perform weighted ensemble
- The discriminant function for class $\boldsymbol{w_j}$ obtained through weighted voting is:

  $$g_j(x) = \sum_{i=1}^{L} b_i \, d_{i,j}$$

  where $\boldsymbol{b_i}$ is the coefficient of importance of classifier $\boldsymbol{D_i}$
- The resulting ensemble decision for class $\boldsymbol{w_k}$ if:

  $$\sum_{i=1}^{L} b_i \, d_{i,k} = \max_{j=1 \to c} \sum_{i=1}^{L} b_i \, d_{i,j}$$

  For convenience, we normalize the coefficients so that $\sum_{i=1}^{C} b_j = 1$
- **Theorem**: Consider an ensemble of L independent classifiers $D_1, \ldots, D_L$, with individual accuracies $p_1, \ldots, p_L$. The outputs are combined by the weighted majority vote. Then the accuracy of ensemble (p^{w}_{maj}) is maximized by assigning weights: $b_i = \log \frac{p_i}{1 - p_i}$

2) Homogenous Ensembles

- Use a single, arbitrary learning algorithm but manipulate training data to make it learn multiple models.
  - $\text{Learner1} = \text{Learner2} = \ldots = \text{Learner}_m$
  - $\text{Data1} \neq \text{Data2} \neq \ldots \neq \text{Data}_m$
- Different methods for changing training data:
  - Bagging: Resample training data
  - Boosting: Reweight training data

2a) Bagging

- Bagging is a "bootstrap" ensemble method that creates individuals for its ensemble by training each classifier on a random redistribution of the training set
  - Draw N items from *D* with replacement (means samples drawn can be repeated)

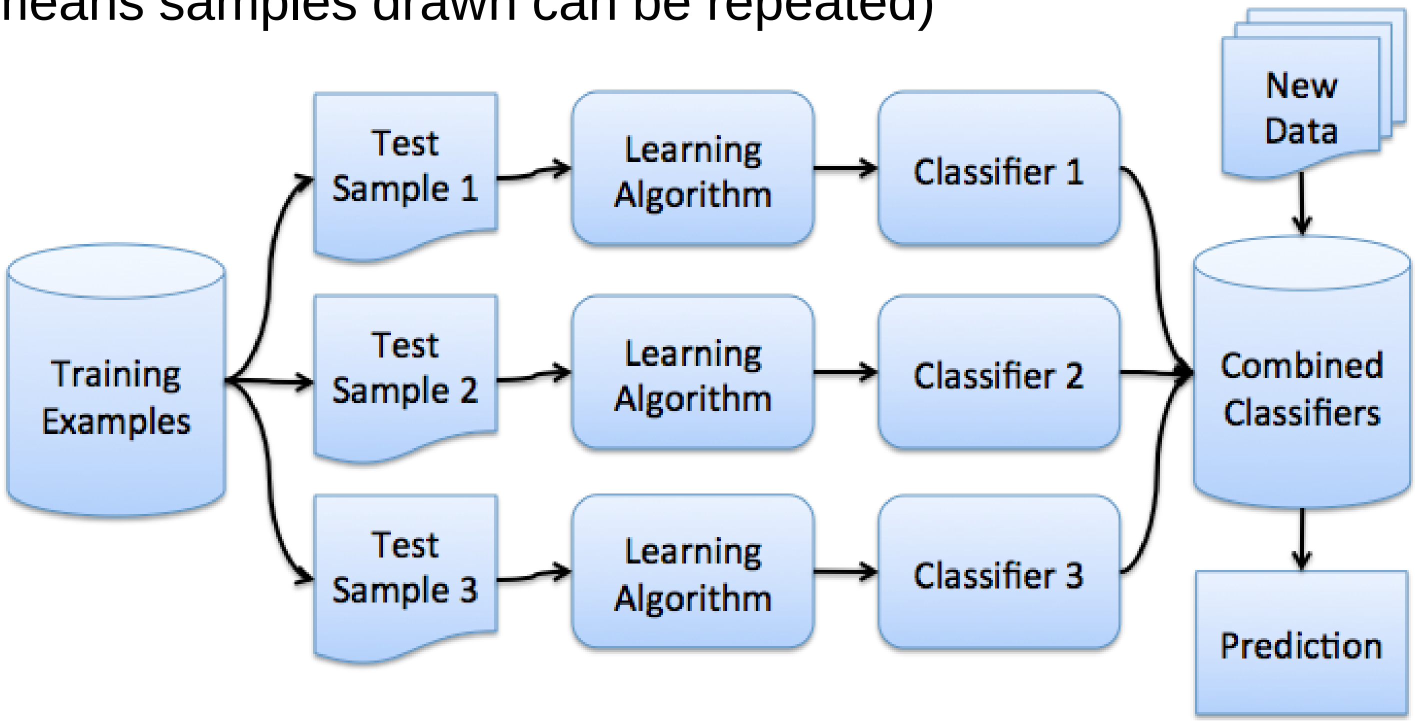


Figure taken from: http://cse-wiki.unl.edu/wiki/index.php/Bagging_and_Boosting

Bagging - Aggregate Bootstrapping

- Create ensembles by "*bootstrap aggregating*", i.e., repeatedly randomly resampling the training data (Brieman, 1996).
- Given a standard training set D of size n
- For i = 1 .. M
  - Draw a sample of size $n^* < n$ from D uniformly and with replacement
  - Learn classifier C_i
- Final classifier is a vote of C_1 .. C_M
  - By simple majority votes
- Increases classifier stability/reduces variance

Properties of Bagging

- Breiman (1996) showed that Bagging is effective on ”unstable" learning algorithms, where small changes in the training set result in large changes in predictions.
  - Examples of unstable learners include decision trees and neural networks)
- It decreases the error by decreasing the variance in the results due to *unstable learners*
- It may slightly degrade the performance of stable learning algorithms, such as kNN.

2b) Boosting

- Weak Learner: only needs to generate a hypothesis with a training accuracy greater than 0.5, i.e., < 50% error over any distribution
- Learners
  - Strong learners are very difficult to construct
  - Constructing weaker Learners is relatively easy
- Question: Can a set of weak learners create a single strong learner?

YES ☺

Boost weak classifiers to a strong learner

Strong and Weak Learners

- Strong Learner → Objective of machine learning
  - Take labeled data for training
  - Produce a classifier which can be *arbitrarily accurate*
- Weak Learner
  - Take labeled data for training
  - Produce a classifier which is more accurate than random guessing

Boosting

- Originally developed by computational learning theorists to guarantee performance improvements on fitting training data for a *weak learner* that only needs to generate a hypothesis with a training accuracy greater than 0.5 (Schapire, 1990).
- Revised to be a practical algorithm, AdaBoost, for building ensembles that empirically improves generalization performance (Freund & Schapire, 1996).
- Key Insights
  - Instead of sampling (as in bagging), re-weigh the examples.
  - Examples are given weights. At each iteration, a new hypothesis is learned (weak learner) and the examples are reweighted to focus on examples that the most recently learned classifier got wrong.
  - Final classification based on weighted vote of weak classifiers

AdaBoost: High Level Algorithm

Algorithm AdaBoost.M1

Input: sequence of m examples $\langle(x_1, y_1), \ldots, (x_m, y_m)\rangle$ with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Initialize $D_1(i) = 1/m$ for all i.

Construct weak classifiers

Do for $t = 1, 2, \ldots, T$

1. Call **WeakLearn**, providing it with the distribution D_t.
2. Get back a hypothesis $h_t : X \to Y$.
3. Calculate the error of h_t: $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$. If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
4. Set $\beta_t = \epsilon_t/(1 - \epsilon_t)$.
5. Update distribution D_t: $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{cases}$

   where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis: $h_{fin}(x) = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}$.

Combine weak classifiers

- Many variants depending on how to set the weights and how to combine the hypotheses.

Construct Weak Classifiers

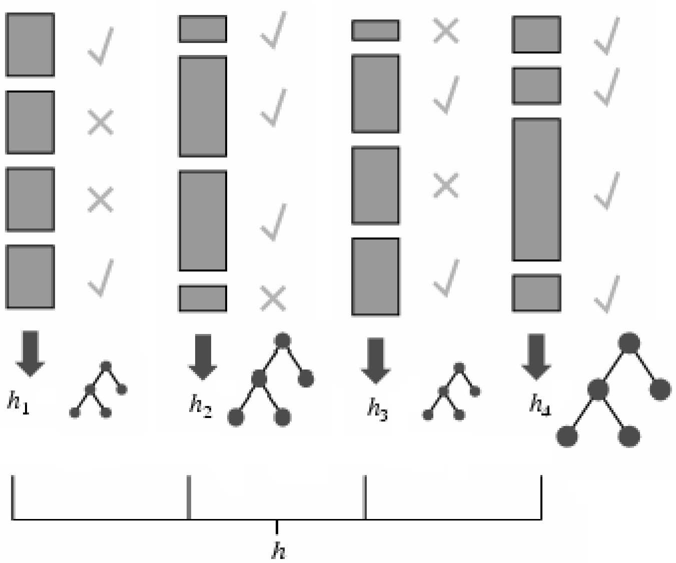
- Using Different Data Distribution
  - Start with uniform weighting
  - During each step of learning
    - Increase weights of the examples which are not correctly learned by the weak learner
    - Decrease weights of the examples which are correctly learned by the weak learner
- Idea
  - Focus on difficult examples which are not correctly classified in the previous steps

Combine Weak Classifiers

- Weighted Voting
  - Construct strong classifier by weighted voting of weak classifiers
- Idea
  - Better weak classifier gets a larger weight
  - Iteratively add weak classifiers
    - Increase accuracy of the combined classifier through minimization of a cost function

How Does Adaptive Boosting Works

- Each rectangle corresponds to an example,
- with weight proportional to its height.
- Crosses correspond to misclassified examples.
- Size of decision tree indicates the weight of that hypothesis in the final ensemble.



Performance of AdaBoost

- Learner = Hypothesis = Classifier
- Weak Learner: < 50% error over any distribution
- M: number of hypothesis in the ensemble.
- If the input learning is a Weak Learner, then AdaBoost will return a hypothesis that classifies the training data perfectly for a large enough M,
- Boosting the accuracy of the original learning algorithm on the training data.
- Strong Classifier: thresholded linear combination of weak learner outputs.

2c) Random Forest

- Ensemble consisting of a bagging of un-pruned decision tree learners with a randomized selection of features at each split.
- Grow many trees on datasets sampled from the original dataset with replacement (a bootstrap sample).
    - Draw K bootstrap samples of a fixed size
    - Grow a DT, randomly sampling a few attributes/dimensions to split on at each internal node
- Average the predictions of the trees for a new query (or take majority vote)
- Random Forests are state of the art classifiers!

Randomness in Random Forests

- Introduce two sources of randomness: "Bagging" and "Random input vectors"
  - Each tree is grown using a bootstrap sample of training data
  - At each node, best split is chosen from random sample of m_{try} variables instead of all variables

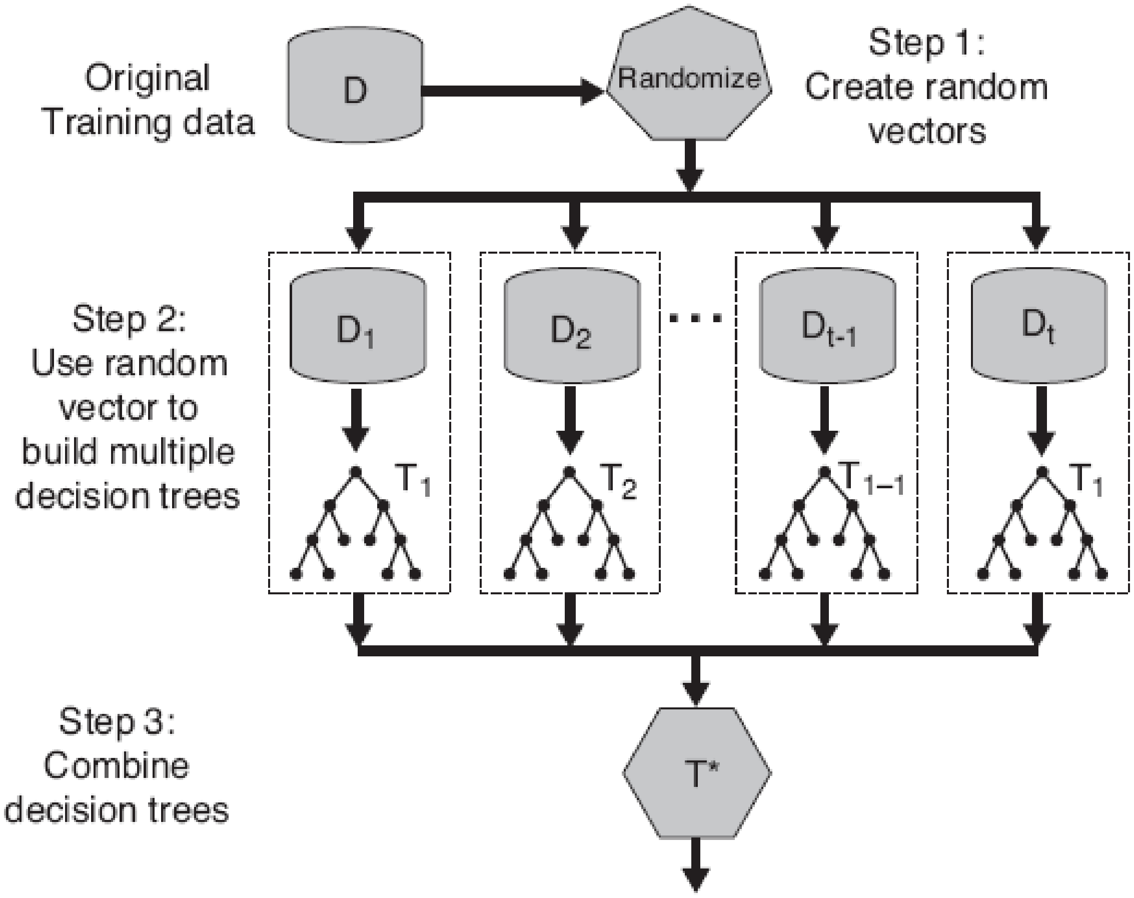


Figure 5.40. Random forests.

Random Forest: practical consideration

- Splits are chosen according to a purity measure:
  - E.g. squared error (regression), Gini index or deviance (classification))
- How to select N?
  - Build trees until the error no longer decreases
- How to select M?
  - Try to recommend defaults, half of them and twice of them and pick the best.

Random Forest:
Features and Advantages

The advantages of random forest are:

- It is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier.
- It runs efficiently on large databases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.

Random Forest:
Features and Advantages

- It has methods for balancing error in class population unbalanced data sets.
- Generated forests can be saved for future use on other data.
- Prototypes are computed that give information about the relation between the variables and the classification.
- It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data.
- The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection.
- It offers an experimental method for detecting variable interactions.

Random Forest:
Disadvantages

- Random forests have been observed to over-fit for some datasets with noisy classification/regression tasks.
- For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Therefore, the variable importance scores from random forest are not reliable for this type of data.

Some Issues to Consider

- Parallelism in Ensembles: Bagging is easily parallelized, while Boosting is not.
- Variants of Boosting to handle noisy data.
- How “weak” should a base-learner for Boosting be?
- Exactly how does the diversity of ensembles affect their generalization performance.
- Combining Boosting and Bagging.

Contents

- Multi-source heterogeneous data
- Data Fusion Techniques
- Evaluation Measures
- Summary

Evaluation Measures

- Importance of Evaluations
- Efficiency vs. effectiveness
  - Efficiency measured using speed and storage overhead
  - Effectiveness measured using relevance
- For both IR and TC, we have:

	Relevant	Non-Relevant
Retrieved	**a**	**b**
Missed	**c**	**d**

N=a+b+c+d ➔ total number of documents DB

Effectiveness:

- **Precision (P) = a / (a+b)**
- **Recall (R) = a / (a+c)**

Evaluation Measures -2

- It is generally more convenient to present a single number:

  $F_\beta = [(\beta^2+1) P R] / [\beta^2 P + R]$

- When both P & R have equal weights, i.e. when β= 1, we have:

  $F_1 = [2 P R] / [P + R]$

  This is popularly used in retrieval evaluations

- Results often presented as:
  - Average F_1 values
  - Tables of average precision values at standard recall intervals (of 0.1 intervals)
  - Recall-Precision graph
- Results over many collections are compared

Evaluation Measures -3

- For classification, we needs to account for skewness in data during evaluation
- Two ways to obtain an overall F_1 value:
  - o MicroF$_1$ – average of F_1 over all test documents
  - o MacroF$_1$ – average over the categories
- Characteristics of these two measures:
  - o MicroF$_1$ tends to be dominated by classifier's performance on common categories
  - o MacroF$_1$ mostly influenced by performance on rare categories (a stricter measure)

Evaluation Measures -4

- For retrieval, the total **# of relevant items is not known**, Average Precision AveP is normally used:

$$\operatorname{AveP} = \frac{\sum_{k=1}^{n} (P(k) \times \operatorname{rel}(k))}{\text{number of relevant documents}}$$

  - **where rel(*k*) =1 if image at rank *k* is relevant, zero otherwise**
  - Note that the average is over all relevant documents

- Example: If returned result is (1 means relevant, 0 irrelevant):

  1, 0, 0, 1, 1, 1

  1/1, 0, 0, 2/4, 3/5, 4/6 ←-- precision @ k

  AveP = (1 + 2/4 + 3/5 + 4/6) / 4 = 0.69

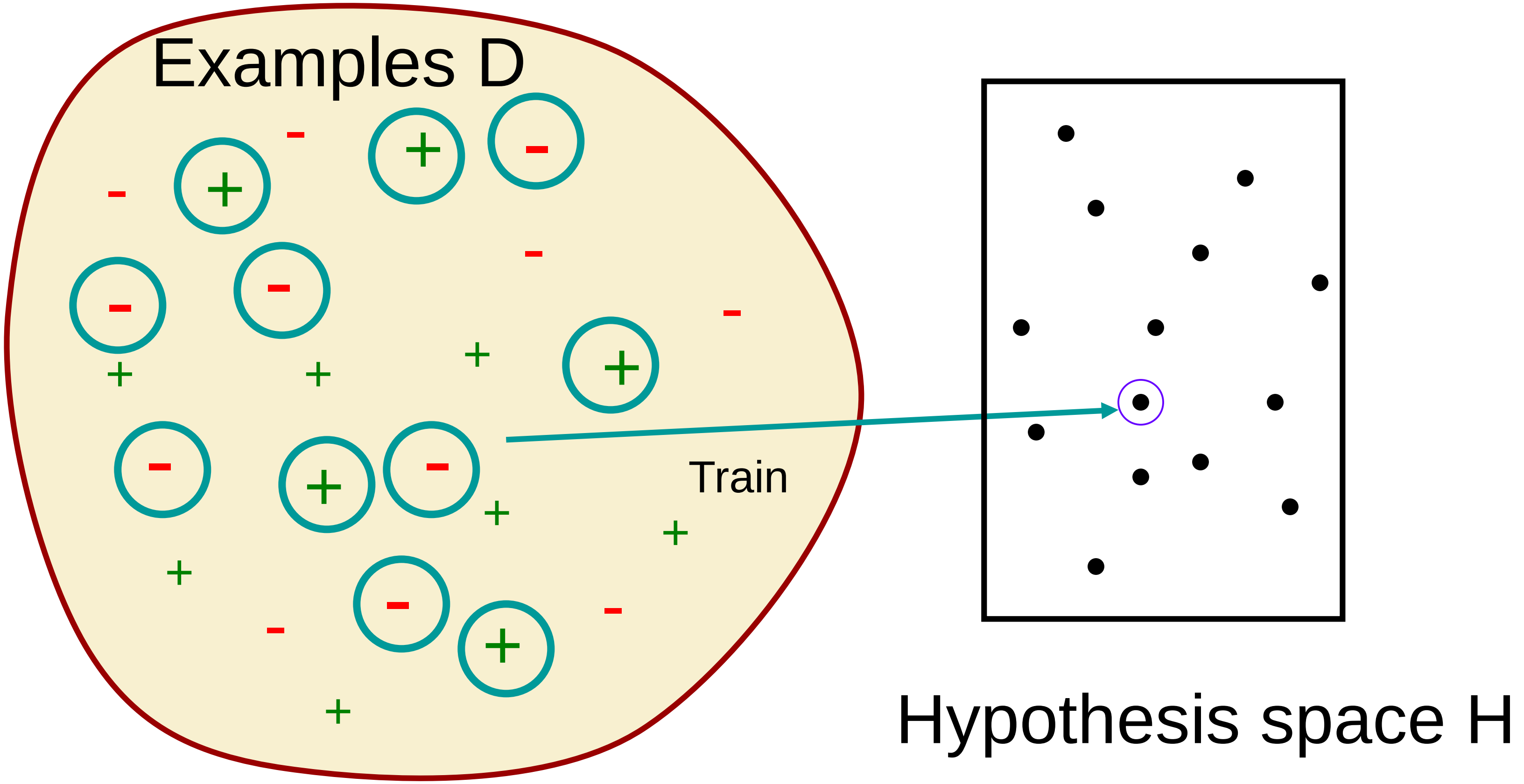
- Mean Average Precision (MAP):

$$\operatorname{MAP} = \frac{\sum_{q=1}^{Q} \operatorname{AveP(q)}}{Q}$$

  - **Average over all queries**

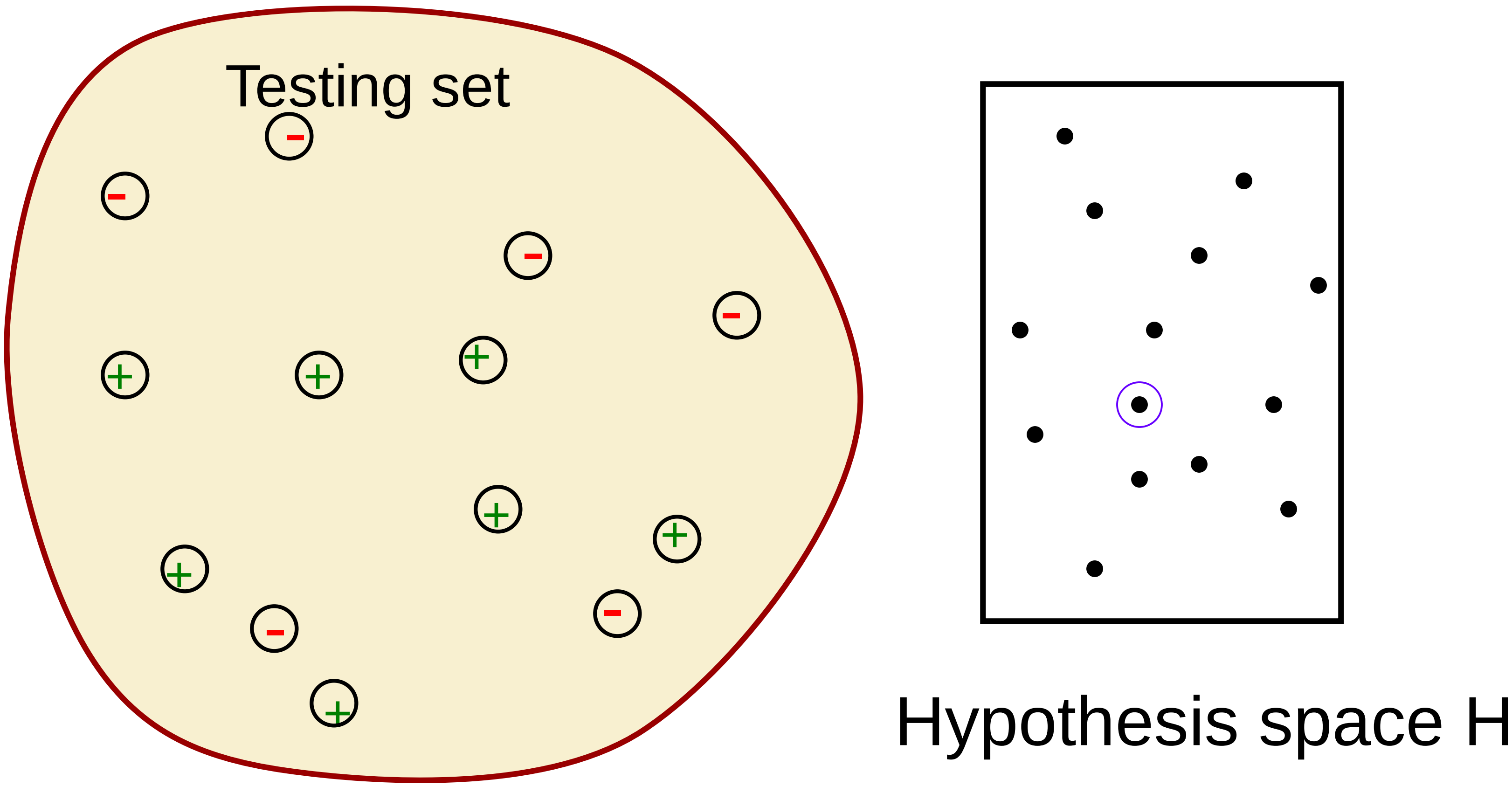
Cross-Validation -1

- Split original set of examples, train



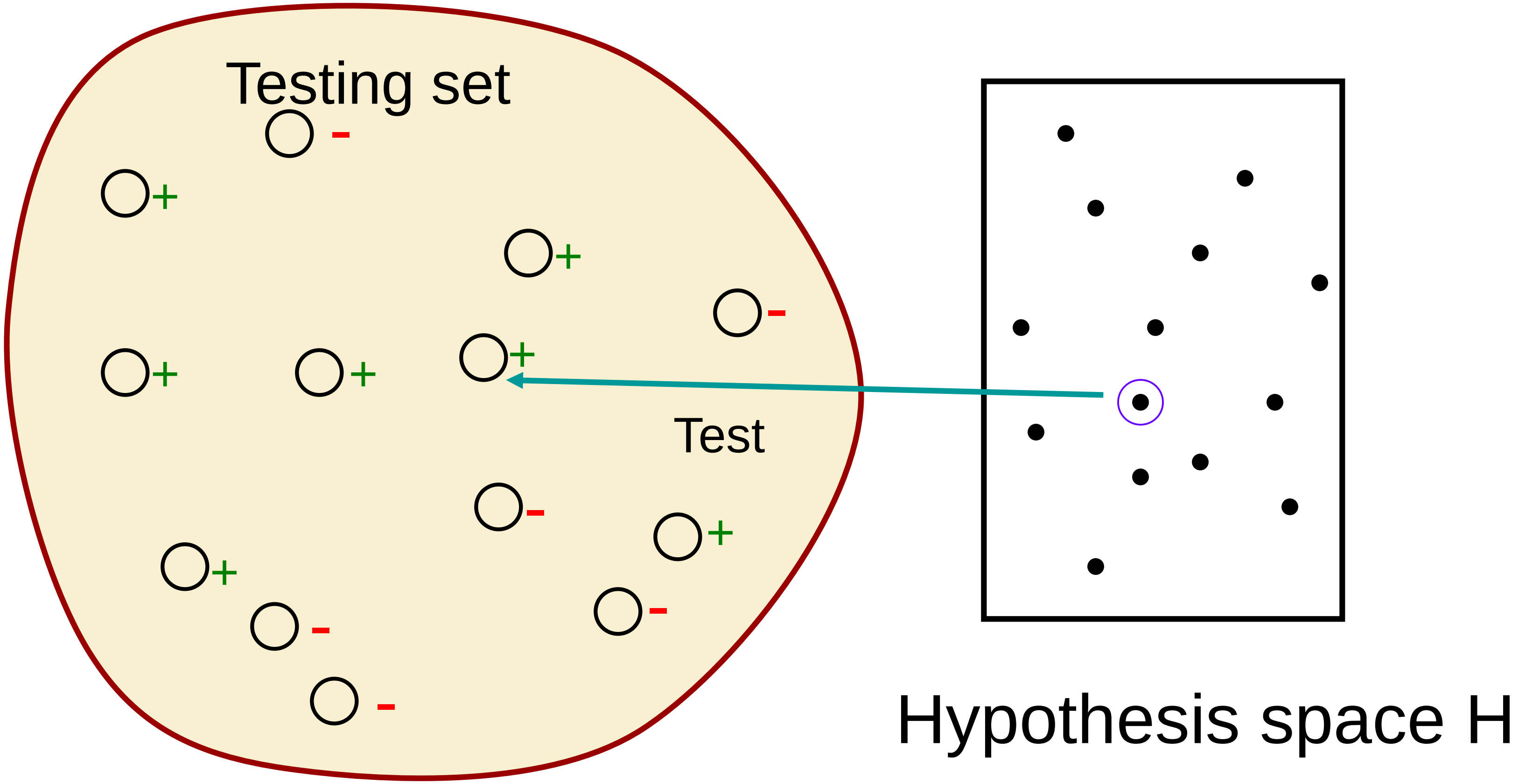
Cross-Validation -1

- Evaluate hypothesis on testing set



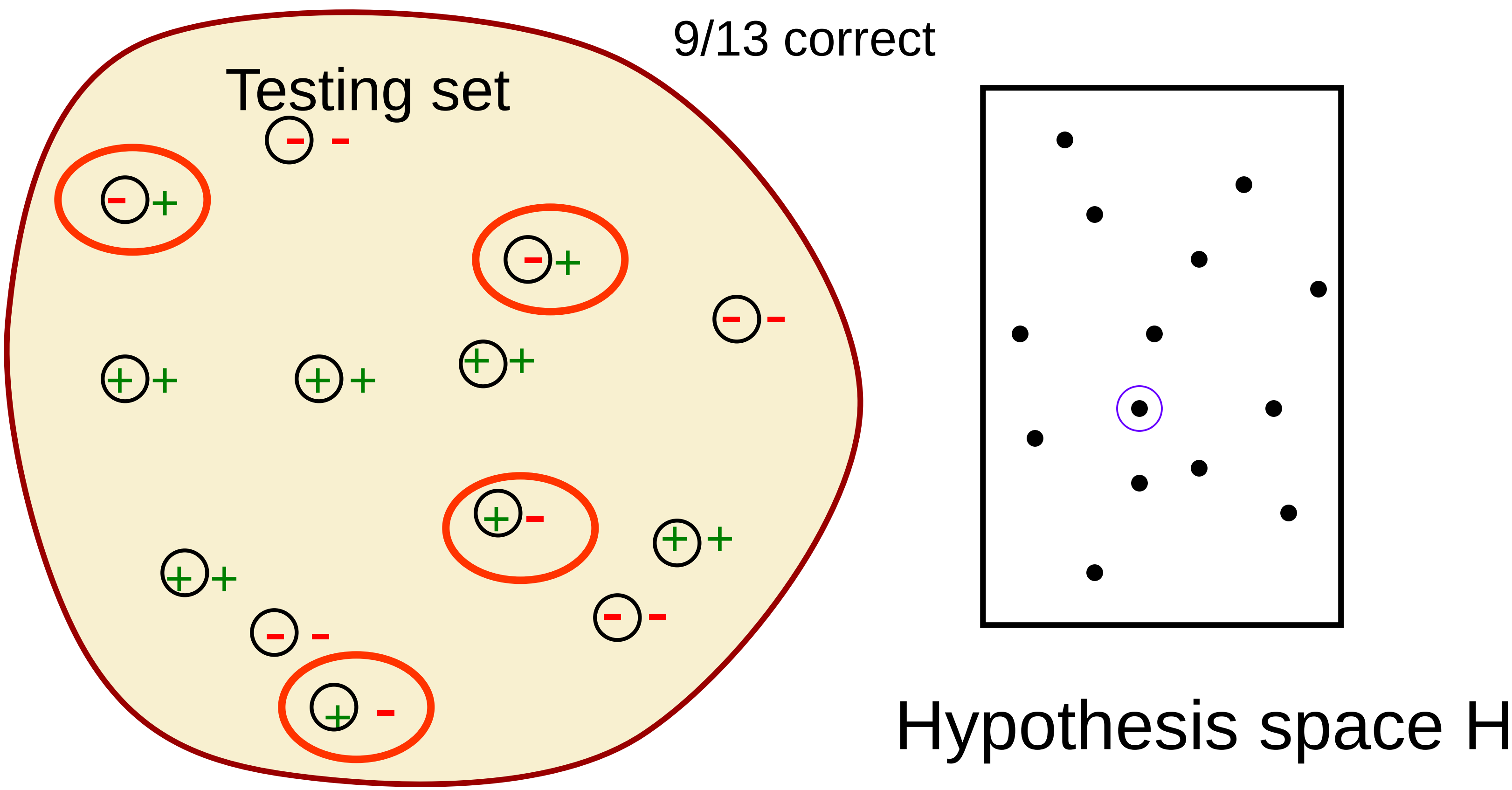
Cross-Validation -1

- Evaluate hypothesis on testing set



Cross-Validation -1

- Compare true concept against prediction



Cross-Validation -2

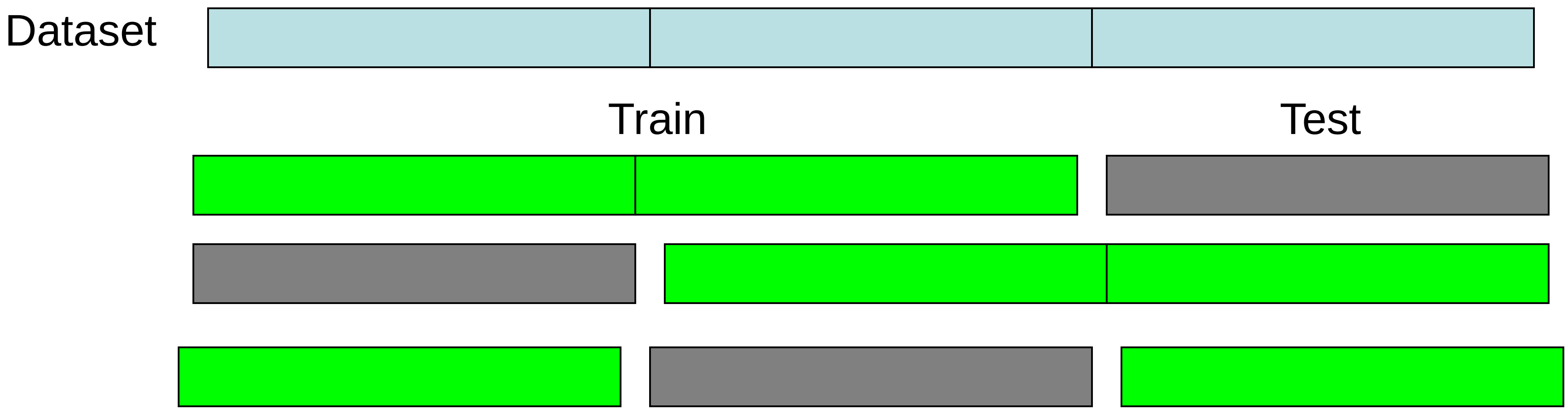
Splitting Strategies

- k-fold cross-validation

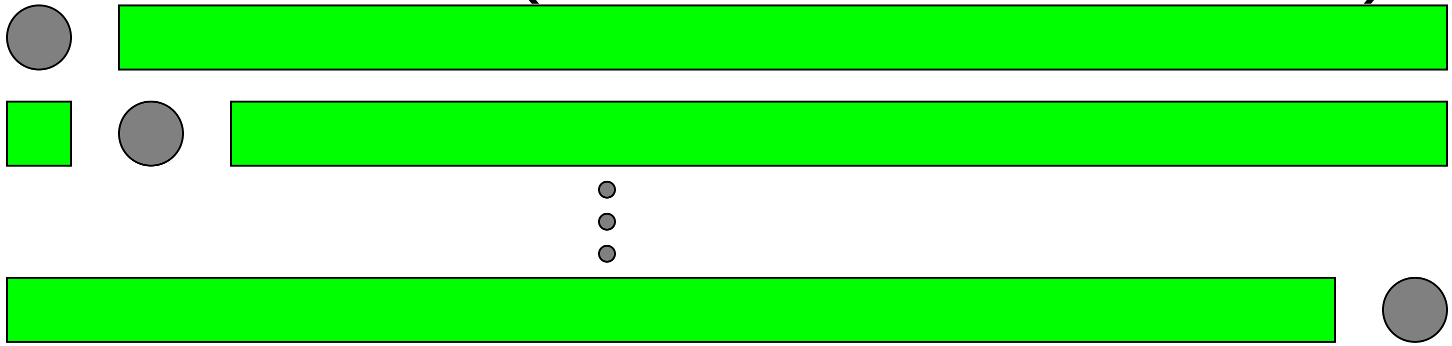
Cross-Validation -2

Splitting Strategies

- k-fold cross-validation



- Leave-one-out (n-fold cross validation)



Contents

Summary

- Data from different sources is heterogeneous in nature.
- Efficient source fusion strategy plays a crucial role in multi-source learning and it is not trivial task
- Simple feature vector concatenation is not always enough.
- Feature selection mechanisms are helpful
- Ensemble learning methods can efficiently fuse multiple data sources.

Next Lesson

- **Case Study**