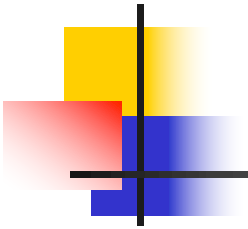
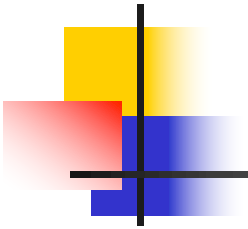


Machine Learning

Semi-Supervised Learning



Semi-Supervised Learning

- Semi-supervised learning refers to learning from data where part contains desired output information and the other part does not
  - Mostly applied to supervised learning problems (classification/regression) with partially labeled data
  - Sometimes partially labeled data is also used to solve unsupervised learning problems (semi-unsupervised learning)
    - Labeled data provides constraints
- Generally unlabeled data is easier to find and more abundant than labeled data

Semi-Supervised Learning

- The goal of most semi-supervised learning is to learn a better classifier/regression function than from only the labeled data
  - Training data can be divided into two sets

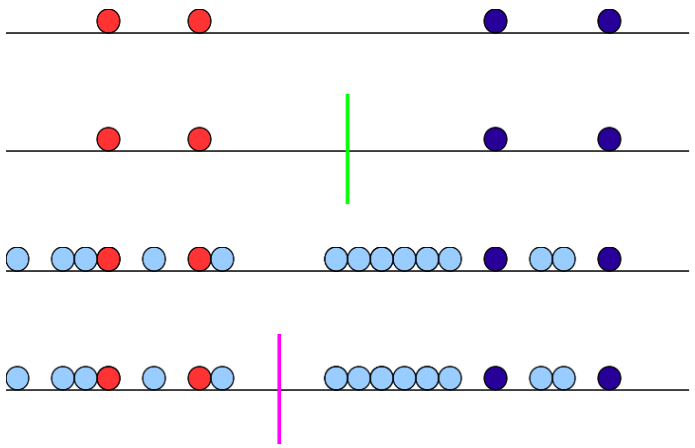
$$D_s = \left\{ \left(x^{(i)}, y^{(i)} \right) : i \in \left[1..n_s \right] \right\}, \; D_u = \left\{ \left(x^{(i)} \right) : i \in \left[n_s + 1..n \right] \right\}$$

$$D = D_s \cup D_u$$

  - Learn $h_D(x)$ that has a higher expected performance on test data than $h_{Ds}(x)$
- Sometimes only labeling of the unlabeled data is required – transductive learning

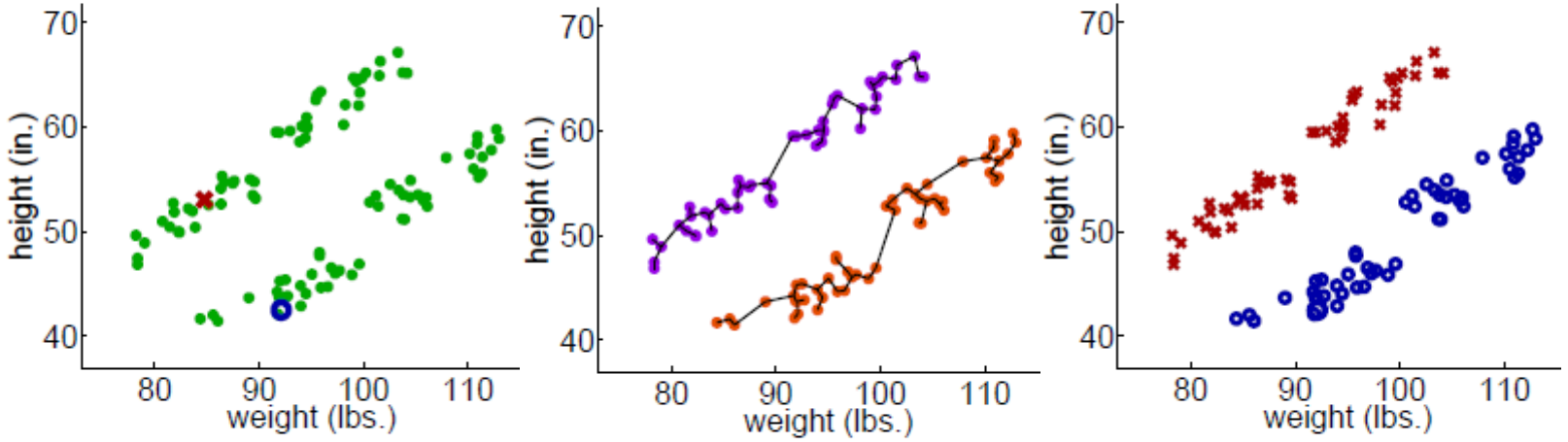
Semi-Supervised Learning

- The main benefit of unsupervised data in semi-supervised learning is by providing information about the structure of the overall data
  - Assumption: labeled and unlabeled data come from the same distribution



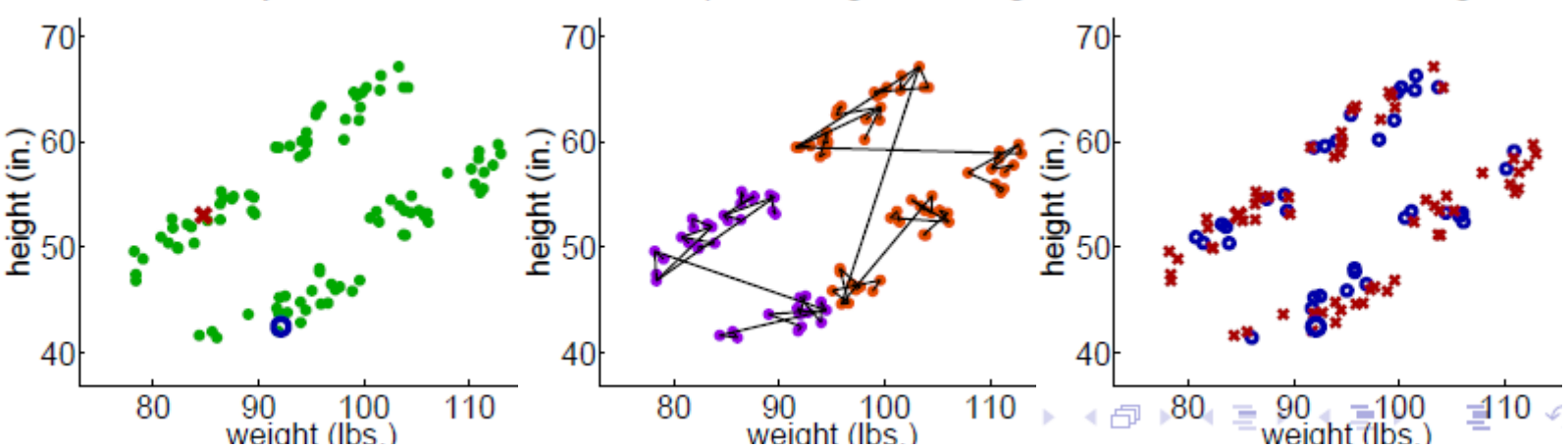
Cluster and Label

- Cluster and Label is the simplest form of semi-supervised learning
  - First apply clustering to all data
  - Then apply supervised learning on the labeled instances in each cluster

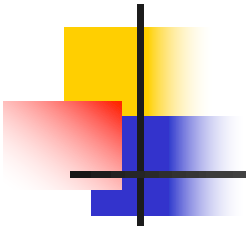


Cluster and Label

- But: works only if the cluster structure matches the underlying class structure



- We can also use unsupervised feature learning to derive new features for supervised learning
  - Requires that feature criteria match class structure

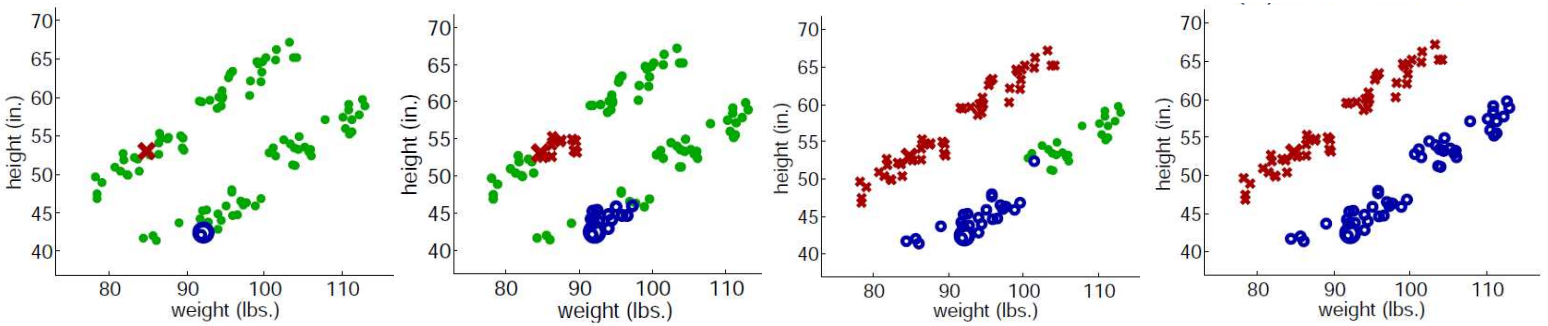


Self-Training

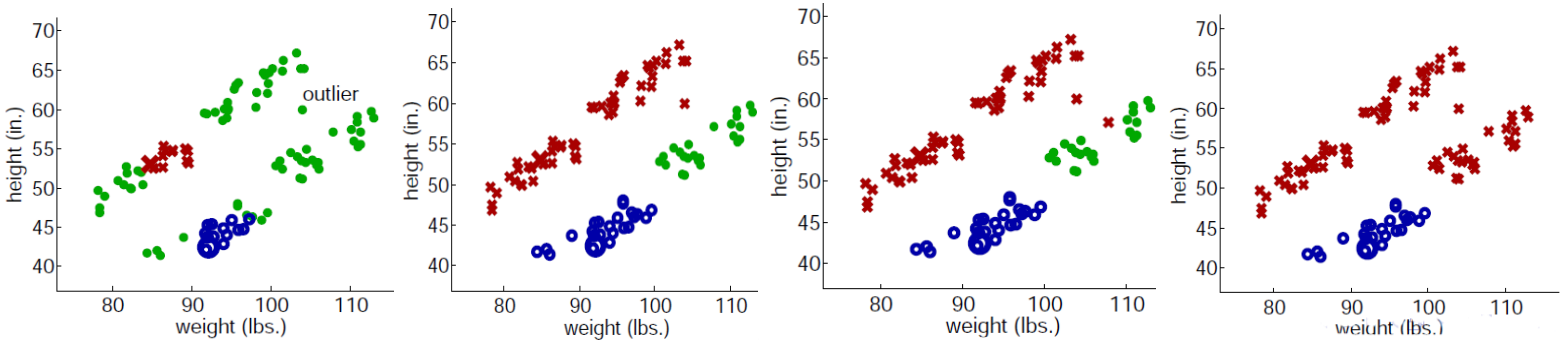
- Self-training can be applied to any supervised learning algorithm that has a measure for confidence of a label assignment
  - E.g.: MLE, MAP, or maximum margin approaches
- Basic idea is to incrementally augment labeled set with self-labeled instances
  - Apply supervised learner to the labeled data
  - Apply learned hypothesis to the unlabeled data
  - Move k highest confidence samples to labeled set
  - Repeat until all data is labeled

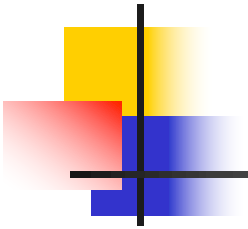
Self-Training

- Self-training basically generates its own training data



- But: self-training is prone to reinforcing mistakes



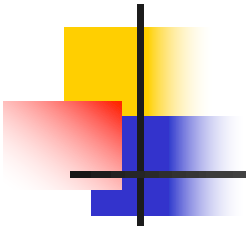


Co-Training

- Self-training is a simple algorithm that often works well
  - Can produce worse results even on labeled data than training only on the labeled data
- Co-training follows a similar idea but tries to reduce the self-reinforcement problem
  - Requires two views of the same data
    - Each data point exists in two representations

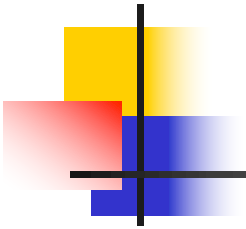
$$x^{(i)} = \left[x_{(1)}^{(i)}, x_{(2)}^{(i)} \right]$$

      - Each view contains all the data but in a different representation (ideally conditionally independent)



Co-Training

- Operation of co-training is similar to self-training
  - In each view, run supervised learner on labeled data
  - In each view, label unlabeled data using hpothesis
  - In each view, identify k most confident unlabeled instances, remove them from unlabeled set in this view and add them to labeled data set of other view
  - Repeat until all unlabeled data has been labeled
  - Form final classifier/regression function by combining the learned classifiers
    - Averaging or voting are most commonly used



Co-Training and Multi-View

- Co-training has fewer problems with reinforcing errors than self-training
  - Lower probability of picking the same item mistakenly in both views
- Multi-view learning generalizes this concepts of learning multiple classifiers with different views
  - Views are achieved either through different data representations or through the use of different supervised learners
  - Use of assigned labels on unlabeled data requires that multiple classifiers agree

Expectation Maximization

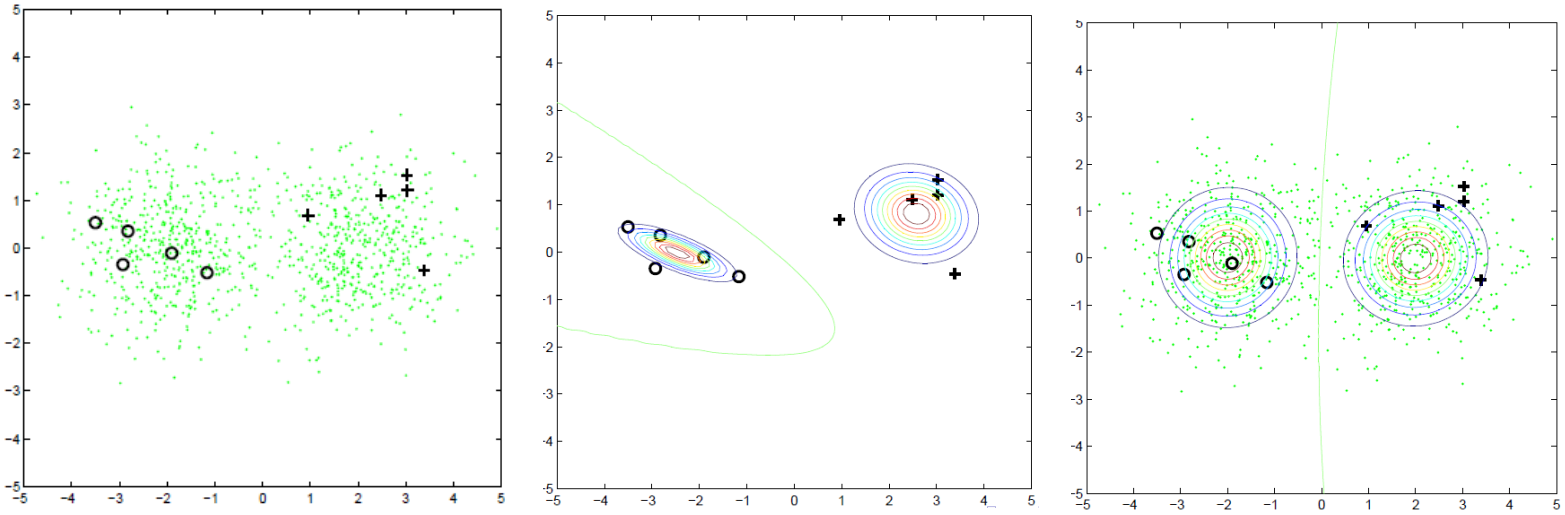
- Another way to limit self-reinforcing errors would be to allow them to be undone later
- Expectation maximization can be used for this with any probabilistic supervised learner

$$\text{argmax}_\theta P(D_s, D_u \mid \theta) = \text{argmax}_\theta \sum_{Y_u} P(D_s, D_u, Y_u \mid \theta)$$

  - Apply supervised learner to labeled data
  - Use current hypothesis to compute the expectation (probability) of the label for the unlabeled data
  - Re-train the supervised learner using labeled and unlabeled data with assigned expected label
  - Repeat from step 2 until convergence

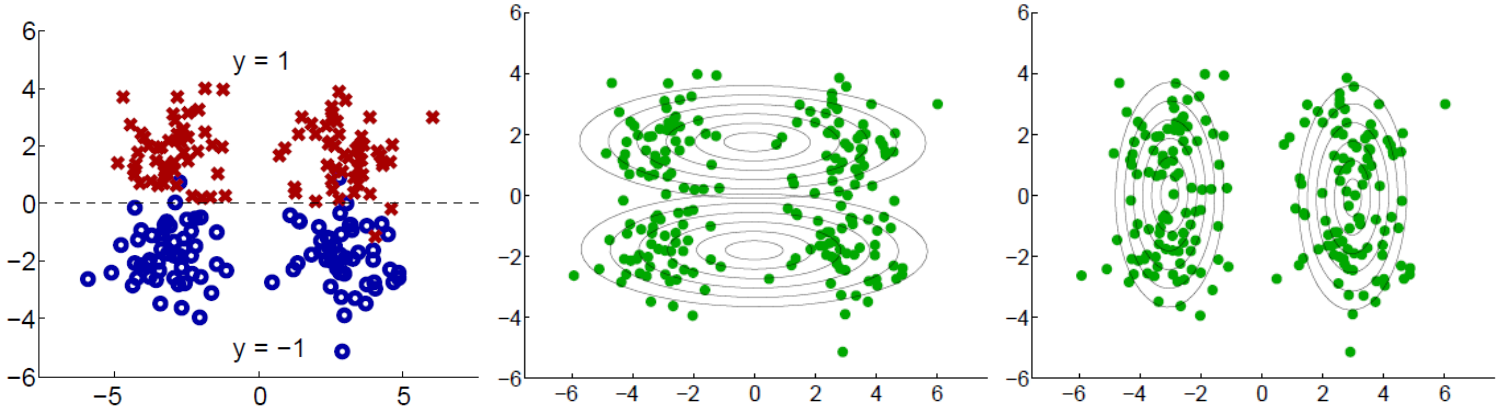
Example: Gaussian Mixture

- Hypothesis space for generative classifier is Gaussian in each class
  - Corresponds to Gaussian mixture model clustering if no labeled data is available

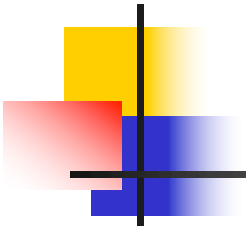


Example: Gaussian Mixture

- Requires that the underlying assumption of the hypothesis space (i.e. in this case of the generative model) is approximately correct
  - If not the final result can be worse

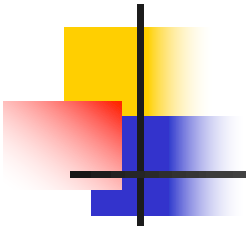


  - Can use mixture within each class
  - Can change weight of unlabeled data in likelihood function



Expectation Maximization

- Expectation maximization will generally work well with probabilistic algorithms if the hypothesis space is appropriate
  - Locally optimizes the marginal data probability starting from the supervised data solution
- A slightly different approach is to look at unsupervised data largely in terms of regularization
  - Unsupervised data is not part of the main performance function but of the regularization term

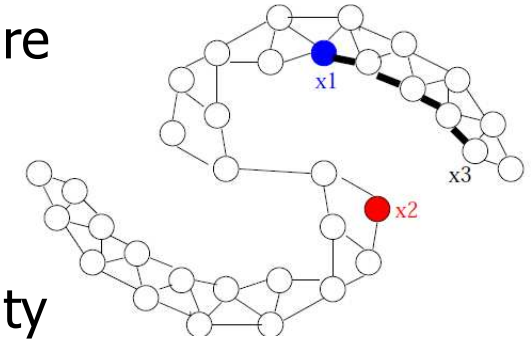


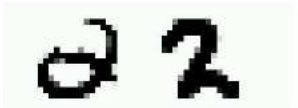
Graph Regularization

- Use label smoothness as a regularization term
  - Label smoothness implies that similar data items should have similar labels
- Graph regularization uses a weighted graph
  - Nodes in the graph are all data points
  - Weighted edges connect nodes and convey similarity between two nodes
    - Fully connected graph with weights representing similarity
    - K-nearest neighbor graph to reduce edge number
    - Weighted ε-distance graph (only close nodes connected)
  - Similarity propagates along the graph structure

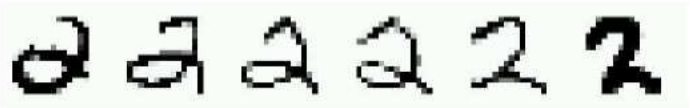
Graph Regularization

- Graph regularization uses the data graph structure to define and propagate similarity
  - Similarity in terms of label can be established / propagated using a sequence of similar nodes
    - Items that are not similar in feature space can be close in the graph
    - Label smoothness over the graph propagates based on data items, not based on pure feature similarity

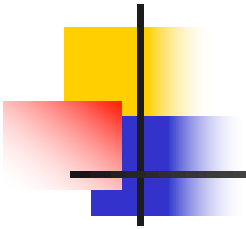




not similar



'indirectly' similar
with stepping stones



Graph Regularization

- Graph regularization represents label smoothness over the graph in the form of a regularization term

- Uses the data graph structure to define and propagate similarity
  - Similarity in terms of label can be established / propagated using a sequence of similar nodes

Graph Regularization

- Graph regularization represents label smoothness over the graph in the form of a regularization term

$$\hat{\theta} = argmin_\theta \left(\sum_{\left(x^{(i)}, y^{(i)}\right) \in D_s} \left(y^{(i)} - h_\theta(x^{(i)}) \right)^2 + \lambda \sum_{x^{(i)}, x^{(j)} \in D} w_{i,j} \left(h_\theta(x^{(i)}) - h_\theta(x^{(j)}) \right)^2 \right)$$

  - Squared error over labeled data with regularization using label smoothness
  - Various algorithms can be used to solve this

Semi-Supervised SVMs

- Semi-supervised SVM (S3VM) or Transductive SVM (TSVM) use intrusion of unlabeled data into the margins as regularization

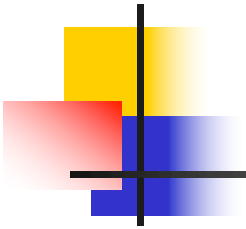
$$\min_{\gamma,w,b} \frac{1}{2}\|w\|^2 + C\left(\sum_{i=1}^{n_s} \xi^{(i)} + \lambda \sum_{i=n_s+1}^{n} \varsigma^{(i)}\right)$$

$$s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi^{(i)}, i = 1,\ldots,n_s$$

$$\left|(w^T x^{(i)} + b)\right| \geq 1 - \varsigma^{(i)}, i = n_s + 1,\ldots,n$$

$$\xi^{(i)}, \varsigma^{(i)} \geq 0$$

  - Keep unlabeled data outside the margins
    - Avoids discriminant through dense regions
  - Optimization harder to solve due to total amount



Semi-Supervised Learning

- Semi-supervised is aimed at improving learning tasks through the use of a mix of labeled and unlabeled data
  - Takes advantage of unlabeled data often being easier to obtain and much more numerous
- Many techniques exist
  - Learning/training regimens that can be used with existing algorithms
    - E.g. self-learning, EM, multiview-learning
  - Modifications of learning through regularization
    - E.g. graph regularization, S3SVM