Unit 4
Regularization for Deep Learning

TFIP-AI Artificial Neural Networks and Deep Learning

Definition

- “Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.”

Norm Penalties

- L1: Encourages sparsity, equivalent to MAP Bayesian estimation with Laplace prior

- Squared L2: Encourages small weights, equivalent to MAP Bayesian estimation with Gaussian prior

L2 Norm Penalties

L2 Parameter Regularization

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha \Omega(\boldsymbol{\theta})$$

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \frac{\alpha}{2} \boldsymbol{w}^\top \boldsymbol{w} + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

$$\nabla_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha \boldsymbol{w} + \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \epsilon \left(\alpha \boldsymbol{w} + \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) \right).$$

$$\boldsymbol{w} \leftarrow (1 - \epsilon \alpha) \boldsymbol{w} - \epsilon \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

L2 Norm Penalties cont…

Quadratic approximation to the objective function

$$\hat{J}(\boldsymbol{\theta}) = J(\boldsymbol{w}^*) + \frac{1}{2}(\boldsymbol{w} - \boldsymbol{w}^*)^\top \boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}^*),$$

The minimum of $\hat{J}$ occurs where its gradient

$$\nabla_{\boldsymbol{w}} \hat{J}(\boldsymbol{w}) = \boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}^*)$$

is equal to $\mathbf{0}$.

By adding the weight decay gradient in the above equation

$$\alpha\tilde{\boldsymbol{w}} + \boldsymbol{H}(\tilde{\boldsymbol{w}} - \boldsymbol{w}^*) = 0$$
$$(\boldsymbol{H} + \alpha\boldsymbol{I})\tilde{\boldsymbol{w}} = \boldsymbol{H}\boldsymbol{w}^*$$
$$\tilde{\boldsymbol{w}} = (\boldsymbol{H} + \alpha\boldsymbol{I})^{-1}\boldsymbol{H}\boldsymbol{w}^*.$$

L2 Norm Penalties cont…

By $\boldsymbol{H} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top$, real and symmetric, we have:

$$\begin{aligned}
\tilde{\boldsymbol{w}} &= (\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top + \alpha\boldsymbol{I})^{-1}\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^* \\
&= \left[\boldsymbol{Q}(\boldsymbol{\Lambda} + \alpha\boldsymbol{I})\boldsymbol{Q}^\top\right]^{-1}\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^* \\
&= \boldsymbol{Q}(\boldsymbol{\Lambda} + \alpha\boldsymbol{I})^{-1}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^*.
\end{aligned}$$

We see that the effect of weight decay is to rescale $\boldsymbol{w}^*$ along the axes defined by the eigenvectors of $\boldsymbol{H}$. Specifically, the component of $\boldsymbol{w}^*$ that is aligned with the i-th eigenvector of $\boldsymbol{H}$ is rescaled by a factor of $\frac{\lambda_i}{\lambda_i + \alpha}$. (You may wish to review how this kind of scaling works, first explained in figure 2.3).

Weight Decay as Constrained Optimization

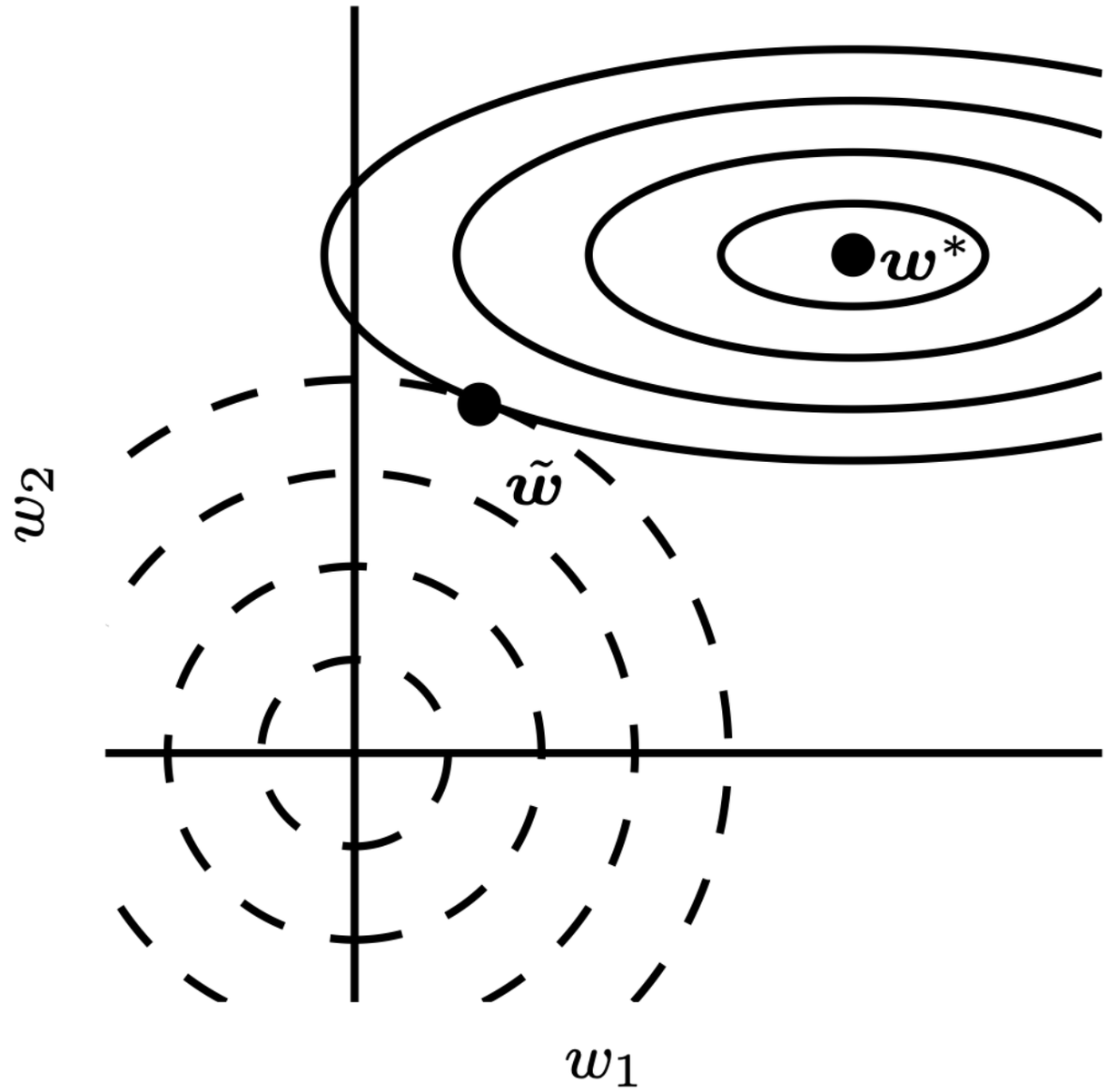


Figure 7.1

Weight Decay as Constrained Optimization cont…

Figure 7.1: An illustration of the effect of L^2 (or weight decay) regularization on the value of the optimal $\boldsymbol{w}$. The solid ellipses represent contours of equal value of the unregularized objective. The dotted circles represent contours of equal value of the L^2 regularizer. At the point $\tilde{\boldsymbol{w}}$, these competing objectives reach an equilibrium. In the first dimension, the eigenvalue of the Hessian of J is small. The objective function does not increase much when moving horizontally away from $\boldsymbol{w}^*$. Because the objective function does not express a strong preference along this direction, the regularizer has a strong effect on this axis. The regularizer pulls w_1 close to zero. In the second dimension, the objective function is very sensitive to movements away from $\boldsymbol{w}^*$. The corresponding eigenvalue is large, indicating high curvature. As a result, weight decay affects the position of w_2 relatively little.

L1 Norm Penalties

L1 Norm, Sparsity, and Feature Selection

$$\Omega(\boldsymbol{\theta}) = ||\boldsymbol{w}||_1 = \sum_i |w_i|,$$

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha||\boldsymbol{w}||_1 + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

$$\nabla_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha \text{sign}(\boldsymbol{w}) + \nabla_{\boldsymbol{w}} J(\boldsymbol{X}, \boldsymbol{y}; \boldsymbol{w})$$

L1 Norm Penalties cont…

Quadratic approximation to the objective function

$$\nabla_{\boldsymbol{w}} \hat{J}(\boldsymbol{w}) = \boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}^*),$$

$$\hat{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{w}^*; \boldsymbol{X}, \boldsymbol{y}) + \sum_i \left[\frac{1}{2} H_{i,i} (\boldsymbol{w}_i - \boldsymbol{w}_i^*)^2 + \alpha |w_i| \right].$$

$$w_i = \text{sign}(w_i^*) \max \left\{ |w_i^*| - \frac{\alpha}{H_{i,i}}, 0 \right\}.$$

L1 Norm Penalties cont…

Quadratic approximation to the objective function

Consider the situation where $w_i^* > 0$ for all i. There are two possible outcomes:

1. The case where $w_i^* \leq \frac{\alpha}{H_{i,i}}$. Here the optimal value of w_i under the regularized objective is simply $w_i = 0$. This occurs because the contribution of $J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})$ to the regularized objective $\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})$ is overwhelmed—in direction i—by the L^1 regularization which pushes the value of w_i to zero.

2. The case where $w_i^* > \frac{\alpha}{H_{i,i}}$. In this case, the regularization does not move the optimal value of w_i to zero but instead it just shifts it in that direction by a distance equal to $\frac{\alpha}{H_{i,i}}$.

A similar process happens when $w_i^* < 0$, but with the L^1 penalty making w_i less negative by $\frac{\alpha}{H_{i,i}}$, or 0.

From a Constained Optimization Perspective

Considering the following generalized Lagrange function:

$$\mathcal{L}(\boldsymbol{\theta}, \alpha; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha(\Omega(\boldsymbol{\theta}) - k).$$

The solution to the constrained problem is given by

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \max_{\alpha, \alpha \geq 0} \mathcal{L}(\boldsymbol{\theta}, \alpha).$$

The above equation satisfied a Karush-Kuhn-Tucker (KKT) condition.

The following equation is exactly the same as the regularised training problem as described in the previously slides:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \alpha^*) = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha^* \Omega(\boldsymbol{\theta}).$$

Other Types of Regularizations

- Data Augmentation
- Robust to Noise
  (1) Noise to Input
  (2) Noise to weights
  (3) Noise to output
- Early Stopping
- Dropout

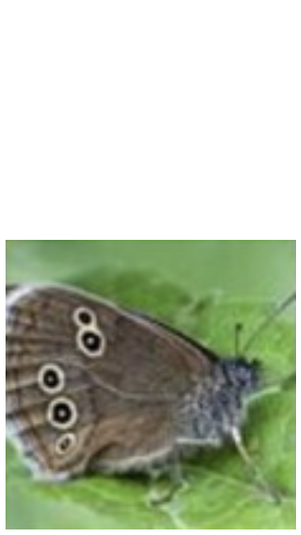
Dataset Augmentation

The best way to make a machine learning model generalised better is to train it on more data.

Of course, in practice, the amount of data we have is limited. One way to get around this problem is to create fake data and add it to the training data set.

Therefore, Dataset Augmentation —>

Dataset Augmentation cont…



Affine Distortion

Noise

Elastic Deformation

Horizontal flip

Random Translation

Hue Shift

Dataset Augmentation cont…

Be careful not to apply transformations that would change the correct class.

For example, optical character recognition tasks require recognizing the difference between “b” and “d” and the difference between “6” and “9”, so horizontal flips and 180^{o} rotations are not appropriate ways of augmenting datasets for these tasks.

Noise Robustness

Regularizing Deep Neural Networks by Injecting Noise

The Bayesian treatment of learning would consider the model weights to be uncertain and representable via a probability distribution that reflects this uncertainty.

Adding noise to the weights is a practical, stochastic way to reflect this uncertainty.

Noise Robustness cont…

Consider the regression setting:

$$J = \mathbb{E}_{p(x,y)} \left[(\hat{y}(\boldsymbol{x}) - y)^2 \right].$$

The training set consists of m labeled examples $\{(\boldsymbol{x}^{(1)}, y^{(1)}), \ldots, (\boldsymbol{x}^{(m)}, y^{(m)})\}$.

Assume that with each input presentation, we also include a random perturbation $\epsilon_{\boldsymbol{W}} \sim \mathcal{N}(\epsilon; \mathbf{0}, \eta \boldsymbol{I})$ of the network weights. We denote the perturbed model as $\hat{y}_{\epsilon_{\boldsymbol{W}}}(\boldsymbol{x})$. The objective function thus becomes:

$$\begin{aligned}
\tilde{J}_{\boldsymbol{W}} &= \mathbb{E}_{p(\boldsymbol{x},y,\boldsymbol{\epsilon_W})} \left[(\hat{y}_{\boldsymbol{\epsilon_W}}(\boldsymbol{x}) - y)^2 \right] \\
&= \mathbb{E}_{p(\boldsymbol{x},y,\boldsymbol{\epsilon_W})} \left[\hat{y}^2_{\boldsymbol{\epsilon_W}}(\boldsymbol{x}) - 2y\hat{y}_{\boldsymbol{\epsilon_W}}(\boldsymbol{x}) + y^2 \right].
\end{aligned}$$

Noise Robustness cont…

The minimization of J with added weight noise is equivalent to minimisation of J with an additional regularisation term:

$$\eta \mathbb{E}_{p(\boldsymbol{x},y)} \left[\| \nabla_{\boldsymbol{W}} \hat{y}(\boldsymbol{x}) \|^2 \right]$$

In the simplified case of linear regression (where, for instance, $\hat{y}(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$), this regularization term collapses into $\eta \mathbb{E}_{p(\boldsymbol{x})} \left[\|\boldsymbol{x}\|^2 \right]$, which is not a function of parameters and therefore does not contribute to the gradient of $\tilde{J}_{\boldsymbol{W}}$ with respect to the model parameters.

Semi-Supervised Learning

In the paradigm of semi-supervised learning, both unlabeled examples from P(x) and labeled examples from P (x, y) are used to estimate P (y | x) or predict y from x.

In the context of deep learning, semi-supervised learning usually refers to learning a representation h = f (x). The goal is to learn a representation so that examples from the same class have similar representations. Unsupervised learning can provide useful cues for how to group examples in representation space.

The application of principal components analysis as a pre-processing step before applying a classifier (on the projected data).

Semi-Supervised Learning cont…

One can then trade-off the supervised criterion –logP(y | x) with the unsupervised or generative one (such as–logP(x) or –logP(x,y)).

By controlling how much of the generative criterion is included in the total criterion, one can find a better trade-off than with a purely generative or a purely discriminative training criterion.

In semi-supervised learning there are two basic assumptions, i.e. the "*cluster assumption*" and the "*manifold assumption*"; both are about data distribution. The former assumes that data have inherent cluster structure, and thus, instances falling into the same cluster have the same class label. The latter assumes that data lie on a manifold, and thus, nearby instances have similar predictions. The essence of both assumptions lies in the belief that similar data points should have similar outputs, whereas unlabeled data can be helpful to disclose which data points are similar.

Semi-Supervised Learning cont…

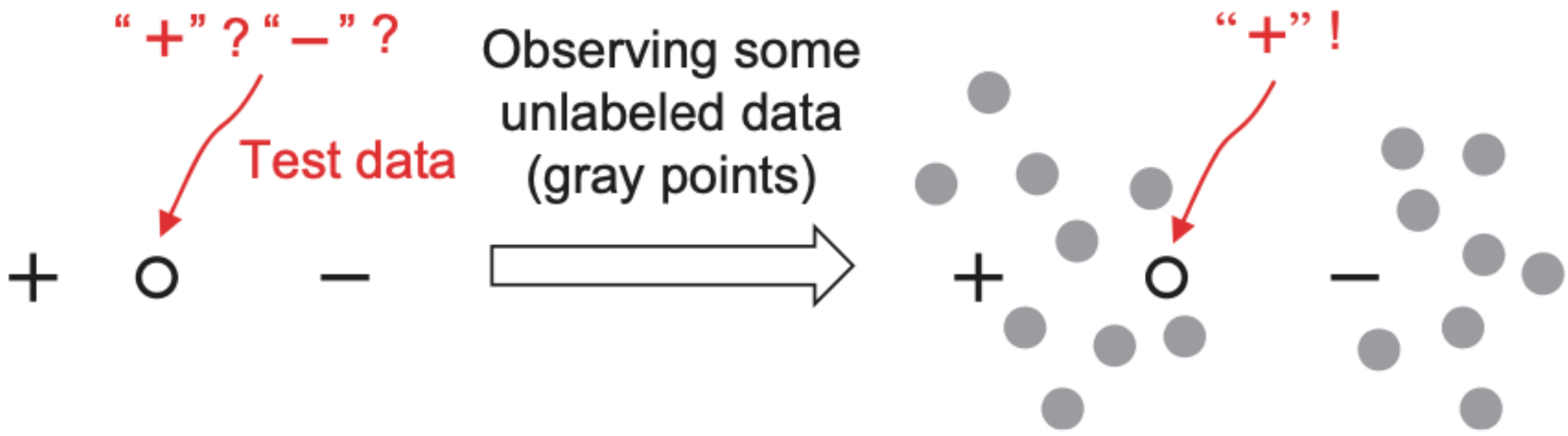


Figure: Illustration of the usefulness of unlabeled data.

If we have to make a prediction based on the only positive and negative points, what we can do is just a random guess because the test data point lies exactly in the middle between the two labeled data points; if we are allowed to observe some unlabeled data points like the gray ones in the figure, we can predict the test data point as positive with high confidence. Here, al- though the unlabeled data points do not explicitly have label information, they implicitly convey some information about data distribution that can be help- ful for predictive modeling.

Semi-Supervised Learning cont…

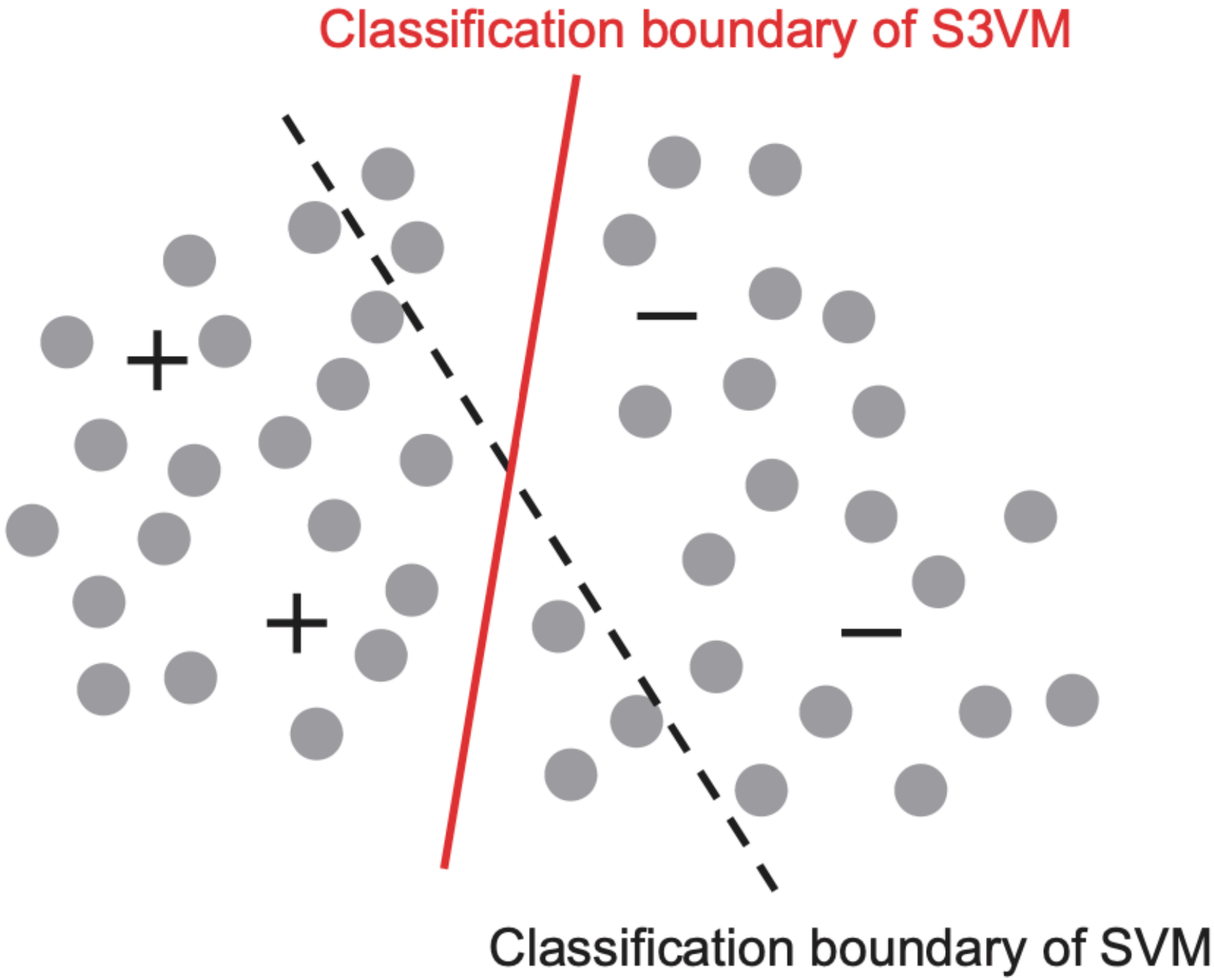


Figure: Illustration of different classification boundaries of SVM which considers only labeled data (”+/−” points), and S3VM (semi-supervised support vector machines) which considers labeled and unlabeled data (gray points).

Semi-Supervised Learning cont…

"Active learning" assumes that the ground-truth labels of unlabeled instances can be queried from an oracle. For simplicity, assume that the labeling cost depends only on the number of queries. Thus, the goal of active learning is to minimize the number of queries such that the labeling cost for training a good model can be minimized.

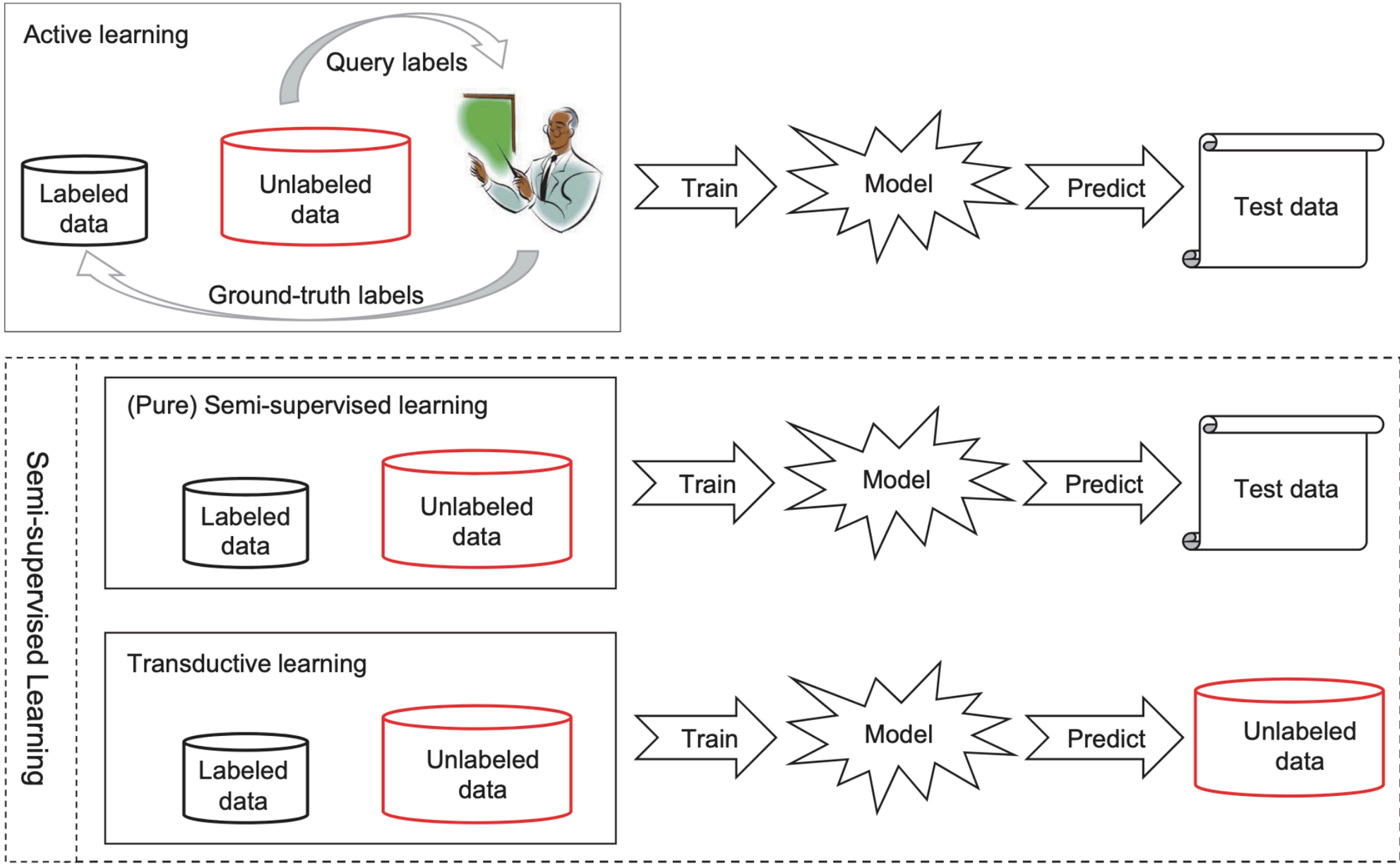


Figure: Active learning, (pure) semi-supervised learning and transductive learning.

Multi-Task Learning

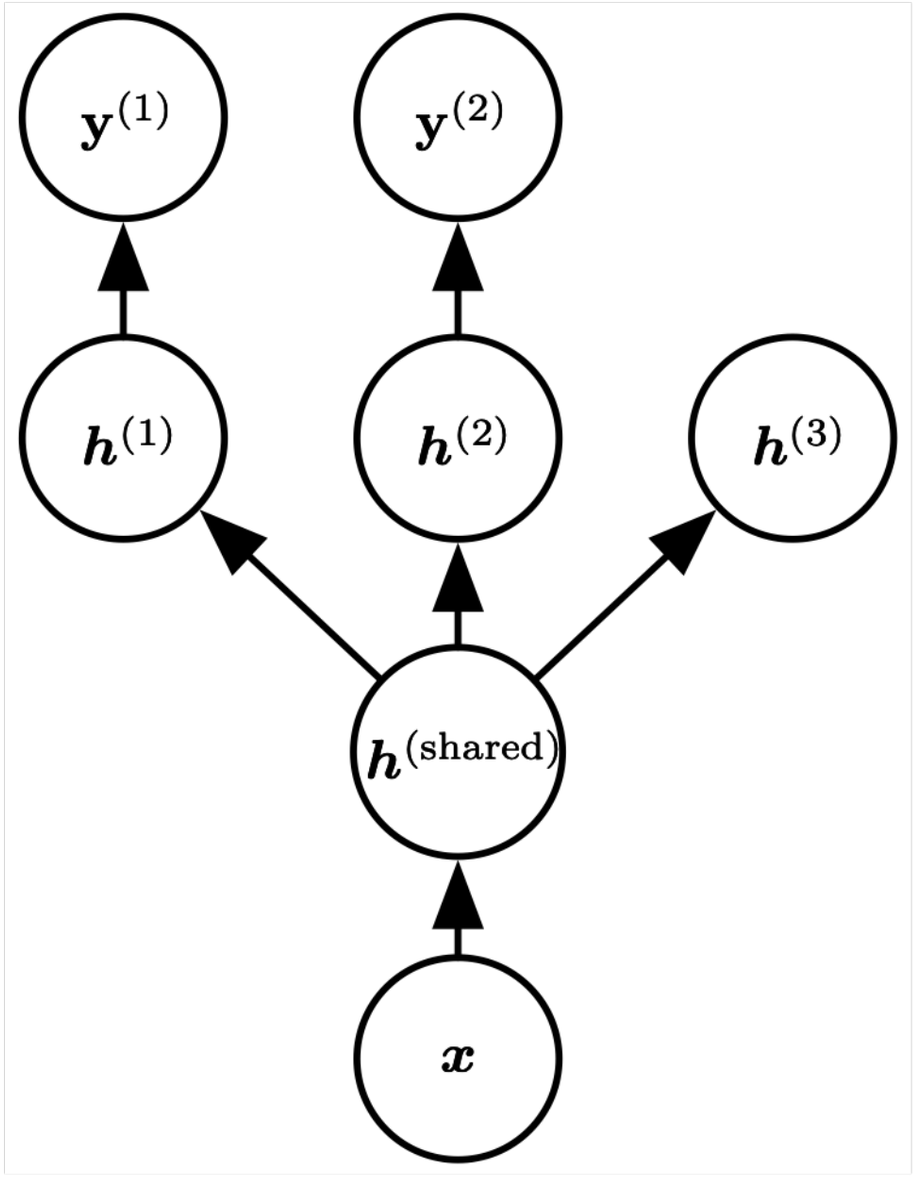


Figure 7.2

Learning Curves

Early stopping: terminate while validation set performance is better

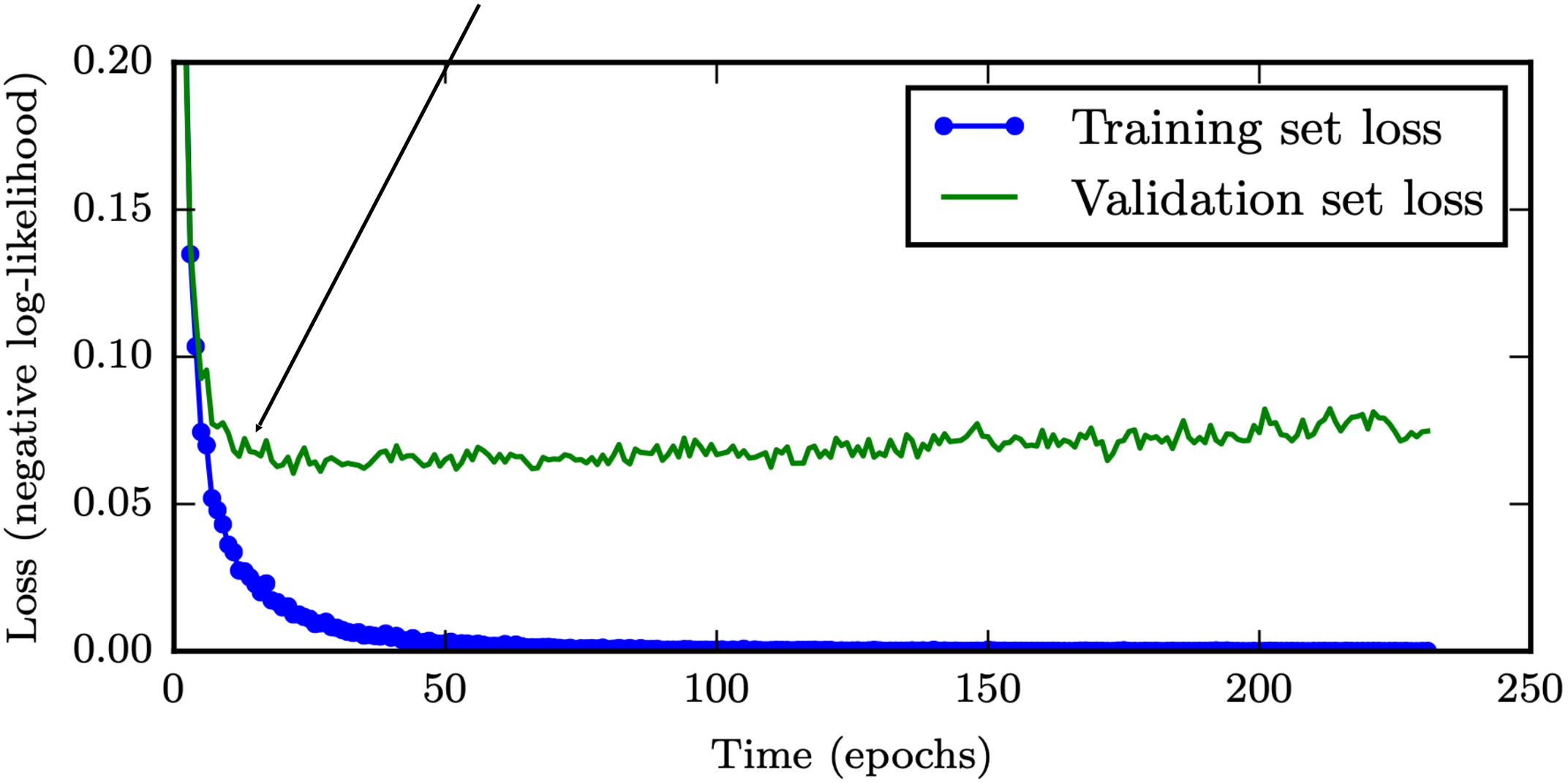


Figure 7.3

Early Stopping and Weight Decay

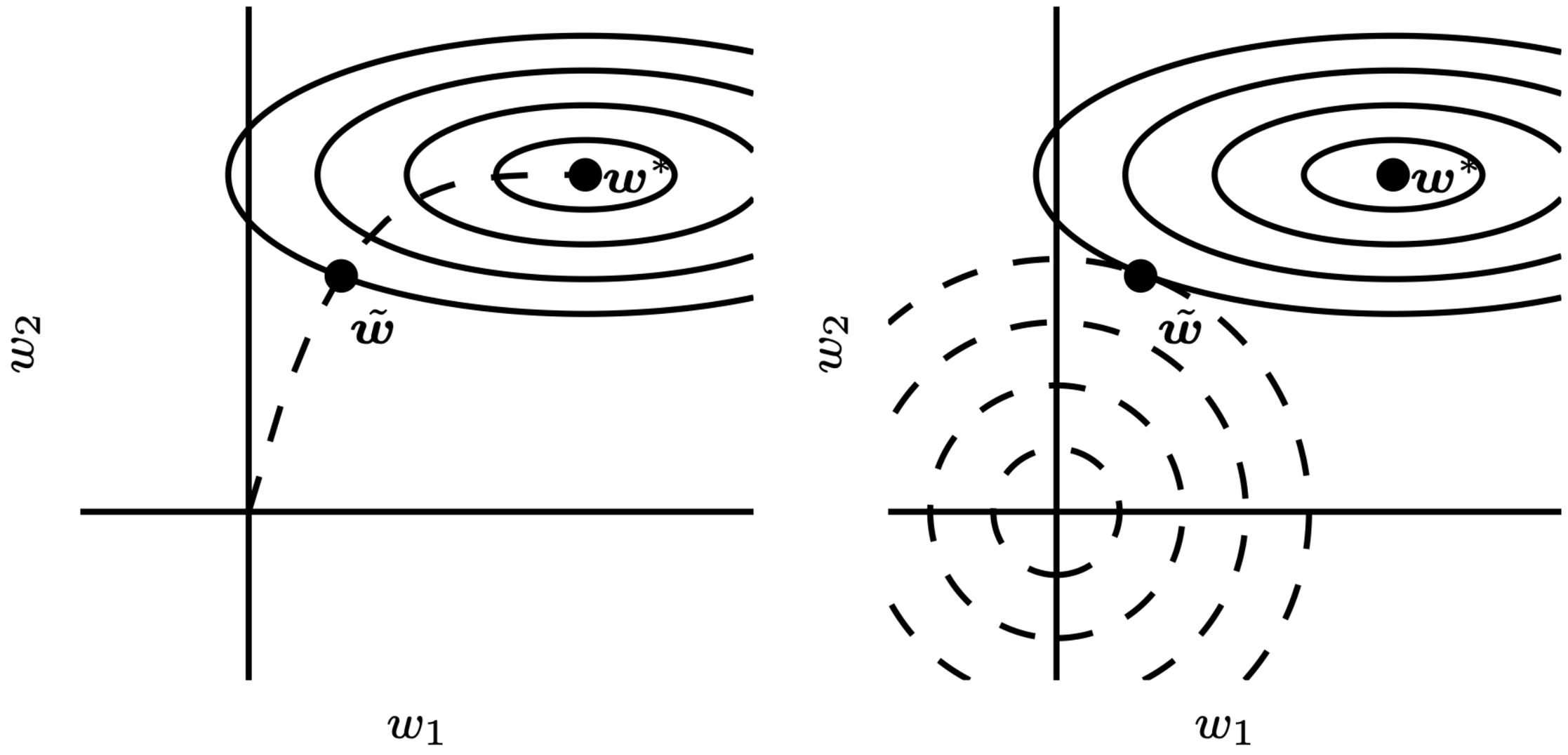


Figure 7.4

Sparse Representations

$$\begin{bmatrix} -14 \\ 1 \\ 19 \\ 2 \\ 23 \end{bmatrix} = \begin{bmatrix} 3 & -1 & 2 & -5 & 4 & 1 \\ 4 & 2 & -3 & -1 & 1 & 3 \\ -1 & 5 & 4 & 2 & -3 & -2 \\ 3 & 1 & 2 & -3 & 0 & -3 \\ -5 & 4 & -2 & 2 & -5 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ -3 \\ 0 \end{bmatrix} \tag{7.47}$$

$\boldsymbol{y} \in \mathbb{R}^m$ $\boldsymbol{B} \in \mathbb{R}^{m \times n}$ $\boldsymbol{h} \in \mathbb{R}^n$

Bagging

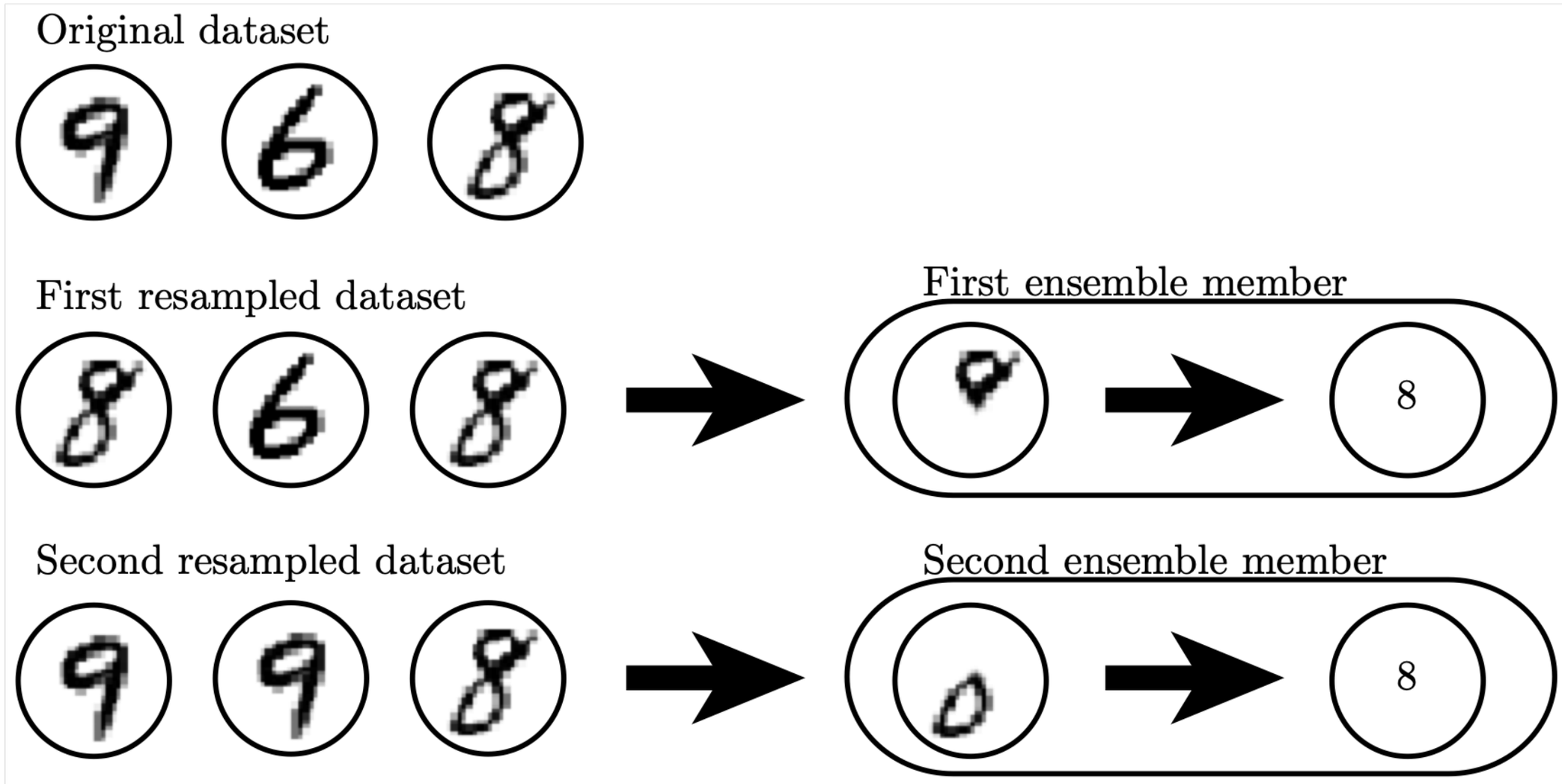


Figure 7.5

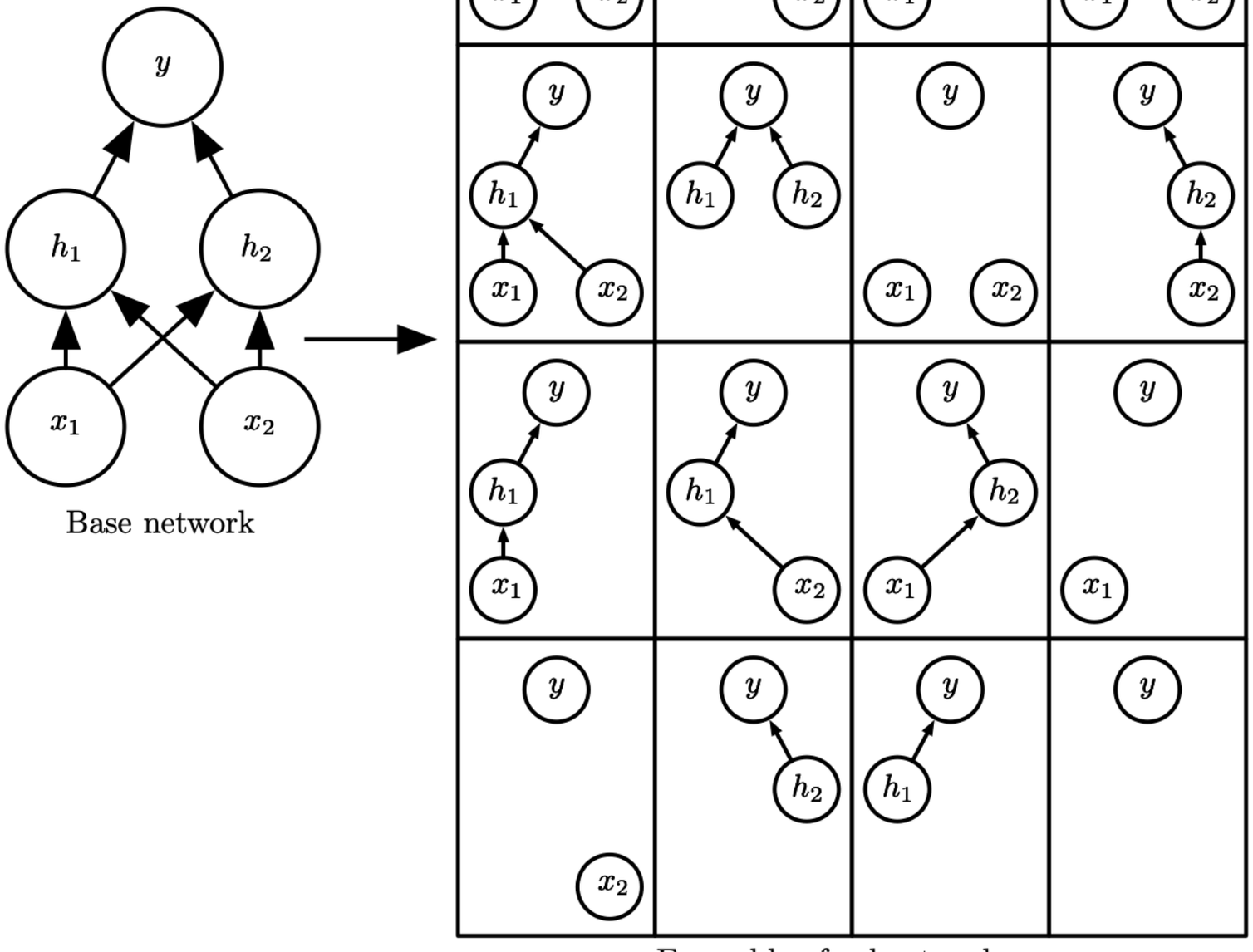
Bagging cont…

Neural networks reach a wide enough variety of solution points that they can often benefit from "model averaging" even if all of the models are trained on the same dataset. Differences in random initialization, random selection of minibatches, differences in hyperparameters, or different outcomes of non-deterministic implementations of neural networks are often enough to cause different members of the ensemble to make partially independent errors.

Model averaging is an extremely powerful and reliable method for reducing generalization error. Its use is usually discouraged when benchmarking algorithms for scientific papers, because any machine learning algorithm can benefit substantially from model averaging at the price of increased computation and memory. For this reason, benchmark comparisons are usually made using a single model.

Dropout

Figure 7.6



Dropout cont…

Figure 7.6: Dropout trains an ensemble consisting of all sub-networks that can be constructed by removing non-output units from an underlying base network. Here, we begin with a base network with two visible units and two hidden units. There are sixteen possible subsets of these four units. We show all sixteen subnetworks that may be formed by dropping out different subsets of units from the original network. In this small example, a large proportion of the resulting networks have no input units or no path connecting the input to the output. This problem becomes insignificant for networks with wider layers, where the probability of dropping all possible paths from inputs to outputs becomes smaller.

Dropout cont…

When extremely few labeled training examples are available, dropout is less effective.

When additional unlabeled data is available, unsupervised feature learning can gain an advantage over dropout.

Adversarial Examples

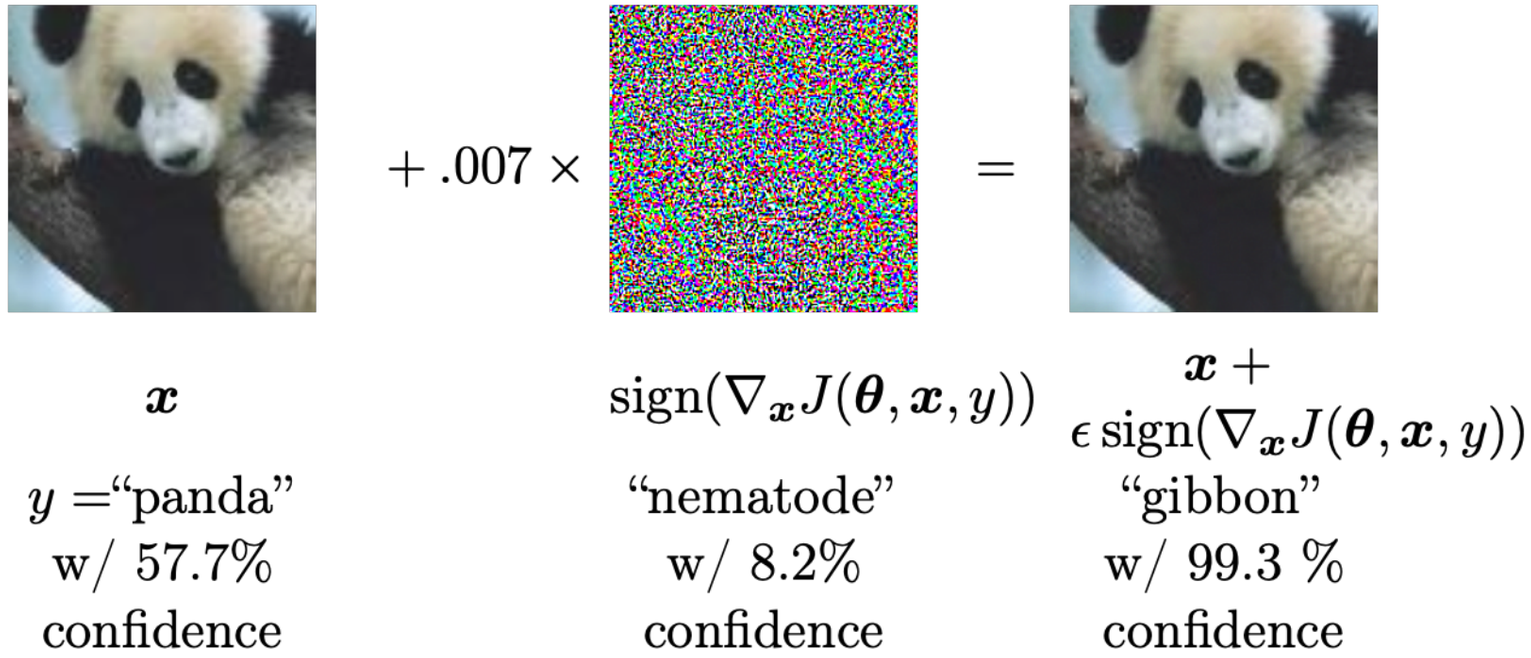


Figure 7.8

Training on adversarial examples is mostly intended to improve security, but can sometimes provide generic regularization.

Adversarial Examples cont…

If we change each input by ε, then a linear function with weights w can change by as much as $\varepsilon \|w\|_1$, which can be a very large amount if w is high-dimensional.

Assumption: different classes usually lie on disconnected manifolds, and a small perturbation should not be able to jump from one class manifold to another class manifold.

Tangent Propagation

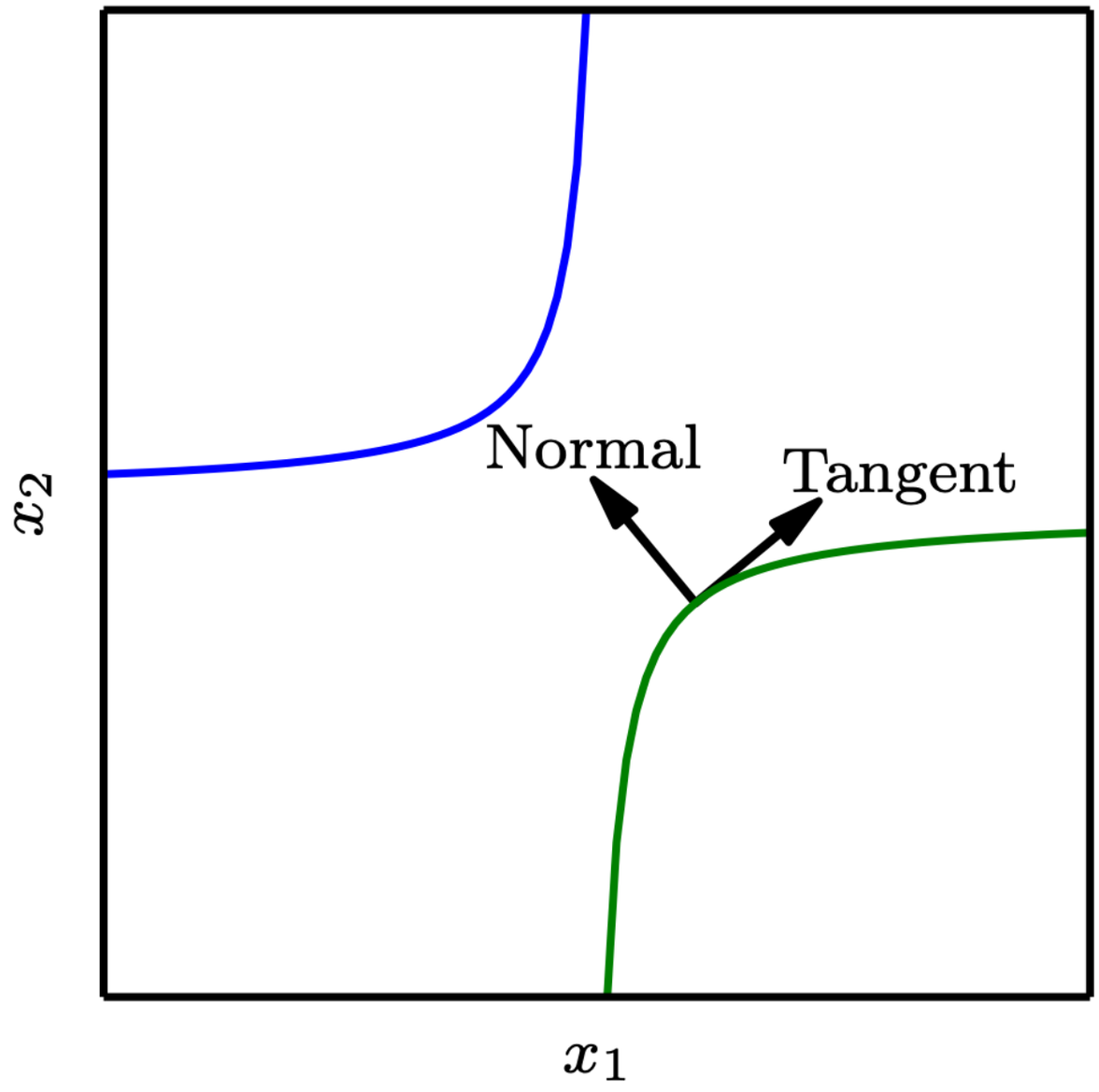


Figure 7.9