Lecture 12 Clustering and High-Dimensional Problems

Key Concepts/Topics of This Lecture
k-center, k-median, k-means
kernel, spectral clustering
sparsity, least absolute shrinkage and selection operator (LASSO)
robustness, Huber's loss
mixing Gaussian model $p(\mathbf{x}) = \sum_{k=1}^{K} \zeta_k \mathcal{N}(\mathbf{x}|\vec{\mu}_k, \vec{\Sigma}_k)$
expectation-maximization (EM) algorithm $\ln p(\mathbf{X}|\mathbf{w}) = \mathcal{L}(q,\mathbf{w}) + \mathrm{KL}(q\|p)$

§12.1 Concept of Clustering

Clustering, which refers to partitioning a set of objects into subsets according to a prescribed criterion, is a fundamental technique in machine learning and data science. For example, given a collection of images of people, one may wish to group them into clusters based on identity or motion patterns. Before applying a clustering algorithm, it is essential to choose an appropriate representation of the data. A common approach is to represent each data object as a vector in R^d, where d real-valued features are extracted from the object. For instance, in document representation, one may adopt the "bag of words" model, where each feature corresponds to a word in the vocabulary and its value records the frequency of occurrence in the document.

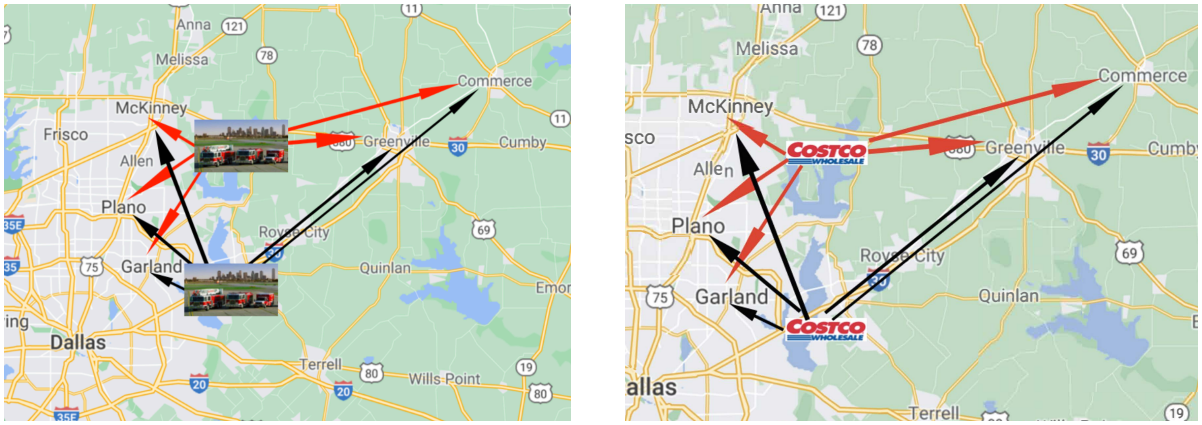

FIG. 12.1: Sketch of the k-center and k-median clustering patterns.

A commonly adopted assumption is that clusters are center-based. Under this assumption, the clustering structure is characterized by K centers $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$, and each data sample is assigned to

the nearest center. However, this assumption alone does not determine which choice of centers is optimal; an explicit objective or optimization criterion is required. Three frequently used criteria are listed as follows:

(a) k-center clustering: Find a partition $\mathcal{C} = \{C_1, C_2, \cdots, C_K\}$ of the data samples into k clusters with corresponding centers $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$ such that the maximum distance between any data point and the center of its cluster is minimized, i.e.,

$$\Phi_{k\,\text{center}}(\mathcal{C}) = \min_{k=1}^{K} \max_{\mathbf{x}^{(i)} \in C_k} d(\mathbf{x}^{(i)}, \vec{\mu}_k). \tag{12.1}$$

This formulation is particularly suitable when clusters correspond to localized regions in space. It is often interpreted as the "firehouse location problem", where one aims to place K fire stations in a city so as to minimize the maximum distance any fire truck must travel. For example, in the left panel of FIG. 12.1, five cities ("Commerce", "McKinney", "Plano", "Greenville", and "Garland" near the Dallas district) and two firehouses are shown. Minimizing the maximum distance suggests that the lower firehouse should be relocated closer to the city "Commerce".

(b) k-median clustering: Find a partition $\mathcal{C} = \{C_1, C_2, \cdots, C_K\}$ of the data samples into K clusters with corresponding centers $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$ so as to minimize the sum of distances between data points and the centers of their clusters, i.e.,

$$\Phi_{k\,\text{median}}(\mathcal{C}) = \min_{k=1}^{K} \sum_{\mathbf{x}^{(i)} \in C_k} d(\mathbf{x}^{(i)}, \vec{\mu}_k). \tag{12.2}$$

Compared with k-center clustering, the k-median objective is more robust to noise since it aggregates distances rather than focusing on the worst case. A small number of outliers typically does not significantly alter the optimal solution unless they are extremely far away or multiple near-optimal solutions coexist. A practical example arises in selecting locations for retail stores such as Costco, where the goal is to minimize the overall cost associated with serving customers, as illustrated in the right panel of FIG. 12.1.

(c) k-means clustering: Find a partition $\mathcal{C} = \{C_1, C_2, \cdots, C_K\}$ of the data samples into K clusters with corresponding centers $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$ to minimize the sum of squares of distances between data samples and the centers of their clusters, i.e.,

$$\Phi_{k\,\text{means}}(\mathcal{C}) = \min_{k=1}^{K} \sum_{\mathbf{x}^{(i)} \in C_k} d^2(\mathbf{x}^{(i)}, \vec{\mu}_k). \tag{12.3}$$

In contrast to k-median clustering, the k-means objective assigns greater weight to outliers because the squared distance amplifies large deviations. In this sense, it lies between k-median and k-center clustering in terms of sensitivity to outliers. Moreover, using squared distances provides certain mathematical advantages when the data lie in R^d, particularly from the perspective of differentiability and optimization. This distinction is also reflected in the choice of optimization algorithms. More generally, one may employ the ℓ-1 loss function together with appropriately designed penalty terms to address sparsity and/or robustness (especially in the presence of outliers).

Among these clustering methods, k-means clustering is most commonly used when the data are points in R^d, whereas k-median clustering is often preferred when dealing with a finite metric space, such as data represented as nodes in a graph with distances defined on edges.

EXERCISE 12-1. Give an example on the k-means clustering.

§12.2 k-means

In this lecture, we mainly focus on the k-means clustering algorithm and develop optimization strategies for handling outliers in data samples, which are closely related to the sparsity and robustness of the problem. In addition, we present examples of spectral clustering, whose central idea is quite intuitive: it is often easier and more effective to perform clustering in a projected space rather than in the original space. The key technique underlying spectral clustering is dimension reduction, which is closely connected to the singular value decomposition (SVD). The essential idea is to identify the subspace spanned by the top K right singular vectors of the data matrix (whose rows correspond to the data samples), project the data onto this subspace, and then perform clustering in the projected space. We will provide a detailed mathematical description of the SVD and the associated clustering procedure in Lecture 13.

Although the mathematical formulation may appear somewhat involved, the basic intuition can be illustrated with simple examples where conventional clustering methods fail. For instance, consider two classes of data samples generated according to $re^{i\theta}$ with different values of the radius r, together with some noise. In this case, the data space is effectively one dimensional rather than two dimensional. By projecting the data onto the one-dimensional subspace characterized by the radius r, clustering can be carried out straightforwardly using r as the discriminating variable. FIG. 12.2 provides a schematic illustration of this projection-based clustering approach, particularly in situations where clustering in the original space is not feasible. We will also discuss the connection between clustering and mixtures of Gaussian distributions, along with general algorithms for addressing such problems. In particular, the expectation-maximization (EM) algorithm will be introduced and discussed in some detail.

We now begin with the k-means algorithm. The general structure of a classification problem is illustrated in FIG. 12.3. A key requirement for successful clustering is that the inter-cluster distance is significantly larger than the intra-cluster distance, namely $d_{\mathrm{cc}} \gg d_{\mathrm{inner}}$. We introduce the centroid of the kth cluster as $\vec{\mu}_k$ for $k = 1 \sim K$, which is not known a priori. The objective is to determine these vectors $\vec{\mu}_k$ so as to minimize the sum of squared distances between the data samples assigned to each cluster and the corresponding centroid, which leads to a standard least-squares problem. For each data sample $\mathbf{x}^{(i)}$, we define a responsibility variable $r_k^{(i)} \in \{0,1\}$ such that $r_k^{(i)} = 1$ if $\mathbf{x}^{(i)}$ belongs to the kth cluster, and $r_j^{(i)} = 0$ for all other clusters. The loss function is defined as

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} r_k^{(i)} \|\mathbf{x}^{(i)} - \vec{\mu}_k\|^2, \tag{12.4}$$

and the corresponding optimization problem is

$$(r_k^{(i)}, \vec{\mu}_k)^* \leftarrow \underset{r_k^{(i)}, \vec{\mu}_k}{\operatorname{argmin}} J. \tag{12.5}$$

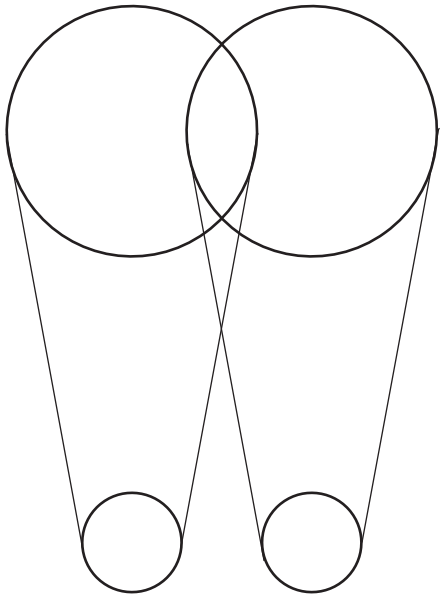

FIG. 12.2: Clusters in the original full space and their projections.

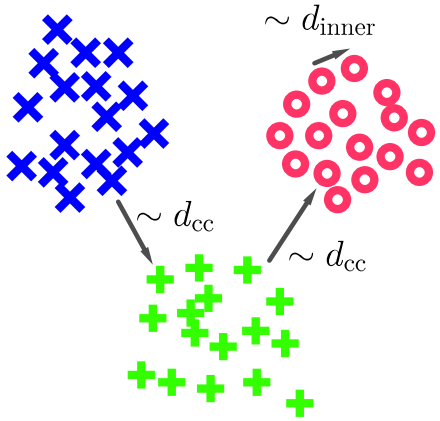

FIG. 12.3: General sketch of a classification problem, where $d_{\text{cc}} \gg d_{\text{inner}}$.

To minimize J, we note that it is linear in $r_k^{(i)}$, which leads to

$$\boxed{r_k^{(i)} = \begin{cases} 1, & k = \operatorname{argmin}_j \|\mathbf{x}^{(i)} - \vec{\mu}_j\|^2, \\ 0, & \text{otherwise.} \end{cases}} \tag{12.6}$$

On the other hand, J is quadratic in $\vec{\mu}_k$, so the optimal value can be obtained via least-squares minimization, i.e., $2\sum_{i=1}^{m} r_k^{(i)} \|\mathbf{x}^{(i)} - \vec{\mu}_k\| = 0$, which yields

$$\boxed{\vec{\mu}_k = \frac{\sum_{i=1}^{m} r_k^{(i)} \mathbf{x}^{(i)}}{\sum_{i=1}^{m} r_k^{(i)}}.} \tag{12.7}$$

The denominator corresponds to the number of samples in cluster k, denoted by m_k. Hence, the vector $\vec{\mu}_k$ represents the mean of cluster k, and is therefore referred to as the cluster center. This also explains the origin of the term "k-means clustering".

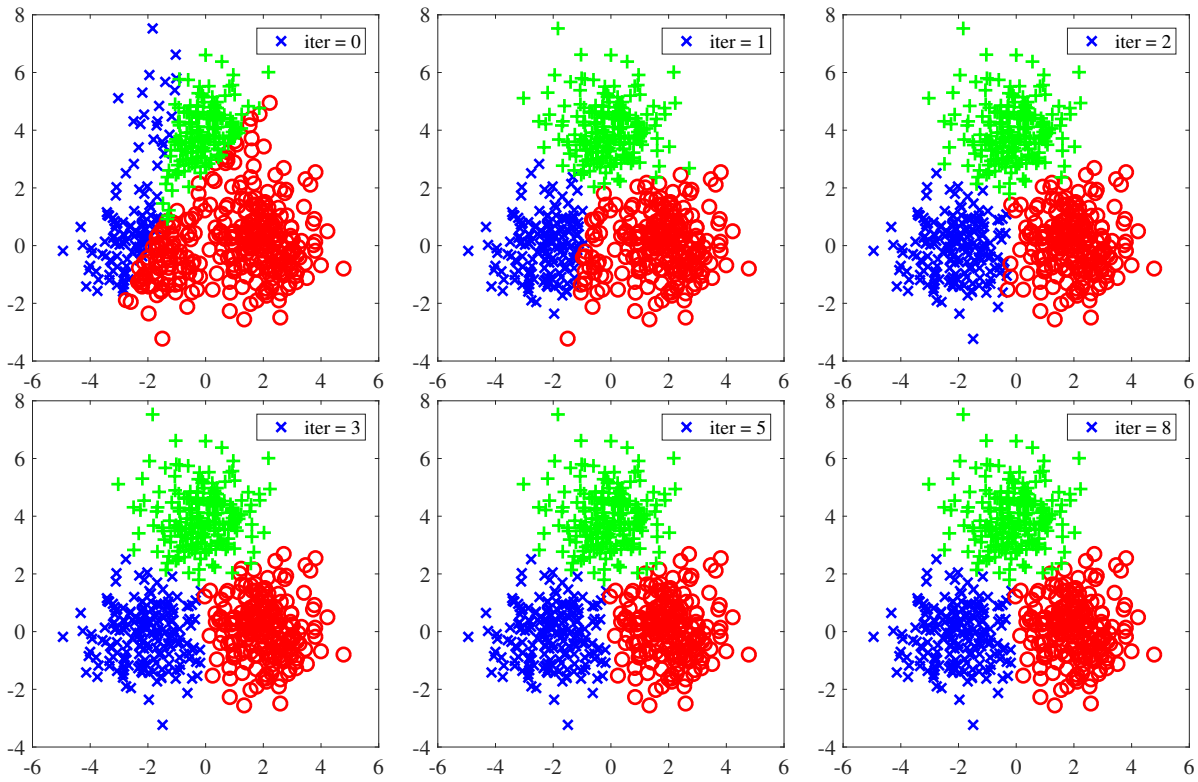

FIG. 12.4: Example of k-means clustering where $K = 3$.

The complete algorithm can be summarized as

$$\min_{k\in\{1,\cdots,K\}} \sum_{i,\mathbf{x}^{(i)}\in C_k}^{m} \sum_{k=1}^{K} \|\mathbf{x}^{(i)} - \vec{\mu}_k\|^2, \quad \vec{\mu}_k = \frac{1}{m_k} \sum_{i,\mathbf{x}^{(i)}\in C_k} \mathbf{x}^{(i)}, \quad \{C_1, C_2, \cdots, C_k\} \leftarrow \underset{k\in\{1,\cdots,K\}}{\operatorname{argmin}} \|\mathbf{x}^{(i)} - \vec{\mu}_k\|^2. \tag{12.8}$$

This procedure is also known as Lloyd's method. It is important to note that the number of clusters K is typically fixed prior to running the algorithm. FIG. 12.4 illustrates an example with two-dimensional data and $K = 3$. The samples are generated from three stochastic models. Initially, there is significant overlap among the clusters. The first iteration of the algorithm produces the most noticeable change in the decision boundaries, while subsequent iterations gradually refine the clustering. After approximately eight iterations, the algorithm converges. The performance of Lloyd's algorithm depends strongly on the initialization. In particular, in some implementations, one or more clusters may become empty, leaving no center from which distances can be computed. Moreover, the algorithm generally converges only to a local optimum, which may not be globally optimal. An example is shown in FIG. 12.5 [∗12-1∗]. Consider clustering the 1D dataset $\{2, 3, 7, 8\}$ into three clusters with initial centers $\{0, 5, 10\}$. In the final configuration, the center at "5" has no assigned samples, resulting in only two clusters instead of the intended three. This highlights the importance of proper initialization.

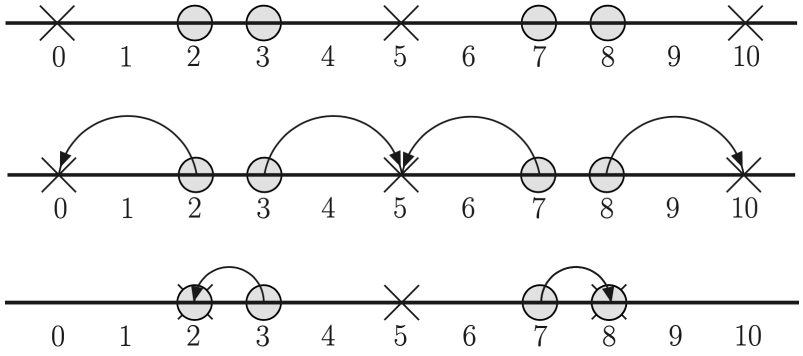

FIG. 12.5: The center at "5" ends up with no items and there are only two clusters {2,8} instead of three.

In certain special cases, such as when all data points lie on a line R^1, the optimal k-means solution can be obtained in polynomial time using dynamic programming. Assume that the data samples $x^{(1)},\cdots,x^{(m)}$ are sorted such that $x^{(1)}\le\cdots\le x^{(m)}$. Suppose that for some $i\ge 1$, we have already computed the optimal k'-means clustering for $x^{(1)},\cdots,x^{(i)}$ for all $k'\le k$. The goal is to extend this solution to $x^{(1)},\cdots,x^{(i+1)}$. Since each cluster consists of consecutive data points, for each k' and each $j\le i+1$, we compute the cost of assigning a single center to $x^{(j)},\cdots,x^{(i+1)}$, defined as the sum of squared distances to their mean. This cost is then combined with the optimal $k'-1$ clustering cost for $x^{(1)},\cdots,x^{(j-1)}$. Taking the minimum over all j yields the optimal solution. The running time for a fixed i is $\mathcal{O}(km)$, and the total complexity is $\mathcal{O}(km^2)$.

In general, when k is part of the input or depends on m, clustering optimization problems are NP-hard. As a result, practical algorithms typically rely on approximations or additional assumptions.

EXERCISE 12-2. Show that in one dimension, the center of a cluster that minimizes the sum of distances of data points to the center is in general not unique. However, if the center minimizes the sum of squared distances to the data points, then it is unique.

EXERCISE 12-3. Let $\mathbf{A}\in\mathrm{R}^{n\times n}$ and $0\le a_{ij}\le 1$ for i and j. A matrix element close to 1/0 indicates that the row and column of the element correspond to related/unrelated items. Develop an algorithm to match each row with a closely related column.

§12.3 Kernel and Spectral Clustering

The Euclidean distance is adopted in (12.8) for searching the clusters. One may also use a non-Euclidean distance to perform a similar task, and in some situations the latter can lead to much better performance. Notice that

$$\begin{aligned}\|\mathbf{x}^{(i)}-\vec{\mu}_k\|^2 &= \left\|\mathbf{x}^{(i)}-\frac{1}{m_k}\sum_{j,\mathbf{x}^{(j)}\in C_k}\mathbf{x}^{(j)}\right\|^2\\ &= \langle\mathbf{x}^{(i)}|\mathbf{x}^{(i)}\rangle-\frac{2}{m_k}\sum_{j,\mathbf{x}^{(j)}\in C_k}\langle\mathbf{x}^{(i)}|\mathbf{x}^{(j)}\rangle+\frac{1}{m_k^2}\sum_{j,j',\mathbf{x}^{(j)},\mathbf{x}^{(j')}\in C_k}\langle\mathbf{x}^{(j)}|\mathbf{x}^{(j')}\rangle,\end{aligned}\tag{12.9}$$

by introducing the kernel, i.e., $k(\mathbf{x},\mathbf{x}') \to \langle \mathbf{x}|\mathbf{x}'\rangle$, one obtains the more general scheme

$$\{C_1, C_2, \cdots, C_k\} \leftarrow \underset{k\in\{1,\cdots,K\}}{\operatorname{argmin}} \left[\langle \mathbf{x}^{(i)}|\mathbf{x}^{(i)}\rangle - \frac{2}{m_k} \sum_{j,\mathbf{x}^{(j)}\in k} \langle \mathbf{x}^{(i)}|\mathbf{x}^{(j)}\rangle + \frac{1}{m_k^2} \sum_{j,j',\mathbf{x}^{(j)},\mathbf{x}^{(j')}\in C_k} \langle \mathbf{x}^{(j)}|\mathbf{x}^{(j')}\rangle \right], \tag{12.10}$$

where $\langle \mathbf{x}^{(i)}|\mathbf{x}^{(i)}\rangle$ does not depend on the cluster label "k". Consequently, the clustering rule can be written as

$$\boxed{\{C_1, C_2, \cdots, C_k\} \leftarrow \underset{k\in\{1,\cdots,K\}}{\operatorname{argmin}} \left[-\frac{2}{m_k} \sum_{j,\mathbf{x}^{(j)}\in C_k} \langle \mathbf{x}^{(i)}|\mathbf{x}^{(j)}\rangle + \frac{1}{m_k^2} \sum_{j,j',\mathbf{x}^{(j)},\mathbf{x}^{(j')}\in C_k} \langle \mathbf{x}^{(j)}|\mathbf{x}^{(j')}\rangle \right].} \tag{12.11}$$

One of the main advantages of the kernel method is that, when the number of data samples is extremely large, techniques such as principal component analysis can be used to reduce the dimension of the problem. One can then apply the traditional clustering algorithm to the dataset in the reduced or projected space. This is essentially the spectral clustering idea. FIG. 12.6 shows an example in which the data samples approximately lie on ellipses with different semi-axes. It is clear that the conventional k-means clustering cannot correctly produce the boundary between the two datasets. In contrast, since spectral clustering effectively performs clustering according to the radius associated with the semi-axes, the boundary can be successfully obtained.

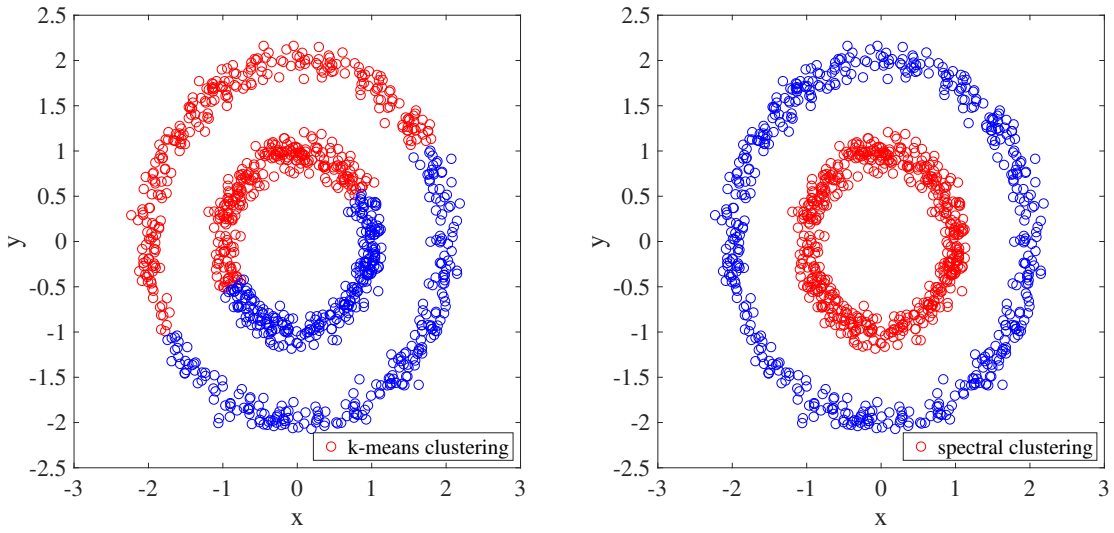

FIG. 12.6: Difference between the conventional k-means clustering and the spectral clustering.

§12.4 ℓ-1 Constraint and LASSO

We do not attempt to give a systematic discussion of the k-median clustering method here. Instead, we introduce the corresponding optimization framework for handling the absolute magnitude of distances. In this context, the regularization term is often taken to be the ℓ-1 constraint, in contrast to the ℓ-2 constraint, which involves the square of the parameters, e.g., $\lambda \mathbf{w}^\top \mathbf{w}$. The corresponding

learning model is given by

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2}\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right)^\top \left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right), \ \|\mathbf{w}\|_1 \le R^2, \tag{12.12}$$

where R is a positive constant and $\|\mathbf{w}\|_1$ is the 1-norm of the vector $\mathbf{w}$ defined as

$$\|\mathbf{w}\|_1 = \sum_{i=1}^{n} |w_i|. \tag{12.13}$$

The geometrical meaning of the ℓ-1 constraint is illustrated in FIG. 12.7. The corresponding regression problem is commonly referred to as the least absolute shrinkage and selection operator (LASSO) problem [∗12-2∗].

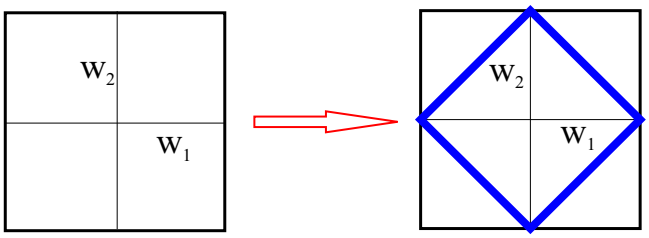

FIG. 12.7: Geometrical meaning of the ℓ-1 constraint.

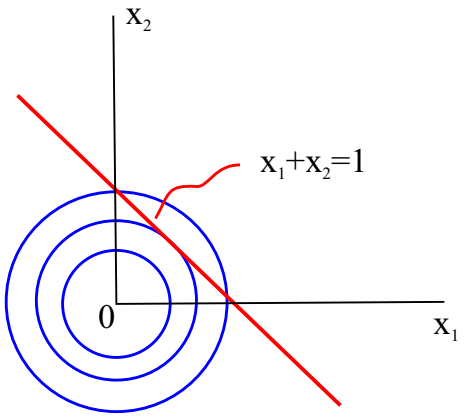

FIG. 12.8: Lagrange multiplier method.

To better understand the difference between the ℓ-2 and ℓ-1 constraints, consider the function $f(x_1, x_2) = 1 - x_1^2 - x_2^2$ with x_1 and x_2 non-negative, under the constraint $g(x_1, x_2) = x_1 + x_2 - 1 = 0$. This problem can be solved using the Lagrange multiplier method by introducing a parameter λ and defining the Lagrange function $L(\mathbf{x}, \lambda) = 1 - x_1^2 - x_2^2 + \lambda(x_1 + x_2 - 1)$, see FIG. 12.8 for its geometrical interpretation. The function satisfies $f \le 1$, represented by the blue circles, while the constraint corresponds to the red line in the figure. The extremum is located at the intersection of the line and the circle, yielding $x_1 = x_2 = 1/2$. One also observes that the extreme value of the function changes from 0 to 1/2 once the constraint g is imposed.

A similar situation arises in the present problem. The function $2^{-1}(\vec{\Phi}\mathbf{w}-\mathbf{y})^{\top}(\vec{\Phi}\mathbf{w}-\mathbf{y})$ corresponds to the unconstrained optimization problem, whose solutions are represented by ellipses, as shown in FIG. 12.9. For general discussions of this loss function, see the relevant sections in Lecture 5. If the constraint is taken to be the ℓ-2 form, the optimal solution corresponds to the intersection point between the red ellipse and the blue circle, where the ellipse/circle represents the solution without/with the ℓ-2 constraint, respectively. In contrast, since the ℓ-1 constraint is sharper, there is a significant probability that the intersection occurs on one of the coordinate axes, implying that some components of the optimal solution $\mathbf{w}^*$ vanish. Solutions with many zero components are referred to as sparse solutions.

This geometrical interpretation shows that the ℓ-1 constraint plays a crucial role in promoting sparsity in machine learning models. It should be emphasized, however, that although the constraint takes the ℓ-1 form, the loss function itself remains of ℓ-2 type, i.e., it is still given by the sum of squared differences between the model prediction and the observed data. Analogous to the standard squared loss problem with ℓ-2 regularization, the squared loss with ℓ-1 regularization is often written as

$$\boxed{J(\mathbf{w})=\frac{1}{2}\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right)^{\top}\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right)+\lambda\|\mathbf{w}\|_1,} \tag{12.14}$$

where λ controls the strength of the regularization.

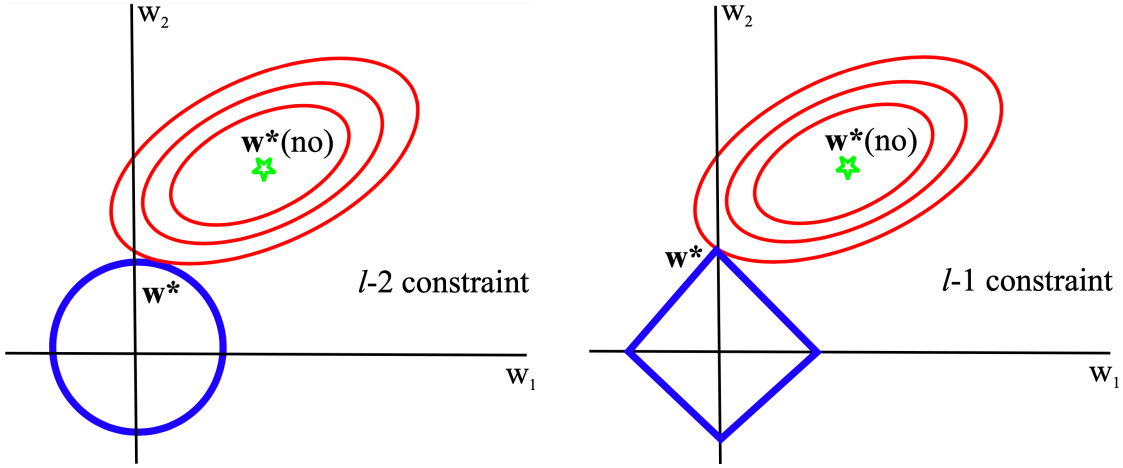

FIG. 12.9: Difference between the ℓ-2 constraint and the ℓ-1 constraint.

There exist standard algorithms for solving the above optimization problem. In these notes, we do not pursue their full derivation; instead, we present the final form of the optimization scheme

$$\min_{\mathbf{w},\mathbf{n}}[f(\mathbf{w})+g(\mathbf{n})],\ \ \mathbf{A}\mathbf{w}+\mathbf{B}\mathbf{n}=\mathbf{d}, \tag{12.15}$$

which leads to the iterative updates

$$\mathbf{w}^{(i+1)}=\operatorname*{argmin}_{\mathbf{w}} L(\mathbf{w},\mathbf{n}^{(i)},\mathbf{u}^{(i)}), \tag{12.16}$$

$$\mathbf{n}^{(i+1)}=\operatorname*{argmin}_{\mathbf{n}} L(\mathbf{w}^{(i+1)},\mathbf{n},\mathbf{u}^{(i)}), \tag{12.17}$$

$$\mathbf{u}^{(i+1)}=\mathbf{u}^{(i)}+\mathbf{A}\mathbf{w}^{(i+1)}+\mathbf{B}\mathbf{n}^{(i+1)}-\mathbf{d}, \tag{12.18}$$

where $\mathbf{u}$ is the Lagrange multiplier and the Lagrange function is

$$L(\mathbf{w},\mathbf{n},\mathbf{u}) = f(\mathbf{w}) + g(\mathbf{n}) + \mathbf{u}^\top(\mathbf{A}\mathbf{w}+\mathbf{B}\mathbf{n}-\mathbf{d}) + \frac{1}{2}\|\mathbf{A}\mathbf{w}+\mathbf{B}\mathbf{n}-\mathbf{d}\|^2. \tag{12.19}$$

This method is known as the alternating direction method of multipliers (ADMM) [∗12-3∗]. It is worth noting that this algorithm does not involve fine-tuning parameters such as a learning rate.

EXERCISE 12-4. If the penalty term is $\lambda\|\mathbf{G}\mathbf{w}\|_1$ with $\mathbf{G}$ being positive-definite, write down the ADMM algorithm.

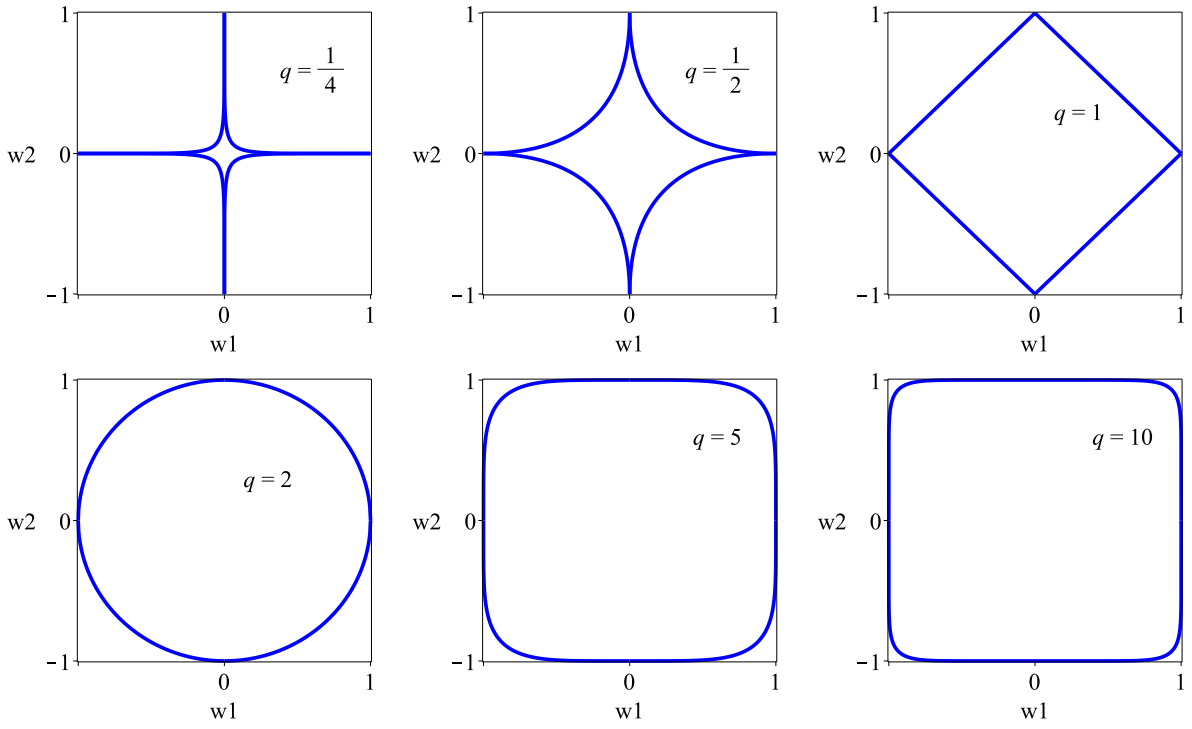

FIG. 12.10: A few examples of the ℓ-q constraint.

More generally, the ℓ-2 and ℓ-1 constraints can be extended to the ℓ-q constraint as $\|\mathbf{w}\|_q \le R^2, 0 < q < \infty$, where $\|\cdots\|_q$ denotes the q-norm defined by $\|\mathbf{w}\|_q = [\sum_{j=1}^m |w_j|^q]^{1/q}$. FIG. 12.10 illustrates several examples of the ℓ-q constraint. From the figure, one observes that solutions to learning problems with an ℓ-2 loss function become sparse when $0 < q \le 1$. In particular, for $q = 0$ and $q \to \infty$, the norms are defined as $\|\mathbf{w}\|_0 = \sum_{j=1}^n \delta(w_j \neq 0)$ and $\|\mathbf{w}\|_\infty = \max(|w_1|,\cdots,|w_n|)$, respectively.

One may also consider hybrid constraints, such as the ℓ-1-2 form $(1-\sigma)\|\mathbf{w}\|_1 + \sigma\|\mathbf{w}\|^2 \le R^2$ with $0 \le \sigma \le 1$, known as the elastic net [∗12-4∗]. The corresponding optimization problem becomes

$$(\mathbf{w}^*,\mathbf{n}^*) = \underset{\mathbf{w},\mathbf{n}}{\operatorname{argmin}}\left[\frac{1}{2}\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right)^\top\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right) + \eta\|\mathbf{w}\|^2 + \lambda\|\mathbf{n}\|_1\right], \tag{12.20}$$

with $\mathbf{w} = \mathbf{n}$, and the optimal solution can be obtained in a similar manner as

$$\boxed{\mathbf{w}^{(i+1)} = \left(\vec{\Phi}^\top\vec{\Phi} + (\eta+1)\vec{1}\right)^{-1}\left(\vec{\Phi}^\top\mathbf{y} + \mathbf{n}^{(i)} - \mathbf{u}^{(i)}\right).} \tag{12.21}$$

We do not pursue further discussion of these extensions here.

§12.5 Huber's Loss Function and Robustness

One may naturally ask what happens if the ℓ-2 loss function itself is replaced by an ℓ-1 loss function. For example, TAB. 12.1 shows eight normal data samples together with one outlier point, where the normal samples are generated by $y^{(i)} \sim \text{Unif}[3.7, 4.3]$, while the outlier is located at $(6.5, -10.20)$. Using the simplest learning model, namely $f_{\mathbf{w}}(x) = w_0$, together with the ℓ-2 loss function, one finds that the optimal learning parameter is $w_0^* = \overline{y^{(i)}} \approx 2.40$.

$x^{(i)}$	1	2	3	4	5	6	7	8	6.5
$y^{(i)}$	3.70	3.75	4.06	4.23	4.28	3.81	4.01	3.94	−10.20

TAB. 12.1: A series of eight normal data samples and an outlier point located at (6.5,−10.20).

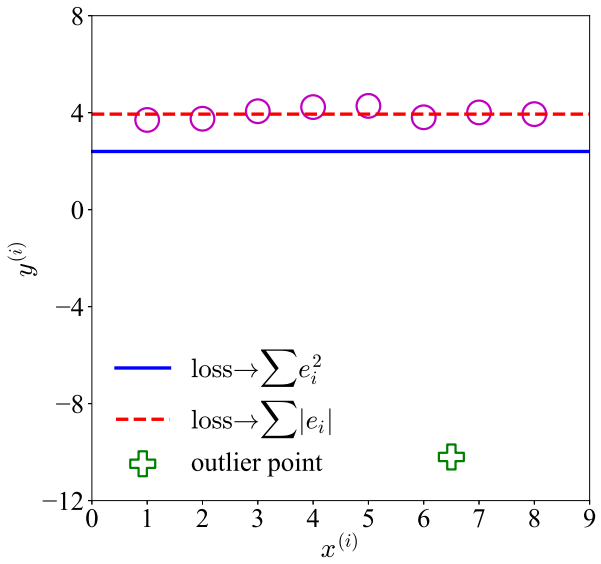

FIG. 12.11: Example on the outlier sample.

The resulting learning curve is represented by the solid blue line in FIG. 12.11. It is clear that, once the outlier is included in the optimization for w_0^*, the final prediction changes significantly. This is not the desired result. The reason why the squared loss function can lead to a large deviation in the prediction is simply that the error itself is amplified by the squaring operation. This raises the question: What type of learning model can avoid such a large deviation in the presence of outliers?

By examining the samples more carefully, one finds that the median of the eight normal samples, without the outlier, is determined by 3.94 and 4.01, giving a median of about 3.98, as indicated by the red dashed line in FIG. 12.11. If the outlier is included, the median changes only to about 3.94. This shows that the change is much smaller than that obtained using the ℓ-2 loss function. Mathematically, the optimal solution given by the median corresponds to the loss function

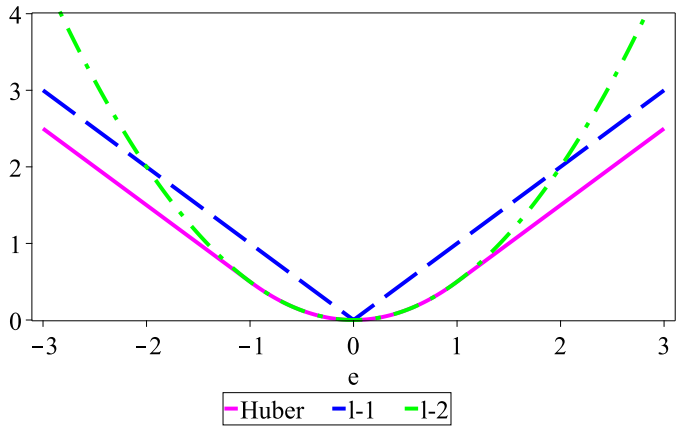

FIG. 12.12: Huber's loss function.

$J(w_0)=\sum_{i=1}^{m}|e_i|, e_i = w_0 - y^{(i)}, w_0^* = \text{argmin}_{w_0} J(w_0)$, namely

$$\boxed{w_0^* = \text{median}(y^{(i)}).} \tag{12.22}$$

The loss function written in terms of the 1-norm is usually called the ℓ-1 loss function, in contrast to the ℓ-2 loss function used so far. For a general learning model $f_{\mathbf{w}}(x)$, the ℓ-1 loss function can be written as

$$\boxed{\mathbf{w}^* = \underset{\mathbf{w}}{\text{argmin}}\, J_{\ell-1}(\mathbf{w}),\quad J_{\ell-1}(\mathbf{w}) = \sum_{i=1}^{m}\left|f_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)}\right|.} \tag{12.23}$$

EXERCISE 12-5. Explain the result (13.9) and give an example.

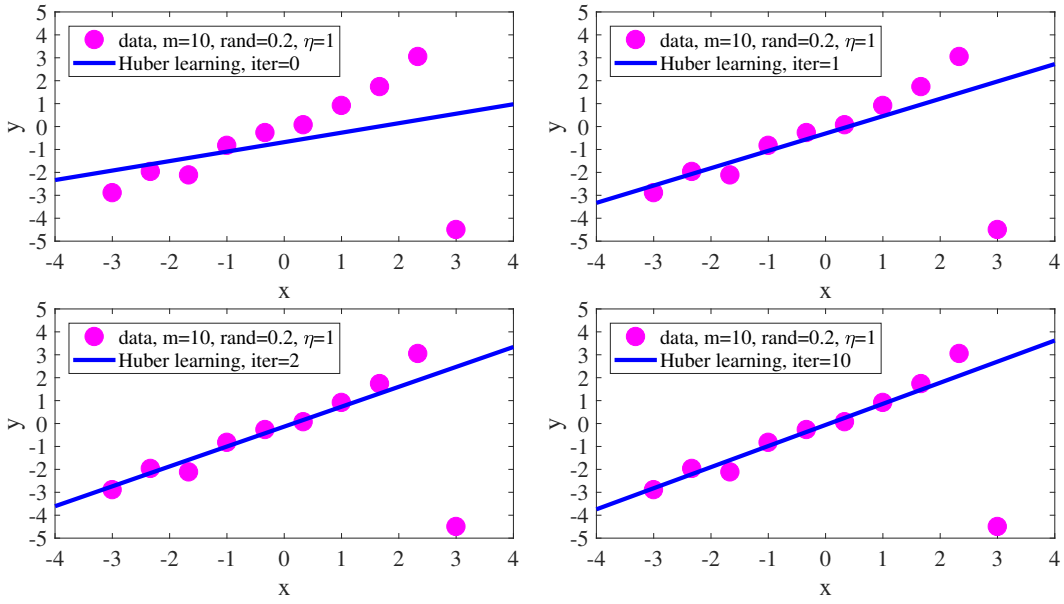

FIG. 12.13: The iterative processes under the Huber's loss function.

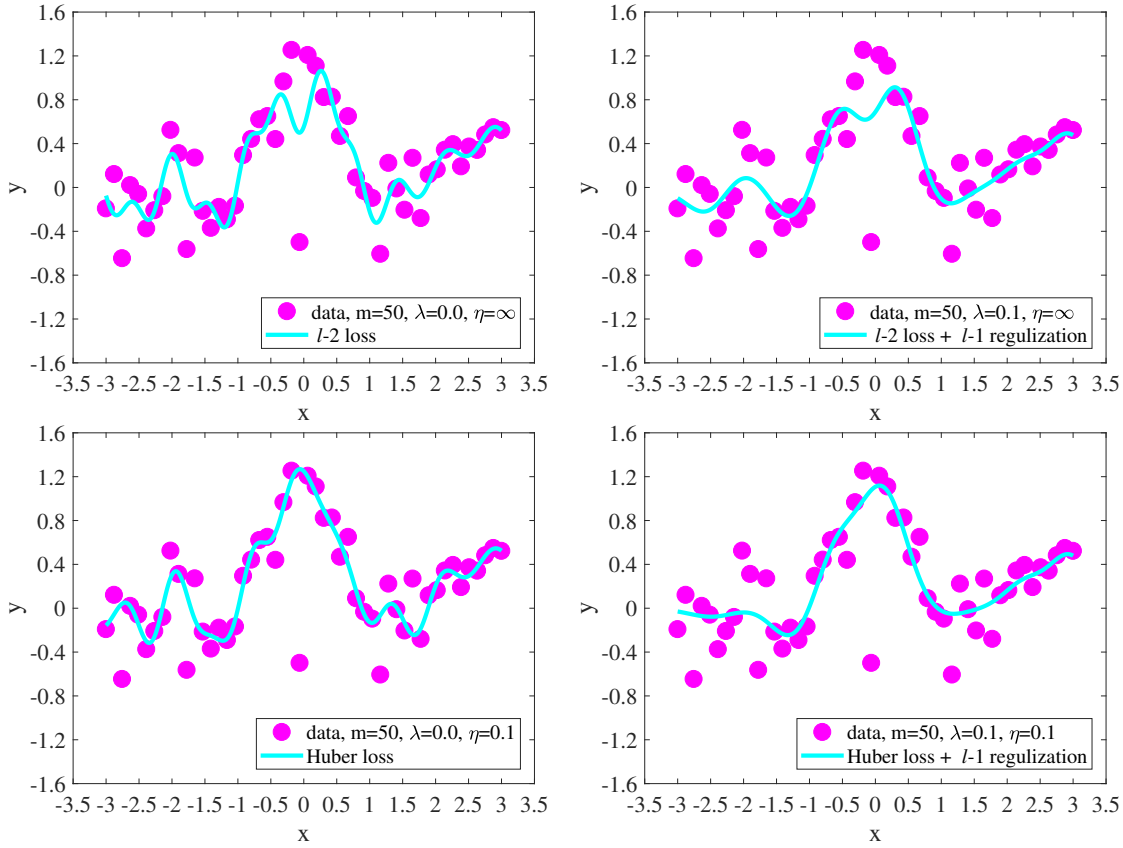

FIG. 12.14: Example using Huber's loss function and the ℓ-1 constraint.

Furthermore, a hybrid loss function composed of both ℓ-1 and ℓ-2 parts is also widely used:

$$\chi^{\mathrm{H}}(e)=\begin{cases} 2^{-1}e^2, & |e|\le\eta; \\ \eta|e|-2^{-1}\eta^2, & |e|>\eta, \end{cases} \tag{12.24}$$

This is known as Huber's loss function. Its geometrical meaning is straightforward: when the magnitude of the error e is smaller than the threshold η, the loss function is dominated by the ℓ-2 form; when the magnitude of the error exceeds η, the loss function becomes effectively ℓ-1. Since the loss is of ℓ-1 type for large errors, the error itself is no longer amplified, while for small errors the squared amplification is still acceptable. FIG. 12.12 shows a sketch of Huber's loss function together with the other two losses.

The solution procedure for Huber's loss function can be summarized as follows: (a) initialize the learning parameter $\mathbf{w}$, for instance by least-squares, $\mathbf{w}\leftarrow(\vec{\Phi}^{\top}\vec{\Phi})^{-1}\vec{\Phi}^{\top}\mathbf{y}$; (b) calculate the weight matrix $\vec{\Theta}$ based on the current $\mathbf{w}$, namely $\vec{\Theta}=\mathrm{diag}(\theta^{(1)},\cdots,\theta^{(m)})$, where $\theta^{(i)}=1$ if $|e_i|\le\eta$ and $\theta^{(i)}=\eta/|e_i|$ if $|e_i|>\eta$, with $e_i=f_{\mathbf{w}}(\mathbf{x}^{(i)})-y^{(i)}$ being the corresponding error; and (c) update the learning parameter according to $\mathbf{w}\leftarrow(\vec{\Phi}^{\top}\vec{\Theta}\vec{\Phi})^{-1}\vec{\Phi}^{\top}\vec{\Theta}\mathbf{y}$. FIG. 12.13 presents an example using Huber's loss function. Here the data are generated by $y^{(i)}=x^{(i)}+\mathrm{rand}\times\delta$ with the learning model $f_{\mathbf{w}}(x)=w_0+w_1x$, where $\delta\sim\mathrm{Unif}[0,1]$ and "rand" characterizes the strength of the random fluctuation. In addition, the outlier point is chosen to be $(3,-4)$. The figure clearly shows that, after roughly two iterations, the learning process has essentially excluded the outlier. The difference between the learning line after two itera-

tions and that after 10 iterations is already quite small, indicating rapid convergence in this example. Moreover, one naturally expects that as the threshold parameter η increases, Huber's loss function eventually approaches the ℓ-2 loss function. In such cases, the algorithm can no longer effectively suppress the influence of outliers.

The ability of Huber's loss function, or effectively the ℓ-1 loss function as in the median case, to suppress the influence of outliers shows that the obtained solution is robust. This property is referred to as robustness. Since the ℓ-1 loss function provides robustness against outliers, while the ℓ-1 constraint promotes sparsity, these two mechanisms are often combined to address more general and more complicated learning problems. FIG. 12.14 shows an example in which the learning problem adopts both Huber's loss function and the ℓ-1 constraint. The learning model used here is the Gauss kernel model, namely $f_{\mathbf{w}}(\mathbf{x}) = \sum_{j=1}^{m} w_j \exp(-\|\mathbf{x}-\mathbf{x}^{(j)}\|^2/2\nu^2)$ where $\nu = 0.4$. Clearly, the limit $\eta \to \infty$ corresponds to the ℓ-2 loss function. In the left panel of the first line, only the ℓ-2 loss function is used, and one sees that the learning model is strongly influenced by the two outlier points located roughly in the lower part of the figure. When the ℓ-1 constraint is included, the performance of the ℓ-2 loss function improves, in the sense that the learning model is less affected by the outliers, as shown in the right panel of the first line. On the other hand, the combination of Huber's loss function and the ℓ-1 constraint not only avoids the influence of outliers, but also smooths the overall learning curve, as shown in the right panel of the second line. This should be compared with the case without the ℓ-1 constraint, shown in the left panel of the second line, where the learning curve avoids the outliers but remains more irregular. The irregularity of the learning curve indicates a large variance, as discussed in Lecture 5.

§12.6 Gaussian Mixing Model and EM Algorithm

As mentioned above, clustering algorithms are closely related to mixtures of several Gaussian distributions. We now make this connection more explicit. We first consider a maximum-likelihood motivation for using the k-means criterion. Suppose that the data samples are generated from an equal-weight mixture of k spherical, well-separated Gaussian densities centered at $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$, each with variance one in every direction. Then the density of the mixture is given by

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}} \frac{1}{K} \sum_{k=1}^{K} e^{-\|\mathbf{x}-\vec{\mu}_k\|^2}. \tag{12.25}$$

Denote by $\vec{\mu}(\mathbf{x})$ the center nearest to $\mathbf{x}$. Since the exponential function decays very rapidly, if $\mathbf{x}$ is noticeably closer to its nearest center than to any other center, then the sum is dominated by its largest term. Thus, we may approximate $\sum_{k=1}^{K} e^{\|\mathbf{x}-\vec{\mu}_k\|^2}$ by $e^{-\|\mathbf{x}-\vec{\mu}(\mathbf{x})\|^2}$, and therefore

$$\boxed{p(\mathbf{x}) \approx \frac{1}{(2\pi)^{d/2} K} \exp\left(-\|\mathbf{x}-\vec{\mu}(\mathbf{x})\|^2\right).} \tag{12.26}$$

The likelihood of drawing samples $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(m)}$ from this mixture, given the centers $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$, is approximately

$$\frac{1}{K^m} \frac{1}{(2\pi)^{md/2}} \prod_{i=1}^{m} \exp\left(-\|\mathbf{x}^{(i)} - \vec{\mu}(\mathbf{x}^{(i)})\|^2\right) = c \exp\left(-\sum_{i=1}^{m} \|\mathbf{x}^{(i)} - \vec{\mu}(\mathbf{x}^{(i)})\|^2\right), \tag{12.27}$$

where c is a constant. Therefore, minimizing the sum of squared distances to the cluster centers gives the maximum-likelihood estimate of $\vec{\mu}_1, \vec{\mu}_2, \cdots, \vec{\mu}_K$. This provides a probabilistic motivation for using the sum of squared distances to the cluster centers, as done above.

A Gaussian mixture model (GMM) is a mixture of several Gaussian distributions [∗12-5∗], namely

$$\boxed{p(\mathbf{x}) = \sum_{k=1}^{K} \zeta_k \mathcal{N}(\mathbf{x}|\vec{\mu}_k, \vec{\Sigma}_k).} \tag{12.28}$$

If the number of components K is sufficiently large, a Gaussian mixture can approximate a very broad class of distributions. Introduce the vector $\mathbf{z} \in \mathrm{R}^K$, defined by

$$z_k \in \{0,1\}, \ \sum_{k=1}^{K} z_k = 1. \tag{12.29}$$

The joint distribution of $\mathbf{x}$ and $\mathbf{z}$ can be obtained from the marginal distribution of $\mathbf{z}$ and the conditional distribution of $\mathbf{x}$ given $\mathbf{z}$, namely $p(\mathbf{x},\mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$. We say that the variable $\mathbf{z}$ determines the component from which $\mathbf{x}$ is generated, and $\mathbf{z}$ is therefore a latent variable. The marginal distribution of the latent variable can be written as $p(\mathbf{z}) = \prod_{k=1}^{K} \zeta_k^{z_k}$, where $p(z_k = 1) = \zeta_k$. Similarly, the conditional distribution is $p(\mathbf{x}|z_k = 1) = \mathcal{N}(\mathbf{x}|\vec{\mu}_k, \vec{\Sigma}_k)$. Consequently, one has $p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\vec{\mu}_k, \vec{\Sigma}_k)^{z_k}$. Finally, the marginal distribution of $\mathbf{x}$ is written as $p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^{K} \zeta_k \mathcal{N}(\mathbf{x}|\vec{\mu}_k, \vec{\Sigma}_k)$. For multiple data samples, the latent variables can be generalized correspondingly to $\mathbf{z}^{(i)}$. Moreover, if the latent variable is continuous, then the summation in these expressions should be replaced by integration, i.e., $\sum_{\mathbf{z}^{(i)}} \to \int \mathrm{d}\mathbf{z}^{(i)}$. Define the responsibility number as

$$\phi(z_k) = p(z_k = 1|\mathbf{x}) = \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{k'} p(z_{k'} = 1)p(\mathbf{x}|z_{k'} = 1)} = \frac{\zeta_k \mathcal{N}(\mathbf{x}|\vec{\mu}_k, \vec{\Sigma}_k)}{\sum_{k'} \zeta_{k'} \mathcal{N}(\mathbf{x}|\vec{\mu}_{k'}, \vec{\Sigma}_{k'})}. \tag{12.30}$$

Thus, one can interpret ζ_k as the prior probability for $z_k = 1$, while $\phi(z_k)$ is the posterior probability after observing the data point $\mathbf{x}$. As an example, FIG. 12.15 shows the calculation for handwritten digits obtained by analyzing a dataset containing about 5000 samples with a $K = 6$ GMM. One expects that as K increases, the prediction becomes better. In fact, the $K = 1$ GMM calculation simply gives the mean image of all images in the dataset.

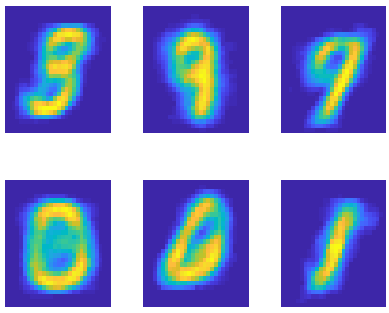

FIG. 12.15: Hand-writing numbers using $K = 6$ Gaussian distribution.

We now maximize the likelihood with respect to the parameters $\vec{\zeta}, \vec{\mu}$ and $\vec{\Sigma}$:

$$\ln p(\mathbf{X}|\vec{\zeta}, \vec{\mu}, \vec{\Sigma}) = \sum_{i=1}^{m} \ln \left[\sum_{k=1}^{K} \zeta_k \mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k, \vec{\Sigma}_k) \right]. \tag{12.31}$$

Taking the derivative with respect to $\vec{\mu}_k$ and setting it to zero gives

$$0 = \sum_{i=1}^{m} \frac{\zeta_k \mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k, \vec{\Sigma}_k)}{\sum_{j=1}^{K} \zeta_j \mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_j, \vec{\Sigma}_j)} \cdot \vec{\Sigma}_k^{-1} \cdot \left(\mathbf{x}^{(i)} - \vec{\mu}_k\right) = \sum_{i=1}^{m} \phi(z_k^{(i)}) \mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k, \vec{\Sigma}_k) \cdot \vec{\Sigma}_k^{-1} \cdot \left(\mathbf{x}^{(i)} - \vec{\mu}_k\right), \tag{12.32}$$

Multiplying both sides of this equation by $\vec{\Sigma}_k$ leads to $\vec{0} = \sum_{i=1}^{m} \phi(z_k^{(i)})(\mathbf{x}^{(i)} - \vec{\mu}_k)$, or

$$\boxed{\vec{\mu}_k = \frac{1}{m_k} \sum_{i=1}^{m} \phi(z_k^{(i)}) \mathbf{x}^{(i)}, \quad m_k = \sum_{i=1}^{m} \phi(z_k^{(i)}),} \tag{12.33}$$

which is very similar to the formula obtained in the k-means clustering algorithm. The difference is that $\phi(z_k^{(i)})$ replaces the responsibility $r_k^{(i)}$. In addition, m_k still represents the effective number of data samples in cluster k.

Next, by taking the derivative of the logarithmic likelihood $\ln p$ with respect to the covariance matrix $\vec{\Sigma}_k$ and setting it to zero, one obtains

$$\boxed{\vec{\Sigma}_k = \frac{1}{m_k} \sum_{i=1}^{m} \phi(z_k^{(i)}) \left(\mathbf{x}^{(i)} - \vec{\mu}_k\right) \left(\mathbf{x}^{(i)} - \vec{\mu}_k\right)^{\top}.} \tag{12.34}$$

Finally, we take the derivative of $\ln p$ with respect to the mixing coefficient ζ_k, while introducing a Lagrange multiplier through $\ln p(\mathbf{X}|\vec{\zeta}, \vec{\mu}, \vec{\Sigma}) + \lambda(\sum_{k=1}^{K} \zeta_k - 1)$:

$$0 = \sum_{i=1}^{m} \frac{\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k, \vec{\Sigma}_k)}{\sum_{j=1}^{K} \zeta_j \mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_j, \vec{\Sigma}_j)} + \lambda = 0. \tag{12.35}$$

Multiplying both sides of this equation by ζ_k and summing over all possible k, one obtains $\lambda = -m$ and therefore $\zeta_k = m_k/m$. Thus, the mixing coefficient ζ_k represents the fraction of the effective number of data points assigned to cluster k relative to the total number of data points. This procedure is known as the expectation-maximization (EM) algorithm [*12-6*].

To present the EM algorithm in a general form, denote the full dataset by $\mathbf{X}$, whose rows are $\mathbf{x}^{(i),\top}$, and denote the full set of latent variables by $\mathbf{Z}$, whose rows are $\mathbf{z}^{(i),\top}$. The joint distribution of $\mathbf{X}, \mathbf{Z}$ is denoted by $p(\mathbf{X}, \mathbf{Z}|\mathbf{w})$, which is also the likelihood of the parameter $\mathbf{w}$. Our goal is to maximize the likelihood $p(\mathbf{X}|\mathbf{w})$, namely the likelihood of the incomplete dataset $\mathbf{X}$. In contrast, $p(\mathbf{X}, \mathbf{Z}|\mathbf{w})$ is the likelihood of the complete dataset $(\mathbf{X}, \mathbf{Z})$. The former, $p(\mathbf{X}|\mathbf{w})$, is usually difficult to calculate directly, while the latter, $p(\mathbf{X}, \mathbf{Z}|\mathbf{w})$, is relatively easier to handle. Therefore, the strategy is to proceed by taking the expectation of the complete-data likelihood. For the GMM, one has

$$p(\mathbf{X}, \mathbf{Z}|\vec{\mu}, \vec{\Sigma}, \vec{\zeta}) = \prod_{i=1}^{m} \prod_{k=1}^{K} \zeta_k^{z_k^{(i)}} \mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k, \vec{\Sigma}_k)^{z_k^{(i)}}, \tag{12.36}$$

$$\ln p(\mathbf{X},\mathbf{Z}|\vec{\mu},\vec{\Sigma},\vec{\zeta}) = \sum_{i=1}^{m}\sum_{k=1}^{K} z_k^{(i)}\big[\ln\zeta_k + \ln\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k,\vec{\Sigma}_k)\big], \tag{12.37}$$

where the logarithm converts the product into a summation. Using $p(\mathbf{z})$ and $p(\mathbf{x}|\mathbf{z})$, together with Bayes' theorem, one obtains the posterior distribution of $\mathbf{Z}$ as

$$p(\mathbf{Z}|\mathbf{X},\vec{\mu},\vec{\Sigma},\vec{\zeta}) \sim \prod_{i=1}^{m}\prod_{k=1}^{K}\big[\zeta_k\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k,\vec{\Sigma}_k)\big]^{z_k^{(i)}}. \tag{12.38}$$

Consequently,

$$\begin{aligned}\mathrm{E}[z_k^{(i)}] &= \left[\sum_{\mathbf{z}^{(i)}} z_k^{(i)}\prod_{j=1}^{K}\big[\zeta_j\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_j,\vec{\Sigma}_j)\big]^{z_j^{(i)}}\right]\Bigg/\left[\sum_{\mathbf{z}^{(i)}}\prod_{j=1}^{K}\big[\zeta_j\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_j,\vec{\Sigma}_j)\big]^{z_j^{(i)}}\right]\\ &= \frac{\zeta_k\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k,\vec{\Sigma}_k)}{\sum_{j=1}^{K}\zeta_j\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_j,\vec{\Sigma}_j)} = \phi(z_k^{(i)}),\end{aligned} \tag{12.39}$$

namely, the expectation of $z_k^{(i)}$ is precisely the responsibility $\phi(z_k^{(i)})$. The expectation of the complete-data log-likelihood is therefore

$$\boxed{\mathrm{E}_{\mathbf{Z}}\big[\ln p(\mathbf{X},\mathbf{Z}|\vec{\mu},\vec{\Sigma},\vec{\zeta})\big] = \sum_{i=1}^{m}\sum_{k=1}^{K}\phi(z_k^{(i)})\big[\ln\zeta_k + \ln\mathcal{N}(\mathbf{x}^{(i)}|\vec{\mu}_k,\vec{\Sigma}_k)\big].} \tag{12.40}$$

If the width of the Gaussian is chosen as $\epsilon^2\vec{1}$, one can write

$$\phi(z_k^{(i)}) = \frac{\zeta_k\exp[-\|\mathbf{x}^{(i)}-\vec{\mu}_k\|^2/2\epsilon]}{\sum_{j=1}^{K}\zeta_j\exp[-\|\mathbf{x}^{(i)}-\vec{\mu}_j\|^2/2\epsilon]}, \tag{12.41}$$

and in this case when $\epsilon\to 0$, the term in the denominator with the smallest value of $\|\mathbf{x}^{(i)}-\vec{\mu}_j\|^2$ decays the slowest and therefore dominates the sum. Denote this term by k'. If $k=k'$, then the numerator has the same dominant contribution and one obtains $\phi(z_k^{(i)})=1$. If $k\neq k'$, the numerator is negligible compared with the denominator and $\phi(z_k^{(i)})=0$. Consequently, we have shown that $\phi(z_k^{(i)})\to r_k^{(i)}$.

Next, considering the structure of the normal distribution, the factor $-2^{-1}\|\mathbf{x}^{(i)}-\vec{\mu}_k\|^2$ appears after taking the logarithm of the likelihood. Thus, in this limit one has

$$\boxed{\mathrm{E}_{\mathbf{Z}}\big[\ln p(\mathbf{X},\mathbf{Z}|\vec{\mu},\vec{\Sigma},\vec{\zeta})\big] \to -\frac{1}{2}\sum_{i=1}^{m}\sum_{k=1}^{K} r_k^{(i)}\|\mathbf{x}^{(i)}-\vec{\mu}_k\|^2,} \tag{12.42}$$

which is the loss function of k-means clustering, see (12.4). Therefore, maximizing the likelihood is equivalent to minimizing the loss function (12.4). This shows that the GMM reduces to k-means clustering in the limit $\epsilon\to 0$.

EXERCISE 12-6. Write down the flow of the EM algorithm.

FIG. 12.16 shows the EM calculation for a GMM with two distinct data clusters [∗12-7∗]. The initial clusters are marked by two circles with the "×" symbol. From the figure, one can see that the blue

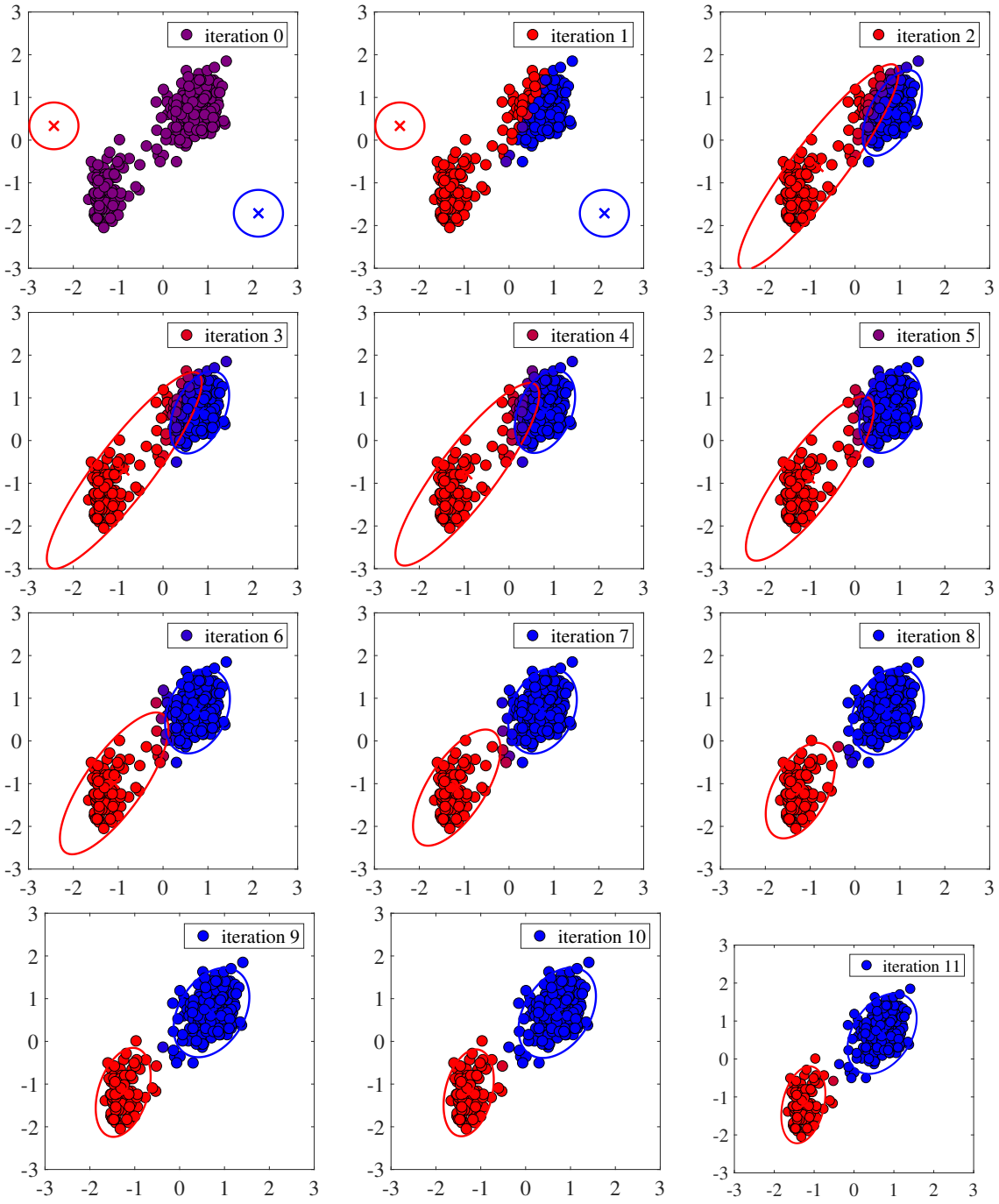

FIG. 12.16: EM calculation for the GMM.

cluster converges much faster than the red cluster. A significant change in the clustering appears after the second iteration, as shown in the third panel of the first line, and after about 10 iterations the final clustering pattern is formed.

EXERCISE 12-7. From FIG. 12.16, can one infer the tendency of the likelihood function?

§12.7 *Mathematical Aspects and Effectiveness of EM Algorithm

Let us discuss the effectiveness of the EM algorithm in more detail. The fundamental aim is to maximize the likelihood function $p(\mathbf{X}|\mathbf{w}) = \sum_{\mathbf{Z}} p(\mathbf{X},\mathbf{Z}|\mathbf{w})$, which is sometimes very difficult to evaluate directly. On the other hand, as mentioned above, the joint distribution for the complete data, $p(\mathbf{X},\mathbf{Z}|\mathbf{w})$, is relatively easier to calculate.

To proceed, we introduce the following two functions:

$$\boxed{\mathcal{L}(q,\mathbf{w}) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z}|\mathbf{w})}{q(\mathbf{Z})},\ \mathrm{KL}(q\|p) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X},\mathbf{w})}{q(\mathbf{Z})}.} \tag{12.43}$$

Here $\mathcal{L}$ is a functional of the distribution $q(\mathbf{Z})$ (which is assumed to be normalized, for example), and is also a function of the parameter $\mathbf{w}$. By rewriting $\mathcal{L}$, one obtains

$$\begin{aligned}
\mathcal{L}(q,\mathbf{w}) &= \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z}|\mathbf{w})}{q(\mathbf{Z})} \\
&= \sum_{\mathbf{Z}} q(\mathbf{Z}) \big[\ln p(\mathbf{X},\mathbf{Z}|\mathbf{w}) - \ln q(\mathbf{Z})\big] \\
&= \sum_{\mathbf{Z}} q(\mathbf{Z}) \big[\ln p(\mathbf{X}|\mathbf{w}) + \ln(\mathbf{Z}|\mathbf{X},\mathbf{w}) - \ln q(\mathbf{Z})\big] \\
&= \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln p(\mathbf{X}|\mathbf{w}) + \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X},\mathbf{w})}{q(\mathbf{Z})} \\
&= \ln p(\mathbf{X}|\mathbf{w}) - \mathrm{KL}(q\|p),
\end{aligned} \tag{12.44}$$

which gives $\ln p(\mathbf{X}|\mathbf{w}) = \mathcal{L}(q,\mathbf{w}) + \mathrm{KL}(q\|p)$. This relation also defines the Kullback–Leibler (KL) divergence. Since the KL divergence is always non-negative, which follows from the convexity property, one has $\mathcal{L}(q,\mathbf{w}) \le \ln p(\mathbf{X}|\mathbf{w})$, where equality holds when $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X},\mathbf{w})$. We therefore call $\mathcal{L}(q,\mathbf{w})$ the lower-bound function of the likelihood $\ln p(\mathbf{X}|\mathbf{w})$, as illustrated in FIG. 12.17.

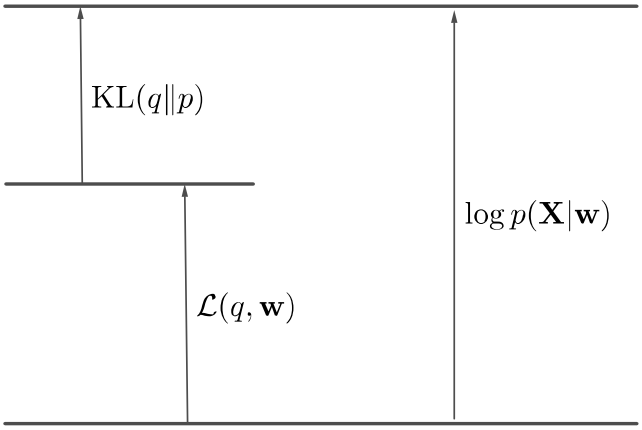

FIG. 12.17: Decomposition of $\ln p(\mathbf{X}|\mathbf{w})$, here $\log \longleftrightarrow \ln$.

In the E-step, we search for the distribution $q(\mathbf{Z})$ that maximizes the lower-bound function. The result is simply $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\mathrm{old}})$, because the KL divergence is then zero, as shown in the left panel

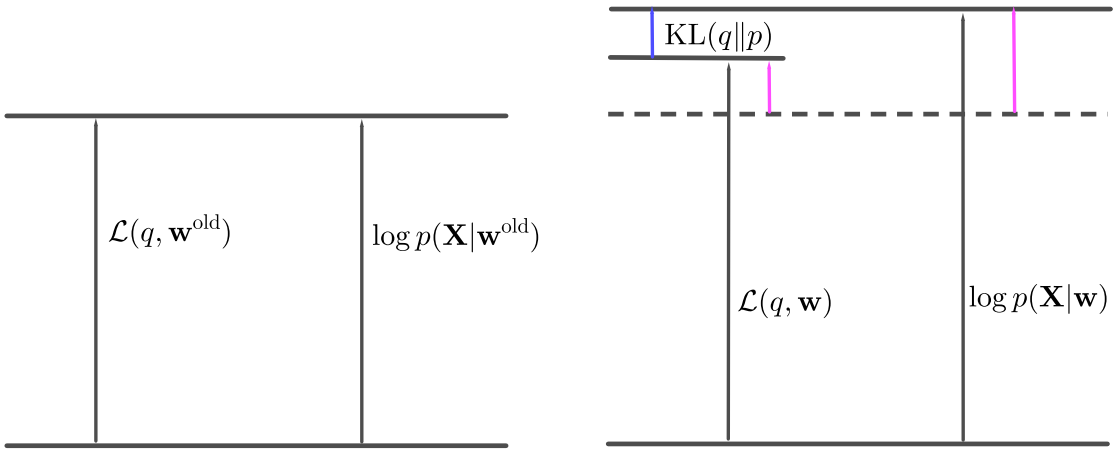

FIG. 12.18: Left: Maximizing the low-limit function $\mathcal{L}(q,\mathbf{w})$, right: Maximizing the likelihood $\ln p(\mathbf{X}|\mathbf{w})$, here $\log \longleftrightarrow \ln$.

of FIG. 12.18. In the M-step, the distribution $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\text{old}})$ is fixed. The goal is to update the parameter $\mathbf{w}$, namely to find a new parameter $\mathbf{w}^{\text{new}}$, so as to increase the lower-bound function, except when it has already reached its maximum. Consequently, the likelihood function also increases. Since $q(\mathbf{Z})$ is fixed during this step, it is generally no longer equal to $p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\text{new}})$, and hence a nonzero KL divergence is generated during the M-step. In this sense, the increase in the lower-bound function is smaller than that in the likelihood, as shown in the right panel of FIG. 12.18. After the E-step, one has

$$\begin{aligned}\mathcal{L}(q,\mathbf{w}) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\text{old}}) \ln p(\mathbf{X},\mathbf{Z}|\mathbf{w}) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\text{old}}) \ln p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\text{old}}) \\ &= Q(\mathbf{w},\mathbf{w}^{\text{old}}) + \text{terms independent of } \mathbf{w}, \end{aligned} \tag{12.45}$$

i.e., the quantity maximized during the M-step is the expectation of the likelihood for the complete dataset. Furthermore, since the KL divergence is zero after the E-step, one naturally has

$$\frac{\partial}{\partial \mathbf{w}} \mathrm{KL}(q\|p) = -\frac{\partial}{\partial \mathbf{w}} \left[\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X},\mathbf{w})}{q(\mathbf{Z})} \right] = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \frac{1}{p(\mathbf{Z}|\mathbf{X},\mathbf{w})} \frac{\partial}{\partial \mathbf{w}} p(\mathbf{Z}|\mathbf{X},\mathbf{w}), \tag{12.46}$$

and the posterior $p(\mathbf{Z}|\mathbf{X},\mathbf{w})$ reaches its maximum at $\mathbf{w}^{\text{old}}$. Consequently, $\partial\, p(\mathbf{Z}|\mathbf{X},\mathbf{w}^{\text{old}})/\partial \mathbf{w} = 0$, indicating that the lower-bound function and the likelihood are tangent to each other. For the function $Q(\mathbf{w},\mathbf{w}^{\text{old}})$, Taylor's expansion gives $Q(\mathbf{w},\mathbf{w}^{\text{old}}) \approx Q(\mathbf{w}',\mathbf{w}^{\text{old}}) + Q'(\mathbf{w},\mathbf{w}^{\text{old}})(\mathbf{w}-\mathbf{w}') + 2^{-1}Q''(\mathbf{w}',\mathbf{w}^{\text{old}})(\mathbf{w}-\mathbf{w}')^2 + \cdots$. Taking $Q'(\mathbf{w},\mathbf{w}^{\text{old}}) = 0$ gives $\mathbf{w}^{\text{new}}$, as illustrated in the left panel of FIG. 12.19. The newly obtained parameter $\mathbf{w}^{\text{new}}$ is then used as the next $\mathbf{w}^{\text{old}}$ in the following E-step. This constitutes the full procedure of the EM algorithm, as shown in the right panel of FIG. 12.19.

The EM algorithm in the continuous case is very similar. For example, for the lower-bound function,

$$\mathcal{L}(q_i(\mathbf{z}^{(i)}),\mathbf{w}) = \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{w})}{q_i(\mathbf{z}^{(i)})} \mathrm{d}\mathbf{z}^{(i)}$$

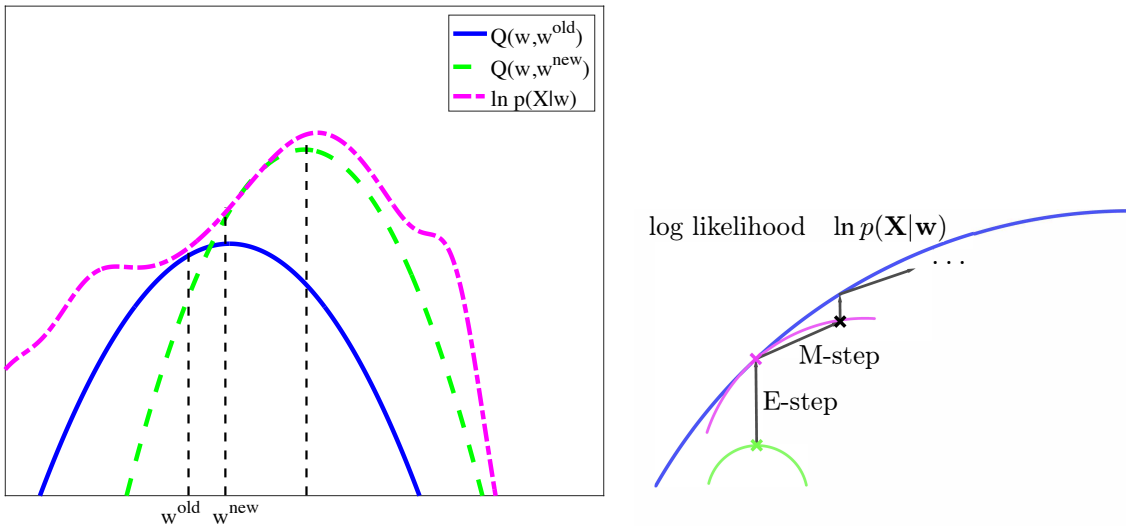

FIG. 12.19: Sketch of the convergence of the EM algorithm (left) and the iteration steps involving the EM algorithm (right).

$$\leq \sum_{i=1}^{m} \ln\left[\int q_i(\mathbf{z}^{(i)}) \frac{p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{w})}{q_i(\mathbf{z}^{(i)})} \mathrm{d}\mathbf{z}^{(i)}\right] = \sum_{i=1}^{m} \ln\left[\int p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{w}) \mathrm{d}\mathbf{z}^{(i)}\right], \tag{12.47}$$

where Jensen's inequality, $\mathrm{E}[\ln x] \leq \ln[\mathrm{E}[x]]$, has been used. In the E-step,

$$\begin{aligned}
\mathcal{L}(q_i(\mathbf{z}^{(i)}),\mathbf{w}) &= \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{w})}{q_i(\mathbf{z}^{(i)})} \mathrm{d}\mathbf{z}^{(i)} = \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)},\mathbf{w}) p(\mathbf{x}^{(i)}|\mathbf{w})}{q_i(\mathbf{z}^{(i)})} \mathrm{d}\mathbf{z}^{(i)} \\
&= \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln p(\mathbf{x}^{(i)}|\mathbf{w}) \mathrm{d}\mathbf{z}^{(i)} - \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{q_i(\mathbf{z}^{(i)})}{p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)},\mathbf{w})} \mathrm{d}\mathbf{z}^{(i)} \\
&= \sum_{i=1}^{m} \ln p(\mathbf{x}^{(i)}|\mathbf{w}) - \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{q_i(\mathbf{z}^{(i)})}{p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)},\mathbf{w})} \mathrm{d}\mathbf{z}^{(i)},
\end{aligned} \tag{12.48}$$

and therefore

$$\begin{aligned}
\widehat{q}_i(\mathbf{z}^{(i)}) &= \max_{q_i(\mathbf{z}^{(i)})} \left[-\ln p(\mathbf{x}^{(i)}|\mathbf{w}) - \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{q_i(\mathbf{z}^{(i)})}{p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)},\mathbf{w})} \mathrm{d}\mathbf{z}^{(i)} \right] \\
&= \min_{q_i(\mathbf{z}^{(i)})} \left[-\ln p(\mathbf{x}^{(i)}|\mathbf{w}) - \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)},\mathbf{w})}{q_i(\mathbf{z}^{(i)})} \mathrm{d}\mathbf{z}^{(i)} \right] \\
&= p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)},\mathbf{w}),
\end{aligned} \tag{12.49}$$

while in the M-step,

$$\mathbf{w} \leftarrow \max_{\mathbf{w}} \mathcal{L}(q_i(\mathbf{z}^{(i)}),\mathbf{w}) = \max_{\mathbf{w}} \left[\sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln \frac{p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{w})}{q_i(\mathbf{z}^{(i)})} \mathrm{d}\mathbf{z}^{(i)} \right]$$

$$= \max_{\mathbf{w}} \left[\sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} | \mathbf{w}) \mathrm{d}\mathbf{z}^{(i)} - \sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln q_i(\mathbf{z}^{(i)}) \mathrm{d}\mathbf{z}^{(i)} \right]$$
$$= \max_{\mathbf{w}} \left[\sum_{i=1}^{m} \int q_i(\mathbf{z}^{(i)}) \ln p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} | \mathbf{w}) \mathrm{d}\mathbf{z}^{(i)} \right]. \tag{12.50}$$

These steps are completely analogous to those in the discrete case.

§12.8 k-nearest-neighbor (Brief)

Finally, we discuss another clustering algorithm, namely the k-nearest-neighbor (KNN) method, whose name sounds very similar to the k-means clustering method. The basic idea of KNN is as follows: For any testing data point $\mathbf{x}$, calculate the distances between $\mathbf{x}$ and the existing data samples, then select the k samples whose distances to $\mathbf{x}$ are the smallest among all samples. Finally, determine which class among these K samples is closest to $\mathbf{x}$ and classify $\mathbf{x}$ into this class. Naturally, if $K = 1$, each data sample tends to define its own local region, indicating that the resulting clustering/classification can be unstable. In FIG. 12.20, we show an example of classification using the KNN algorithm with $K = 1$, where all data samples are separated. This gives the famous Voronoi diagram.

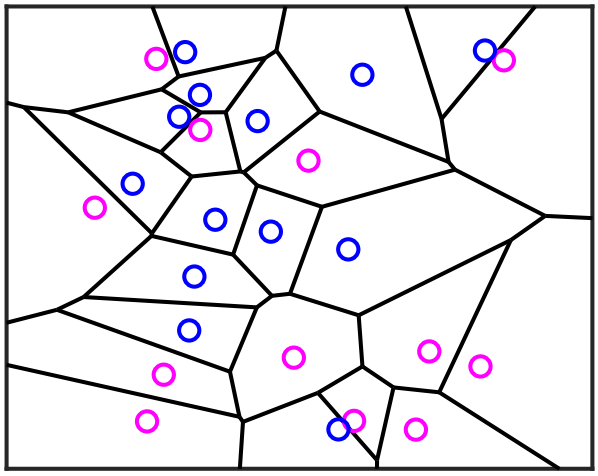

FIG. 12.20: KNN result for $K = 1$ (Voronoi diagram).

From a mathematical perspective, the probability density $p(\mathbf{x})$ of finding $\mathbf{x}$ in a generalized region R can be approximated as $p(\mathbf{x}) \approx P/V$, where $P = \int_R p(\mathbf{x}) \mathrm{d}\mathbf{x}$ and $V = \int_R \mathrm{d}\mathbf{x}$ are the probability and the volume over R, respectively. Moreover, one may approximate P by K/m, where K is an effective parameter and m is the number of samples. Comparing these two approximations, one immediately obtains $p(\mathbf{x}) \approx K/mV$. For a d-dimensional sphere with radius r, its volume is

$$\boxed{V(d) = \frac{\pi^{d/2} r^d}{\Gamma(d/2+1)} = \frac{2\pi^{d/2} r^d}{d\Gamma(d/2)},} \tag{12.51}$$

which can be derived as follows. Denote $d = n+2$ and introduce the spherical coordinates of the sphere equation as $(r, \theta_1, \theta_2, \cdots, \theta_n, \varphi)$, i.e.,

$$x_1 = r\prod_{i=1}^{n}\sin\theta_i\cos\varphi,\quad x_2 = r\prod_{i=1}^{n}\sin\theta_i\sin\varphi,\quad x_\beta = r\prod_{j=\beta-1}^{n}\sin\theta_j\cos\theta_{\beta-2}, \tag{12.52}$$

where the index β takes values from 3 to $n+1 = d-1$, and $x_{n+2} = r\cos\theta_n$. The volume element is given by $\mathrm{d}V_{n+2} = r^{n+1}\prod_{j=1}^{n}\sin^j\theta_j$, and after straightforward integration one obtains (12.51). An important observation is that $\lim_{d\to\infty} V(d) = 0$. The fact that the volume of a sphere (ball) approaches zero as $d\to\infty$ is one of its remarkable high-dimensional properties. We will discuss the volume of spheres in high dimensions in more detail in the next lecture.

Using the formula (12.51), the probability density estimate for the KNN algorithm is

$$\boxed{\widehat{p}_{\mathrm{KNN}}(\mathbf{x}) \approx \frac{K\Gamma(d/2+1)}{m\pi^{d/2}r^d}.} \tag{12.53}$$

There are naturally some basic requirements on the parameter K. For example, as m increases, K should also increase correspondingly. In fact, one often requires $\lim_{m\to\infty} K = \infty, \lim_{m\to\infty}(K/m) = 0$, and $K = \sqrt{m}$ is one possible choice satisfying these conditions. Estimating the probability density $\widehat{p}$ is an important problem. The kernel density estimator (KDE) is often used for continuous density estimation. The KDE is defined as $\widehat{p}_{\mathrm{KDE}}(\mathbf{x}) = m^{-1}h^{-d}\sum_{i=1}^{m}K[(\mathbf{x}-\mathbf{x}^{(i)})/h]$ under the basic requirements $K(\mathbf{x}) \geq 0$ and $\int_{\mathrm{all}} K(\mathbf{x})\mathrm{d}\mathbf{x} = 1$. Here K denotes the kernel, rather than the number of clusters in the KNN algorithm. For example, the Gaussian KDE is simply $K(\mathbf{x}) = (2\pi)^{-d/2}\exp(-2^{-1}\mathbf{x}^\top\mathbf{x})$. More generally, the KDE takes the form

$$\boxed{\widehat{p}_{\mathrm{KDE}}(\mathbf{x}) = \frac{1}{m\det\mathbf{H}}\sum_{i=1}^{m}K\left(\mathbf{H}^{-1}(\mathbf{x}-\mathbf{x}^{(i)})\right),} \tag{12.54}$$

where d is the dimension and $\mathbf{H}\in\mathrm{R}^{d\times d}$ is the bandwidth matrix. If K is the Gaussian kernel, then $\mathbf{H}\mathbf{H}^\top$ is the covariance matrix.

As an example, consider $\mathbf{h} = \mathrm{diag}(h_1, \cdots, h_d)$ and assume that the physical distribution is Gaussian. Then the optimal bandwidth estimate is

$$\widehat{h}_j = \left(\frac{4}{m}\frac{1}{d+2}\right)^{\frac{1}{d+4}}\sigma_j, \tag{12.55}$$

where σ_j is the jth standard deviation of the data sample $\mathbf{x}$. Since it is unknown in practice, it can be estimated from the samples as

$$\boxed{\widehat{\sigma}_j = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}\left(x_j^{(i)} - \frac{1}{m}\sum_{i=1}^{m}x_j^{(i)}\right)^2}.} \tag{12.56}$$

EXERCISE 12-8. Plot (12.51) as a function of d.

§12.9 Problems for This Lecture

Theoretical

1. Point out the geometrical meanings of the following relations in three dimensions:

$$\sqrt{w_0^2+w_1^2+w_2^2}\le R,\ \sqrt{w_0^2}+\sqrt{w_1^2}+\sqrt{w_2^2}\le R,\ \sqrt{w_0^2+w_1^2}+\sqrt{w_2^2}\le R. \tag{12.57}$$

2. Denote $\mathbf{t}=m^{-1}\sum_{i=1}^{m}\mathbf{x}^{(i)}$ as the cluster center of a series of unit vectors $\mathbf{x}^{(i)}$'s, and define the similarity between two points $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ as their dot product. Prove that the average cluster similarity $m^{-2}\sum_{i,j}\mathbf{x}^{(i)}\mathbf{x}^{(j),\top}$ is the same whether it is computed by averaging over all pairs or by computing the average similarity of each point with the centroid of the cluster.

Computational/Programming

3. Tukey's loss function is defined as

$$\chi^{\text{Tuk}}(e)=\begin{cases}[1-(1-e^2/\eta^2)^3]\eta^2/6, & |e|\le\eta,\\ \eta^2/6, & |e|>\eta,\end{cases} \tag{12.58}$$

   where η is the threshold. Apply Tukey's loss function to the learning process for the data $y^{(i)}=x^{(i)}+\text{rand}\times\delta$ under the learning model $f_{\mathbf{w}}(x)=w_0+w_1x$.
4. Implement the EM algorithm for discrete data problems.

References and Notes for Lecture 12

[∗12-1∗] This example is taken from A. Blum, J. Hopcroft, and R. Kannan, *Foundations of Data Science*, Cambridge University Press, 2020, Chap. 7.

[∗12-2∗] R. Tibshirani, *Regression Shrinkage and Selection via the lasso*, Journal of the Royal Statistical Society. Series B **58**, 267 (1996).

[∗12-3∗] See, e.g., S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, Foundations and Trends in Machine Learning, **3**,1 (2011).

[∗12-4∗] H. Zou and T. Hastie, *Regularization and Variable Selection via the Elastic Net*, J. Roy. Stat. Soc. B, **67**, 301 (2005).

[∗12-5∗] C. Bishop, *Pattern Recognition and Machine Learning*, 2006, Springer, Chap. 9.

[∗12-6∗] The original paper on the EM algorithm was, A. Dempster, N. Laird, and D. Rubin, *Maximum Likelihood from Incomplete Data via the EM Algorithm*, J. Roy. Stat. Soc. B, **39**, 1(1977).

[∗12-7∗] This example is taken from, C. Bishop, *Pattern Recognition and Machine Learning*, 2006, Springer, Chap. 9.