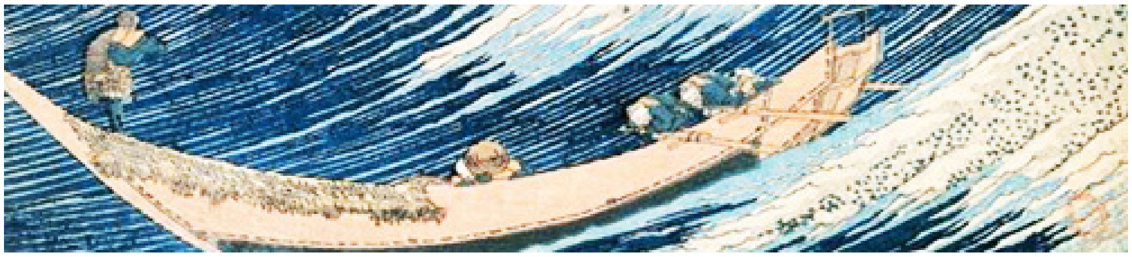
Mathematical Foundations of Data Sciences



Gabriel Peyré
CNRS & DMA
École Normale Supérieure
gabriel.peyre@ens.fr
www.gpeyre.com
www.numerical-tours.com

November 15, 2017

Chapter 16

Machine Learning

This chapter gives a rapid overview of the main concepts in machine learning. The goal is not to be exhaustive, but to highlight representative problem to insist on the distinction between unsupervised (vizualization and clustering) and supervised (regression and classification) problems. We also shed light on the tight connexions between machine learning and inverse problems.

16.1 Unsupervised Learning

In unsupervised learning, one observed n points $(x_i)_{i=1}^n$. For simplicity, we assume the data are point in Euclidean space $x_i \in \mathbb{R}^p$ (p is the so-called number of features). These points are conveniently stored as the rows of a matrix $X \in \mathbb{R}^{n \times d}$. The problem is now to infer some properties for this points, typically for vizualization or unsupervised classication (often called clustering).

16.1.1 Dimensionality Reduction and PCA

Dimensionality reduction is useful for vizualization. It can also be understood as the problem of feature extraction (determining which are the relevant parameters), and this can be later used for doing other tasks more efficiently (faster and/or with better performances).

In order to display in 2-D or 3-D the data, dimensionality reduction is needed. The simplest method is the Principal Component Analysis (PCA), which perform an orthogonal linear projection on the principal axis (eigenvector) of the covariance matrix.

The empirical mean is defined as

$$\hat{m} \overset{\text{def.}}{=} \frac{1}{n} \sum_{i=1}^n x_i \in \mathbb{R}^p$$

and covariance

$$\hat{C} \overset{\text{def.}}{=} \frac{1}{n} \sum_{i=1}^n (x_i - m)(x_i - m)^\top \in \mathbb{R}^{p \times p}. \tag{16.1}$$

Denoting $\tilde{X} = X - 1_p \hat{m}^\top$, one has $\hat{C} = \tilde{X}^\top \tilde{X}$.

Note that if points $(x_i)_i$ are actually i.i.d. variable, and denoting x one of these random variable, one has using the law of large number, the almost surely convergence as $n \to +\infty$

$$\hat{m} \to m \overset{\text{def.}}{=} \mathbb{E}(x) \quad \text{and} \quad \hat{C} \to C \overset{\text{def.}}{=} \mathbb{E}((x-m)(x-m)^\top). \tag{16.2}$$

Denoting μ the distribution (Radon measure) on $\mathbb{R}^p$ of x, one can alternatively write

$$m = \int_{\mathbb{R}^p} x \mathrm{d}\mu(x) \quad \text{and} \quad C = \int_{\mathbb{R}^p} (x-m)(x-m)^\top \mathrm{d}\mu(x).$$

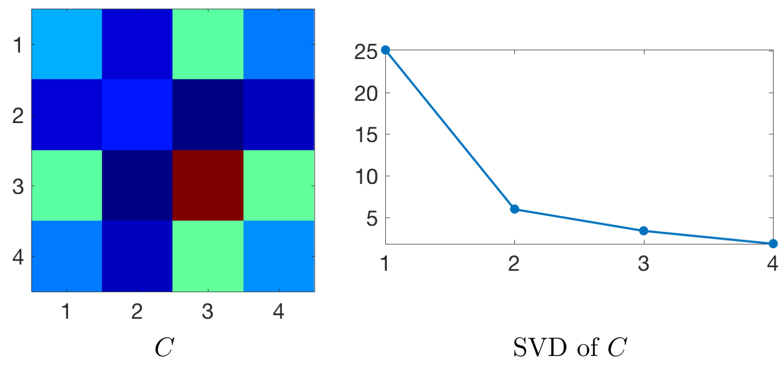


Figure 16.1: Empirical covariance of the data and its associated singular values.

The PCA ortho-basis, already introduced in Section 20, corresponds to the right singular vector of the centred data matrix using the (reduce) SVD decomposition

$$\tilde{X} = U \operatorname{diag}(\sigma) V$$

where $U \in \mathbb{R}^{n \times r}$ and $V \in \mathbb{R}^{p \times r}$, and where $r = \operatorname{rank}(\tilde{X}) \leqslant \min(n, p)$. In usual settings, $n \gg p$, and one assume having enough samples so that the approximation (16.2). We denote $V = (v_k)_{k=1}^r$ the orthogonal columns $v_k \in \mathbb{R}^p$. The intuition is that they are the main axis of "gravity" of the cloud X. We assume the singular are order, $\sigma_1 \geqslant \ldots \geqslant \sigma_r$, so that the first singular values capture most of the variance.

Figure 16.1 displays an example of covariance and its associated spectrum σ. The points $(x_i)_i$ corresponds to the celebrated IRIS dataset[1] of Fisher. This dataset consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four $d = 4$ were measured from each sample: the length and the width of the sepals and petals, in centimetres.

The PCA dimensionality reduction embedding $x_i \in \mathbb{R}^p \mapsto z_i \in \mathbb{R}^d$ in dimension $d \leqslant p$ is obtained by projecting on the first d singular vector

$$z_i \overset{\text{def.}}{=} (\langle x_i - m, \, v_k \rangle)_{k=1}^d \in \mathbb{R}^d.$$

From these low-dimensional embedding, one can reconstruct back an approximation as

$$\tilde{x}_i \overset{\text{def.}}{=} m + \sum_k z_{i,k} v_k \in \mathbb{R}^p.$$

One has that $\tilde{x}_i = \operatorname{Proj}_{\tilde{T}}(x_i)$ where $\tilde{T} \overset{\text{def.}}{=} m + \operatorname{Span}_{k=1}^d(v_k)$ is an affine space. The following proposition shows that PCA is optimal in term of ℓ^2 distance if one consider only affine spaces.

Proposition 52. *One has*

$$(\tilde{x}, \tilde{T}) \in \underset{(\bar{x}, \bar{T})}{\operatorname{argmin}} \left\{ \sum_i \|x_i - \bar{x}_i\|^2 \; ; \; \forall \, i, \bar{x}_i \in \bar{T} \right\}$$

where $\bar{T}$ is constrained to be a d-dimensional affine space.

Figure 16.2 shows an example of PCA for 2-D and 3-D vizualization.

16.1.2 Clustering and k-means

A typical unsupervised learning task is to infer a class label $y_i \in \{1, \ldots, k\}$ for each input points x_i, and this is often called a clustering problems (since the set of points associated to a given label can be thought as a cluster).

[1] https://en.wikipedia.org/wiki/Iris_flower_data_set

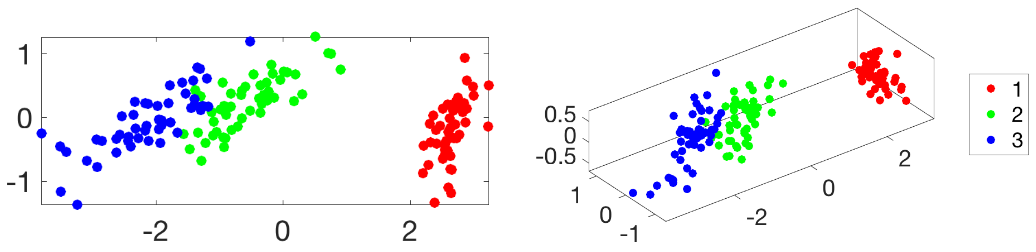


Figure 16.2: 2-D and 3-D PCA vizualization of the input clouds.

***k*-means** A way to infer these label is by assuming that the cluster are compact, and optimizing a compactness criterion. Assuming for simplicity that the data are in Euclidean space (which can be relaxed to an arbitrary metric space, although the computation become more complicated), the k-means approach minimizes the distance between the points and their class centroids $c = (c_\ell)_{\ell=1}^k$, where each $c_\ell \in \mathbb{R}^p$. The corresponding variational problem becomes

$$\min_{(y,c)} \mathcal{E}(y,c) \overset{\text{def.}}{=} \sum_{\ell=1}^{k} \sum_{i:y_i=\ell} \|x_i - c_\ell\|^2.$$

The k-means algorithm can be seen as a block coordinate relaxation, which alternatively update the class label and the centroids. The centroids c are first initialized (more on this later), for instance, using a well-spread set of points from the samples. For a given set c of centroids, minimizing $y \mapsto \mathcal{E}(y,c)$ is obtained in closed form by assigning as class label the index of the closest centroids

$$\forall\, i \in \{1,\dots,n\}, \quad y_i \leftarrow \underset{1\leqslant \ell \leqslant k}{\operatorname{argmin}} \|x_i - c_\ell\|. \tag{16.3}$$

For a given set y of centroids, minimizing $c \mapsto \mathcal{E}(y,c)$ is obtained in closed form by computing the barycenter of each class

$$\forall\, \ell \in \{1,\dots,k\}, \quad c_\ell \leftarrow \frac{\sum_{i:y_i=\ell} x_i}{|\{i\,;\, y_i = \ell\}|} \tag{16.4}$$

If during the iterates, one of the cluster associated to some c_ℓ become empty, then one can either decide to destroy it and replace k by $k-1$, or try to "teleport" the center c_ℓ to another location (this might increase the objective function $\mathcal{E}$ however).

Since the energy $\mathcal{E}$ is decaying during each of these two steps, it is converging to some limit value. Since there is a finite number of possible labels assignments, it is actually constant after a finite number of iterations, and the algorithm stops.

Of course, since the energy is non-convex, little can be said about the property of the clusters output by k-means. To try to reach lower energy level, it is possible to "teleport" during the iterations centroids c_ℓ associated to cluster with high energy to location within cluster with lower energy (because optimal solutions should somehow balance the energy).

Figure 16.3 shows an example of k-means iterations on the Iris dataset.

***k*-means++** The crucial point to obtain good results when using k-means is to have an efficient initialization. In practice, the best results are obtained by seeding them as far as possible from one another (a greedy strategy works great in practice).

Quite surprisingly, there exists a randomized seeding strategy which can be shown to be closed to optimal, even without running the k-means iterations (although in practice it is still required to obtain descent results). The corresponding k-means++ initialization is obtained by selecting c_1 uniformly at random among the x_i, and then assuming c_ℓ has been seeded, drawing $c_{\ell+1}$ according to the probability $\pi^{(\ell)}$ on $\{1,\dots,n\}$

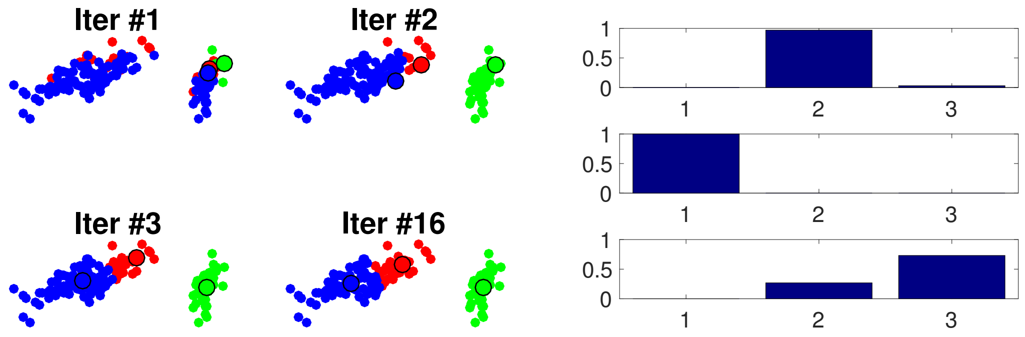


Figure 16.3: Left: iteration of k-means algorithm. Right: histogram of points belonging to each class.

proportional to the square of the distance to the previously seeded points

$$\forall\, i \in \{1,\ldots,n\}, \quad \pi_i^{(\ell)} \overset{\text{def.}}{=} \frac{d_i^2}{\sum_{j=1}^n d_j^2} \quad \text{where} \quad d_j \overset{\text{def.}}{=} \min_{1\leqslant r\leqslant \ell-1} \|x_i - c_r\|^2$$

The following results, due to David Arthur and Sergei Vassilvitskii shows that this seeding is optimal up to log factor on the energy. Note that finding a global optimum is known to be NP-hard.

Theorem 50. *For the centroids $c^\star$ defined by the k-means++ strategy, denoting $y^\star$ the associated nearest neighbor labels defined as in (16.3), one has*

$$\mathbb{E}(\mathcal{E}(y^\star, c^\star)) \leqslant 8(2+\log(k)) \min_{(y,c)} \mathcal{E}(y,v).$$

where the expectation is on the random draws of performed by the algorithm.

Lloyd algorithm and continuous densities. The k-means iterations are also called "Lloyd" algorithm, which also find popular application to optimal vector quantization for compression. It can also be used in the "continuous" setting where the empirical samples $(x_i)_i$ are replaced by an arbitrary measure over $\mathbb{R}^P$. The energy to minimize becomes

$$\min_{(\mathcal{V},c)} \sum_{\ell=1}^k \int_{\mathcal{V}_\ell} \|x - c_\ell\|^2 \mathrm{d}\mu(x)$$

where $(\mathcal{V}_\ell)_\ell$ is a partition of the domain. Step (16.3) is replaced by the computation of a Voronoi cell

$$\forall\, \ell \in \{1,\ldots,k\}, \quad \mathcal{V}_\ell \overset{\text{def.}}{=} \{x \,;\, \forall\, \ell' \neq \ell, \|x - c_\ell\| \leqslant \|x - c_{\ell'}\|\}.$$

These Voronoi cells are polyhedra delimited by segments of mediatrix between centroids, and this Voronoi segmentation can be computed efficiently using tools from algorithmic geometry in low dimension. Step (16.4) are then replaced by

$$\forall\, \ell \in \{1,\ldots,k\}, \quad c_\ell \leftarrow \frac{\int_{\mathcal{C}_\ell} x \mathrm{d}\mu(x)}{\int_{\mathcal{C}_\ell} \mathrm{d}\mu(x)}.$$

In the case of μ being uniform distribution, optimal solution corresponds to the hexagonal lattice. Figure 16.4 displays two examples of Lloyd iterations on 2-D densities on a square domain.

16.1.3 Supervised Learning: Regression

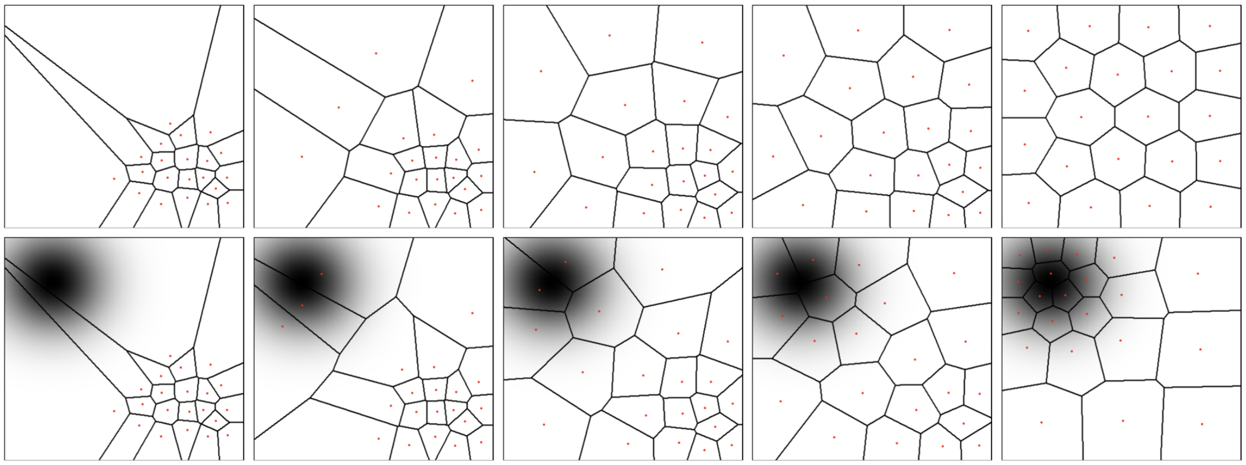


Figure 16.4: Iteration of k-means algorithm (Lloyd algorithm) on continuous densities μ. Top: uniform. Bottom: non-uniform (the densities of μ with respect to the Lebesgue measure is displayed as a grayscale image in the background).

In supervised learning, one has access to training data, consisting in pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. Here $\mathcal{X} = \mathbb{R}^p$ for simplicity. The goal is to infer some relationship, typically of the form $y_i \approx f(x_i)$ for some deterministic function $f : \mathcal{X} \to \mathcal{Y}$, in order, when some un-observed data x without associated value in $\mathcal{Y}$ is given, one can "predict" the value using $y = f(x)$.

If the set $\mathcal{Y}$ is discrete and finite, then this problem is called a supervised classification problem, and this is specifically studied in Section 16.2. The simplest example being the binary classification case, $\mathcal{Y} = \{0, 1\}$, for instance in medical diagnosis, $y_i = 0$ indicate a healthy subject, why $y_i = 0$ a pathological one. If $\mathcal{Y}$ is continuous (the typical example being $\mathcal{Y} = \mathbb{R}$), then this problem is called a regression problem.

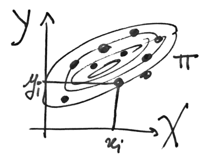


Figure 16.5: Probabilistic modelling.

16.1.4 Empirical Risk Minimization

In order to make the problem tractable computationally, and also in order to obtain efficient prediction score, it is important to restrict the fit to the data $y_i \approx f(x_i)$ using a restricted class of functions. Intuitively, in order to avoid overfitting, the "size" of this class of function should grows with the number n of samples.

Empirical risk. Denoting $\mathcal{F}_n$ some class of function, the most usual way to do the learning is to perform an empirical risk minimization (ERM)

$$\hat{f} \in \underset{f \in \mathcal{F}_n}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x_i), y_i). \tag{16.5}$$

Here $\mathcal{L} : \mathcal{Y}^2 \to \mathbb{R}^+$ is the so-called loss function, and it should typically satisfies $\mathcal{L}(y, y') = 0$ if and only if $y = y'$. The specifics of $\mathcal{L}$ depends on the applications (in particular, one should use different loss for classification and regression tasks). To highlight the dependency of $\hat{f}$ on n, we occasionally write $\hat{f}_n$.

Prediction risk and prediction consistency. When doing a mathematically analysis, one usually assume that (x_i, y_i) are drawn from a distribution π on $\mathcal{X} \times \mathcal{Y}$, and the large n limit defines the ideal estimator

$$\bar{f} \in \underset{f \in \mathcal{F}_n}{\operatorname{argmin}} \int_{\mathcal{X}\times\mathcal{Y}} \mathcal{L}(f(x), y)\mathrm{d}\pi(x,y) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\pi}(\mathcal{L}(f(\mathbf{x}), \mathbf{y}). \tag{16.6}$$

Intuitively, one should have $\hat{f}_n \to \bar{f}$ as $n \to +\infty$, which can be captured in expectation of the prediction error over the samples $(x_i, y_i)_i$, i.e.

$$E_n \overset{\text{def.}}{=} \mathbb{E}(\tilde{\mathcal{L}}(\hat{f}_n(\mathbf{x}), f(\mathbf{x}))) \longrightarrow 0.$$

One should be careful that here the expectation is over both $\mathbf{x}$ (distributed according to the marginal $\pi_{\mathcal{X}}$ of π on $\mathcal{X}$), and also the n i.i.d. pairs $(x_i, y_i) \sim \pi$ used to define $\hat{f}_n$ (so a better notation should rather be $(\mathbf{x}_i, \mathbf{y}_i)_i$. Here $\bar{\mathcal{L}}$ is some loss function on $\mathcal{Y}$ (one can use $\bar{\mathcal{L}} = \mathcal{L}$ for instance). One can also study convergence in probability, i.e.

$$\forall\, \varepsilon > 0, \quad E_{\varepsilon,n} \overset{\text{def.}}{=} \mathbb{P}(\tilde{\mathcal{L}}(\hat{f}_n(\mathbf{x}), f(\mathbf{x})) > \varepsilon) \to 0.$$

If this holds, then one says that the estimation method is consistent (in expectation or in probability). The question is then to derive convergence rates, i.e. to upper bound E_n or $E_{\varepsilon,n}$ by some explicitly decay rate.

Note that when $\tilde{\mathcal{L}}(y, y') = |y - y'|^r$, then convergence in expectation is stronger (implies) than convergence in probability since using Markov's inequality

$$E_{\varepsilon,n} = \mathbb{P}(|\hat{f}_n(\mathbf{x}) - f(\mathbf{x})|^r \geqslant \varepsilon) \leqslant \frac{1}{\varepsilon}\mathbb{E}(|\hat{f}_n(\mathbf{x}) - f(\mathbf{x})|^r) = \frac{E_n}{\varepsilon}.$$

Parametric approach and regularization. Instead of directly defining the class $\mathcal{F}_n$ and using it as a constraint, it is possible to rather use a penalization using some prior to favor "simple" or "regular" functions. This can also be enforced using parametric models $y \approx f(x, \beta)$ where $\beta \in \mathcal{B}$ parametrizes the function $f(\cdot, \theta) : \mathcal{X} \to \mathcal{Y}$. The empirical risk minimization procedure (16.7) now becomes

$$\hat{\beta} \in \underset{\beta \in \mathcal{B}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x_i, \beta), y_i) + \lambda_n J(\beta). \tag{16.7}$$

where J is some regularization function, for instance $J = \|\cdot\|_2^2$ or $J = \|\cdot\|_1$. Here $\lambda_n > 0$ is a regularization parameter, and it should tend to 0 when $n \to 0$.

Then one similarly defines the ideal parameter $\bar{\beta}$ as in (16.6) so that the limiting estimator as $n \to +\infty$ is of the form $\bar{f} = f(\cdot, \bar{\beta})$.

In this parametric approach, one could be interested in also studying how close $\hat{\theta}$ is close to $\bar{\theta}$. This can be measure by controlling how fast some estimation error $\|\hat{\theta} - \bar{\theta}\|$ (for some norm $\|\cdot\|$) goes to zero. Note however that in most cases, controlling the estimation error is much more difficult than doing the same for the prediction error. Doing a good parameter estimation implies doing a good prediction, but the converse is not true.

Testing set. In practice, E_n or $E_{\varepsilon,n}$ are ideal quantity which cannot be accessed. One thus rather ressort to a second set of data $(\bar{x}_j, \bar{y}_j)_{j=1}^{\bar{n}}$, called "testing set". From a modelling perspective, this set should also be distributed i.i.d. according to π. The validation (or testing) risk is then

$$R_{\bar{n}} = \frac{1}{\bar{n}} \sum_{j=1}^{\bar{n}} \mathcal{L}(\hat{f}(\bar{x}_j), \bar{y}_j). \tag{16.8}$$

Minimizing $R_{\bar{n}}$ to setup to some meta-parameter of the method (for instance the regularization parameter λ_n) is called "cross validation" in the litterature.

16.1.5 Linear Regression

We now specialized the empirical risk minimization approach to regression problem, and even more specifically, we consider $\mathcal{Y} = \mathbb{R}$ and use a quadratic loss $\mathcal{L}(y, y') = \frac{1}{2}|y - y'|^2$.

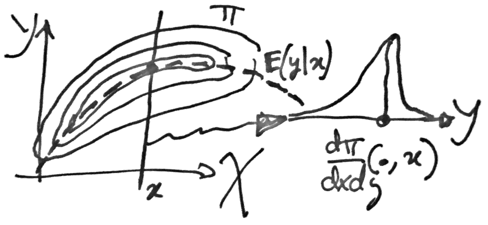


Figure 16.6: Conditional expectation.

Least square and conditional expectation. If one do not put any constraint on f (beside being measurable), then the optimal limit estimator $\bar{f}(x)$ (16.6) is simply averaging the values y sharing the same , which is the so-called conditional expectation. Assuming for simplicity that π has some density $\frac{\mathrm{d}\pi}{\mathrm{d}x\mathrm{d}y}$ with respect to a tensor product measure $\mathrm{d}x\mathrm{d}y$ (for instance the Lebegues mesure), one has

$$\forall\, x \in \mathcal{X}, \quad \bar{f}(x) = \mathbb{E}(\mathbf{y}|\mathbf{x} = x) = \frac{\int_{\mathcal{Y}} y \frac{\mathrm{d}\pi}{\mathrm{d}x\mathrm{d}y}(x,y)\mathrm{d}y}{\int_{\mathcal{Y}} \frac{\mathrm{d}\pi}{\mathrm{d}x\mathrm{d}y}(x,y)\mathrm{d}y}$$

where $(\mathbf{x}, \mathbf{y})$ are distributed according to π.

In the simple case where $\mathcal{X}$ and $\mathcal{Y}$ are discrete, denoting $\pi_{x,y}$ the probability of $(\mathbf{x} = x, \mathbf{y} = y)$, one has

$$\forall\, x \in \mathcal{X}, \quad \bar{f}(x) = \frac{\sum_y y\pi_{x,y}}{\sum_y \pi_{x,y}}$$

and it unspecified if the marginal of π along $\mathcal{X}$ vanishes at x.

The main issue is that the estimator $\hat{f}$ performs very poorly on finite sample, and $f(x)$ is actually undefined if there is no sample x_i equal to x. This is due to the fact that the class of functions is too large, and one should impose some regularity or simplicity on the set of admissible f.

Penalized linear models. A very simple class of model is obtained by imposing that f is linear, and set $f(x, \beta) = \langle x,\, \beta\rangle$, for parameter $\beta \in \mathbb{R}^p$. Note that one can also treat this way affine function by remarking that $\langle x,\, \beta\rangle + \beta_0 = \langle (x, 1),\, (\beta, \beta_0)\rangle$ and replacing x by $(x, 1)$.

Under the square loss, the regularized ERM (16.7) is conveniently rewritten as

$$\hat{\beta} \in \underset{\beta\in\mathcal{B}}{\operatorname{argmin}}\ \frac{1}{2}\langle \hat{C}\beta,\, \beta\rangle - \langle \hat{u},\, \beta\rangle + \lambda_n J(\beta) \tag{16.9}$$

where we introduced the empirical correlation (already introduced in (16.1)) and observations

$$\hat{C} \overset{\text{def.}}{=} \frac{1}{n}X^*X = \frac{1}{n}\sum_{i=1}^n x_i x_i^* \quad \text{and} \quad \hat{u} \overset{\text{def.}}{=} \frac{1}{n}\sum_{i=1}^n y_i x_i = \frac{1}{n}X^*y \in \mathbb{R}^p.$$

As $n \to 0$, under weak condition on π, one has with the law of large number the almost sure convergence

$$\hat{C} \to C \overset{\text{def.}}{=} \mathbb{E}(\mathbf{x}^*\mathbf{x}) \quad \text{and} \quad \hat{u} \to C \overset{\text{def.}}{=} \mathbb{E}(\mathbf{y}\mathbf{x}). \tag{16.10}$$

When considering $\lambda_n \to 0$, in some case, one can shows that in the limit $n \to +\infty$ (under some additional condition on the decay), one retrieves the following ideal parameter

$$\bar{\beta} \in \underset{\beta}{\operatorname{argmin}}\ \{J(\beta)\ ;\ C\beta = u\}\,.$$

Problem (16.9) is equivalent to the regularized resolution of inverse problems (10.9), with $\hat{C}$ in place of Φ and $\hat{u}$ in place of $\Phi^* y$.

The major, and in fact only difference between machine learning and inverse problems is that the linear operator is also noisy since $\hat{C}$ can be viewed as a noisy version of C. The "noise level", in this setting is $1/\sqrt{n}$ in the sense that

$$\mathbb{E}(\|\hat{C} - C\|) \sim \frac{1}{\sqrt{n}} \quad \text{and} \quad \mathbb{E}(\|\hat{u} - u\|) \sim \frac{1}{\sqrt{n}},$$

under mild moments assumption on π.

The general take home message is that it is simple to generalize Theorems 31, 42 and 43 to cope with the noise on the covariance matrix to obtain prediction convergence rates of the form

$$\mathbb{E}(|\langle \hat{\beta}, \mathbf{x} \rangle - \langle \bar{\beta}, \mathbf{x} \rangle|^2) = O(n^{-\kappa})$$

and estimation rates of the form

$$\mathbb{E}(\|\hat{\beta} - \bar{\beta}\|^2) = O(n^{-\kappa'})$$

under some suitable source condition involving C and u. Since the noise level is roughly $n^{-\frac{1}{2}}$, the ideal cases are $\kappa = \kappa' = 1$, which is the so-called linear rate regime. It is also possible to derive sparsistency theorems by extending theorem 44.

Ridge regression (quadratic penalization). For $J = \|\cdot\|^2/2$, the estimator (16.9) is obtained in closed form as

$$\hat{\beta} = (X^* X + n\lambda_n \mathrm{Id}_p)^{-1} X^* y = (\hat{C} + n\lambda_n \mathrm{Id})^{-1} \hat{u}. \tag{16.11}$$

This is often called ridge regression in the litterature. Note that thanks to the Woodbury formula, this estimator can also be re-written as

$$\hat{\beta} = X^* (X X^* + n\lambda_n \mathrm{Id}_n)^{-1} y. \tag{16.12}$$

If $n \gg p$ (which is the usual setup in machine learning), then (16.12) is preferable. In some cases however (in particular when using RKHS technics), it makes sense to consider very large p (even infinite dimensional), so that (16.11) must be used.

If $n\lambda_n \to 0$, then using (16.10) one has the convergence in expectation and probability

$$\hat{\beta} \to \bar{\beta} = C^+ u.$$

Theorems 31 and 42 can be extended to this setting and one obtains the following result.

Theorem 51. *If*

$$\bar{\beta} = C^\gamma z \quad \textit{where} \quad \|z\| \leqslant \rho \tag{16.13}$$

for $0 < \gamma \leqslant 2$, *then*

$$\mathbb{E}(\|\hat{\beta} - \bar{\beta}\|^2) \leqslant C \rho^{2\frac{1}{\gamma+1}} n^{-\frac{\gamma}{\gamma+1}} \tag{16.14}$$

for a constant C depending only on γ.

If is important to note that, since $\bar{\beta} = C^+ u$, the source condition (16.13) is always satisfied, but what matters here is that the rate (16.14) does not depends on the dimension p of the features, but rather only on ρ, which can be much smaller. This theoretical analysis actually works perectly fine in infinite dimension $p = \infty$ (which is the setup considered when dealine with RKHS bellow).

In contrast, when the dimensionality p of the feature is very large and there is little data, the second is faster. Furthermore, this second expression is generalizable to Kernel Hilbert space setting, corresponding possibly to $p = +\infty$ for some kernels.

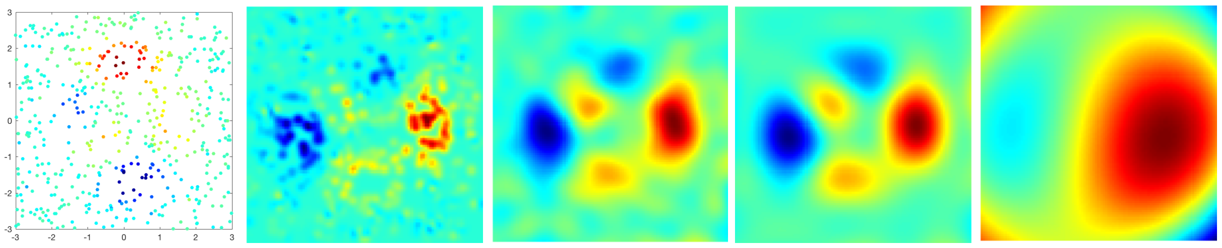


Figure 16.7: Regression using a Gaussian kernel.

16.1.6 Kernelized Ridge Regression

In order to perform non-linear and non-parametric regression, it is possible to use kernelization. It is non-parametric in the sense that the number of parameter grows with the number n of samples (while for the initial linear method, the number of parameter is p). This allows in particular to generate estimator of arbitrary complexity.

Given a kernel $\kappa(x, z) \in \mathbb{R}$ defined for $(x, z) \in \mathbb{R}^p \times \mathbb{R}^p$, the kernelized method replace the linear approximation functional $f(x, \beta) = \langle x, \beta \rangle$ by a sum of kernel centred on the samples

$$f(x, \alpha) = \sum_{i=1}^{n} \alpha_i k(x_i, x) \tag{16.15}$$

where $\alpha \in \mathbb{R}^n$ is the unknown vector of weight to find. Note that now, the dimension of the parameter depends on n. When using the linear kernel $\kappa(x, y) = \langle x, y \rangle$, one retrieves the previously studied linear method.

The gaussian kernel is the most well known and used kernel

$$\kappa(x, y) \stackrel{\text{def.}}{=} e^{-\frac{\|x-y\|^2}{2\sigma^2}}.$$

The bandwidth parameter $\sigma > 0$ is crucial and controls the locality of the model. It is typically tuned through cross validation.

Once evaluated on grid points, the kernel define a matrix

$$K = (\kappa(x_i, x_j))_{i,j=1}^{n} \in \mathbb{R}^{n \times n}. \tag{16.16}$$

Valid kernels are those that gives rise to positive symmetric matrices K. The linear and Gaussian kernel are valid kernel functions. Other popular kernels include the polynomial kernel $\langle x, y \rangle^a$ for $a \geqslant 1$ and the Laplacian kernel $\exp(-\|x - y\|^2/\sigma)$.

The weights $\alpha \in \mathbb{R}^n$ are solutions of

$$\min_{\alpha} \|K\alpha - y\|^2 + \lambda_n \langle K\alpha, \alpha \rangle \tag{16.17}$$

and hence can be computed by solving a linear system

$$\alpha = (K + \lambda_n \mathrm{Id}_n)^{-1} y$$

Figure (16.7) displays the function $f(\cdot, \alpha)$ for a varying bandwidth σ for the Gaussian kernel.

This minimization (16.17) can be related to an infinite dimensional optimization problem where one minimizes directly over the function f. This is shown to be equivalent to the above finite-dimenisonal optimization problem thanks to the theory of RKHS.

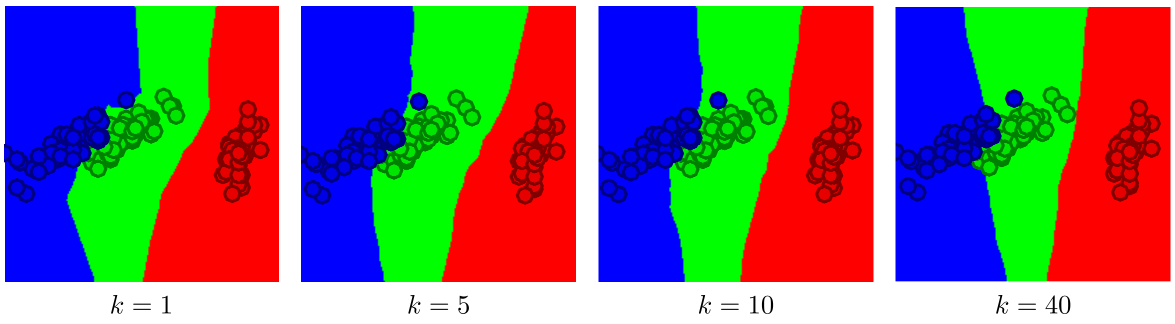


Figure 16.8: k-nearest-neighbor classification boundary function.

16.2 Supervised Learning: Classification

We now focus on the case of discrete labels $y_i \in \mathcal{Y} = \{1, \ldots, k\}$, which is the classification setup. We now detail two popular classification methods: logistic and nearest neighbors. It is faire to say that a very large class of successful applications of machine learning technics consists in using one of these two approaches, which should be considered as baselines. Note that the nearest neighbor approach, while popular for classification could as well be used for regression.

16.2.1 Nearest Neighbors Classification

Probably the simplest method for supervised classification is R nearest neighbors (R-NN), where R is a parameter indexing the number of neighbors. Increasing R is important to cope with noise and obtain smoother decision boundary, and hence better generalization performance. It should typically decrease as the number of training sample n increases.

The class $\hat{f}(x) \in \mathcal{Y}$ predicted for a point x is the one which is the most represented among the R points $(x_i)_i$ which are the closed to x. This is a non-parametric method, and $\hat{f}$ depends on the numbers n of samples (its "complexity" increases with n).

One first compute the Euclidean distance between this x and all other x_i in the training set. Sorting the distances generates an indexing σ (a permutation of $\{1, \ldots, n\}$) such that

$$\|x - x_{\sigma(1)}\| \leqslant \|x - x_{\sigma(2)}\| \leqslant \ldots \leqslant \|x - x_{\sigma(n)}\|.$$

For a given R, one can compute the histogram of class apparition

$$h_\ell(x) \stackrel{\text{def.}}{=} \frac{1}{R} \left\{ i \; ; \; y_{\sigma(i)} \in \{1, \ldots, R\} \right\}.$$

The decision class for x is then the maximum of the histogram

$$\hat{f}(x) \stackrel{\text{def.}}{=} \underset{\ell}{\operatorname{argmax}}\, h_\ell(x).$$

In practice, the parameter k can be decided through cross-validation, by minimizing the testing risk $R_{\bar{n}}$ defined in (16.8). Of course the method extends to arbitrary metric space in place of Euclidean space $\mathbb{R}^p$ for the features. Note also that instead of explicitly sorting all the Euclidean distance, one can use fast nearest neighbor search methods.

Figure 16.8 shows, for the IRIS dataset the classification domain (i.e. $\{x \; ; \; f(x) = \ell\}$ for $\ell = 1, \ldots, k$) in using a 2-D projection for vizualization. Increasing R leads to smoother class boundaries. Despite its simplicity, k-NN is surprisingly successful in practice, specially in low dimension p.

16.2.2 Two Classes Logistic Classification

The logistic classification method (for 2 classes and multi-classes) is one of the most popular (maybe "the" most) popular machine learning technics. This is due in large part of both its simplicity and because it also output a probability of belonging to each class (in place of just a class membership), which is useful to quantify (somehow ...) the "uncertainty" of the estimation. Note that logistic classification is actually called "logistic regression" in the literature, but it is in fact a classification method.

Another very popular (and very similar) approach is support vector machine (SVM). SVM is both more difficult to train (because the loss is non-smooth) and does not give class membership probability, so the general rule of thumb is that logistic classification is preferable.

To simplify the expression, classes indexes are set to $y_i \in \mathcal{Y} = \{-1, 1\}$ in the following. Note that for logistic classification, the precision function $f(\cdot, \beta) \in [0, 1]$ output the probability of belonging to the first class, and not the class indexes. With a slight abuse of notation, we still denote it as f.

Logistic classification can be understood as a linear model as introduced in Section 16.1.5, although the decision function $f(\cdot, \beta)$ is not. Indeed, one needs to "remap" the linear value $\langle x,\, \beta\rangle$ in the interval $[0, 1]$. In logistic classification, we define the predicted probability of x belonging to class 1 (thus label -1) as

$$f(x, \beta) \overset{\text{def.}}{=} \theta(\langle x,\, \beta\rangle) \quad \text{where} \quad \theta(s) \overset{\text{def.}}{=} \frac{e^s}{1+e^s} = (1+e^{-s})^{-1}.$$

Note that the probability of belonging to the second class is $1 - f(x, \beta)$.

Note that $f(x, \beta)$ can be interpreted as a single layer perceptron with a logistic (sigmoid) rectifying unit.

Since the (x_i, y_i) are modelled as i.i.d. variables, it makes sense to define $\hat\beta$ from the observation using a maximum likelihood, assuming that each y_i conditionned on x_i is a Bernoulli variable with associated probability $(p_i, 1-p_i)$ with $p_i = f(x_i, \beta)$. The probability of oberving $y_i \in \{0, 1\}$ is thus, denoting $s_i = \langle x_i,\, \beta\rangle$

$$\mathbb{P}(\mathbf{y} = y_i | \mathbf{x} = x_i) = p_i^{\bar y_i}(1-p_i)^{1-\bar y_i} = \left(\frac{e^{s_i}}{1+e^{s_i}}\right)^{1-\bar y_i} \left(\frac{1}{1+e^{s_i}}\right)^{\bar y_i}$$

where we denoted $\bar y_i = \frac{y_i+1}{2} \in \{0, 1\}$.

One can then minimize minus the sum of the log of the likelihoods, which reads

$$\hat\beta \in \underset{\beta \in \mathbb{R}^p}{\text{argmin}} \; -\sum_{i=1}^n \log(\mathbb{P}(\mathbf{y} = y_i|\mathbf{x} = x_i)) = \sum_{i=1}^n -(1-\bar y_i)\log\frac{e^{s_i}}{1+e^{s_i}} - \bar y_i \log\frac{1}{1+e^{s_i}}$$

Some algebraic manipulations shows that this equivalent to an ERM-type form (16.7) with a logistic loss function

$$\hat\beta \in \underset{\beta \in \mathbb{R}^p}{\text{argmin}}\; E(\beta) = \frac{1}{n}\sum_{i=1}^n \mathcal{L}(\langle x_i,\, \beta\rangle, y_i) \tag{16.18}$$

where the logistic loss reads

$$\mathcal{L}(s, y) \overset{\text{def.}}{=} \log(1 + \exp(-sy)).$$

Problem (16.18) is a smooth convex minimization. If X is injective, E is also strictly convex, hence it has a single global minimum.

Figure (16.9) compares the binary (ideal) 0-1 loss, the logistic loss and the hinge loss (the one used for SVM).

Re-writting the energy to minimize

$$E(\beta) = \tilde{\mathcal{L}}(X\beta, y) \quad \text{where} \quad \tilde{\mathcal{L}}(s, y) = \frac{1}{n}\sum_i \mathcal{L}(s_i, y_i),$$

its gradient reads

$$\nabla E(\beta) = X^\top \nabla\tilde{\mathcal{L}}(X\beta, y) \quad \text{where} \quad \nabla\tilde{\mathcal{L}}(s, y) = \frac{y}{n} \odot \theta(-y \odot s),$$

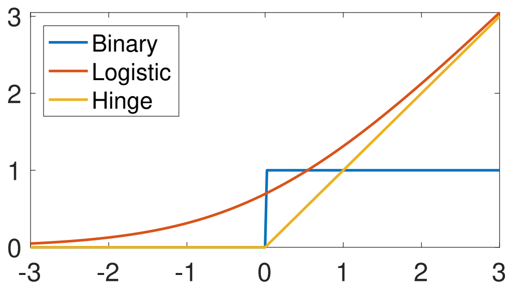


Figure 16.9: Comparison of loss functions.

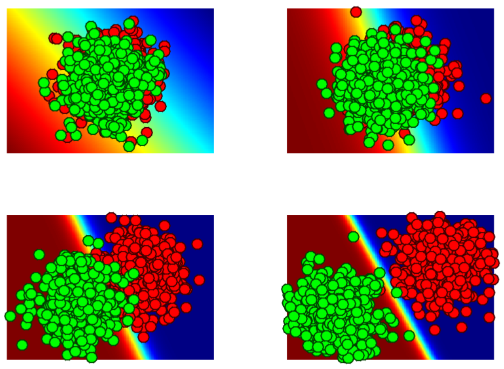


Figure 16.10: Influence on the separation distance between the class on the classification probability.

where $\odot$ is the pointwise multiplication operator, i.e. `.*` in Matlab. Once $\beta^{(\ell=0)} \in \mathbb{R}^p$ is initialized (for instance at 0_p), one step of gradient descent (13.2) reads

$$\beta^{(\ell+1)} = \beta^{(\ell)} - \tau_\ell \nabla E(\beta^{(\ell)}).$$

To understand the behavior of the method, in Figure 16.10 we generate synthetic data distributed according to a mixture of Gaussian with an overlap governed by an offset ω. One can display the data overlaid on top of the classification probability, this highlight the separating hyperplane $\{x \; ; \; \langle \beta, \, x \rangle = 0\}$.

16.2.3 Kernelized Logistic Classification

Logistic classification tries to separate the classes using a linear separating hyperplane $\{x \; ; \; \langle w, \, x \rangle = 0\}$.

In order to generate a non-linear decision boundary, one can replace the parametric linear model by a non-linear non-parametric model, thanks to kernelization. It is non-parametric in the sense that the number of parameter grows with the number n of sample (while for the basic method, the number of parameter is p). This allows in particular to generate decision boundaries of arbitrary complexity.

The downside is that the numerical complexity of the method grows (at least) quadratically with n.

The good news however is that thanks to the theory of reproducing kernel Hilbert spaces (RKHS), one can still compute this non-linear decision function using (almost) the same numerical algorithm.

Similarly to (16.15), given a kernel $\kappa(x,z) \in \mathbb{R}$ defined for $(x,z) \in \mathbb{R}^p$, the kernelized method replace the linear decision functional $f(x) = \langle x, \, \beta \rangle$ by a sum of $f(x, \alpha)$ of kernel centered on the samples

$$f(x,\alpha) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$$

where $\alpha \in \mathbb{R}^n$ is the unknown vector of weight to find.

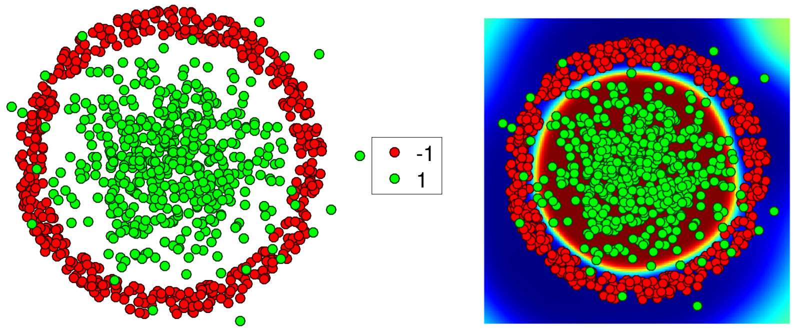


Figure 16.11: Non-linear classification using a Gaussian kernel.

The kernelized Logistic minimization reads

$$\min_{\alpha \in \mathbb{R}^n} F\mathcal{F}(\alpha) \overset{\text{def.}}{=} \tilde{\mathcal{L}}(K\alpha, y)$$

where K is the sampled kernel (16.16). One can then use a gradient descent to minimize $\mathcal{F}(\alpha)$.

Once this optimal α has been found, class probability at a point x are obtained as

$$(\theta(f(x,\alpha)), 1 - \theta(f(x,\alpha))$$

where $f(\cdot, \alpha)$ is defined in (16.15).

Note that it is of course possible to add a regularization to this problem

$$\min_{\alpha} \tilde{\mathcal{L}}(K\alpha, y) + \lambda R(\alpha),$$

where for instance $R = \|\cdot\|_2^2$ or $R = \|\cdot\|_1$.

16.2.4 Multi-Classes Logistic Classification

The logistic classification method is extended to an arbitrary number k of classes by considering a family of weight vectors $\beta = (\beta_\ell)_{\ell=1}^k$, which are conveniently stored as columns of matrix $\beta \in \mathbb{R}^{p \times k}$.

This allows one to model probabilistically the belonging of a point $x \in \mathbb{R}^p$ to the classes using an exponential model

$$h(x) = \left(\frac{e^{-\langle x,\, \beta_\ell \rangle}}{\sum_m e^{-\langle x,\, \beta_m \rangle}} \right)_\ell$$

This vector $h(x) \in [0,1]^k$ describes the probability of x belonging to the different classes, and $\sum_\ell h(x)_\ell = 1$.

The computation of β is obtained by solving a maximum likelihood estimator

$$\max_{\beta \in \mathbb{R}^{p \times k}} \frac{1}{n} \sum_{i=1}^n \log(h(x_i)_{y_i})$$

where we recall that $y_i \in \mathcal{Y} = \{1, \ldots, k\}$ is the class index of the point x_i.

This is conveniently rewritten as

$$\min_{\beta \in \mathbb{R}^{p \times k}} \mathcal{E}(\beta) \overset{\text{def.}}{=} \sum_i \text{LSE}(X\beta)_i - \langle X\beta,\, D \rangle$$

where $D \in \{0,1\}^{n \times k}$ is the binary class index matrices

$$D_{i,\ell} = \begin{cases} 1 & \text{if} \quad y_i = \ell, \\ 0\text{otherwise.} \end{cases}$$

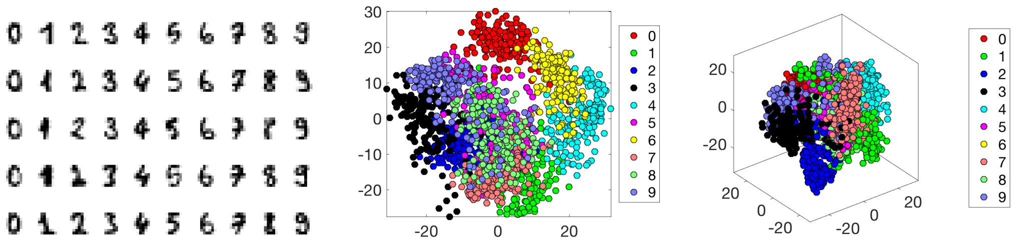


Figure 16.12: 2-D and 3-D PCA vizualization of the digits images.

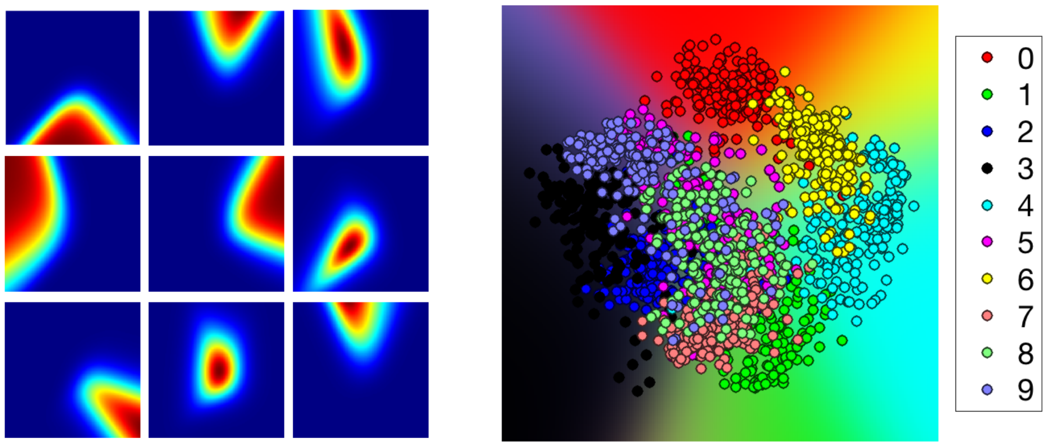


Figure 16.13: Results of digit classification Left: probability of belonging to each of the 9 first classes (displayed over a 2-D PCA space). Right: colors reflect probability of belonging to classes.

and LSE is the log-sum-exp operator

$$\mathrm{LSE}(S) = \log\left(\sum_\ell \exp(S_{i,\ell})\right) \in \mathbb{R}^n.$$

The computation of LSE is unstable for large value of $S_{i,\ell}$ (numerical overflow, producing NaN), but this can be fixed by subtracting the largest element in each row, since $\mathrm{LSE}(S+a) = \mathrm{LSE}(S)+a$ if a is constant along rows. This is often referred to as the "LSE trick".

The gradient of the LSE operator is the soft-max operator

$$\nabla \mathrm{LSE}(S) = \mathrm{SM}(S) \overset{\text{def.}}{=} \left(\frac{e^{S_{i,\ell}}}{\sum_m e^{S_{i,m}}}\right)$$

Similarly to the LSE, it needs to be stabilized.

We load a dataset of n images of size $p = 8 \times 8$, representing digits from 0 to 9 (so there are $k = 10$ classes). Figure 16.12 displays a few representative examples as well as 2-D and 3-D PCA projections.

Once D matrix is computed, the gradient of $\mathcal{E}$ is computed as

$$\nabla \mathcal{E}(\beta) = \frac{1}{n} X^\top (\mathrm{SM}(X\beta) - D).$$

and one can minimize $\mathcal{E}$ using for instance a gradient descent scheme. Figure (16.13) displays the "fuzzy" decision boundary by vizualizing the value of $h(x)$ using colors on an image regular grid.

Bibliography

[1] P. Alliez and C. Gotsman. Recent advances in compression of 3d meshes. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Advances in multiresolution for geometric modelling*, pages 3–26. Springer Verlag, 2005.

[2] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. Recent advances in remeshing of surfaces. In *AIM@SHAPE repport*. 2005.

[3] Amir Beck. *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*. SIAM, 2014.

[4] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[6] E. Candès and D. Donoho. New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities. *Commun. on Pure and Appl. Math.*, 57(2):219–266, 2004.

[7] E. J. Candès. The restricted isometry property and its implications for compressed sensing. *Compte Rendus de l'Académie des Sciences*, Serie I(346):589–592, 2006.

[8] E. J. Candès, L. Demanet, D. L. Donoho, and L. Ying. Fast discrete curvelet transforms. *SIAM Multiscale Modeling and Simulation*, 5:861–899, 2005.

[9] A. Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vis.*, 20:89–97, 2004.

[10] Antonin Chambolle, Vicent Caselles, Daniel Cremers, Matteo Novaga, and Thomas Pock. An introduction to total variation for image analysis. *Theoretical foundations and numerical methods for sparse recovery*, 9(263-340):227, 2010.

[11] Antonin Chambolle and Thomas Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.

[12] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[13] F. R. K. Chung. Spectral graph theory. *Regional Conference Series in Mathematics, American Mathematical Society*, 92:1–212, 1997.

[14] Philippe G Ciarlet. Introduction à l'analyse numérique matricielle et à l'optimisation. 1982.

[15] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *SIAM Multiscale Modeling and Simulation*, 4(4), 2005.

[16] P. Schroeder et al. D. Zorin. Subdivision surfaces in character animation. In *Course notes at SIGGRAPH 2000*, July 2000.

[17] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. on Pure and Appl. Math.*, 57:1413–1541, 2004.

[18] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. *J. Fourier Anal. Appl.*, 4(3):245–267, 1998.

[19] D. Donoho and I. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, Dec 1994.

[20] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.

[21] M. Figueiredo and R. Nowak. An EM Algorithm for Wavelet-Based Image Restoration. *IEEE Trans. Image Proc.*, 12(8):906–916, 2003.

[22] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Advances in multiresolution for geometric modelling*, pages 157–186. Springer Verlag, 2005.

[23] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*, volume 1. Birkhäuser Basel, 2013.

[24] I. Guskov, W. Sweldens, and P. Schröder. Multiresolution signal processing for meshes. In Alyn Rockwood, editor, *Proceedings of the Conference on Computer Graphics (Siggraph99)*, pages 325–334. ACM Press, August8–13 1999.

[25] A. Khodakovsky, P. Schröder, and W. Sweldens. Progressive geometry compression. In *Proceedings of the Computer Graphics Conference 2000 (SIGGRAPH-00)*, pages 271–278, New York, July 23–28 2000. ACMPress.

[26] L. Kobbelt. $\sqrt{3}$ subdivision. In Sheila Hoffmeyer, editor, *Proc. of SIGGRAPH'00*, pages 103–112, New York, July 23–28 2000. ACMPress.

[27] M. Lounsbery, T. D. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Trans. Graph.*, 16(1):34–73, 1997.

[28] S. Mallat. *A Wavelet Tour of Signal Processing, 3rd edition.* Academic Press, San Diego, 2009.

[29] Stephane Mallat. *A wavelet tour of signal processing: the sparse way.* Academic press, 2008.

[30] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Commun. on Pure and Appl. Math.*, 42:577–685, 1989.

[31] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[32] Gabriel Peyré. *L'algèbre discrète de la transformée de Fourier.* Ellipses, 2004.

[33] Gabriel Peyré and Marco Cuturi. Computational optimal transport. 2017.

[34] J. Portilla, V. Strela, M.J. Wainwright, and Simoncelli E.P. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Trans. Image Proc.*, 12(11):1338–1351, November 2003.

[35] E. Praun and H. Hoppe. Spherical parametrization and remeshing. *ACM Transactions on Graphics*, 22(3):340–349, July 2003.

[36] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.

[37] Filippo Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, 2015.

[38] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, Frank Lenzen, and L Sirovich. *Variational methods in imaging.* Springer, 2009.

[39] P. Schröder and W. Sweldens. Spherical Wavelets: Efficiently Representing Functions on the Sphere. In *Proc. of SIGGRAPH 95*, pages 161–172, 1995.

[40] P. Schröder and W. Sweldens. Spherical wavelets: Texture processing. In P. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95.* Springer Verlag, Wien, New York, August 1995.

[41] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[42] A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis.*, 2(2):105–171, 2006.

[43] Jean-Luc Starck, Fionn Murtagh, and Jalal Fadili. *Sparse image and signal processing: Wavelets and related geometric multiscale analysis.* Cambridge university press, 2015.

[44] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computation Harmonic Analysis*, 3(2):186–200, 1996.

[45] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1997.