Modern Statistical Methods

Rajen D. Shah r.shah@statslab.cam.ac.uk

Course webpage:
http://www.statslab.cam.ac.uk/~rds37/modern_stat_methods.html

The field of statistics has undergone profound changes in recent decades. Firstly, the types of datasets that statisticians are asked to analyse have transformed dramatically. In the past, we typically dealt with datasets containing many observations and a modest number of carefully chosen variables. Today, by contrast, it is common to encounter datasets with thousands of variables—sometimes even far exceeding the number of observations. For instance, in genomics, we might measure the expression levels of several thousand genes but only across a few hundred tissue samples. Classical statistical methods are often simply not applicable in these "high-dimensional" settings. As the scale of data collection has expanded, so too has the scope of the questions we seek to answer. Whereas statistics was once primarily concerned with uncovering associations between variables, we are now increasingly interested in understanding the causal structure of data. And rather than focusing solely on prediction, we often aim to predict the effects of interventions. At the same time, the rapid rise of machine learning has provided us with powerful new tools. In this course, we will explore how these advances can be harnessed to tackle some of the modern statistical challenges outlined above. The selection of material is heavily biased towards my own interests, but I hope it will nevertheless give you a flavour of some of the most important recent methodological developments in statistics.

The course is divided into 4 chapters (of unequal size). Our **first chapter** will start by introducing ridge regression, a simple generalisation of ordinary least squares. Our study of this will lead us to some beautiful connections with functional analysis and ultimately one of the most successful and flexible classes of learning algorithms: kernel machines. The **second chapter** concerns the Lasso and its extensions. The Lasso has been at the centre of much of the developments that have occurred in high-dimensional statistics, and will allow us to perform regression in the seemingly hopeless situation when the number of parameters we are trying to estimate is larger than the number of observations. In the **third chapter** we will study graphical modelling and provide an introduction to the exciting field of causal inference. Where the previous chapters consider methods for relating a particular response variable to a potentially large collection of (explanatory) variables, in the third chapter, we will study how to infer relationships between the variables themselves and answer causal questions using so-called *double/debiased machine learning* approaches. In the **final chapter**, we will turn to the problem of multiple testing which concerns handling settings where we may be performing thousands of hypothesis tests at the same time.

Before we begin the main content of the course, we will briefly review two key classical statistical methods: ordinary least squares and maximum likelihood estimation. This will help to set the scene and provide a warm-up for the modern methods to come later.

Classical statistics

Ordinary least squares

Imagine data are available in the form of observations $(Y_i, x_i) \in \mathbb{R} \times \mathbb{R}^p$, $i = 1, \ldots, n$, and the aim is to infer a simple *regression function* relating the average value of a *response*, Y_i, and a collection of *predictors* or *variables*, x_i. This is an example of regression analysis, one of the most important tasks in statistics.

A *linear model* for the data assumes that it is generated according to

$$Y = X\beta^0 + \varepsilon, \tag{1}$$

where $Y \in \mathbb{R}^n$ is the vector of responses; $X \in \mathbb{R}^{n \times p}$ is the predictor matrix (or design matrix) with ith row $x_i^\top$; $\varepsilon \in \mathbb{R}^n$ represents random error; and $\beta^0 \in \mathbb{R}^p$ is the unknown vector of coefficients.

Provided $p \ll n$, a sensible way to estimate β is by ordinary least squares (OLS). This yields an estimator $\hat{\beta}^{\mathrm{OLS}}$ with

$$\hat{\beta}^{\mathrm{OLS}} := \arg\min_{\beta \in \mathbb{R}^p} \|Y - X\beta\|_2^2 = (X^\top X)^{-1} X^\top Y, \tag{2}$$

provided X has full column rank.

Under the assumptions that (i) $\mathbb{E}(\varepsilon_i) = 0$ and (ii) $\mathrm{Var}(\varepsilon) = \sigma^2 I$, we have that:

- $\mathbb{E}_{\beta^0,\sigma^2}(\hat{\beta}^{\mathrm{OLS}}) = \mathbb{E}\{(X^\top X)^{-1} X^\top (X\beta^0 + \varepsilon)\} = \beta^0$.
- $\mathrm{Var}_{\beta^0,\sigma^2}(\hat{\beta}^{\mathrm{OLS}}) = (X^\top X)^{-1} X^\top \mathrm{Var}(\varepsilon) X (X^\top X)^{-1} = \sigma^2 (X^\top X)^{-1}$.

The Gauss–Markov theorem states that OLS is the best linear unbiased estimator in our setting: for any other estimator $\tilde{\beta}$ that is linear in Y (so $\tilde{\beta} = AY$ for some fixed matrix A), we have

$$\mathrm{Var}_{\beta^0,\sigma^2}(\tilde{\beta}) - \mathrm{Var}_{\beta^0,\sigma^2}(\hat{\beta}^{\mathrm{OLS}})$$

is positive semi-definite.

Maximum likelihood estimation

The method of least squares is just one way to construct as estimator. A more general technique is that of maximum likelihood estimation. Here given data $y \in \mathbb{R}^n$ that we take as a realisation of a random variable Y, we specify its density $f(y; \theta)$ up to some unknown vector of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$, where Θ is the parameter space. The likelihood function is a function of θ for each fixed y given by

$$L(\theta) := L(\theta; y) = c(y) f(y; \theta),$$

where $c(y)$ is an arbitrary constant of proportionality. The maximum likelihood estimate of θ maximises the likelihood, or equivalently it maximises the log-likelihood

$$\ell(\theta) := \ell(\theta; y) = \log f(y; \theta) + \log(c(y)).$$

A very useful quantity in the context of maximum likelihood estimation is the *Fisher information* matrix with jkth ($1 \leq j, k \leq d$) entry

$$i_{jk}(\theta) := -\mathbb{E}_\theta\biggl\{\frac{\partial^2}{\partial\theta_j\partial\theta_k}\ell(\theta)\biggr\}.$$

It can be thought of as a measure of how hard it is to estimate θ when it is the true parameter value. The Cramér–Rao lower bound states that if $\tilde{\theta}$ is an unbiased estimator of θ, then under regularity conditions,

$$\mathrm{Var}_\theta(\tilde{\theta}) - i^{-1}(\theta)$$

is positive semi-definite.

A remarkable fact about maximum likelihood estimators (MLEs) is that (under quite general conditions) they are asymptotically normally distributed, asymptotically unbiased and asymptotically achieve the Cramér–Rao lower bound.

Assume that the Fisher information matrix when there are n observations, $i^{(n)}(\theta)$ (where we have made the dependence on n explicit) satisfies $i^{(n)}(\theta)/n \to I(\theta)$ for some positive definite matrix I. Then denoting the maximum likelihood estimator of θ when there are n observations by $\hat{\theta}^{(n)}$, under regularity conditions, as the number of observations $n \to \infty$ we have

$$\sqrt{n}(\hat{\theta}^{(n)} - \theta) \overset{d}{\to} N_d(0, I^{-1}(\theta)).$$

Returning to our linear model, if we assume in addition that $\varepsilon_i \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2)$, then the log-likelihood for (β, σ^2) is

$$\ell(\beta, \sigma^2) = -\frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^n (y_i - x_i^\top\beta)^2.$$

We see that the maximum likelihood estimate of β and OLS coincide. It is easy to check that

$$i(\beta, \sigma^2) = \begin{pmatrix} \sigma^{-2}X^\top X & 0 \\ 0 & n\sigma^{-4}/2 \end{pmatrix}.$$

The general theory for MLEs would suggest that approximately $\sqrt{n}(\hat{\beta} - \beta) \sim N_p(0, \sigma^2(n^{-1}X^\top X)^{-1})$; in fact it is straight-forward to show that this distributional result is exact.

Chapter 1

Kernel machines

Let us revisit the linear model with

$$Y_i = x_i^\top \beta^0 + \varepsilon_i.$$

For unbiased estimators of β^0, their variance gives a way of comparing their quality in terms of squared error loss. For a potentially biased estimator, $\tilde{\beta}$, the relevant quantity is the mean-squared error (MSE),

$$\begin{aligned}\mathbb{E}_{\beta^0,\sigma^2}\{(\tilde{\beta}-\beta^0)(\tilde{\beta}-\beta^0)^\top\} &= \mathbb{E}[\{\tilde{\beta}-\mathbb{E}(\tilde{\beta})+\mathbb{E}(\tilde{\beta})-\beta^0\}\{\tilde{\beta}-\mathbb{E}(\tilde{\beta})+\mathbb{E}(\tilde{\beta})-\beta^0\}^\top] \\ &= \mathrm{Var}(\tilde{\beta})+\{\mathbb{E}(\tilde{\beta}-\beta^0)\}\{\mathbb{E}(\tilde{\beta}-\beta^0)\}^\top,\end{aligned}$$

a sum of squared bias and variance terms. A crucial part of the optimality arguments for OLS and MLEs was *unbiasedness*. Do there exist biased methods whose variance is is reduced compared to OLS such that their overall prediction error is lower? Yes—in fact the use of biased estimators is essential in dealing with settings where the number of parameters to be estimated is large compared to the number of observations. In the first two chapters we will explore two important methods for variance reduction based on different forms of penalisation: rather than forming estimators via optimising a least squares or log-likelihood term, we will introduce an additional penalty term that encourages estimates to be shrunk towards 0 in some sense. This will allow us to produce reliable estimators that work well when classical MLEs are infeasible, and in other situations can greatly outperform the classical approaches.

1.1 Ridge regression

One way to reduce the variance of $\hat{\beta}^{\mathrm{OLS}}$ is to shrink the estimated coefficients towards 0. *Ridge regression* [Hoerl and Kennard, 1970] does this by solving the following optimisation problem

$$(\hat{\mu}_\lambda^{\mathrm{R}}, \hat{\beta}_\lambda^{\mathrm{R}}) = \underset{(\mu,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\arg\min}\,\{\|Y-\mu\mathbf{1}-X\beta\|_2^2+\lambda\|\beta\|_2^2\}.$$

Here $\mathbf{1}$ is an n-vector of 1's. We see that the usual OLS objective is penalised by an additional term proportional to $\|\beta\|_2^2$. The parameter $\lambda \geq 0$, which controls the severity of the penalty and therefore the degree of the shrinkage towards 0, is known as a *regularisation parameter* or *tuning parameter*. We have explicitly included an intercept term which is not penalised. The reason for this is that were the variables to have their origins shifted so e.g. a variable representing temperature is given in units of Kelvin rather than Celsius, the fitted values would not change. However, $X\hat{\beta}$ is not invariant under scale transformations of the variables so it is standard practice to centre each column of X (hence making them orthogonal to the intercept term) and then scale them to have ℓ_2-norm $\sqrt{n}$.

It is straightforward to show that after this standardisation of X, $\hat{\mu}_\lambda^{\mathrm{R}} = \bar{Y} := \sum_{i=1}^n Y_i/n$, so we may assume that $\sum_{i=1}^n Y_i = 0$ by replacing Y_i by $Y_i - \bar{Y}$ and then we can remove μ from our objective function. In this case

$$\hat{\beta}_\lambda^{\mathrm{R}} = (X^\top X + \lambda I)^{-1} X^\top Y.$$

In this form, we can see how the addition of the λI term helps to stabilise the estimator. Note that when X does not have full column rank (such as in high-dimensional situations), we can still compute this estimator. On the other hand, when X does have full column rank, we have the following theorem.

Theorem 1. *For λ sufficiently small (depending on β^0 and σ^2),*

$$\mathbb{E}(\hat{\beta}^{\mathrm{OLS}} - \beta^0)(\hat{\beta}^{\mathrm{OLS}} - \beta^0)^\top - \mathbb{E}(\hat{\beta}_\lambda^{\mathrm{R}} - \beta^0)(\hat{\beta}_\lambda^{\mathrm{R}} - \beta^0)^\top$$

is positive definite.

Proof. First we compute the bias of $\hat{\beta}_\lambda^{\mathrm{R}}$. We drop the subscript λ and superscript R for convenience.

$$\begin{aligned}\mathbb{E}(\hat{\beta}) - \beta^0 &= (X^\top X + \lambda I)^{-1} X^\top X \beta^0 - \beta^0 \\ &= (X^\top X + \lambda I)^{-1}(X^\top X + \lambda I - \lambda I)\beta^0 - \beta^0 \\ &= -\lambda (X^\top X + \lambda I)^{-1}\beta^0.\end{aligned}$$

Now we look at the variance of $\hat{\beta}$.

$$\begin{aligned}\mathrm{Var}(\hat{\beta}) &= \mathbb{E}\{(X^\top X + \lambda I)^{-1} X^\top \varepsilon\}\{(X^\top X + \lambda I)^{-1} X^\top \varepsilon\}^\top \\ &= \sigma^2 (X^\top X + \lambda I)^{-1} X^\top X (X^\top X + \lambda I)^{-1}.\end{aligned}$$

Thus $\mathbb{E}(\hat{\beta}^{\mathrm{OLS}} - \beta^0)(\hat{\beta}^{\mathrm{OLS}} - \beta^0)^\top - \mathbb{E}(\hat{\beta} - \beta^0)(\hat{\beta} - \beta^0)^\top$ is equal to

$$\sigma^2 (X^\top X)^{-1} - \sigma^2 (X^\top X + \lambda I)^{-1} X^\top X (X^\top X + \lambda I)^{-1} - \lambda^2 (X^\top X + \lambda I)^{-1} \beta^0 \beta^{0\top} (X^\top X + \lambda I)^{-1}.$$

After some simplification, we see that this is equal to

$$\lambda (X^\top X + \lambda I)^{-1} [\sigma^2 \{2I + \lambda (X^\top X)^{-1}\} - \lambda \beta^0 \beta^{0\top}] (X^\top X + \lambda I)^{-1}.$$

Thus $\mathbb{E}(\hat{\beta}^{\mathrm{OLS}} - \beta^0)(\hat{\beta}^{\mathrm{OLS}} - \beta^0)^\top - \mathbb{E}(\hat{\beta} - \beta^0)(\hat{\beta} - \beta^0)^\top$ is positive definite for $\lambda > 0$ if and only if

$$\sigma^2\{2I + \lambda(X^\top X)^{-1}\} - \lambda\beta^0\beta^{0\top}$$

is positive definite, which is true for $\lambda > 0$ sufficiently small (we can take $0 < \lambda < 2\sigma^2/\|\beta^0\|_2^2$). □

The theorem says that $\hat{\beta}_\lambda^{\mathrm{R}}$ outperforms $\hat{\beta}^{\mathrm{OLS}}$ provided λ is chosen appropriately. To be able to use ridge regression effectively, we need a way of selecting a good λ—we will come to this very shortly. What the theorem doesn't really tell us is in what situations we expect ridge regression to perform well. To understand that, we will turn to one of the key matrix decompositions used in statistics, the singular value decomposition (SVD).

1.1.1 The singular value decomposition and principal components analysis

The singular value decomposition (SVD) is a generalisation of an eigendecomposition of a square matrix. We can factorise any $X \in \mathbb{R}^{n\times p}$ into its SVD

$$X = UDV^\top.$$

Here the $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{p\times p}$ are orthogonal matrices and $D \in \mathbb{R}^{n\times p}$ has $D_{11} \geq D_{22} \geq \cdots \geq D_{mm} \geq 0$ where $m := \min(n,p)$ and all other entries of D are zero. To compute such a decomposition requires $O(np\min(n,p))$ operations. The rth columns of U and V are known as the rth left and right singular vectors of X respectively, and D_{rr} is the rth singular value.

When $n > p$, we can replace U by its first p columns and D by its first p rows to produce another version of the SVD (sometimes known as the thin SVD). Then $X = UDV^\top$ where $U \in \mathbb{R}^{n\times p}$ has orthonormal columns (but is no longer square) and D is square and diagonal. There is an equivalent version for when $p > n$.

Let us take $X \in \mathbb{R}^{n\times p}$ as our matrix of predictors and suppose $n \geq p$. Using the (thin) SVD we may write the fitted values from ridge regression as follows.

$$\begin{aligned}
X\hat{\beta}_\lambda^{\mathrm{R}} &= X(X^\top X + \lambda I)^{-1}X^\top Y \\
&= UDV^\top(VD^2V^\top + \lambda I)^{-1}VDU^\top Y \\
&= UD(D^2 + \lambda I)^{-1}DU^\top Y \\
&= \sum_{j=1}^p U_j \frac{D_{jj}^2}{D_{jj}^2 + \lambda} U_j^\top Y.
\end{aligned}$$

Here we have used the notation (that we shall use throughout the course) that U_j is the jth column of U. For comparison, the fitted values from OLS (when X has full column rank) are

$$X\hat{\beta}^{\mathrm{OLS}} = X(X^\top X)^{-1}X^\top Y = UU^\top Y.$$

Both OLS and ridge regression compute the coordinates of Y with respect to the columns of U. Ridge regression then shrinks these coordinates by the factors $D_{jj}^2/(D_{jj}^2+\lambda)$; if D_{jj} is small, the amount of shrinkage will be larger.

To interpret this further, note that the SVD is intimately connected with Principal Components Analysis (PCA). Consider $v \in \mathbb{R}^p$ with $\|v\|_2 = 1$. Since the columns of X have had their means subtracted, the sample variance of $Xv \in \mathbb{R}^n$, is

$$\frac{1}{n} v^\top X^\top X v = \frac{1}{n} v^\top V D^2 V^\top v.$$

Writing $a = V^\top v$, so $\|a\|_2 = 1$, we have

$$\frac{1}{n} v^\top V D^2 V^\top v = \frac{1}{n} a^\top D^2 a = \frac{1}{n} \sum_j a_j^2 D_{jj}^2 \leq \frac{1}{n} D_{11} \sum_j a_j^2 = \frac{1}{n} D_{11}^2.$$

As $\|XV_1\|_2^2/n = D_{11}^2/n$, V_1 determines the linear combination of the columns of X which has the largest sample variance, when the coefficients of the linear combination are constrained to have ℓ_2-norm 1. $XV_1 = D_{11}U_1$ is known as the first principal component of X. Subsequent principal components $D_{22}U_2, \ldots, D_{pp}U_p$ have maximum variance D_{jj}^2/n, subject to being orthogonal to all earlier ones—see example sheet 1 for details.

Returning to ridge regression, we see that it shrinks Y most in the smaller principal components of X. Thus it will work well when most of the signal is in the large principal components of X. We now turn to the problem of choosing λ.

1.2 v-fold cross-validation

Cross-validation is a general technique for selecting a good regression method from among several competing regression methods. We illustrate the principle with ridge regression, where we have a family of regression methods given by different λ values.

So far, we have considered the matrix of predictors X as fixed and non-random. However, in many cases, it makes sense to think of it as random. Let us assume that our data are i.i.d. pairs (x_i, Y_i), $i = 1, \ldots, n$. Then ideally, we might want to pick a λ value such that

$$\mathbb{E}\{(Y^* - x^{*\top} \hat{\beta}_\lambda^{\mathrm{R}}(X, Y))^2 | X, Y\} \tag{1.1}$$

is minimised. Here $(x^*, Y^*) \in \mathbb{R}^p \times \mathbb{R}$ is independent of (X, Y) and has the same distribution as (x_1, Y_1), and we have made the dependence of $\hat{\beta}_\lambda^{\mathrm{R}}$ on the training data (X, Y) explicit. This λ is such that conditional on the original *training* data, it minimises the expected prediction error on a new observation drawn from the same distribution as the training data.

A less ambitious goal is to find a λ value to minimise the expected prediction error,

$$\mathbb{E}[\mathbb{E}\{(Y^* - x^{*\top} \hat{\beta}_\lambda^{\mathrm{R}}(X, Y))^2 | X, Y\}] \tag{1.2}$$

where compared with (1.1), we have taken a further expectation over the training set.

We still have no way of computing (1.2) directly, but we can attempt to estimate it. The idea of v-fold cross-validation is to split the data into v groups or folds of roughly equal size: $(X^{(1)}, Y^{(1)}), \ldots, (X^{(v)}, Y^{(v)})$. Let $(X^{(-k)}, Y^{(-k)})$ be all the data except that in the kth fold. For each λ on a grid of values, we compute $\hat{\beta}^{\mathrm{R}}_{\lambda}(X^{(-k)}, Y^{(-k)})$: the ridge regression estimate based on all the data except the kth fold. Writing $\kappa(i)$ for the fold to which (x_i, Y_i) belongs, we choose the value of λ that minimises

$$\mathrm{CV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\{Y_i - x_i^{\top}\hat{\beta}^{\mathrm{R}}_{\lambda}(X^{(-\kappa(i))}, Y^{(-\kappa(i))})\}^2. \tag{1.3}$$

Writing λ_{CV} for the minimiser, our final estimate of β^0 can then be $\hat{\beta}^{R}_{\lambda_{\mathrm{CV}}}(X, Y)$.

Note that for each i,

$$\mathbb{E}\{Y_i - x_i^{\top}\hat{\beta}^{\mathrm{R}}_{\lambda}(X^{(-\kappa(i))}, Y^{(-\kappa(i))})\}^2 = \mathbb{E}[\mathbb{E}\{Y_i - x_i^{\top}\hat{\beta}^{\mathrm{R}}_{\lambda}(X^{(-\kappa(i))}, Y^{(-\kappa(i))})\}^2 | X^{(-\kappa(i))}, Y^{(-\kappa(i))}]. \tag{1.4}$$

This is precisely the expected prediction error in (1.2) but with the training data X, Y replaced with a training data set of smaller size. If all the folds have the same size, then $\mathrm{CV}(\lambda)$ is an average of n identically distributed quantities, each with expected value as in (1.4). However, the quantities being averaged are not independent as they share the same data.

Thus cross-validation gives a biased estimate of the expected prediction error. The amount of the bias depends on the size of the folds, the case when the $v = n$ giving the least bias—this is known as leave-one-out cross-validation. The quality of the estimate, though, may be worse as the quantities being averaged in (1.3) will be highly positively correlated. Typical choices of v are 5 or 10.

Cross-validation aims to allow us to choose the single best λ (or more generally regression procedure); we could instead aim to find the best weighted combination of regression procedures. Returning to our ridge regression example, suppose λ is restricted to a grid of values $\lambda_1 > \lambda_2 > \cdots > \lambda_L$. We can then minimise

$$\frac{1}{n}\sum_{i=1}^{n}\left\{Y_i - \sum_{l=1}^{L} w_l x_i^{\top}\hat{\beta}^{\mathrm{R}}_{\lambda_l}(X^{(-\kappa(i))}, Y^{(-\kappa(i))})\right\}^2$$

over $w \in \mathbb{R}^L$ subject to $w_l \geq 0$ for all l. This is a non-negative least-squares optimisation, for which efficient algorithms are available. This is known as *stacking* [Wolpert, 1992, Breiman, 1996] and it can often outperform cross-validation.

1.3 The kernel trick

The fitted values from ridge regression are

$$X(X^{\top}X + \lambda I)^{-1}X^{\top}Y. \tag{1.5}$$

An alternative way of writing this is suggested by the following

$$\begin{aligned} X^\top (XX^\top + \lambda I) &= (X^\top X + \lambda I)X^\top \\ (X^\top X + \lambda I)^{-1}X^\top &= X^\top (XX^\top + \lambda I)^{-1} \\ X(X^\top X + \lambda I)^{-1}X^\top Y &= XX^\top (XX^\top + \lambda I)^{-1}Y. \end{aligned} \tag{1.6}$$

Two remarks are in order:

- Note while $X^\top X$ is $p \times p$, $XX^\top$ is $n \times n$. Computing fitted values using (1.5) would require roughly $O(np^2 + p^3)$ operations. If $p \gg n$ this could be extremely costly. However, our alternative formulation would only require roughly $O(n^2p + n^3)$ operations, which could be substantially smaller.
- We see that the fitted values of ridge regression depend only on inner products $K = XX^\top$ between observations (note $K_{ij} = x_i^\top x_j$).

Now suppose that we believe the signal depends quadratically on the predictors:

$$Y_i = x_i^\top \beta + \sum_{k,l} x_{ik}x_{il}\theta_{kl} + \varepsilon_i.$$

We can still use ridge regression provided we work with an enlarged set of predictors

$$x_{i1}, \ldots, x_{ip}, x_{i1}x_{i1}, \ldots, x_{i1}x_{ip}, x_{i2}x_{i1}, \ldots, x_{i2}x_{ip}, \ldots, x_{ip}x_{ip}.$$

This will give us $O(p^2)$ predictors. Our new approach to computing fitted values would therefore have complexity $O(n^2p^2 + n^3)$, which could be rather costly if p is large.

However, rather than first creating all the additional predictors and then computing the new K matrix, we can attempt to directly compute K. To this end consider

$$\begin{aligned} (1 + x_i^\top x_j)^2 &= \left(1 + \sum_k x_{ik}x_{jk}\right)^2 \\ &= 1 + 2\sum_k x_{ik}x_{jk} + \sum_{k,l} x_{ik}x_{il}x_{jk}x_{jl}. \end{aligned}$$

Observe this amounts to an inner product between vectors of the form

$$(1, \sqrt{2}x_{i1}, \ldots, \sqrt{2}x_{ip}, x_{i1}x_{i1}, \ldots, x_{i1}x_{ip}, x_{i2}x_{i1}, \ldots, x_{i2}x_{ip}, \ldots, x_{ip}x_{ip})^\top. \tag{1.7}$$

Thus if we set

$$K_{ij} = (1 + x_i^\top x_j)^2 \tag{1.8}$$

and plug this into the formula for the fitted values, it is *exactly* as if we had performed ridge regression on an enlarged set of variables given by (1.7). Now computing K using (1.8) would require only p operations per entry, so $O(n^2p)$ operations in total. It thus seems we have improved things by a factor of p using our new approach. This is a nice computational trick, but more importantly for us it serves to illustrate some general points.

- Since ridge regression only depends on inner products between observations, rather than fitting non-linear models by first mapping the original data $x_i \in \mathbb{R}^p$ to $\phi(x_i) \in \mathbb{R}^d$ (say) using some *feature map* ϕ (which could, for example introduce quadratic effects), we can instead try to directly compute $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$.
- In fact rather than thinking in terms of feature maps, we can instead try to think about an appropriate measure of similarity $k(x_i, x_j)$ between observations. Modelling in this fashion is sometimes much easier.

We will now formalise and extend what we have learnt with this example.

1.4 Kernels

We have seen how a model with quadratic effects can be fitted very efficiently by replacing the inner product matrix (known as the *Gram matrix*) $XX^\top$ in (1.6) with the matrix in (1.8). It is then natural to ask what other non-linear models can be fitted efficiently using this sort of approach.

We won't answer this question directly, but instead we will try to understand the sorts of similarity measures k that can be represented as inner products between transformations of the original data.

That is, we will study the similarity measures $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ from the input space $\mathcal{X}$ to $\mathbb{R}$ for which there exists a *feature map* $\phi : \mathcal{X} \to \mathcal{H}$ where $\mathcal{H}$ is some (real) inner product space with

$$k(x, x') = \langle \phi(x), \phi(x') \rangle. \tag{1.9}$$

Recall that an inner product space is a real vector space $\mathcal{H}$ endowed with a map $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ that obeys the following properties.

(i) Symmetry: $\langle u, v \rangle = \langle v, u \rangle$.

(ii) Linearity: for $a, b \in \mathbb{R}$ $\langle au + bw, v \rangle = a\langle u, v \rangle + b\langle w, v \rangle$.

(iii) Positive-definiteness: $\langle u, u \rangle \geq 0$ with equality if and only if $u = 0$.

Definition 1. A *positive definite kernel* or more simply a *kernel* (for brevity) k is a symmetric map $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for which for all $n \in \mathbb{N}$ and all $x_1, \ldots, x_n \in \mathcal{X}$, the matrix K with entries

$$K_{ij} = k(x_i, x_j)$$

is positive semi-definite.

A kernel is a little like an inner product, but need not be bilinear in general. However, a form of the Cauchy–Schwarz inequality does hold for kernels.

Proposition 2.

$$k(x, x')^2 \leq k(x, x)k(x', x').$$

Proof. The matrix

$$\begin{pmatrix} k(x,x) & k(x,x') \\ k(x',x) & k(x',x') \end{pmatrix}$$

must be positive semi-definite so in particular its determinant must be non-negative. □

First we show that any inner product of feature maps will give rise to a kernel.

Proposition 3. *k defined by $k(x,x') = \langle \phi(x), \phi(x') \rangle$ is a kernel.*

Proof. Let $x_1, \ldots, x_n \in \mathcal{X}$, $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and consider

$$\begin{aligned} \sum_{i,j} \alpha_i k(x_i, x_j) \alpha_j &= \sum_{i,j} \alpha_i \langle \phi(x_i), \phi(x_j) \rangle \alpha_j \\ &= \left\langle \sum_i \alpha_i \phi(x_i), \sum_j \alpha_j \phi(x_j) \right\rangle \geq 0. \end{aligned}$$

□

Showing that every kernel admits a representation of the form (1.9) is slightly more involved, and we delay this until after we have studied some examples.

1.4.1 Examples of kernels

Proposition 4. *Suppose $k_1, k_2, \ldots$ are kernels.*

(i) If $\alpha_1, \alpha_2 \geq 0$ then $\alpha_1 k_1 + \alpha_2 k_2$ is a kernel. If $\lim_{m \to \infty} k_m(x,x') =: k(x,x')$ exists for all $x, x' \in \mathcal{X}$, then k is a kernel.

(ii) The pointwise product $k = k_1 k_2$ is a kernel.

Linear kernel. $k(x,x') = x^\top x'$.

Polynomial kernel. $k(x,x') = (1 + x^\top x')^d$. To show this is a kernel, we can simply note that $1 + x^\top x'$ gives a kernel owing to the fact that 1 is a kernel and (i) of Proposition 4. Next (ii) and induction shows that k as defined above is a kernel.

Gaussian kernel. The highly popular Gaussian kernel is defined by

$$k(x,x') = \exp\left(- \frac{\|x - x'\|_2^2}{2\sigma^2} \right).$$

For x close to x' it is large whilst for x far from x' the kernel quickly decays towards 0. The additional parameter σ^2 known as the *bandwidth* controls the speed of the decay to zero. Note it is less clear how one might find a corresponding feature map and indeed any feature map that represents this must be infinite dimensional.

To show that it is a kernel first decompose $\|x - x'\|_2^2 = \|x\|_2^2 + \|x'\|_2^2 - 2x^\top x'$. Note that by Proposition 3,

$$k_1(x, x') = \exp\biggl(- \frac{\|x\|_2^2}{2\sigma^2}\biggr) \exp\biggl(- \frac{\|x'\|_2^2}{2\sigma^2}\biggr)$$

is a kernel. Next writing

$$k_2(x, x') = \exp(x^\top x'/\sigma^2) = \sum_{r=0}^{\infty} \frac{(x^\top x'/\sigma^2)^r}{r!}$$

and using (i) of Proposition 4 shows that k_2 is a kernel. Finally observing that $k = k_1 k_2$ and using (ii) shows that the Gaussian kernel is indeed a kernel.

Sobolev kernel. Take $\mathcal{X}$ to be $[0, 1]$ and let $k(x, x') = \min(x, x')$. Note this is the covariance function of Brownian motion so it must be positive definite.

Jaccard similarity kernel. Take $\mathcal{X}$ to be the set of all subsets of $\{1, \ldots, p\}$. For $x, x' \in \mathcal{X}$ with $x \cup x' \neq \emptyset$ define

$$k(x, x') = \frac{|x \cap x'|}{|x \cup x'|}$$

and if $x \cup x' = \emptyset$ then set $k(x, x') = 1$. Showing that this is a kernel is left to the example sheet.

1.4.2 Reproducing kernel Hilbert spaces

Theorem 5. *For every kernel k there exists a feature map ϕ taking values in some inner product space $\mathcal{H}$ such that*

$$k(x, x') = \langle \phi(x), \phi(x') \rangle. \tag{1.10}$$

Proof. We will take $\mathcal{H}$ to be the vector space of functions of the form

$$f(\cdot) = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i), \tag{1.11}$$

where $n \in \mathbb{N}$, $x_i \in \mathcal{X}$ and $\alpha_i \in \mathbb{R}$. Our feature map $\phi : \mathcal{X} \to \mathcal{H}$ will be

$$\phi(x) = k(\cdot, x). \tag{1.12}$$

We now define an inner product on $\mathcal{H}$. If f is given by (1.11) and

$$g(\cdot) = \sum_{j=1}^{m} \beta_j k(\cdot, x'_j) \tag{1.13}$$

we define their inner product to be

$$\langle f, g\rangle = \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_i \beta_j k(x_i, x'_j). \tag{1.14}$$

We need to check this is well-defined as the representations of f and g in (1.11) and (1.13) need not be unique. To this end, note that

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_i \beta_j k(x_i, x'_j) = \sum_{i=1}^{n} \alpha_i g(x_i) = \sum_{j=1}^{m} \beta_j f(x'_j). \tag{1.15}$$

The first equality shows that the inner product does not depend on the particular expansion of g whilst the second equality shows that it also does not depend on the expansion of f. Thus the inner product is well-defined.

First we check that with ϕ defined as in (1.12) we do have relationship (1.10). Observe that

$$\langle k(\cdot, x), f\rangle = \sum_{i=1}^{n} \alpha_i k(x_i, x) = f(x), \tag{1.16}$$

so in particular we have

$$\langle \phi(x), \phi(x')\rangle = \langle k(\cdot, x), k(\cdot, x')\rangle = k(x, x').$$

It remains to show that it is indeed an inner product. It is clearly symmetric and (1.15) shows linearity. We now need to show positive definiteness.

First note that

$$\langle f, f\rangle = \sum_{i,j} \alpha_i k(x_i, x_j)\alpha_j \geq 0 \tag{1.17}$$

by positive definiteness of the kernel. Now from (1.16),

$$f(x)^2 = (\langle k(\cdot, x), f\rangle)^2.$$

If we could use the Cauchy–Schwarz inequality on the right-hand side, we would have

$$f(x)^2 \leq \langle k(\cdot, x), k(\cdot, x)\rangle \langle f, f\rangle, \tag{1.18}$$

which would show that if $\langle f, f\rangle = 0$ then necessarily $f = 0$; the final property we need to show that $\langle \cdot, \cdot\rangle$ is an inner product. However, in order to use the traditional Cauchy–Schwarz inequality we need to first know we're dealing with an inner product, which is precisely what we're trying to show!

Although we haven't shown that $\langle \cdot, \cdot\rangle$ is an inner product, we do have enough information to show that it is itself a kernel. We may then appeal to Proposition 2 to obtain (1.18). With this in mind, we argue as follows. Given functions $f_1, \ldots, f_m$ and coefficients $\gamma_1, \ldots, \gamma_m \in \mathbb{R}$, we have

$$\sum_{i,j} \gamma_i \langle f_i, f_j\rangle \gamma_j = \left\langle \sum_i \gamma_i f_i, \sum_j \gamma_j f_j \right\rangle \geq 0$$

where we have used linearity and (1.17), showing that it is a kernel. □