Statistical Learning in Practice (Lent 2020)

Course Notes

Dr Alberto J Coca[1]

March 7, 2020

[1] These notes originate from those of Dr Tengyao Wang's from 2018–2019; I thank him for sharing the original version with me. Please, email me if you find any typos/errors.

Contents

Chapter 1

Regression models

In regression models we assume that the data consists of pairs of observations $(x_1, y_1), \dots, (x_n, y_n)$, where the $y_i, i = 1, \dots, n$ (we refer to n as the sample size), are often called the response, target or dependent variables (or labels in classification models), and the x_i are known as predictors, covariates, independent variables or explanatory variables. Unless otherwise mentioned, we assume that $y_i \in \mathcal{Y}$ and $x_i \in \mathcal{X}$ for some $\mathcal{Y} \subseteq \mathbb{R}$ and $\mathcal{X} \subseteq \mathbb{R}^p, p \geq 1$. Regression models relate the responses to the covariates through a functional relationship and our goal is to estimate or perform inference on this so-called *regression function*. We should always keep in mind that models are an attempt to approximate reality and that, traditionally, they have been as simple and interpretable as possible to facilitate computability and usefulness.

1.1 Generalised linear models

The most important regression model is the linear model. Recall that it postulates that the responses are realisations of the random variables

$$Y_i = x_i^T \beta + \varepsilon_i$$

for some $\beta = (\beta_1, \dots, \beta_p)^T \in \mathbb{R}^p$ and some random errors ε_i (representing, e.g., measurement errors or our incomplete understanding of the world) satisfying

$$\mathbb{E}(\varepsilon_i) = 0, \qquad \mathrm{Var}(\varepsilon_i) = \sigma^2, \qquad \mathrm{Cov}(\varepsilon_i, \varepsilon_j) = 0,\ i \neq j.$$

The model is also expressed in matrix form as

$$Y = X\beta + \varepsilon, \qquad Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}, \ X = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix}, \ \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$

where X is called the *design matrix*. When $\varepsilon \sim N(0, \sigma^2 I_n)$, where I_n is the $n \times n$ identity matrix, the model is known as the *normal linear model*.

Despite its importance, the linear model may not be a good model for many applications: e.g. when the responses are strictly positive (prices), correspond to counting data (number of a certain event), or are binary ("yes" or "no"). Generalised linear models extend the normal linear model so that the distribution of the Y_i are from an exponential dispersion family, and so that the relationship between $\mathbb{E}\, Y_i$ and the linear predictor $x_i^T\beta$ is more general than linear.

1.1.1 Exponential dispersion families

Let m be a non-degenerate[1] σ-finite measure on $\mathbb{R}^d, d \geq 1$, such that $\Theta := \{\theta \in \mathbb{R}^d : K(\theta) < \infty\}$ is open and $K''(\theta)$ exists on Θ, where $K(\theta) := \log \int_{\mathbb{R}^d} e^{\theta^\top z} m(dz), \theta \in \Theta$. Let

$$\Phi := \left\{ \phi > 0 : K(\cdot)/\phi = \log \int_{\mathbb{R}^d} e^{\cdot^\top z} m_\phi(dz) \text{ for some non-degenerate } \sigma\text{-finite measure } m_\phi \right\}.$$

Definition 1. For some K as above and known, $\mathcal{P} := \{P_{\theta,\phi} : \theta \in \Theta, \phi \in \Phi\}$ is an exponential dispersion family of distributions with natural and dispersion parameters (and spaces) θ and ϕ (and Θ and Φ) if the density f of $P_{\theta,\phi}$ with respect to some dominating measure (e.g. m_ϕ) satisfies

$$f(z; \theta, \phi) = h(z, \phi) \exp\left\{ \frac{1}{\phi} \big(\theta^\top z - K(\theta)\big) \right\}, \quad z \in \mathbb{R}^d,$$

for some function h.

From now on we take $d = 1$. By computing the cumulative generating function of $P_{\theta,\phi}$, it is easy to check that for $Z \sim P_{\theta,\phi}$, we have $\mathbb{E}\, Z = K'(\theta)$ and $\operatorname{Var} Z = \phi K''(\theta)$. Due to the latter and the non-degeneracy of m_ϕ, it follows that $K' : \theta \mapsto \mu := \mathbb{E}Z$ is invertible.

[1] I.e., m is not a point mass or Dirac delta

Thus, we may reparametrise $\mathcal{P}$ using the *mean parameter* $\mu = \mu(\theta) = K'(\theta)$ in place of θ; naturally, the *mean parameter space* is $\mathcal{M} := \{\mu(\theta) : \theta \in \Theta\}$. We also write $\theta = \theta(\mu) = (K')^{-1}(\mu)$. With this notation, $\operatorname{Var} Z = \phi K''(\theta(\mu)) =: \phi V(\mu)$, where V is called the *variance function* for this exponential dispersion family.

Example 1. The univariate normal distribution $N(\gamma, \sigma^2)$ has density

$$f(z;\gamma,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\biggl\{-\frac{(z-\gamma)^2}{2\sigma^2}\biggr\} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\biggl\{-\frac{z^2}{2\sigma^2}\biggr\} \exp\biggl\{\frac{z\gamma - \gamma^2/2}{\phi}\biggr\}$$

with respect to the Lebesgue measure. Hence,

$$\bigl\{N(\mu, \sigma^2) : \gamma \in \mathbb{R}, \sigma^2 > 0\bigr\}$$

is an exponential dispersion family with natural parameter $\theta = \gamma$ and dispersion parameter $\phi = \sigma^2$. Also, $K(\theta) = \theta^2/2$, so the mean parameter and the variance function satisfy $\mu = \theta\ (= \gamma)$ and $V(\mu) = 1$.

Example 2. The Poisson distribution $\operatorname{Poi}(\lambda)$ has density

$$f(z;\lambda) = \frac{e^{-\lambda}\lambda^z}{z!} = \frac{1}{z!} \exp\{z \log\lambda - \lambda\}.$$

with respect to the counting measure. Hence,

$$\bigl\{\operatorname{Poi}(\lambda) : \lambda > 0\bigr\}$$

is an exponential dispersion family with natural parameter $\theta = \log\lambda$ and dispersion parameter $\phi = 1$. Also, $K(\theta) = e^\theta$, so the mean parameter and the variance function satisfy $\mu = e^\theta\ (= \lambda)$ and $V(\mu) = \mu$.

Example 3. The binomial distribution $\operatorname{Bin}(m, p)$ has density

$$f(z;m;p) = \binom{m}{z} p^z (1-p)^{m-z} = \binom{m}{z} \exp\biggl\{z \log\Bigl(\frac{p}{1-p}\Bigr) + m\log(1-p)\biggr\}$$

with respect to the counting measure. Hence,

$$\biggl\{\frac{1}{m}\operatorname{Bin}(m,p) : m \in \mathbb{N}, p \in (0,1)\biggr\}$$

is an exponential dispersion family (note the rescaling) with natural parameter $\theta = \operatorname{logit} p := \log(p/(1-p))$ and dispersion parameter $\phi = 1/m$. Also, $K(\theta) = \log(e^\theta + 1)$, so the mean parameter and the variance function satisfy $\mu = \operatorname{expit}(\theta) := e^\theta/(1+e^\theta)\ (= p)$ and $V(p) = p(1-p)$.

Note that in this course we will not worry about checking that K satisfies the assumptions at the beginning of the section.

1.1.2 The generalised linear model

For some K as in the previous section, let $\{P_{\mu,\phi} : \mu \in \mathcal{M}, \phi \in \Phi\}$ be a given exponential dispersion family parametrised by the mean and dispersion parameters.

Definition 2. The *generalised linear model* assumes that

$$Y_i \overset{\text{ind.}}{\sim} P_{\mu_i,\phi_i}, \quad i = 1, \ldots, n,$$

where

- $g(\mu_i) = x_i^\top \beta$ for some vector of coefficients $\beta \in \mathbb{R}^p$, with $g : \mathcal{M} \to \mathbb{R}$ the *link function* (known); and,
- $\phi_i = a_i \phi$, with $a_1, \ldots, a_n > 0$ known (some times referred to as weights) and $\phi > 0$ possibly unknown (referred to as the dispersion parameter).

Unless otherwise stated, in generalised linear models we assume $p \leq n$, $X = (x_1, \ldots, x_n)^\top$ full rank and g strictly monotonic for identifiability, and g twice differentiable for computations. Common choices of the link function are $g(\mu) = \theta(\mu)$ (called the *canonical link*) or one such that $g(\mathcal{M}) = \mathbb{R}$ or one that results in a simple interpretation of β.

1.1.3 Estimation of β and ϕ

From Definition 2 we see that the only parameter(s) to be estimated in a generalised linear model are β and ϕ (the latter only if unknown).

If $Y = (Y_1, \ldots, Y_n)^\top$ follows a generalised linear model, the log-likelihood function is given by

$$\ell(\beta, \phi) = \ell(\beta, \phi; Y) = \sum_{i=1}^n \frac{1}{a_i \phi} \left\{ \theta\left(g^{-1}(x_i^\top \beta)\right) Y_i - K\left[\theta\left(g^{-1}(x_i^\top \beta)\right)\right] \right\} + \sum_{i=1}^n \log h(Y_i, \phi).$$

The maximum likelihood estimator for β, denoted by $\hat{\beta}$, has a very particular characterisation (within generalised linear models) under the canonical link.

Proposition 1. *Let Y follow a generalised linear model with the canonical link function. Then, the log-likelihood function is strictly concave in β. Consequently, the maximum* *likelihood estimator $\hat{\beta}$ always exists, is unique and, writing $\hat{\mu}_i = \mu(x_i^\top \hat{\beta}), i = 1, \dots, n$, it solves by the score equations*

$$\sum_{i=1}^n \frac{x_i}{a_i}(Y_i - \hat{\mu}_i) = 0.$$

Proof. Under the canonical link, we have

$$\ell(\beta) = \text{const} + \sum_{i=1}^n \frac{1}{a_i \phi}\{x_i^\top \beta y - K(x_i^\top \beta)\}.$$

Differentiating with respect to β, we have

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^n \frac{x_i}{a_i \phi}\{y_i - K'(x_i^\top \beta)\} \qquad \frac{\partial^2 \ell}{\partial \beta \partial \beta^\top} = -\sum_{i=1}^n \frac{x_i x_i^\top}{a_i \phi}\{K''(x_i^\top \beta)\}.$$

Since $K''(x_i^\top \beta) > 0$ and X has full rank, the Hessian is negative definite and thus $\ell(\beta)$ is strictly concave. Setting the gradient to zero gives the desired score equation. □

Two methods to approximate $\hat{\beta}$ are Newton–Raphson and Fisher scoring (a.k.a. iteratively reweighted least squares or IRLS algorithm); we look at them in more detail in the practicals.

When ϕ is unknown, we could estimate it using maximum likelihood estimation too, but such estimator does not necessarily enjoy analogous characterisation and properties. A simple estimator of ϕ is given instead by the *generalised Pearson statistic*, namely

$$\hat{\phi} := \frac{1}{n-p} \sum_{i=1}^n \frac{(Y_i - \hat{\mu}_i)^2}{a_i V(\hat{\mu}_i)}.$$

This is motivated by the fact that $\text{Var}(Y_i) = a_i \phi V(\mu_i)$. If ϕ is known, we set $\hat{\phi} := \phi$ for economy of notation.

We remark that the canonical link for the normal linear model, Poisson model and binomial model are the identity function, logarithmic function and logistic function respectively. For this reason, the binomial model is also commonly known as the logistic model.

1.1.4 Asymptotic guarantees

Recall that the residual sum of squares plays a central role in inferential procedures for the normal linear model. A generalisation of it for generalised linear models is the so-called deviance. Let $\tilde{\ell}$ be the log-likelihood function parametrised by (μ, ϕ).

Definition 3. The *deviance* of a generalised linear model is defined as

$$D(Y; \hat{\mu}) := 2\phi\big\{\tilde{\ell}(Y, \phi; Y) - \tilde{\ell}(\hat{\mu}, \phi; Y)\big\}.$$

Note that if $\tilde{f}$ is the density of the generalised linear model parametrised by (μ, ϕ),

$$D(Y; \hat{\mu}) = 2\phi \log \frac{\max_{\mu \in \mathbb{R}^n} \tilde{f}(Y; \mu, \phi)}{\max_{\mu \in \mathbb{R}^n : g(\mu_i) = x_i^\top \beta} \tilde{f}(Y; \mu, \phi)}.$$

Thus, $\phi^{-1} D(Y; \hat{\mu})$ corresponds to the likelihood ratio test statistic for the model at hand versus its saturated version, where the latter is the same model but with $\mu \in \mathbb{R}^n$ unrestricted.

Theorem 1 (Large sample asymptotics)**.** *In a generalised linear model, assume $n^{-1} X^\top W X \to \Sigma$ as $n \to \infty$, where $W^{-1} = \operatorname{diag}(a_i V(\mu_i) g'(\mu_i)^2)$ and Σ is non-degenerate. Then, as $n \to \infty$,*

(i) $\sqrt{n/\hat{\phi}}\left(\hat{\beta} - \beta\right) \xrightarrow{d} N\left(0, \Sigma^{-1}\right)$*; and,*

(ii) $\hat{\phi} \xrightarrow{a.s.} \phi$.

Write $\beta = (\beta_0^\top, \beta_1^\top)^\top$, where $\beta_0 \in \mathbb{R}^{p_0}$ for some $1 \leq p_0 \leq p$, and let ω_0 be the model where $\beta_1 = 0$. Let $\breve{\mu}$ be the maximum likelihood estimator for μ under ω_0. Then, as $n \to \infty$ and under ω_0,

(iii) $\hat{\phi}^{-1}\big\{D(Y; \breve{\mu}) - D(Y; \hat{\mu})\big\} \xrightarrow{d} \chi^2_{p-p_0}$.

Theorem 2 (Small dispersion asymptotics)**.** *Adopt the notation of Theorem 1. Then, as $\phi \to 0$,*

(i)

$$\frac{1}{\hat{\phi}^{1/2}}\left(\hat{\beta} - \beta\right) \xrightarrow{d} \begin{cases} N\left(0, \left(X^\top W X\right)^{-1}\right) & \text{if } \phi \text{ known,} \\ t_{n-p}\left(0, \left(X^\top W X\right)^{-1}\right) & \text{if } \phi \text{ unknown;} \end{cases}$$

(ii)

$$\frac{\hat{\phi}}{\phi} \xrightarrow{d} \frac{1}{n-p}\chi^2_{n-p}; \quad \textit{and,}$$

(iii) *under* ω_0,

$$\frac{1}{\hat{\phi}}\left\{D(Y;\breve{\mu}) - D(Y;\hat{\mu})\right\} \xrightarrow{d} \begin{cases} \chi^2_{p-p_0} & \textit{if } \phi \textit{ known,} \\ (p-p_0)F_{p-p_0,n-p} & \textit{if } \phi \textit{ unknown.} \end{cases}$$

The asymptotic statements in the above theorems can be used to construct confidence intervals and tests for β and ϕ, to compare two nested models and to test the goodness-of-fit of the generalised linear model. Small dispersion asymptotics can be applied, for instance, to normal linear models with large signal-to-noise ratio and to binomial models with large number of trials m_i. It does not directly apply for Poisson regression since the dispersion parameter in Poisson model is always 1, but it can be shown that when all the λ_i are large, the same asymptotic results are valid. We will refer to the Poisson model large count asymptotic result also as small dispersion asymptotics. See Jørgensen (1987) for more details. Lastly, note that even when the dispersion is not small, the distributional approximations provided by Theorem 2 are generally better than those from Theorem 1.

Post scriptum: although not explicitly mentioned in the statements, Proposition 1 and Theorems 1 and 2 require the usual regularity assumptions for the asymptotic normality of the maximum likelihood estimator in parametric models; see Chapter 10 of the classical Casella and Berger (2001) for the theory for i.i.d. observations.

1.1.5 Model selection and regularisation

In statistics, we often have a set of possible models and must choose between them. For example, we may consider generalised linear models with different exponential dispersion families or, more commonly, even when the family is fixed we may have many predictors and face the decision of which main effects and interactions to include in the model. Model selection aims to provide a disciplined way of choosing the best model according to the purpose we wish to use it for.

So far we have fitted models through maximum likelihood estimation, so with the purpose of explaining the data as well as possible. For this purpose, the last parts of Theorems 1 and 2 allow us to choose between two nested models. However, this does not allow for comparison of two or more not necessarily nested models.

Another purpose we may use a model for is prediction. We introduce this important topic at the heart of machine learning through an example.

Example 4. Assume we have observations $(x_1, Y_1), \dots, (x_n, Y_n)$ with $Y_i = f(x_i) + \varepsilon_i$, where $\varepsilon_1, \dots, \varepsilon_n$ are uncorrelated and satisfy $\mathbb{E}\,\varepsilon_i = 0$ and $\operatorname{Var} \varepsilon_i = \sigma^2 > 0$, and let $\hat{f}$ be an estimate of f. The *prediction* (or *generalisation* or *test*) *error* of $\hat{f}$ at a new and independent observation (x^*, Y^*) is

$$\begin{aligned}\mathbb{E}\bigl\{\bigl(Y^* - \hat{f}(x^*)\bigr)^2\bigr\} &= \mathbb{E}\bigl\{\bigl(Y^* - f(x^*)\bigr)^2\bigr\} + \bigl(f(x^*) - \mathbb{E}\hat{f}(x^*)\bigr)^2 + \mathbb{E}\bigl\{\bigl(\hat{f}(x^*) - \mathbb{E}\hat{f}(x^*)\bigr)^2\bigr\} \\ &=: \sigma^2 + \operatorname{Bias}^2\bigl(\hat{f}(x^*)\bigr) + \operatorname{Var}\bigl(\hat{f}(x^*)\bigr).\end{aligned}$$

Thus, the prediction error can be decomposed into three terms:

(i) a stochastic error term σ^2 that is caused by the random nature of the data-generating mechanism;
(ii) a squared bias term that measures how well the mean value of our prediction $\hat{f}$ approximates the true mean of the response at the new data point; and,
(iii) a variance term that measures the expected squared error due to variability in $\hat{f}$.

Of the three terms, the stochastic error is out of our control and cannot be reduced even if we know the true model. However, we have control over the second and the third terms through choosing the estimator. For instance, in the linear model (so $f(x) = x^\top \beta$) we may choose $\hat{f}(x^*) = (x^*)^\top \hat{\beta}$, where $\hat{\beta}$ is the least square estimator, in which case we have that the bias is zero and, by the Gauss–Markov theorem, $\mathrm{Var}((x^*)^\top \hat{\beta}) \leq \mathrm{Var}((x^*)^\top \tilde{\beta})$ for any other linear and unbiased estimator $\tilde{\beta}$. Therefore, if the true model includes all the covariates, $\hat{\beta}$ is the linear and unbiased estimator inducing the smallest prediction error. However, a key observation of modern statistics is that it may well be that a slightly biased estimator has smaller variance so that its prediction error is smaller than that of $\hat{\beta}$. A possible way to achieve this is by fitting the least squares estimator to a model with a subset of the covariates, so that some bias is introduced and the variance is decreased because there are less parameters to estimate. Of course, if we choose a very small subset, the bias may be too large and the prediction error may increase. Thus, we see that there is a trade-off between bias and variance. Lastly, we note that if the true model only includes $p_0 < p$ covariates and if the columns of the design matrix X are mutually orthogonal then $\mathrm{Var}((x^*)^\top \hat{\beta}) = \sum_{i=1}^p (x_i^*)^2 / \|X_{\cdot,i}\|^2$, so even if we wish to use the least squares estimator, we would like to select a submodel before fitting it.

Crucially, the insights presented in the example are not specific to the linear model or to generalised linear models. They even apply to the general situation when we do not know the data generating mechanism and this is much more complex than any model we propose: by increasing the complexity of the latter, we can better pick up the structure of the former hence decreasing the bias but, at the same time, more complex models are also more likely to fit into the noise, leading to a higher variance term; we see again the bias-variance trade-off.

Lastly, we mention that when $p > n$, i.e. when the number of collected covariates is larger than the number of observations, we will also have to select between submodels. In particular, between those whose number of covariates is less than or equal to n.

Consequently, how to select between submodels (or models in general) is a key question in statistics and machine learning. In the next two subsections we consider the general setting in which we wish to select a model from a collection $\mathbb{M} := \{\mathcal{M}_1, \ldots, \mathcal{M}_K\}$, where $\mathcal{M}_k := \{f_k(\cdot;\theta_k) : \theta_k \in \Theta_k\}$ and θ_k is identifiable. Let the data be $Z := (Z_1, \ldots, Z_n) \sim f_0$ (not necessarily in $\mathbb{M}$) and write $\hat{\theta}_k$ for the maximum likelihood estimator for θ_k in model $\mathcal{M}_k$ calculated from Z.

1.1.5.1 Information Criteria

Definition 4. The Akaike and the Bayesian Information Criteria (AIC and BIC, respectively) for $\mathcal{M}_k$ are defined as

$$\mathrm{AIC}(\mathcal{M}_k) := -2\log f_k(Z;\hat{\theta}_k) + 2\,\mathrm{dim}\Theta_k, \quad \mathrm{BIC}(\mathcal{M}_k) := -2\log f_k(Z;\hat{\theta}_k) + \log n\,\mathrm{dim}\Theta_k.$$

Then,

$$\mathrm{AIC}(\mathbb{M}) := \arg\min_k \mathrm{AIC}(\mathcal{M}_k) \qquad \text{and} \qquad \mathrm{BIC}(\mathbb{M}) := \arg\min_k \mathrm{BIC}(\mathcal{M}_k).$$

A common misconception is that AIC or BIC can only be applied to a sequence of nested models. However, AIC was derived by Akaike (1973) as an asymptotic approximation of Kullback–Leibler divergence between the model of interest and the truth, while BIC was derived by Schwarz (1978) as a large-sample approximation to Bayesian maximum *a posteriori* model selection given the data. Therefore, both can be used to compare different models, as long as we *compute the likelihood based on the same data.* Nonetheless, applying AIC or BIC on the same set of models does make the model selection more stable.

Compared to AIC, BIC imposes a harsher penalty for each additional parameter. Consequently, BIC tends to choose simpler models. If $f_0 \in \mathbb{M}$ it can be shown that BIC will select the true model asymptotically as $n \to \infty$ under suitable conditions, whilst AIC is often too conservative in the sense that an unnecessarily large model is chosen, even asymptotically. Thus, BIC is useful in finding the best *explanatory* model, whereas AIC is more useful in finding the best *predictive* model (even if the true data generating mechanism is not included in $\mathbb{M}$).

1.1.5.2 Cross-validation

If we want to choose a model with optimal predictive performance, we would ideally like to have a separate test sample. In the absence of a test sample, we can withhold a portion of the original data for testing and train on the remaining data. This can be done repeatedly by holding aside a different subset of data each time. Such a technique is known as *cross-validation.* In a V-fold cross-validation, the original data is partitioned into V subsets of roughly equal sizes. Each time, we use $V-1$ subsets to estimate the model parameter and test the fitted model on the remaining subset. This is repeated V times (folds) and the average validation error from the V folds is reported as the cross-validation error.

For concreteness, we illustrate below how V-fold cross-validation error can be computed in a generalised linear model $\mathcal{M}_k$: let $Z_i = (x_i, Y_i), i = 1, \ldots, n$; then,

1. partition (possibly randomly) the data into V subsets of almost equal size $\{(x_i, Y_i) : i \in I_v\}, \quad v = 1, \ldots V$, where $\sqcup_{v=1}^{V} I_v = \{1, \ldots, n\}$;
2. for each $v = 1, \ldots, V$
   (a) use $\{(x_i, Y_i) : i \notin I_v\}$ as training data to estimate parameter $\hat{\beta}_k^{(v)}$; and
   (b) evaluate the mean squared error on the remaining data
   $$\mathrm{err}_k^{(v)} := \frac{1}{|I_v|} \sum_{i \in I_v} D\left(Y_i; \mu\left(x_i^\top \hat{\beta}_k^{(v)}\right)\right);$$
3. lastly, aggregate over all V folds to obtain an overall cross-validation error
   $$\mathrm{err}_{\mathrm{CV}}(\mathcal{M}_k) := \frac{1}{V} \sum_{v=1}^{V} \mathrm{err}_k^{(v)}.$$

Then, if we wish to choose a model from a collection $\mathbb{M}$, we perform cross-validation for every model and select the one with the smallest cross-validation error, i.e., we choose it according to $V\text{-CV}(\mathbb{M}) := \arg\min_k \mathrm{err}_{\mathrm{CV}}(\mathcal{M}_k)$.

Typical choices of V include $V = 5, 10, n$. The last one is also known as leave-one-out cross-validation. Clearly, V-fold cross-validation (and cross-validation in general) is constructed for prediction purposes. Under certain conditions, it can be shown that leave-one-out cross-validation is asymptotically equivalent to AIC, so it confirms that AIC is useful in finding predictive models and also implies that leave-one-out cross-validation tends to overfit. Indeed, Shao (1993) showed that leave-one-out cross-validation can be inconsistent when the true data generating mechanism is included in the set of models considered.

1.1.5.3 Best subset, forward and backward selection

Please, see practical 2.

1.1.5.4 Regularisation

One of the most successful way to trade-off bias and variance in regression settings is through regularisation. Many times it is also useful in model selection.

The general idea is that we wish to minimise an objective function of the data and of the model that comprises two terms: i) a data-fidelity term, and ii) a regularisation or penalisation term. By minimising the first we calibrate the model parameters to the data, whilst by minimising the latter we achieve certain properties in the model such as regularity or sparsity.

For generalised linear models, the objective function to minimise usually comprises i) the negative log-likelihood or the deviance, and ii) a constant times the q^{th} power of the L_q-norm of the vector of coefficients β, i.e. $\lambda \sum_{j=1}^{p} |\beta_j|^q$ for some $\lambda, q \geq 0$. The constant λ is called the regularisation or smoothing parameter and its value determines how much importance each term gets: the larger it is, the more importance the second term has so the more regularisation will be performed, and the more bias and the less variance will be induced; the opposite occurs as λ is decreased. Note that by penalising more complex models, we generally prevent overfitting and obtain better generalisation. λ must be calibrated from the data: generally, this is achieved by proposing a set of values $\{\lambda_1, \ldots, \lambda_m\}$ and applying cross-validation to choose from them. Note that when $q = 0$, the regularisation term equals $\lambda \text{dim}(\beta)$ where we take the convention that the dimension of β is the number of non-zero coordinates. You may have realised that we have already seen two examples of regularised generalised linear models with i) the negative log-likelihood and ii) $q = 0$: best subset selection (from models with $\text{dim}(\beta) \leq n$) using AIC and BIC; for these we had $\lambda = 2$ and $\lambda = \log n$, respectively. As discussed in Section 1.1.5.3, these L_0-norm (and non-convex) regularisation procedures can be very expensive. To alleviate the cost, q is usually chosen such that $q \geq 1$ so that the penalisation term is convex (and strictly convex if $q > 1$). The most common choices are $q = 1$ and $q = 2$. We present these in the framework of linear models.

Prior to doing so we make a change to the ongoing notation: note that for generalised linear models, the negative log-likelihood and the deviance depend on β through the mean parameter μ only; thus, in the likely case that an intercept is included in the model, we do not wish to include it in the penalisation term so that shifts in the responses can be accounted for well; therefore we differentiate the intercept from the rest of the coefficients by denoting it α and, with some abuse of notation, denote by β, X and p

the three quantities after removing the intercept from them (so, e.g., $p-1$ becomes p). Consequently, we may centre the columns of X without loss of generality. Additionally, note that when $q > 0$, the penalisation term is meaningful only if the columns of X have the same scale. Hence, throughout this section we assume they have been centred and rescaled to have unit variance.

Definition 5. For any $\lambda \geq 0$, the Lasso (least absolute shrinkage and selection operator) estimator and the ridge regularisation estimator are defined, respectively, as

$$(\hat{\alpha}^{(L)}, \hat{\beta}_\lambda^{(L)}) := \underset{(\alpha,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\arg\min} \left\{ \|Y - \alpha\mathbf{1}_n - X\beta\|_2^2 + \lambda\|\beta\|_1 \right\}$$

and

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

Hence, the Lasso and ridge estimators are, respectively, the minimisers of an ℓ_1 and ℓ_2 penalised residual sum of squares. Note that for their definition does not require the responses to follow a linear model (although one generally assumes it to show theoretical guarantees).

Lasso has no closed-form solution in general, whilst

$$\hat{\alpha}^{(R)} = \frac{1}{n}\sum_{i=1}^n Y_i \qquad \text{and} \qquad \hat{\beta}_\lambda^{(R)} = \left(X^\top X + \lambda I_p\right)^{-1} X^\top \left(Y - \hat{\alpha}^{(R)}\mathbf{1}_n\right).$$

It follows that the ridge estimator always exists and is unique, even when $X^\top X$ is not full-rank such as when $p \geq n$. Lasso typically exists and is unique but it does not necessarily do.

We can get more intuition about their behaviour. Notice that objective functions minimised by Lasso and ridge are the Lagrangian formulations of the constrained optimisation problems

$$\min_{(\alpha,\beta)\in\mathbb{R}\times\mathbb{R}^p} \|Y - \alpha\mathbf{1}_n - X\beta\|_2^2 \qquad \text{subject to } \|\beta\|_1 \leq t \text{ or } \|\beta\|_2^2 \leq t,$$

respectively, where $t \geq 0$ is in one-to-one correspondence with λ. See Figure 1.1 for an illustration of the Lasso and ridge estimators for $p = 2 < n$. It becomes clear that since the contours of the ℓ_1 norm, i.e. the set of those β with the same ℓ_1 norm, have sharp corners, Lasso zeros out many coordinates of the estimator and encourages *sparse* solutions. And since the contours of the ℓ_2 norm are smooth, the probability that ridge zeros out any coordinates of the estimator may well be zero. Indeed, the Lasso estimator automatically performs model selection.

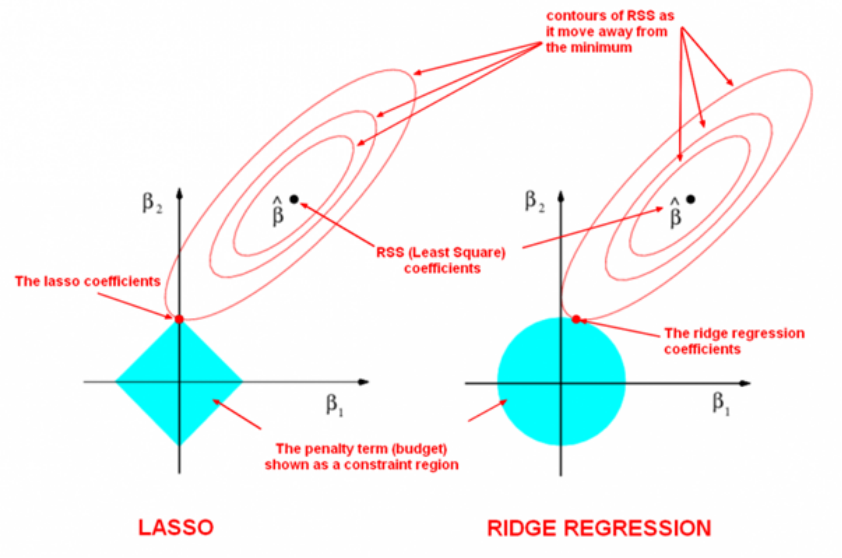

contours of RSS as it move away from the minimum
β_2
$\hat{\beta}$
β_2
$\hat{\beta}$
RSS (Least Square) coefficients
The lasso coefficients
The ridge regression coefficients
β_1
β_1
The penalty term (budget) shown as a constraint region
LASSO
RIDGE REGRESSION

Figure 1.1: Visual illustration of the Lasso and ridge estimators for $p = 2 < n$ from their Lagrangian formulation

Principal component analysis (PCA) .

Let $p < n$ herein. By the singular value decomposition of X we can represent it as

$$X = UDV^\top,$$

where $U \in \mathbb{R}^{n\times p}$ is such that $U^\top U = I_p$, $D \in \mathbb{R}^{p\times p}$ is a diagonal matrix with $D_{1,1} \geq D_{2,2} \geq \cdots \geq D_{p,p} \geq 0$ and $V \in \mathbb{R}^{p\times p}$ is a unitary matrix. Then, since the columns of X are centred, the sample variance of Xw for some $w \in \mathbb{R}^p$ is

$$\frac{1}{n}w^\top X^\top Xw = \frac{1}{n}w^\top VD^2V^\top w = \frac{1}{n}(V^\top w)^\top D^2(V^\top w).$$

If w is the i^{th} column of V, i.e. $w = V_{\cdot,i}$, the right hand side equals $D^2_{i,i}/n$ and $V_{\cdot,i}$ are the coefficients of the (unitary) linear combination of the columns of X with i^{th} largest sample variance. As a result and due to $XV = UD$, $U_{\cdot,i}$ is the direction with the i^{th} maximum variation within the span of X, or the so-called i^{th} principal component of X. This whole analysis is called *principal component analysis*. Note that it is common to keep the first j principal components, where $j \leq p$ is the first natural number for which

$$\frac{\sum_{i=1}^{j} D^2_{i,i}}{\sum_{i=1}^{p} D^2_{i,i}} \geq 80\% \text{ or } 90\% \text{ or } 95\% \text{ typically.}$$

Comparison between OLS, Ridge, PCA and Lasso Let $\operatorname{diag}(d_i)$ be the diagonal matrix with diagonal entries $d_1, \ldots, d_p$. Using the singular value decomposition of X, it is simple to check that

$$\hat{\beta}_\lambda^{(R)} = V \operatorname{diag}\left(\frac{D_{ii}}{D_{ii}^2 + \lambda}\right) U^\top Y \qquad \text{and} \qquad \hat{\beta} = \hat{\beta}_0^{(R)}.$$

so

$$\hat{Y}_\lambda^{(R)} := X\hat{\beta}_\lambda^{(R)} = \sum_{i=1}^{p} \frac{D_{ii}^2}{D_{ii}^2 + \lambda} U_{\cdot,i} \langle U_{\cdot,i}, Y\rangle \qquad \text{and} \qquad \hat{Y} := X\hat{\beta} = \hat{Y}_0^{(R)}.$$

For PCA$_j$, i.e. PCA when only the first j principal components have been kept, we can regress Y on $U_{\cdot,1}, \ldots, U_{\cdot,j}$ and, since the principal components are orthonormal, the resulting projection is

$$\hat{Y}^{(\mathrm{PCA}_j)} = \sum_{i=1}^{j} U_{\cdot,i} \langle U_{\cdot,i}, Y\rangle = U \operatorname{diag}\left(\mathbf{1}_{i \le j}\right) U^\top Y = XVD^{-1} \operatorname{diag}\left(\mathbf{1}_{i \le j}\right) U^\top Y.$$

Consequently,

$$\hat{\beta}^{(\mathrm{PCA}_j)} = V \operatorname{diag}\left(D_{ii}^{-1}\mathbf{1}_{i \le j}\right) U^\top Y$$

and, analogously to above, $\hat{Y}^{(\mathrm{PCA}_j)} = X\hat{\beta}^{(\mathrm{PCA}_j)}$. We note in passing that

$$\hat{\beta}^{(\mathrm{PCA}_j)} = \lim_{\substack{\lambda_1, \ldots, \lambda_j \to 0 \\ \lambda_{j+1}, \ldots, \lambda_p \to \infty}} \operatorname*{arg\,min}_{(\alpha,\beta) \in \mathbb{R} \times \mathbb{R}^p} \left\{ \|Y - \alpha\mathbf{1}_n - X\beta\|_2^2 + \sum_{i=1}^{p} \lambda_i |\beta_i|^2 \right\},$$

so $\hat{\beta}^{(\mathrm{PCA}_j)}$ is the limit of a regularised procedure (in particular, a generalised ridge estimator).

As the expressions for $\hat{Y}_\lambda^{(R)}, \hat{Y}$ and $\hat{Y}^{(\mathrm{PCA}_j)}$ show, the ordinary least squares estimator (OLS) projects Y onto the span of $\operatorname{span}\left(\{U_{\cdot,1}, \ldots, U_{\cdot,p}\}\right) = \operatorname{span}(X)$ (as known from the usual theory of the linear model), the PCA$_j$ estimator projects Y onto the span of $\operatorname{span}\left(\{U_{\cdot,1}, \ldots, U_{\cdot,j}\}\right)$, and the ridge estimator projects Y onto the span of $\operatorname{span}\left(\{U_{\cdot,1}, \ldots, U_{\cdot,p}\}\right)$ shrinking each coordinate according to the respective sample variance. We also infer from the expressions for $\hat{Y}_\lambda^{(R)}$ and $\hat{Y}^{(\mathrm{PCA}_j)}$ that for them to work well the responses should vary most in the directions of highest variability of the covariates. This is a reasonable assumption, since covariates are generally chosen to explain variability in the responses, but in some data-generating mechanisms it may fail to hold, in which case these estimators are unlikely to do a good job. In a linear model, this is the case and, as anticipated at the beginning of the section, we have the following result.

Theorem 3. *Assume that the data follows a linear model so, in particular, $Y = \alpha\mathbf{1}_n + X\beta + \varepsilon$. Then, under some assumptions including that λ is small enough, the difference of covariance matrices*

$$\mathbb{E}\left[\left(\begin{pmatrix}\hat{\alpha}\\ \hat{\beta}\end{pmatrix} - \begin{pmatrix}\alpha\\ \beta\end{pmatrix}\right)\right]\left[\left(\begin{pmatrix}\hat{\alpha}\\ \hat{\beta}\end{pmatrix} - \begin{pmatrix}\alpha\\ \beta\end{pmatrix}\right)\right]^\top - \mathbb{E}\left[\left(\begin{pmatrix}\hat{\alpha}^{(R)}_\lambda\\ \hat{\beta}^{(R)}_\lambda\end{pmatrix} - \begin{pmatrix}\alpha\\ \beta\end{pmatrix}\right)\right]\left[\left(\begin{pmatrix}\hat{\alpha}\\ \hat{\beta}\end{pmatrix} - \begin{pmatrix}\alpha\\ \beta\end{pmatrix}\right)\right]^\top$$

is positive definite.

It follows that the prediction error of the ridge estimator is smaller than that of the OLS estimator if λ is small enough.

Assume now on top of the linear model that $\beta = (\beta_0^\top, \beta_1^\top)^\top$ for some $\beta_0 \in \mathbb{R}^{p_0}, 1 \leq p_0 \leq p$, and $\beta_1 \equiv 0$. Let $\hat{\beta}^{(O)}$ be the Oracle estimator, i.e. the OLS estimator applied knowing p_0. Then, under some assumptions including that λ is small enough, the prediction error of the Lasso estimator is comparable to that of the Oracle estimator.

1.2 Generalising generalised linear models

Recall that if $Y \sim P_{\mu,\phi}$, where $P_{\mu,\phi}$ is from an exponential dispersion family, then $\operatorname{Var} Y = \phi V(\mu)$. This is very useful in applications where we expect a certain mean-variance relationship, so that we can choose the model appropriately. However, it can also be very limiting, particularly so when ϕ is not a free parameter as in the Poisson and binomial families: the variability of the data may well be larger (smaller) than that predicted by the model. This phenomenon, known as over(under)dispersion, is common and not specific to generalised linear models; note, however, that in some generalised linear models like in the normal linear model, ϕ is free and we can fit the observed data up to the first two moments.

Over(under)dispersion suggests that our modelling assumptions do not hold and, thus, inferential procedures following from our model may not be valid (e.g. over(under)optimistic confidence regions). A common reason is that we missed to model some of the heterogeneity in the data. This may be the result of missing important covariates, of certain dependencies between observations or of wrong distributional assumptions, among other reasons. Note that in practice, overdispersion is much more common than underdispersion—e.g., given our limited understanding of the world, we generally fail to include some predictors and this will result in responses that, for the same set of covariates, vary more than anticipated by the model. In the rest of the section we introduce some generalisations of generalised linear models to account for over- and underdispersion.

1.2.1 Negative binomial model

The first model we consider arises directly from generalised linear models and allows to handle overdispersion in the Poisson regression model. As shown in the example sheet, the family of negative binomial distributions $\{\mathrm{NB}(r,p) : r \in \mathbb{N}, p \in (0,1)\}$ with r is known is an exponential dispersion family with

$$\mu = r\frac{p}{1-p}, \quad \phi = 1 \quad \text{and} \quad V(\mu) = \mu + \frac{\mu^2}{r}.$$

We reparametrise and generalise this family and, with some abuse of notation, say that $Y \sim \mathrm{NB}(\mu, \tau), \mu, \tau > 0$, if the probability mass function of Y is

$$f(u; \mu, \tau) = \frac{\Gamma(y + 1/\tau)}{\Gamma(y+1)\Gamma(1/\tau)} \left(\frac{1/\tau}{1/\tau + \mu}\right)^{1/\tau} \left(\frac{\mu}{1/\tau + \mu}\right)^{y}.$$

It follows that $\mathbb{E}\, Y = \mu$ and $\operatorname{Var} Y = \mu + \tau\mu^2$.

Definition 6. The negative binomial model assumes $Y_i \overset{\text{ind.}}{\sim} \mathrm{NB}(\mu_i, \tau)$ where $g(\mu_i) = x_i^\top \beta$ with $g = \log$ generally, so that $\operatorname{Var} Y_i = \mu_i + \tau\mu_i^2$ for $\mu_i = \exp\left(x_i^\top \beta\right)$. The parameter τ is called the overdispersion parameter.

In general, τ is unknown, so the negative binomial model is not a generalised linear model, but it is a generalisation of Poisson regression that can handle overdispersion. We may generalise it further by letting the overdispersion parameter to change with the observations.

Joint estimation of (β, τ) is required. The log-likelihood equation can be easily obtained from the above probability mass function and we may estimate the parameters by their maximum likelihood estimators $(\hat{\beta}, \hat{\tau})$. Assuming $\tau > 0$, and under certain regularity conditions, it can be shown that $\sqrt{n}(\hat{\beta} - \beta, \hat{\tau} - \tau)$ is asymptotically normally distributed with zero mean and covariance matrix given by the inverse of the Fisher information matrix.

It is natural to test H_0 : Poisson model versus H_1 : negative binomial model. We can do so using the usual likelihood ratio test statistic. However, the Poisson regression model sits at the boundary of all possible negative binomial models with $\tau > 0$ (informally, the former corresponds to $\tau = 0$). Thus, the resulting hypothesis test may not fit into the classical Neyman–Pearson framework. Indeed, if ℓ_0 and ℓ_1 are the respective log-likelihoods[2], Lawless (1987) showed that, under H_0,

$$\ell_0(\hat{\beta}; Y) - \ell_1(\hat{\beta}, \hat{\tau}; Y) \xrightarrow{\mathrm{d}} \frac{1}{2}\delta_0 + \frac{1}{2}\chi^2_1 \qquad \text{as } n \to \infty.$$

1.2.2 Zero-inflated models

One possible form of overdispersion arises when many more zero values are observed for the response variable than expected from the model. This may be called zero-inflated data. One possible interpretation is that the data comes from two different populations, for one of which the model is valid while for the other the data is always zero (or the event is so rare that it is not registered in the observation window). If this is indeed the case, generalised linear models or the negative binomial model may give a poor fit. To handle this situation we consider Zero-Inflated Models (ZIMs), where a non-degenerate model is used for the “active” population.

Definition 7. Let $\mathcal{M}_1, \dots, \mathcal{M}_n$ be given non-degenerate models. A Zero-Inflated Model assumes that $Y_1, \dots, Y_n$ are independently distributed with

$$Y_i \sim \begin{cases} \delta_0 & \text{with probability } p_i, \\ \mathcal{M}_i & \text{with probability } 1 - p_i, \end{cases}$$

where $h(p_i) = x_i^\top \gamma$ for some $h : (0,1) \to \mathbb{R}$ known and γ unknown. In particular, a Zero-Inflated Generalised Linear Model (ZIGLM) assumes $\mathcal{M}_i = P_{\mu_i, \phi_i}$ with $g(\mu_i) = x_i^\top \beta$ and $\phi_i = a_i \phi$ as usual in a GLM. The definition of the Zero-Inflated Negative Binomial Model follows analogously.

Herein we focus on ZIGLMs. In general, $h = \text{logit}$ and $\dim(\beta), \dim(\gamma) < p$, i.e. the covariates that affect the probability of belonging to one of the two distributions may or may not be the same that affect the mean of the GLM. A special case is when the probability $p_i = p$ is constant, so when only an intercept is included to model the zero probabilities.

[2] Typo: this result applies to log-likelihoods and not deviances, since the likelihood-ratio test statistic is not the difference of the deviances herein (the respective saturated models may be different).

The parameters β, γ may be estimated by maximum likelihood estimation using Newton-type optimisation algorithms. However, since the log-likelihood in these models can be quite involved, the algorithms may not converge. Note that we could have formulated the definition of zero-inflated models as $Y_i = 0$ if $Z_i = 0$ and $Y_i \sim \mathcal{M}_i$ if $Z_i = 1$, where $Z_i \overset{\text{ind.}}{\sim} \text{Bin}(1, p_i)$ are latent variables. An algorithm to maximise the likelihood of models with unobserved variables that is more costly than Newton-type ones but that, under mild conditions, it is guaranteed to converge to a local maximum (cf. Wu (1983))is the expectation-maximisation (EM) algorithm. We now describe it in more generality; for its application to a ZIGLM see the example sheets.

Let Y be a vector of observed variables and Z a vector of latent variables and θ a vector of unknown parameters of interest. Let $\ell(\theta; Y)$ be the log-likelihood of the model for Y and $\ell(\theta; Y, Z)$ be the log-likelihood of the (so-called augmented) model for (Y, Z). The EM algorithm searches for a $\hat{\theta}$ that maximises $\ell(\theta; Y, Z)$ as follows:

1. Initialise the parameter to $\hat{\theta}^{(0)}$;
2. *Expectation step*: at the $k = 1, 2, \ldots$ step, compute the function
$$Q(\cdot, \hat{\theta}^{(k)}) = \mathbb{E}_{Z|Y,\hat{\theta}^{(k)}}\big\{\ell_0(\cdot; Y, Z)\big\}; \tag{1.1}$$
3. *Maximisation step*: set
$$\hat{\theta}^{(k+1)} = \arg\max_{\theta} Q(\theta, \hat{\theta}^{(k)});$$
4. Iterate steps 2 and 3 until numerical convergence.

1.2.3 Generalised linear mixed effect models

We continue explicitly modelling heterogeneity, but now arising as a result of dependencies in the data. Indeed, some data can naturally be split into groups of observations that are dependent within the group and independent between different groups. For instance, it is common to collect repeated measurements for the same subject under different conditions and/or at different moments. An important type of study assuming collection of data is this way is a longitudinal study. This is a study in which we collect data for different subjects and, for each, we do so at different points in time. It is sensible to assume that the observations of each subject will be dependent and that, perhaps, observations of different subjects are independent. The presence of such dependencies can give rise to overdispersion.

Generalised Linear Mixed effect Models are a popular way of modelling such heterogeneity in the data. In them, we assume that the data consists of triplets $(x_{ij}, Y_{ij}, z_{ij}), i = 1, \ldots, I, j = 1, \ldots, J_i$, satisfying that, for $n = \sum_{i=1}^{I} J_i$, $X = (x_{11}, \ldots, x_{IJ_I})^\top \in \mathbb{R}^{n \times p}$ and $Z = (z_{11}, \ldots, z_{IJ_I})^\top \in \mathbb{R}^{n \times q}$ are rank $p, q \leq n$ matrices of covariates.

Definition 8. A Generalised Linear Mixed effect Model assumes

$$Y_{ij} \mid u_i \overset{\text{ind.}}{\sim} P_{\mu_{ij}, \phi_{ij}}, i = 1, \ldots, I, j = 1, \ldots, J_i,$$

where $g(\mu_{ij}) = x_{ij}^\top \beta + z_{ij}^\top u_i$ for β an unknown vector of *fixed effects* and $u_i \sim P(\alpha)$, for some distribution P parametrised by an unknown parameter $\alpha \in \mathbb{R}^r$, is a vector of *random effects*. We may say that the distribution of the Y_{ij} mixes its underlying exponential dispersion distribution with the distribution P.

It should be clear form the definition that we model the change in the mean between subjects through realisations of a random variable. This naturally models the aforementioned heterogeneity, but it can also model heterogeneity arising from, e.g., missing important covariates or measurement errors in the covariates.

The fact that the number of observations per subject J_i changes with i allows us to model, for example, longitudinal studies in which a subject does not finish the study. However, it is common to assume that $J_i = J$ for some $J \in \mathbb{N}$ for simplicity. It is also common to assume that the u_i's are independent and that $u_i \sim N(0, \Sigma)$ for some non-degenerate covariance matrix Σ. Examples of models assuming the last two properties of the random effects are the normal linear mixed model (more details below) and the logistic- and Poisson-normal models, in which the canonical link is used.

A particularly common instance of Generalised Linear Mixed effect Models is the random intercept model, in which the z_{ij} have dimension 1 and are equal to 1 for all i, j. For $J = 1$ we have already seen an example (see example sheet): the negative binomial model, which mixes a Poisson distributions with a log-gamma distribution and is expressed in a marginal formulation. Due to $J = 1$, it is mainly used to model heterogeneity of the type of missing important covariates.

The parameters α, β can be estimated by maximum likelihood estimation. However, this is not straightforward in general: if $Y = (Y_1, \ldots, Y_J)^\top$ denotes a generic set of observations for a given subject, its marginal likelihood is given by

$$L(\alpha, \beta; Y) = \int f(Y \mid u; \beta) P(u; \alpha) du;$$

thus, there is no closed form in general and it must be approximated by numerical methods and then maximised.

Note that if we wish to predict the response for a given set of covariates we will need to anticipate the value of the random effects. However, this is unknown. We estimate it using Bayes theorem: recall that, if $Y_i = (Y_{i1}, \dots, Y_{iJ})^\top$,

$$f(u_i \mid Y_i, \alpha, \beta) \propto f(Y_i \mid u_i, \alpha, \beta) f(u_i \mid \alpha);$$

hence, we can estimate u_i by substituting α, β by their approximate maximum likelihood estimators $\hat{\alpha}, \hat{\beta}$ and maximising the right hand side with respect to u_i, resulting in the MAP (maximum a posteriori) estimator $\tilde{u}_i$.

Regarding testing, we can use the usual likelihood-ratio test statistic for nested generalised linear mixed effect models. Yet, if we wish to test H_0 : GLM vs. H_1 : GLMM then, often, we choose the alternative model to generalise the null and H_0 will be at the boundary of the parameter space of the GLMM. E.g., in a random-intercept model with $u_i \overset{\text{ind.}}{\sim} N(0, \sigma^2), \sigma^2 > 0$. Generally, if the GLM and the GLMM only differ by such a parameter, the limiting distribution for the likelihood-ratio test statistic under the null model will be $\frac{1}{2}\delta_0 + \frac{1}{2}\chi_1^2$, just as when testing the Poisson model versus the negative binomial model.

Note that the limiting results we have been using so far to perform inference are only approximations generally. For small sample sizes or in complicated models like GLMMs, these approximations may be poor. Furthermore, we may not know the limiting distribution of the statistic of interest. A better alternative in practice is parametric bootstrap. We describe it in a more general setting.

Parametric bootstrap Let $\{P_\theta : \theta \in \Theta\}$ be a family of distributions and $Y \sim P_{\theta_0}$ for some θ_0 unknown. Let $\psi = \psi(\theta)$ be a functional of interest (e.g., $\psi =$ *identity* or $\psi =$ *deviance*) and let $\hat{\psi} = \psi(\hat{\theta})$ be the "plug-in" MLE for $\psi_0 = \psi(\theta_0)$, where $\hat{\theta}$ is the MLE for θ_0 computed from data Y. In particular $\hat{\theta} = \hat{\theta}(Y)$, so the distribution of $\hat{\psi}$ is determined by that of Y. The latter is unknown (through θ_0) but can be estimated by $P_{\hat{\theta}}$ which, in turn, can be used to estimate the former distribution:

1. sample $Y^{(b)} \overset{\text{ind.}}{\sim} P_{\hat{\theta}}$, $b = 1, \dots, B$, for some $B \in \mathbb{N}$, and for each compute the maximum likelihood estimator $\hat{\theta}^{(b)}$ and $\hat{\psi}^{(b)} = \psi(\hat{\theta}^{(b)})$; then,

2. approximate the law of $\hat{\psi}$ by the empirical distribution $\mathbb{P}^{(B)} := \frac{1}{B}\sum_{b=1}^{B} \delta_{\hat{\psi}^{(b)}}$.

Approximate confidence intervals and hypothesis tests for[3] $\hat{\psi}$ can be computed using $\mathbb{P}^{(B)}$ appropriately: assume, without loss of generality, that $\hat{\psi}^{(b)}$ are in increasing order and $B = 100$; then, e.g., if $\psi : \Theta \to \mathbb{R}$, $[\hat{\psi}^{(3)}, \hat{\psi}^{(97)}]$ is a 95% confidence interval for ψ_0, and, in a random-intercept model with $u_i \overset{\text{ind.}}{\sim} N(0, \sigma^2), \sigma^2 > 0$, taking $\theta = (\sigma^2, \beta)$ and $\psi(\theta) = \sigma^2$, we do not reject $H_0 : \sigma^2 = 0$ against the alternative $H_1 : \sigma^2 > 0$ with significance level 0.05 if $\hat{\sigma}^2 \leq \hat{\psi}^{(95)}$, where the parameter $\hat{\theta}$ used in the distribution of the bootstrap samples is either $(0, \hat{\beta})$, where $(\hat{\sigma}^2, \hat{\beta})$ is the MLE for the alternative model, or the MLE in the model of the null-hypothesis (under the null-hypothesis, both will give the same results as the sample size tends to infinity), whilst the $\hat{\psi}^{(b)}$ must of course be the MLEs for the alternative model. For more details, see the classic textbook by Efron and Tibshirani (1994).

Linear mixed effect models As mentioned when overdispersion was introduced, overdispersion is not a phenomenon present in linear models. Nonetheless, the normal instance of GLMMs is of significant importance, as it models the heterogeneity mentioned above.

The *linear mixed effect model* assumes that

$$Y = X\beta + \tilde{Z}u + \epsilon,$$

where

- β is an unknown vector of fixed effects (parameters),
- $\tilde{Z}$ is the block-diagonal matrix with i^{th} block given by $(z_{i1}, \ldots, z_{iJ_i})^\top$
- $u = (u_1^\top, \ldots, u_I^\top)^\top$ with $u_i \overset{\text{ind.}}{\sim} N(0, \Sigma)$ are the unknown random effects, and
- $\epsilon \sim N(0, \sigma^2 I_n)$ and is independent of u.

Alternatively, it is possible to define the model through the conditional distribution of the response vector (this is also known as *hierarchical formulation*):

$$Y \mid u \sim N(X\beta + \tilde{Z}u, \sigma^2 I_n), \qquad u = (u_1^\top, \ldots, u_I^\top)^\top, \quad u_i \overset{\text{ind.}}{\sim} N(0, \Sigma).$$

[3] Typo: inference is for $\hat{\psi}$ than ψ_0

In particular, $Y \sim N(X\beta, V)$, where $V = \tilde{Z}\Sigma\tilde{Z}^\top + \sigma^2 I_n$ (this is called *marginal formulation* of the model). Given that we know the likelihood of this marginal formulation in closed-form, for this model we can say more about estimation and inference for the parameters than in general GLMMs: the log-likelihood of the model is

$$\ell(\beta, \sigma^2, \Sigma; Y) = \text{const} - \frac{1}{2}\log\det(V) - \frac{1}{2}(Y - X\beta)^\top V^{-1}(Y - X\beta);$$

the maximiser for β is

$$\hat{\beta} = \hat{\beta}(V) = (X^\top V^{-1} X)^{-1} X^\top V^{-1} Y,$$

and we can substitute it the penultimate display to obtain the maximum likelihood estimators for σ^2, Σ by maximising the resulting expression with respect to σ^2, Σ; we can then obtain the maximum likelihood estimator for β by introducing those for σ^2, Σ into $\hat{\beta}(V)$.

However, the MLE estimator for (σ^2, Σ) has a couple of drawbacks. First, it is biased (as in the normal linear model) and the error on the random effects covariance may be large. Second, the covariance being constrained to be positive definite, we may have numerical instability when the maximum of the likelihood corresponds to negative definite matrices and the optimum is reached on the boundary of the permissible domain.

To bypass these problems, Corbeil and Searle (1976) proposed to use a *restricted maximum likelihood* (REML) approach. This consists in estimating the covariance matrix V in a way not dependent on β and then introduce it into $\hat{\beta}$ as above. To estimate V, we consider linear combinations a_k of the observations such that $a_k^\top X = 0$. With this in mind, let $a_1, \ldots, a_{n-p}$ be an orthonormal basis spanning the orthogonal complement of the column space of X and write $A = (a_1, \ldots, a_{n-p})$. Then we have $W := A^\top Y \sim N(0, A^\top V A)$, and we can estimate (σ^2, Σ) by maximising the log-likelihood arising from W.

Note that, while the difference between MLE and REML lies in the way the covariance parameters are estimated, they also lead to different estimates for the β, since its estimator depends on V.

1.2.4 Quasi-likelihood methods

So far we have discussed models that include variational assumptions on the responses. This is the right thing to do if we believe certain distributions are good approximations.

Nonetheless, it may be that, as in generalised linear models, these impose some restrictions that give rise to modelling limitations such as overdispersion. In other occasions, we may not have sufficient insights to propose a given distribution for the responses and may wish to be agnostic about it, whilst making weaker assumptions such as mean-variance relationships.

In quasi-likelihood methods we only assume that the responses $Y_1, \dots, Y_n$ satisfy the second moment conditions

$$\mathbb{E}Y_i = \mu_i = g^{-1}(x_i^\top \beta) \qquad \text{and} \qquad \operatorname{Var} Y_i = \nu(\mu_i) \tag{1.2}$$

for some known functions g and ν. This, of course, allows us to extend GLMs that suffer from overdispersion to account for it: e.g., instead of the Poisson model we may assume the above with $\nu(\mu_i) = \phi\mu_i$ for a multiplier $\phi > 0$; then, overdispersion will be accounted for when $\phi > 1$, i.e. by inflating the variance.

How can we consistently estimate β? Wedderburn (1974) introduced the *quasi-score function*[4]

$$u(\beta; Y) := \sum_{i=1}^n x_i \frac{Y_i - \mu_i(\beta)}{\nu(\mu_i(\beta))} \mu_i'(\beta).$$

Then, the *quasi-likelihood estimator* $\check{\beta}$ is its zero. If the mean-relationship assumption in (1.2) is true, $\check{\beta}$ is an asymptotically efficient estimator of β with covariance matrix $(X^\top W X)^{-1}$, where W is diagonal with $W_{ii} = (\mu_i')^2/\nu(\mu_i)$ (cf. McCullagh (1983)). In particular, if $\nu(\mu_i) = \phi V^*(\mu_i)$ for some V^* as in the inflated-variance example above, the variance of each entry of $\check{\beta}$ gets multiplied by ϕ. In this case, we can estimate the multiplier ϕ using the same rationale as to construct the generalised Pearson statistic:

$$\check{\phi} := \frac{1}{n-p} \sum_{i=1}^n \frac{(Y_i - \mu_i(\check{\beta}))^2}{V^*(\mu_i(\check{\beta}))}.$$

When the first assumption in (1.2) holds but the second does not, $\check{\beta}$ is still consistent. Note that, even though we do not discuss the details here, the quasi-score function can be modified to account for dependencies between the responses.

[4] NB: herein, $\mu_i'(\beta) = (g^{-1})'(x_i^\top \beta)$

Chapter 2

Classification

In this chapter we will look at the problem of statistical classification. From image recognition to medical diagnosis, classifying objects into discrete categories is of fundamental importance in modern applications. Classification concerns the task of assigning objects to one of two or more groups, on the basis of a sample of training data.

Herein, we focus on *supervised* classification, i.e. when the training data is labelled. Thus, a generic data point is a pair $(X, Y) \in \mathcal{X} \times \mathcal{L}$, where $\mathcal{X}$ is the sample space and which, unless otherwise mentioned, satisfies $\mathcal{X} = \mathbb{R}^p$, and $\mathcal{L}$ is the set of labels which we assume to be finite with L elements, so $\mathcal{L} = \{1, \ldots, L\}$ without loss of generality. A *classifier* is a Borel measurable function $\psi : \mathcal{X} \to \mathcal{L}$ with the interpretation that we assign a point $x \in \mathcal{X}$ to the class $\psi(x)$. Note that, mathematically, classification is just a special case of a regression, where we modelled $\mathbb{E}\, Y \mid X$. If $L = 2$, assume[1] without loss of generality that $\mathcal{L} = \{0, 1\}$, so $\mathbb{E}\, Y \mid X = P(Y = 1 \mid X) = 1 - P(Y = 0 \mid X)$ and this is still a natural quantity to model. However, when $L > 2$, it is no longer sensible to model $\mathbb{E}\, Y \mid X$ (recall that $\mathcal{L} = \{1, \ldots, L\}$ by convention, but the labels could have meanings which are not numerical, e.g. image descriptions, and for which it is not sensible to average as scalars). So what is(are) the right quantity(ies) to model? We can resort to decision theory to answer this.

Let $(X, Y) \sim f$ for some probability density f on $\mathcal{X} \times \mathcal{L}$. Then, the risk or the test error

[1] Typo: I changed $\mathcal{L} = \{1, 2\}$ to $\mathcal{L} = \{0, 1\}$ as in class for this to make sense.

of a classifier ψ is

$$R(\psi) := \int_{\mathcal{X}\times\mathcal{L}} \mathbb{1}_{\psi(x)\neq\ell}\, f(x,\ell)\, dx\, d\ell = \int_{\mathcal{X}} \sum_{\ell\in\mathcal{L}} \mathbb{1}_{\psi(x)\neq\ell}\, f(x,\ell)\, dx.$$

Then, the test error is minimised by the classifier $\psi^*(x) \in \arg\max_\ell f(x,\ell)$. As usual, we will model the data-generating structures. We could attempt to model $f(x,\ell)$ and ψ^* would follow. However, $f(x,\ell)$ is not the most natural quantity to model and we seek to write it in terms of others. Let $\pi(\ell) := \mathbb{P}(Y = \ell)$ denote the marginal probability mass function of the label Y and let f_ℓ be conditional density of $X \mid Y = \ell$ for $\ell \in \mathcal{L}$. Moreover, we write g_X for the marginal distribution of X and $\eta_x(\ell) := \mathbb{P}(Y = \ell \mid X = x)$ for the conditional probability mass function. Then,

$$f(x,\ell) = f_\ell(x)\pi(\ell) = \eta_x(\ell) g_X(x),$$

and, consequently,

$$\psi^*(x) \in \arg\max_\ell \eta_x(\ell) = \arg\max_\ell f_\ell(x)\pi(\ell).$$

Recall that a classifier ψ is π-Bayes for some distribution π on $\mathcal{L}$ if it minimises the π-Bayes risk

$$R_\pi(\psi) := \int_{\mathcal{L}} \int_{\mathcal{X}} \mathbb{1}_{\psi(x)\neq\ell} f_\ell(x)\, dx\, \pi(\ell)\, d\ell.$$

Therefore, it follows from the second equality in the penultimate display that $\psi^* = \psi_\pi^{\text{Bayes}}$, i.e. it is π-Bayes. Such decision rules enjoy desirable properties and so we can model $\eta_x(\ell)$ or $\pi(\ell)$ and $f_\ell(x)$, and obtain the Bayes estimator by using the penultimate display. Note that by Bayes theorem,

$$\eta_x(\ell) = \frac{f_\ell(x)\pi(\ell)}{\sum_{\ell'\in\mathcal{L}} f_{\ell'}(x)\pi(\ell')},$$

so even if we model $\pi(\ell)$ and $f_\ell(x)$ we will be modelling $\eta_x(\ell)$ implicitly. Indeed, $\eta_x(\ell) = P(Y = \ell \mid X = x)$ is also known as the regressor and, once modelled, for a given $x \in \mathcal{X}$ we choose the label that maximises it. Hence, it is an example of a discriminant function. Given the form of $\eta_x(\ell)$, we may realise that if rather than regarding Y as a label in $\mathcal{L}$ we think of it as $Y := (Y_1, \ldots, Y_L)^\top \sim \text{Multinomial}(1, p_1, \ldots, p_L)$ for some $\sum_{\ell=1}^L p_\ell = 1$, modelling $\eta_x(\ell)$ is no more than assuming $p_\ell = p_\ell(x)$ and modelling $p_\ell(x) = P(Y_\ell = 1 \mid X = x) = \mathbb{E}[Y_\ell \mid X = x]$. I.e. we are modelling $\mathbb{E}[Y \mid X = x]$ and have never left the modelling strategy of regression. We will use both forms of Y interchangeably.

In what follows, we present some of the most currently popular classification algorithms (in their basic forms) and we do so in, roughly, decreasing order of interpretability or explainability. The first two are models in the traditional sense in statistics, i.e. plausible representations of reality built using some structural understanding of the data-generating mechanism. Afterwards, we will present algorithms that are increasingly black-box type of procedures which are trained from the data $(x_1, y_1), \ldots, (x_n, y_n)$ mainly to have low risk or prediction error. These have been very successful in practice but their lack of explainability is now limiting their use in sensitive applications such as self-driving cars or some medical applications. As a result, providing them with interpretability is a very active area of research, especially for (deep) neural networks.

2.1 Linear classifiers

Definition 9. Given a classifier ψ, its preimages $\{\psi^{-1}(\ell) : \ell \in \mathcal{L}\}$ form a partition of the sample space. The boundaries of these preimages are called *decision* or *classification boundaries*. We say a classifier is *linear* if all decision boundaries are piecewise affine, and *non-linear* otherwise.

2.1.1 Linear discriminant analysis

Linear discriminant analysis (LDA) assumes that each class density f_ℓ is multivariate Gaussian with mean μ_ℓ and a common covariance matrix Σ, so

$$f_\ell(x) = \frac{1}{(2\pi)^{p/2}(\det \Sigma)^{1/2}} e^{-\frac{1}{2}(x-\mu_\ell)\Sigma^{-1}(x-\mu_\ell)}. \tag{2.1}$$

Then, for a given π

$$\psi_\pi^{\text{Bayes}}(x) = \arg\max_\ell \delta_\ell(x),$$

where, by the monotonicity of the logarithm, the discriminant function δ_ℓ is

$$\delta_\ell(x) = -\frac{1}{2}(x-\mu_\ell)^\top \Sigma^{-1}(x-\mu_\ell) + \log \pi(\ell).$$

As a consequence, the decision boundary between classes k and ℓ is $\{x \in \mathcal{X} : \delta_k(x) = \delta_\ell(x)\}$, and one can show that the decision boundaries are

$$\left\{ x \in \mathcal{X} : \log \frac{\eta_x(k)}{\eta_x(\ell)} = 0 \text{ for some } k \neq \ell \right\},$$

where[2]

$$\log\frac{\eta_x(k)}{\eta_x(\ell)} = \log\frac{\pi(k)f_k(x)}{\pi(\ell)f_\ell(x)} = \log\frac{\pi(k)}{\pi(\ell)} - \frac{1}{2}(\mu_k+\mu_\ell)\Sigma^{-1}(\mu_k-\mu_\ell) + x^\top\Sigma^{-1}(\mu_k-\mu_\ell).$$

In practice, of course, we need to estimate the parameters from the data.

Definition 10. The LDA classifier ψ^{LDA} is given by

$$\psi^{\text{LDA}}(x) = \arg\max_\ell \delta_\ell^{\text{LDA}}(x), \qquad \delta_\ell^{\text{LDA}}(x) = -\frac{1}{2}(x-\hat\mu_\ell)^\top\hat\Sigma^{-1}(x-\hat\mu_\ell) + \log\hat\pi(\ell),$$

where

- $\hat\pi(\ell) := N_\ell/n$, for N_ℓ the number of class-ℓ observations and n the sample size;
- $\hat\mu_\ell = N_\ell^{-1}\sum_{i:y_i=\ell} x_i$ is the class centroid; and,
- $\hat\Sigma = (n-L)^{-1}\sum_{\ell\in\mathcal{L}}\sum_{i:y_i=\ell}(x_i-\hat\mu_\ell)(x_i-\hat\mu_\ell)^\top$ is the estimated common covariance matrix.

We remark that the natural estimator $\hat\pi(\ell)$ is consistent, and so are $\hat\mu_\ell$ and $\hat\Sigma$, as they are the maximum likelihood estimators for μ_ℓ and Σ (the latter rescaled by $n/(n-L)$ for unbiasedness) . By analogous calculations to the data-less ones above, ψ^{LDA} is linear.

2.1.2 Logistic regression classifier

Recall that logistic regression can be used to model binary outcomes by representing their log-odds as a linear function of the covariates. We can turn the logistic regression into a classifier by classifying a new data point according to whether the predicted log-odds is larger or smaller than 0. We extend this idea to multiple classes $\mathcal{L} = \{1,\dots,L\}$ by modelling $Y_i \overset{\text{ind.}}{\sim} \text{Multinomial}(1, p_1, \dots, p_L)$ with

$$p_\ell = p_\ell\left(x; \underline{\beta}_1, \dots, \underline{\beta}_{L-1}\right) = \frac{e^{(1,x^\top)\underline{\beta}_\ell}}{\sum_{\ell'=1}^L e^{(1,x^\top)\underline{\beta}_{\ell'}}}, \quad \underline{\beta}_\ell = (\beta_{\ell 0}, \beta_\ell^\top)^\top \in \mathbb{R}^{1+p}, \underline{\beta}_L \equiv 0.$$

In other words, the class label has a multinomial distribution with a probability vector proportional to $(e^{(1,x^\top)\underline{\beta}_1}, \dots, e^{(1,x^\top)\underline{\beta}_L})^\top$. We have taken by convention $\beta_L = 0$, though this choice is arbitrary and another class as the baseline will lead to exactly the same classifier. Equivalently, if $Y_i \in \{1,\dots,L\}$, we can model

$$\log\frac{\mathbb{P}(Y_i = \ell \mid X = x)}{\mathbb{P}(Y_i = L \mid X = x)} = (1, x^\top)\underline{\beta}_\ell, \quad \ell \in \mathcal{L}, \ \underline{\beta}_L \equiv 0.$$

[2] Typo: changed $\Sigma^{-1/2}$ by Σ^{-1} in the second summand on the right hand side below

Definition 11. Let $\hat{\underline{\beta}}_1, \ldots, \hat{\underline{\beta}}_{L-1}$ be the maximum likelihood estimators for $\underline{\beta}_1, \ldots, \underline{\beta}_{L-1}$. Then, the logistic regression classifier ψ^{LRC} is given by

$$\psi^{\mathrm{LRC}}(x) = \arg\max_{\ell} \delta_\ell^{\mathrm{LRC}}(x), \qquad \delta_\ell^{\mathrm{LRC}}(x) = (1, x^\top)\hat{\underline{\beta}}_\ell.$$

We remark that the logistic regression model provides more than the predicted class label for x. In fact, the vector

$$\frac{\left(e^{(1,x^\top)\hat{\underline{\beta}}_1}, \ldots, e^{(1,x^\top)\hat{\underline{\beta}}_L}\right)^\top}{\sum_{\ell=1}^L e^{(1,x^\top)\hat{\underline{\beta}}_\ell}}$$

gives the estimated probability that x belongs to each of the L classes.

Note that, if $Y_i \in \{1, \ldots, L\}$, the log-likelihood function is

$$\operatorname{loglik}(\beta_1, \ldots, \beta_{L-1}) = \sum_{i=1}^n \sum_{\ell=1}^L \mathbb{1}_{\{Y_i = \ell\}} \log \left(\frac{e^{(1,x_i^\top)\underline{\beta}_\ell}}{\sum_{\ell'=1}^L e^{(1,x_i^\top)\underline{\beta}_{\ell'}}} \right).$$

Thus, if L, p or n are large, Newton–Raphson iterations can be costly and numerically unstable. We may instead, we may solve them via a *gradient descent* method.

Gradient descent Suppose we want to minimise a differentiable function $f : \mathbb{R}^p \to \mathbb{R}$ over $\mathbb{R}^d$. *Gradient descent* is the following iterative algorithm:

1. choose an initial point $\theta^{(0)} \in \mathbb{R}^d$; and,
2. set $\theta^{(t)} = \theta^{(t-1)} - \alpha_t \nabla f(\theta^{(t-1)})$ for $t = 1, 2, \ldots$ until numerical convergence.

Recall that when f is differentiable at a point θ_0, we can approximate f locally by the linear function $f(\theta + \theta_0) \approx f(\theta_0) + \theta^\top \nabla f(\theta_0)$. Thus, the negative gradient $-\nabla f$ represents the direction in which the function value decreases most sharply. Of course, the local approximation breaks down in the long range. Hence, we need to choose the step size or learning rate α_t carefully. It can be shown that under mild conditions, the gradient descent algorithm always converges towards a local minimum of f (see, e.g., Nesterov (2013)).

2.1.3 Optimal separating hyperplane and support vector machines

The classifiers in this section are two-class classifiers, so without loss of generality we assume $\mathcal{L} = \{-1, 1\}$. Multiclass classification can be achieved by performing repeated one-versus-rest classifications, or by performing classification for all pairs of class labels and classify a new data point via a majority voting scheme.

In the two-class setting, both LDA and LRC classify a new observation $x \in \mathbb{R}^p$ as ±1 if

$$\pm(1, x^\top)\underline{\hat{v}} > 0, \quad \text{respectively, for some fitted vector } \underline{\hat{v}} = (\hat{v}_0, \hat{v}^\top)^\top.$$

In LDA this comes as a result of assuming Gaussianity with common covariance matrices for the class densities. In LRC we assume less: we simply postulate that the regressor is proportional to an exponentiated linear combination of the predictors with intercept. The parameters of both models are then fitted by maximum likelihood estimation. In terms of interpretability, LDA and LRC give clear meanings to their parameters. However, even in the simplest case when the training data points of the two classes are completely separated, they provide no immediate interpretation for how the classes are separated or, even, if they are guaranteed to be separated. An alternative then is to begin by making the same mild modelling assumption as LRC, but to fit the parameters so that we explicitly require certain separation properties between perfectly separated classes. This is what the optimal separating hyperplane classifier and the support vector machines do, requiring that the boundary separates the two classes as much as possible.

Mathematically, the optimal separating hyperplane chooses $\underline{\hat{v}} = \underline{\hat{\beta}} = (\hat{\beta}_0, \hat{\beta}^\top)^\top$ in the last display satisfying $\|\hat{\beta}\|_2 = 1$ and that the hyperplane $H := \{x : (1, x^\top)\underline{\hat{\beta}} = 0\}$ has the largest margin between the classes. I.e., since $(1, x^\top)\underline{\hat{\beta}}$ is the distance between x and H,

$$\underline{\hat{\beta}} := \underset{\substack{\beta_0 \in \mathbb{R} \\ \beta \in \mathbb{R}^p : \|\beta\|_2 = 1}}{\arg\max}\ M \qquad \text{subject to } \ y_i(\beta_0 + x_i^\top \beta) \geq M,\ i = 1, \ldots, n. \tag{2.2}$$

The optimal M will be the distance between the separating hyperplane H and the closest training point in each of the classes, so the margin is twice this optimal value of M.

In general, data points from different classes are not perfectly separated, in which case the optimisation problem above has no solution. A way to circumvent this is to allow for some misclassification: for some slack variables $\xi_1, \ldots, \xi_n$ and a given $N \geq 0$, we let

$$\underline{\hat{\beta}} := \mathop{\arg\max}_{\substack{\beta_0 \in \mathbb{R} \\ \beta \in \mathbb{R}^p : \|\beta\|_2 = 1}} M \quad \text{subject to} \quad y_i(\beta_0 + x_i^\top \beta) \geq M(1 - \xi_i),\ \xi_i \geq 0,\ i = 1, \ldots, n, \sum_{i=1}^{n} \xi_i \leq N. \tag{2.3}$$

Therefore, $\xi_i \in (0, 1]$ indicates that x_i is allowed to be within distance M to the decision boundary, whilst $\xi_i > 1$ happens if $y_i \neq \operatorname{sgn}(\hat{\beta}_0 + x_i^\top \hat{\beta})$, i.e. if x_i is on the wrong side of the decision boundary. As a result, constraint $\sum_i \xi_i \leq N$ means that at most N training points are allowed to be misclassified. The observations x_i for which the first inequality in the last display is an equality are called the support vectors.

Definition 12. The support vector machine (SVM) classifier or soft margin SVM is given by

$$\psi^{\text{SVM}}(x) = \operatorname{sgn}((1, x^\top)\underline{\hat{\beta}}),$$

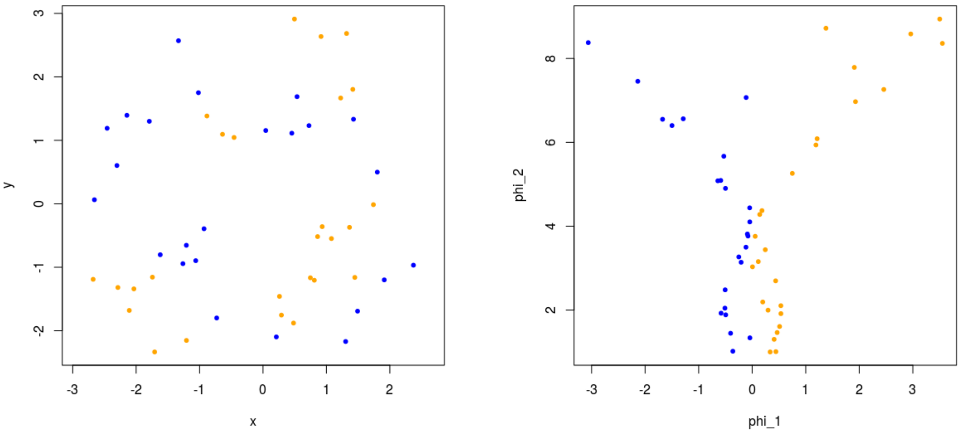
where $\underline{\hat{\beta}}$ is as in (2.3) with $N > 0$. If $N = 0$, this classifier is known as the optimal separating hyperplane classifier or the hard margin SVM.

The optimisation problem in (2.3) was first introduced by Cortes and Vapnik (1995) and can be rephrased as a convex optimisation problem with a quadratic objective and linear constraints. We do this in the corresponding practical and refer to Section 12.2.1 in Hastie, Tibshirani, and Friedman (2009) for its closed-form solution.

2.2 Non-linear classifiers

2.2.1 The kernel method

In this section we take $\mathcal{X}$ general unless otherwise specified. Linear classifiers are simple and are considerably interpretable. However, they are quite limited in their classification power in real-life applications. Notice that if the data is embedded into a sufficiently high dimensional space (possibly infinite dimensional), we will generally be able to separate data points from different classes. See, e.g., the data on the left panel in the figure below, which becomes separable in higher dimensions by adding in some carefully chosen non-linear features: $(x_1, x_2) \mapsto (x_1, x_2, x_1^2 + x_2^2, x_1 x_2(\sqrt{x_1^2 + x_2^2} - 2))$. The right panel below plots $x_1^2 + x_2^2$ against $x_1 x_2(\sqrt{x_1^2 + x_2^2} - 2)$.



Therefore, a simple way to obtain non-linear classifiers is to map them via a *feature map* $\phi : \mathcal{X} \to \mathcal{F}$, where $\dim(\mathcal{F})$ may be infinite (and is usually greater than p if $\mathcal{X} = \mathbb{R}^p$), and to fit linear classifiers to the transformed training data $(\phi(x_1), y_1), \ldots, (\phi(x_n), y_n)$. We will assume that $\phi(x) = (\phi_1(x), \phi_2(x), \ldots), x \in \mathcal{X}$, for some real-valued functions $\phi_1, \phi_2, \ldots$ known as *features.*

A natural question is how to choose good features and how many features shall we choose. Also, if we choose to map $x_1, \ldots, x_n$ into some very high-dimensional space, how do we fit the parameters of the linear classifier efficiently? As the theorem below shows, we can partly circumvent the problem by working with so-called kernels. See the corresponding practical for some details on how to choose an appropriate kernel in practice.

Definition 13. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a *positive definite kernel* on $\mathcal{X}$ if $(k(x_i, x_j))_{i,j=1}^m$ is a positive semidefinite matrix for any $x_1, \ldots, x_m \in \mathcal{X}$ and $m \in \mathbb{N}$.

Recall that a *Hilbert space* is an inner product space where every Cauchy sequence (with respect to the norm associated with the inner product) has a limit. You do not need to know this for this course, but simply that it can be viewed as a generalisation of the Euclidean space to possibly infinite dimensions.

Theorem 4. *If $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel, then there exists a Hilbert space $\mathcal{H}$ and a map $\phi : \mathcal{X} \to \mathcal{H}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product on $\mathcal{H}$. Furthermore, if $L : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}^+$ is a loss function and $\lambda \geq 0$,*

$$\min_{h \in \mathcal{H}} \left\{ \sum_{i=1}^{n} L(y_i, h(x_i)) + \lambda \langle h, h \rangle_{\mathcal{H}} \right\} = \min_{b \in \mathbb{R}^n} \left\{ \sum_{i=1}^{n} L(y_i, K_{i\cdot} b) + \lambda b^\top K b \right\},$$

where $K := (k(x_i, x_j))_{i,j=1}^n$.

The first part of the theorem is part of Moore–Aronszajn's Theorem and it tells us that any positive definite kernel can be realised as the inner product in some (possibly infinite-dimensional) feature space. The second half, which can be found in, e.g., Hastie, Tibshirani, and Friedman (2009), complements the first by saying that the possibly infinite dimensional optimisation problem arising from embedding our data into $\mathcal{H}$ reduces to a finite dimensional optimisation problem. This is known as the *kernel property* and using it to construct new procedures from existing ones is called the *kernel method*. Therefore, the theorem tells us that, if our goal is only prediction, we can propose a positive definite kernel k without needing to know exactly what ϕ is or where it maps to, and the new procedure can be solved efficiently: if h^* and b^* are the optimal values of the optimisation problems in the theorem, $h^* = \sum_{i=1}^n b_i^* K(x, x_i)$. Indeed, LDA, LRC and SVM can all be kernelised. For a more detailed overview of this general area, known as Reproducing Kernel Hilbert Spaces (RKHS), see the Part III course notes on Modern Statistical Methods.

2.2.2 Artificial neural networks

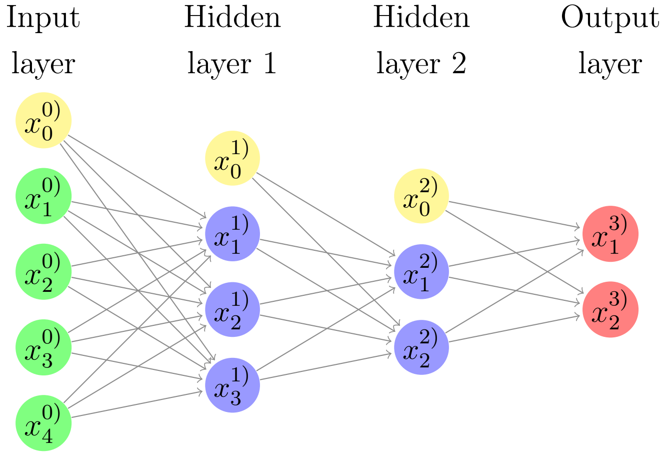
Artificial neural networks originated as models of how neurons in the brain act collectively to process complex information, although they also correspond to other non-linear statistical models developed simultaneously without this application in mind. They can be used both for regression and classification. In the context of classification, where we recall that a generic data point $(x, Y), x \in \mathbb{R}^p$, can be modelled as $Y \sim \text{Multinomial}(1, \eta_x(1), \ldots, \eta_x(L))$, they model the regressors $\eta_x(\ell)$ through the composition of a series of simple, non-linear functions that can be visually described by a directed graph. A neural network consists an *architecture*, an *activation rule* and an *output rule*. We explain these first and then introduce the mathematical model.

Architecture. The architecture of a neural network can be described as a directed graph whose vertices are called *neurons*, *units* or *nodes*. The set of nodes with no incoming edges (except the so-called bias nodes, defined later) are called *input nodes*, and correspondingly the set of nodes with no outgoing edges are called *output nodes*. We will focus on the so-called *feedforward neural network*, where we can put nodes into *layers* such that every directed edge is going from one layer to the next (mathematically, this basically says that the architecture is a directed acyclic graph or DAG). We assume that all input nodes (and the first bias node) make up the first layer, the *input layer*, and all output nodes make up the final layer, called the *output layer*. All other layers (including their respective bias nodes) are called *hidden layers*. This graphical interpretation allows us to have a better understanding of the construction of the model.

Activation function. For a given observation $x \in \mathbb{R}^p$, the input layer nodes receives a value from the coordinate values of x. Like neurons in the brain, every node in the hidden layers of a feedforward neural network receives input from its predecessors (analogue of synapsed neurons) and sends an output to its successors. The precise dependence is described by an *activation rule*, which generally corresponds to the image of a linear combination of the values of the predecessors through an *activation function*. In this case, we interpret the activation rule as a two-step process: first linearly aggregate output values from predecessors to and, second, apply the activation function to this value.

Output rule. At an output node, after taking a linear combination of its predecessors' values we do not apply the activation function, but rather use an *output rule* or sometimes *function* to aggregate information across these linear combinations so as to respect the structure of the data we are trying to model (in classification, this is that the regressors should add up to 1 for any $x \in \mathbb{R}^p$).

Model. We denote the j^{th} unit in the h^{th} layer by $x_j^{h)}$, where $j = 0, \ldots, p_h$ for $h = 0$ (input layer) and $h = 1, \ldots, H$ (hidden layers), and $j = 1, \ldots, p_h$ for $h = H + 1$ (output layer). The units with $j = 0$ are called the *bias nodes*. See the figure below for an example where $H = 2, p_0 = 4, p_1 = 3, p_2 = 2, p_3 = 2$ and all nodes have as many connections as possible, i.e. it is a fully connected neural network to perform binary classification from 4 covariates with two hidden layers of widths 3 and 2, respectively.



For each unit $x_j^{h)}, j = 1, \ldots, p_h, h = 1, \ldots, H + 1$, we introduce the parameters or weights $\underline{\beta}_j^{h)} \in \mathbb{R}^{1+p_{h-1}}$ and, writing $\underline{x}^{h)} := \left(x_j^{h)}\right)_{j=0}^{p_h}$ for $h = 0, \ldots, H$, we define the linear combination

$$s_j^{h)} = \underline{x}^{h-1)\top} \underline{\beta}_j^{h)} = \sum_{k=0}^{p_{h-1}} x_k^{h-1)} \beta_{jk}^{h)}.$$

Note that a missing edge from $x_k^{h-1)}$ to $x_j^{h)}$ for some $j = 1, \ldots, p_h, k = 0, \ldots, p_{h-1}, h = 1, \ldots, H + 1$, is modelled by $\beta_{jk}^{h)} = 0$. Then, taking $x_0^{h)} = 1$ for all $h = 0, \ldots, H$ by convention, $p_0 = p$ (number of covariates), $x^{0)} = x_j$ (j^{th} coordinate of the covariates vector x) and $p_{H+1} = L$, a feedforward neural network models the regressors $(\eta_x(1), \ldots, \eta_x(L))$ through $\underline{x}^{H+1)} := \left(x_j^{H+1)}\right)_{j=1}^{p_{H+1}}$, where

$$x_j^{H+1)} = f_j\left(s_1^{H+1)}, \ldots, s_{p_{H+1}}^{H+1)}\right), \quad j = 1, \ldots, p_{H+1},$$

and

$$x_j^{h)} = g_h(s_j^{h)}), \quad j = 1, \ldots, p_h, \; h = 1, \ldots, H,$$

for some output function or rule $(f_1, \ldots, f_{p_{H+1}})$ and activation functions g_h. We could of course have proposed this model without resorting to a network diagram, but doing so aids the understanding. Given the model clearly see the role of the bias nodes: just as in the case of regression, we may want to include an intercept term when computing a unit's value from its predecessors and we achieve this by simply adding a bias node with constant value 1 in each layer. A commonly used output function is the *softmax* function, where

$$f_j(s_1, \dots, s_L) = \frac{\exp(s_j)}{\sum_{\ell=1}^{L} \exp(s_\ell)}$$

and one may take $\underline{\beta}_L^{H+1)} \equiv 0$ by convention, and commonly used activation functions include the *sigmoid* (also known as *expit*, or *logistic*) function $g(s) = \frac{e^s}{1+e^s}$ and the *rectifier* or *ReLU* function $g(s) = \max\{s, 0\}$. Note that, writing $\theta := \left(\underline{\beta}_j^{h)}\right)_{\substack{j=1,\dots,p_h \\ h=1,\dots,H+1}}$ for all the parameters, we have $\underline{x}^{H+1)} = \underline{x}^{H+1)}(\theta, x)$ so we may treat $\underline{x}^{H+1)}$ and its coordinates as functions of θ or x indistinctively.

So far we have only introduced the model for the regressors. Given data $(x_1, Y_1), \dots, (x_n, Y_n)$ and a loss function $L : \mathbb{R}^L \times \mathbb{R}^L \to \mathbb{R}^+$, we train an artificial neural network or, more specifically, fit its parameters θ by minimising

$$R(\theta) = \sum_{i=1}^{n} L\left(y_i, \underline{x}^{H+1)}(\theta)\right).$$

Neural networks used in practice often have many more weights than training data points. Consequently, some form of regularisation is necessary, and it is common to add to $R(\theta)$ a penality term $P(\theta)$, e.g. $P(\theta) = \lambda \|\theta\|_q^q, \lambda \geq 0, q \in \{1, 2\}$. Under the usual assumption $Y_i \overset{\text{ind.}}{\sim}$ Multinomial$(1, x_1^{H+1)}(x_i), \dots, x_L^{H+1)}(x_i))$, a natural loss function is the negative log-likelihood or the *cross-entropy loss*

$$L(y, x) = -\sum_{\ell=1}^{L} \mathbf{1}_{\{y_\ell = 1\}} \log x_\ell.$$

An alternative is the *squared error loss* which is given by

$$L(y, x) = \|y - x\|_2^2 = \sum_{\ell=1}^{L} (y_\ell - x_\ell)^2.$$

Definition 14. Let $\hat{\underline{x}}^{H+1)}$ be the trained feedforward neural network, i.e., if $\hat{\theta}$ are the fitted parameters, $\hat{\underline{x}}^{H+1)} = \underline{x}^{H+1)}(\hat{\theta})$. Then, the resulting feedforward neural network classifier is

$$\psi^{FNN}(x) = \arg\max_{\ell} \hat{x}_\ell^{H+1)}(x).$$

Note that when no hidden layers are present, a fully-connected neural network with sigmoid activation and softmax output trained with the cross-entropy loss is simply a logistic regression classifier. As such, a feedforward neural network with sigmoid activations may be viewed as a hierarchical model of logistic regressions. Generally, an artificial neural network is a non-linear classifier.

Part of the success neural networks lies in the composition of non-linear activation functions and the ability to shift and shrink/dilate these arbitrarily. The following theorem implies that we can approximate any binary-class data-generating structure arbitrarily well by a single-hidden layer trained feedforward neural network with sigmoid activation and a sufficiently large number of weights and samples. We omit some of the mild assumptions it requires.

Theorem 5. *Assume* $Y_i \overset{ind.}{\sim}$ *Multinomial*$(1, \delta_1(x_i), \delta_2(x_i))$ *where* $\delta = (\delta_1, \delta_2)^\top : \mathbb{R}^p \to \{(z_1, z_2) \in \mathbb{R}^2_{\geq 0} : z_1 + z_2 = 1\}$ *satisfies some mild regularity assumptions (related to continuity/differentiability). Then, there exists a* $C > 0$ *and a fully-connected feedforward neural network with* $H = 1, p_H = m$*, softmax output function and sigmoid activation functions, trained using the squared error loss, such that*

$$\mathbb{E}\, \|\delta - \underline{\hat{x}}^{H+1)}\|_2^2 \leq C \left(\frac{1}{m} + m \frac{p \log n}{n} \right).$$

In particular, taking $m \sim \sqrt{n/(p \log n)}$*, the left hand side is at most of order* $\sqrt{(p \log n)/n}$*.*

This result can be found in Barron (1994) and is derived from the classical result in Barron (1993). The latter studies the bias error of the feedforward neural networks in the theorem above, which equals the first term on the right hand side of the display, i.e. C/m. Since this earlier result implies that feedforward neural networks can approximate continuous functions (on compact sets) arbitrarily well, neural networks are said to be *universal approximators* in the machine learning community. While these results are reassuring, they can be slightly misleading: in practice, the number m of weights needed to attain a certain precision will be much larger than the total number of weights required from a network with several hidden layers so that it attains the same precision.

Note that in the modern applications where neural networks are used, $\dim(\theta)$ and n are usually quite large. Therefore, it is common to use stochastic versions of gradient descent methods to train the networks, i.e., to minimise $R(\theta)$.

Stochastic gradient descent

Suppose we wish to minimise a differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ that can be expressed as the average of functions

$$f(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w).$$

Each f_i is an approximation to f and can obviously be computed much faster. Then, to reduce the cost of evaluating the gradient of f we can replace it by each of the (cheaper) f_i and increase the number of gradient steps to compensate for the drop in quality of the computed gradient. *Stochastic gradient descent* exploits this insight:

1. choose an initial point $w^{(0,n)} \in \mathbb{R}^p$; and,
2. for $e = 1, \ldots, E$ (the so-called *epochs*), repeat the following:
   (a) set $w^{(e,0)} \leftarrow w^{(e-1,n)}$;
   (b) randomly reorder indices $1, \ldots, n$; and,
   (c) for $i = 1, \ldots, n$, update $w^{(e,i)} \leftarrow w^{(e,i-1)} - \alpha_e \nabla f_i(w^{(e,i-1)})$.

It can be shown that, for slowly decaying learning rates, SGD converges almost surely to a local minimum (cf. Lelong (2005)). Note that one entire epoch of stochastic gradient descent takes roughly the same computational time as one step in the gradient descent. However, the stochastic gradient descent estimator after T epochs is typically closer to a good local minimum optimum than the gradient descent estimator after T steps. Furthermore, and although not fully understood theoretically, early stopping in stochastic gradient descent can also be viewed as a form of regularisation: note that the loss function of a neural network has (possibly infinitely) many minima and we are not interested in finding the global minimum, as it will generally correspond to an overfitted solution; then, instead of executing the stochastic gradient descent until it convergence to a global minimum, it is typically run only for a few epochs, guided by validation error from a validation set held aside.

Given the motivation of stochastic gradient descent above, we could approximate f by the average of some of the f_i's rather than by a single one. *Mini-batch stochastic gradient descent* proceeds as above, replacing step 2(c) by the update

$$w^{(e,k)} \leftarrow w^{(e,k-1)} - \alpha_e \frac{1}{n/K} \sum_{i=\frac{n}{K}(k-1)+1}^{\frac{n}{K}k} \nabla f_i(w^{(e,k-1)})$$

for $k = 1, \dots, K$, where $K \in (1, n)$ is such that $n/K \in \mathbb{N}$. Due to this generalisation, the first algorithm presented above is also known as *vanilla stochastic gradient descent*, and gradient descent is sometimes referred to as *batch stochastic gradient descent.*

2.2.3 Nearest neighbour classifiers

The nearest neighbour classifiers are based on the simple and intuitive idea that observations close in the sample space are likely to belong to the same class. Therefore, given a new observation, we classify it according to the class labels of its nearest neighbours. The nearest neighbour classifier places no distributional assumption on the data generating mechanism and has been shown to achieve good performance in a wide range of applications.

2.2.3.1 k-nearest neighbour classifier

Suppose we have training data $(X_1, Y_1), \dots, (X_n, Y_n) \in \mathbb{R}^p \times \{1, \dots, L\}$. Fix a point $x \in \mathbb{R}^p$ and let $(X_{(1)}, Y_{(1)}), \dots, (X_{(n)}, Y_{(n)})$ be a permutation of the training data satisfying

$$\|X_{(1)} - x\|_2 \leq \|X_{(2)} - x\|_2 \leq \dots \leq \|X_{(n)} - x\|_2.$$

.

Definition 15. The *k-nearest neighbour classifier* (kNN classifier) is defined to be

$$\psi^{k\mathrm{NN}}(x) \in \operatorname*{arg\,max}_{\ell=1,\dots,L} \frac{1}{k} \sum_{i=1}^{k} \mathbb{1}_{\{Y_{(i)}=\ell\}},$$

with ties broken arbitrarily.

When $k = 1$, the 1NN classifier simply classifies a new observation according to the label of its nearest neighbour in the training set. In general, kNN amounts to a majority voting scheme among the k nearest neighbours. Note that the classifier is implicitly estimating the class probability distributions by $\left(k^{-1} \sum_{i=1}^{k} \mathbb{1}_{\{Y_{(i)}=\ell\}}\right)_{\ell=1}^{L}$. Indeed, if we take the alternative formulation of classification where for a generic label Y and covariate vector X are related by $Y \sim \text{Multinomial}(1, \eta_X(1), \dots, \eta_X(L))$ for some regressors, kNN is modelling $\eta_x(\ell) = \mathbb{E}\, Y_\ell \,|_X$ as $k^{-1} \sum_{i=1}^{k} Y_{(i),\ell}$. If some covariate vectors from the training data were equal to X, taking the average of their respective labels would result in a very intuitive approximation. Since it is not guaranteed that the training data contains such covariate vectors, we thus see that the mild (and non-parametric) modelling assumption the kNN classifier makes is that the regressors are well approximated locally by piecewise constant functions. We expect this very intuitive classifier to have good theoretical properties. Indeed, let the risk of a classifier ψ trained with n data points be

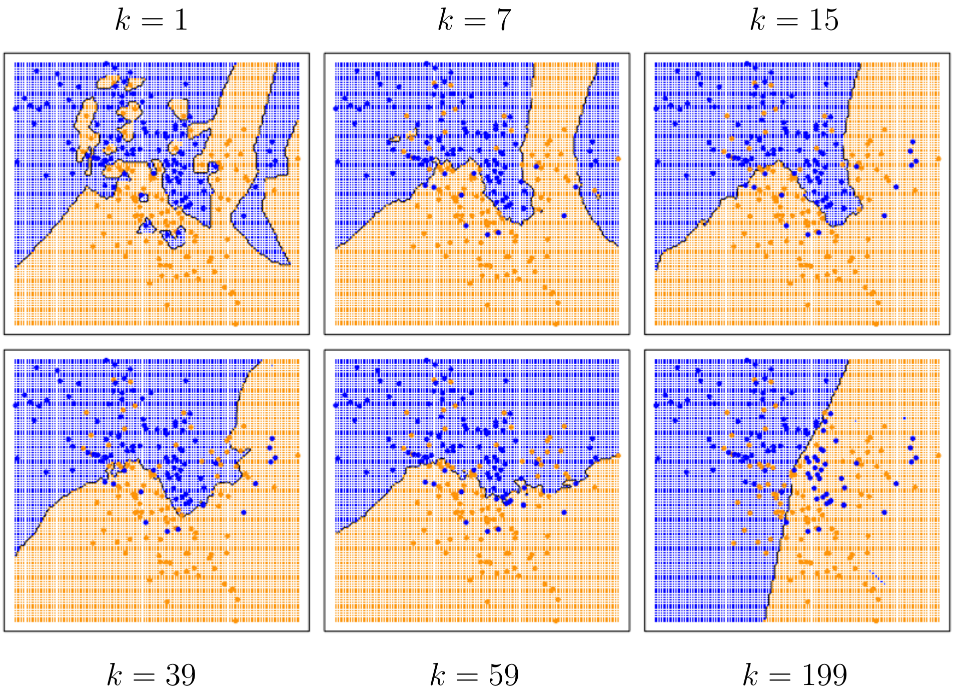
$$R_n(\psi) := \mathbb{E}[\psi(X) \neq Y],$$

where the expectation is taken over both the training data and the new independent data point.

Theorem 6. *In an L-class classification problem, we have*

$$R(\psi^{Bayes}) \leq \lim_{n\to\infty} R_n(\psi^{1NN}) \leq 2R(\psi^{Bayes}) - \frac{L}{L-1} R(\psi^{Bayes})^2.$$

The theorem was shown by Cover and Hart (1967), and it says that the 1NN classifier performs almost as well as the Bayes classifier. In particular, the risk of the 1NN is bounded above by twice the risk of the Bayes classifier, which may be interpreted as saying that, as the sample size increases, at least half of the classification information in it is contained in the nearest neighbour. As Theorem 5, this theorem is reassuring but possibly misleading: as $n \to \infty$ the bias of the 1NN classifier vanishes, whilst for finite samples it (and the variance) may be large. In general, increasing k has the effect of reducing the variance of the kNN classifier by averaging over more data, but this comes at the cost of decreasing the complexity of the classification boundaries, thus increasing the bias of the kNN classifier. The following pictures show how the classification boundaries change for various choices of k.



$k = 1$ $k = 7$ $k = 15$

$k = 39$ $k = 59$ $k = 199$

Many theoretical works have been devoted to find the optimal scaling of k. For example, Hall, Park and Samworth (2008) showed that the optimal k choice k^* should grow at a rate of $C_p n^{4/(p+4)}$ as $n \to \infty$, with the leading constant C_p depending on p and the (unknown) class densities. In practice, cross-validation is the preferred method of choosing k.

2.2.4 Bootstrap aggregation and weighted nearest neighbours

Bootstrap aggregation, or "bagging", was introduced by Breiman (1996) to improve the performance of a predictor through bootstrap resampling from the empirical distribution. For a given training data set of size n, the bootstrap aggregation nearest neighbour classifier is obtained by applying the 1-nearest neighbour classifier to many resamples from the training data.

Definition 16. Let $B, m \in \mathbb{N}$. Then, the bootstrap aggregation nearest neighbour classifier is constructed as follows:

1. for $b = 1, 2, \ldots, B$,

   (a) sample a subset of m points $(X_1^{(b)}, Y_1^{(b)}), \ldots, (X_m^{(b)}, Y_m^{(b)})$ from the training data uniformly with replacement;

(b) compute $\psi^{(b)}(\cdot)$, i.e., the 1-nearest neighbour classifier based on the training data $(X_1^{(b)}, Y_1^{(b)}), \ldots, (X_m^{(b)}, Y_m^{(b)})$; and,

2. aggregate the 1NN classifiers to form

$$\psi_B^{\mathrm{bNN}}(x) = \operatorname*{arg\,max}_{\ell=1,\ldots,L} \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}_{\{\psi^{(b)}(x)=\ell\}}.$$

For $m << n$, the bagged nearest neighbour classifier reduces the variance of 1NN classifier by averaging over many bootstrap samples, whereas the kNN classifier reduces variance by averaging over more nearby points. Furthermore, ψ_B^{bNN} only slightly increases the bias with respect to the 1NN classifier, thus having a lower risk overall.

Theorem 7. *Under mild assumptions and if $m \to \infty$ and $m/n \to 0$ appropriately,*

$$\psi_\infty^{bNN}(x) := \lim_{B\to\infty} \psi_B^{bNN}(x) = \operatorname*{arg\,max}_{\ell=1,\ldots,L} \sum_{i=1}^{n} w_i^{bNN} \mathbb{1}_{\{Y_{(i)}=\ell\}}$$

for some $\big(w_i^{bNN}\big)_{i=1}^n$ satisfying $\sum_{i=1}^n w_i^{bNN} = 1$, and

$$R\big(\psi_\infty^{bNN}\big) \le R\big(\psi^{k^*NN}\big) \quad \text{for } p \ge 2 \text{ (strict inequality for } p > 2\text{)}.$$

Hence, both kNN and bagged nearest neighbour classifiers are special examples of *weighted nearest neighbour classifiers*, defined to be of the form

$$\psi^{\mathrm{wNN}}(x) = \operatorname*{arg\,max}_{\ell=1,\ldots,L} \sum_{i=1}^{n} w_i \mathbb{1}_{\{Y_{(i)}=\ell\}}$$

for some weights $(w_i)_{i=1}^n$ such that $\sum_{i=1}^n w_i = 1$. The theorem above was shown by Samworth (2012), who also proved that $R\big(\psi_\infty^{\mathrm{bNN}}\big)$ is not optimal, but converges to the optimal weighting scheme as $p \to \infty$, and that any unweighted kNN classifier can be modified into a weighted nearest neighbour classifier with improved performance.

Chapter 3

Time series

A time series is a set of observations recorded over time or, mathematically, a stochastic process $X := (X_t)_{t \in T}$ indexed by a time index set T. You can think for example of the GDP of a country over the years (or quarters) or the hourly measurements of temperature over a month. Time series present a challenge for the statistical analysis because of the obvious correlation introduced by the sampling of adjacent points in time. Throughout we assume $T = \mathbb{Z}$ or $\mathbb{N}$.

The objective of time series analysis is to understand the process that generates the sample time series and to predict (forecast) the response variable at future times. A complete description of the time series X is given by the joint distribution function of all its elements. In practice, some modelling assumptions on the form of this joint distribution will be needed. An important object in the analysis of time series with $\mathrm{Var}(X_t) < \infty$ for all $t \in T$ is the *autocovariance function* (ACVF),

$$\gamma_X(s,t) := \mathrm{Cov}(X_s, X_t) = \mathbb{E}(X_s X_t) - \mathbb{E}X_s \mathbb{E}X_t, \quad s, t \in T.$$

The autocovariance measures the *linear* dependence between the response at two different times. It is often more convenient to consider instead the *autocorrelation function* (ACF)

$$\rho_X(s,t) := \mathrm{corr}(X_s, X_t) = \frac{\gamma_X(s,t)}{\sqrt{\gamma_X(s,s)\gamma_X(t,t)}}, \quad s, t \in T.$$

3.1 Stationary time series

Without any assumptions on the process generating the time series it would be impossible to carry out any meaningful statistical analysis because, as we will assume without loss of generality, we only observe one replicate from the random vector $(X_1, \dots, X_n)$. However, in many applications there is some structure (e.g. regularity or smoothness) in the underlying process, which allows us to borrow information across the time series to investigate the characteristics of the process. An important class of these processes are *stationary* time series.

Definition 17. A time series is called *strictly stationary* if the joint distribution of $(X_{t_1}, \dots, X_{t_k})$ is equal to the joint distribution of the time shifted set $(X_{t_1+h}, \dots, X_{t_k+h})$, for every $k \in \mathbb{N}$, $t_1, \dots, t_k, h \in T$.

This definition implies that, for a strictly stationary series X, X_s and X_t are identically distributed for any $s, t \in T$. Moreover, if $\mathrm{Var}(X_t) < \infty$ for all $t \in T$, the autocovariance function satisfies $\gamma_X(s,t) = \gamma_X(s+h, t+h)$ for all $s, t, h \in T$. Therefore the covariance between two time points depends only on their time shift or *lag h*. Strict stationarity is often quite a strong assumption and, furthermore, it is hard to check from a single sample. Thus, we may wish to retain only its first and second order properties and model these. This is akin to what we did in regression, where we were interested in modelling the mean and variance of the responses, although these were assumed to be independent in most cases therein.

Definition 18. A time series is called *weakly stationary* or *second-order stationary* or simply *stationary* if

1. the mean $\mathbb{E}X_t =: \mu$ is constant for all $t \in T$;
2. X_t has finite variance for all $t \in T$; and,
3. the autocovariance function $\gamma_X(s,t)$ depends on s and t only through the absolute value of their lag $h = s - t$.

Strict stationarity and finite variance imply (weak) stationarity, but the opposite is not true. We may also simplify the notation for the autocovariance and autocorrelation functions: with a slight abuse of notation, we write $\gamma_X(|s-t|) = \gamma_X(s,t)$.

Example 5. A simple type of generating process for an observed time series is a collection of uncorrelated random variables $W := (W_t)_{t \in T}$, each with mean zero and constant variance $0 < \sigma^2 < \infty$. In engineering, this process is usually called *white noise*, denoted by $W \sim \mathrm{WN}(0, \sigma^2)$.

The white noise process is clearly stationary, because $\operatorname{Var} W_t = \sigma^2$, $\mathbb{E}W_t = 0$ and $\gamma_W(h) = 0$ if $h \neq 0$ and $\gamma_W(0) = \sigma^2$. If $W_t \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2)$ (*Gaussian white noise*), then the series is strictly stationary.

Naturally, we wish to go beyond white noise and incorporate dependencies into our temporal models. We now motivate and introduce some of the most popular parametric models for stationary series. Throughout we take $T = \mathbb{Z}$ and refer to the practicals and example sheets for the case $T = \mathbb{N}$, and we assume (unless otherwise stated) that any (stationary) time series has mean zero.

Proposition 2. *Let $Y := (Y_t)_{t\in\mathbb{Z}}$ be stationary and, for some $(\psi_j)_{j\in\mathbb{Z}} \in \mathbb{R}^{\mathbb{Z}}$, define*

$$X_t := \sum_{j\in\mathbb{Z}} \psi_j Y_{t-j}, \quad t \in \mathbb{Z}\,.$$

Then, $X := (X_t)_{t\in\mathbb{Z}}$ is stationary if $\sum_{j\in\mathbb{Z}} |\psi_j| < \infty$.

Definition 19. A time series $X := (X_t)_{t\in\mathbb{Z}}$ is linear or a linear process if, for some $(\psi_j)_{j\in\mathbb{Z}} \in \mathbb{R}^{\mathbb{Z}}$ such that $\sum_{j\in\mathbb{Z}} |\psi_j| < \infty$ and $\varepsilon := (\varepsilon_t)_{t\in\mathbb{Z}} \sim \mathrm{WN}(0, \sigma^2)$ with $\sigma^2 > 0$,

$$X_t = \sum_{j\in\mathbb{Z}} \psi_j \varepsilon_{t-j}, \quad t \in \mathbb{Z}\,.$$

Note that, by definition, a linear process is stationary. A celebrated result that we mention informally is Wold's decomposition: any stationary time series is either linear or it is a sum of a linear process and a deterministic process. We omit the definition of the latter, but we emphasise that Wold's decomposition places linear processes at the heart of stationary processes.

Autoregressive moving average (ARMA) processes are the textbook example of linear processes. This is not only due to their intuitive form, parametric nature and the simplicity of their fitting and prediction procedures, but also due to the remarkable property that they can model autocovariance functions *exactly* for any fixed number of lags: let $\gamma : \mathbb{N} \to \mathbb{R}$ be any autocovariance function; then, for any $H \in \mathbb{N}$, there exist an ARMA process X such that $\gamma_X(h) = \gamma(h)$ for all $h = 0, \ldots, H$. This makes them a very useful class of models, since stationary processes are defined through their first and second moments and we have assumed they have zero mean. Indeed, we focus on ARMA processes from now on.

3.1.1 Autoregressive models

Definition 20. A time series $X := (X_t)_{t\in\mathbb{Z}}$ follows an autoregressive model of order p, denoted by $X \sim \mathrm{AR}(p)$, if it is stationary and it satisfies

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t, \quad t \in \mathbb{Z}, \tag{3.1}$$

for some $\phi_1, \ldots, \phi_p \in \mathbb{R}, \phi_p \neq 0$, and $\varepsilon := (\varepsilon_t)_{t\in\mathbb{Z}} \sim \mathrm{WN}(0, \sigma^2), \sigma^2 > 0$.

As the name suggests, we model the time series at any time as depending on some of its values. More particularly, each coordinate depends on its most recent p past values, which determines the autocovariance function of X and its geometric-decay property (see example sheet). A useful quantity to study the ACVF—and the time series in general—is the *backshift operator* B. It acts on the time series at any time t as $BX_t = X_{t-1}$, and can also be extended to powers by $B^h X_t = X_{t-h}, h \in \mathbb{N}$. Then, the *autoregressive operator* is defined as

$$\Phi(B) := 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p,$$

and can write (3.1) more succinctly as

$$\Phi(B) X_t = \epsilon_t, \quad t \in \mathbb{Z}\,.$$

The autoregressive operator decorrelates an AR time series into a white noise series and it determines its properties.

Let us consider the AR(1) model to gain intuition on the main properties of AR processes. We denote the model parameter by $\phi = \phi_1 \neq 0$ and consider a time series X (not necessarily stationary) satisfying (3.1). Iterating this expression backward j times, we get

$$X_t = \phi^j X_{t-j} + \sum_{k=0}^{j-1} \phi^k \varepsilon_{t-k}, \quad t \in \mathbb{Z}, j \in \mathbb{N}\,.$$

Provided that $|\phi| < 1$, we can represent X_t as

$$X_t = \sum_{k=0}^{\infty} \phi^k \varepsilon_{t-k}, \quad t \in \mathbb{Z},$$

where the equality is in almost sure sense, or, formally written in terms of B and Φ,

$$X_t = \frac{1}{1 - \phi B} \varepsilon_t =: \Phi^{-1}(B) \varepsilon_t.$$

Indeed, for any $j \in \mathbb{N}$,

$$\mathbb{E}\left|X_t - \sum_{k=0}^{\infty} \phi^k \varepsilon_{t-k}\right| \leq |\phi|^j \, \mathbb{E}\,|X_{t-j}| + \sum_{k=j}^{\infty} |\phi|^k \, \mathbb{E}\,|\varepsilon_{t-j}| \leq |\phi|^j \sqrt{\operatorname{Var} X_1} + \sigma^2 \frac{|\phi|^j}{1-|\phi|},$$

and so the left hand side must be zero and the conclusion follows. Consequently, X is a linear process, so it must be stationary, and, by similar arguments,

$$\gamma_X(h) = \sigma^2 \frac{\phi^h}{1-\phi^2}, \quad h \in \mathbb{N}\,.$$

By inspecting the arguments above, it is clear that when $|\phi| > 1$ we cannot write X_t in terms of infinite past values of the white noise. Indeed, the process is explosive because the values of the series can quickly become large in magnitude. We can, nonetheless, iterate to the future:

$$X_t = \phi^{-1} X_{t+1} - \phi^{-1} \epsilon_{t+1} = \cdots = -\sum_{k=1}^{\infty} \phi^{-k} \epsilon_{t+k}, \quad t \in \mathbb{Z},$$

where the last equality is in almost sure sense. Again, this means that X is linear and stationary, but it shows that this model useless for practical purposes as the solution is future dependent. Lastly, when $|\phi| = 1$ we can neither iterate to the future or the past. Indeed, no stationary time series X satisfies $X_t = \phi X_{t-1} + \varepsilon_t, t \in \mathbb{Z}$, and the AR(1) definition implicitly rules out $|\phi| = 1$.

Let us generalise these properties to AR(p) processes.

Proposition 3. *A stationary solution to* (3.1) *exists if and only if* $\Phi(z) \neq 0$ *for all* $z \in \mathbb{C}$ *such that* $|z| = 1$. *If a solution exists, it is unique.*

The condition of the proposition is generally referred to as $\Phi(z), z \in \mathbb{C}$, having no unit roots or, simply, that X has no unit root or that it is not unit root.

Definition 21. A time series $X := (X_t)_{t\in\mathbb{Z}} \sim \mathrm{AR}(p)$ is causal if it satisfies the definition of linearity with $\psi_{-1} = \psi_{-2} = \ldots = 0$.

Proposition 4. *A time series* $X := (X_t)_{t\in\mathbb{Z}} \sim AR(p)$ *is causal if and only if* $\Phi(z) \neq 0$ *for all* $z \in \mathbb{C}$ *such that* $|z| \leq 1$

3.1.2 Moving average models

Definition 22. A time series $X := (X_t)_{t\in\mathbb{Z}}$ follows an moving average model of order q, denoted by $X \sim \mathrm{MA}(q)$, if it satisfies

$$X_t = \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q}, \quad t \in \mathbb{Z}, \tag{3.2}$$

for some $\theta_1, \ldots, \theta_q \in \mathbb{R}, \theta_q \neq 0$, and $\varepsilon := (\varepsilon_t)_{t\in\mathbb{Z}} \sim \mathrm{WN}(0, \sigma^2), \sigma^2 > 0$.

Assuming that X is stationary is not needed in the definition of moving average models as (3.2) immediately guarantees so. The *moving average operator* is defined as

$$\Theta(B) := 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q,$$

and we may write (3.2) as

$$X_t = \Theta(B)\epsilon_t, \quad t \in \mathbb{Z}\,.$$

Let us consider the MA(1) model to introduce properties of MA processes. Denoting the model parameter by $\theta = \theta_1 \neq 0$, it is easy to check that

$$\gamma_X(h) = \begin{cases} (1+\theta^2)\sigma^2 & h = 0, \\ \theta\sigma^2 & h = 1, \\ 0 & h \geq 2, \end{cases}$$

i.e., the time series at any time is uncorrelated with any time coordinates away from it by more than $q = 1$ time unit. We will see in the example sheet that this extends to MA(q) models with $q > 1$ and, furthermore, this phenomenon characterises MA processes within stationary processes (see below). Note that the ACVF does not change when θ and σ^2 are replaced by $1/\theta$ and $\theta^2\sigma^2$, respectively. Thus, the model is not identifiable from its first two moment properties. In particular, if $\varepsilon_t \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2)$, the model is not identifiable at all. As a result, for statistical purposes we should choose between $|\theta| \leq 1$ or $|\theta| \geq 1$. Notice that $\varepsilon_t = \Theta^{-1}(B)X_t$ if and only if $|\theta| < 1$. In other words, for this parameter space, $\Theta^{-1}(B)$ decorrelates X into white noise, and we say that X is invertible. As we will see, invertibility is useful for statistical purposes, so it is common to take $|\theta| < 1$.

Definition 23. A time series $X := (X_t)_{t\in\mathbb{Z}} \sim \mathrm{MA}(q)$ is invertible if, for some $(\pi_j)_{j\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}}$ such that $\sum_{j\in\mathbb{N}} |\pi_j| < \infty$,

$$\varepsilon_t = \sum_{j=0}^{\infty} \pi_j X_{t-j}.$$

Proposition 5. *A time series $X := (X_t)_{t \in \mathbb{Z}} \sim MA(q)$ is invertible if and only if $\Theta(z) \neq 0$ for all $z \in \mathbb{C}$ such that $|z| \leq 1$.*

Definition 24. A time series $X := (X_t)_{t \in \mathbb{Z}}$ is q-correlated if X is stationary and $\gamma_X(h) = 0$ for all $h > q$.

Proposition 6. *If a time series $X := (X_t)_{t \in \mathbb{Z}}$ is q-correlated, then $X \sim MA(q)$.*

3.1.3 Autoregressive (integrated) moving average models

We can combine AR and MA models to form autoregressive moving average models.

Definition 25. A time series $X := (X_t)_{t \in \mathbb{Z}}$ follows an autoregressive moving average model of orders p and q, denoted by $X \sim \mathrm{ARMA}(p, q)$, if it is stationary and it satisfies

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}, \quad t \in \mathbb{Z}, \tag{3.3}$$

for some $\phi_1, \ldots, \phi_p, \theta_1, \ldots, \theta_q \in \mathbb{R}, \phi_p, \theta_q \neq 0$, and $\varepsilon := (\varepsilon_t)_{t \in \mathbb{Z}} \sim \mathrm{WN}(0, \sigma^2), \sigma^2 > 0$, where, without loss of generality, we assume that the resulting AR and MA operators Φ and Θ have no common roots over $\mathbb{C}$.

We remark that Definitions 21 and 23, together with Propositions 3, 4 and 5, apply to ARMA models without modifications.

So far, we have focused on stationary processes, whilst many real-world time series data are not stationary. Indeed, we typically preprocess the data by applying certain transformations to it until the resulting time series is stationary. The most common transformations seek to remove global and/or seasonal trends. In addition to specific transformations for each type of trend (see practical), a popular one that typically results in stationarity is *differencing*. It is performed by applying the *differencing operator* $\Delta := 1 - B$ and its powers Δ^d to each time coordinate of the time series at hand. Note that differencing may also convert trend-less non-stationary processes into stationary: the prototypical example is the random walk process $X_t = X_{t-1} + \varepsilon_t$, satisfying (3.1) with $p = 1$ and $\phi_1 = 1$. Differencing gives rise to autoregressive integrated moving average models, which encompass a wide range of non-stationary processes.

Definition 26. A time series $X := (X_t)_{t \in \mathbb{Z}}$ follows an autoregressive integrated moving average model of orders p, d and q, denoted by $X \sim \mathrm{ARIMA}(p, d, q)$, if $(\Delta^d X_t)_{t \in \mathbb{Z}} \sim \mathrm{ARMA}(p, q)$

Bibliography

Agresti, A. (2015) *Foundations of Linear and Generalized Linear Models.* Wiley.

Akaike, H. (1973) Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, eds Petrov, B. N. and Cáski, F., pp. 267–281. Akademiai Kaidó, Budapest. Reprinted in *Selected Papers of Hirotugu Akaike*, eds Parzen, E., Kitagawa, G. and Tanabe, K. (1998), pp. 199–213. Springer, New York.

Barron, A. R. (1993) Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. Inform. Theory*, **39**, 930–945.

Barron, A. R. (1994) Approximation and estimation bounds for artificial neural networks. *Machine learning*, **14**(1), 115–133.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992) A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 144–152.

Breiman, L. (1996) Bagging predictors. *Machine Learning*, **24**, 123–140.

Brockwell, P.J. and Davis, R.A. (1996) *Introduction to time series and forecasting.* Springer.

Casella G. and Berger, R.L. (2001) *Statistical Inference.* Duxbury.

Corbeil, R. R. and Searle, S. R. (1976) Restricted maximum likelihood (REML) estimation of variance components in the mixed model. *Technometrics*, **18**, 31–38.

Cortes, C. and Vapnik, V. (1995) Support-vector networks. *Machine learning*, **20**, 273–297.

Cover, T., and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Trans. Inform. Theory*, **13**, 21–27.

Dobson, A. J. and Barnett A. (2008) *An Introduction to Generalized Linear Models.* Third edition. Chapman & Hall/CRC.

Efron, B. and Tibshirani, R. J. (1994) *An Introduction to the Bootstrap*. CRC Press, Boca Raton.

Hall, P., Park, B. U. and Samworth, R. J. (2008) Choice of neighbor order in nearest-neighbor classification. *Ann. Statist.*, **36**, 2135–2152.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning.* Second Edition. Springer.

Jørgensen, B. (1987). Exponential dispersion models (with discussion). *J. Roy. Statist. Soc., Ser. B*, **49**, 127–162.

Lawless, J. F. (1987) Negative binomial and mixed Poisson regression. *Canad. J. Statist.*, **15**, 209–225.

Lelong, J. (2005) A central limit theorem for Robbins Monro algorithms with projections. *Technical report.*

McCullagh, P. (1983) Quasi-likelihood functions. *Ann. Statist.*, **11**, 59–67.

Nelson, M. (2017) *Neural Networks and Deep Learning.* Available at http://neuralnetworksanddeeplearning.com/index.html.

Nesterov, Yu. (2013) *Introductory Lectures on Convex Optimization: A Basic Course.* Springer Science and Business Media.

Samworth, R. J. (2012) Optimal weighted nearest neighbour classifiers. *Ann. Statist.*, **40**, 2733–2763

Time Series Analysis and Its Applications: with R Examples. Springer Science & Business Media.

Schwarz, G. (1978) Estimating the dimension of a model. *Ann. Statist.*, **6**, 461–464.

Shao, J. (1993) Linear model selection by cross-validation. *J. Amer. Statist. Assoc.*, **88**, 486–494.

Wedderburn, R. W. (1974) Quasi-likelihood functions, generalized linear models, and the Gauss–Newton method. *Biometrika*, **61**, 439–447.

Wu, C. J. (1983) On the convergence properties of the EM algorithm. *Ann. Statist.*, 95–103.