

STATISTICAL LEARNING

Recall $\rightarrow$ estimator $\hat{\theta}$ of θ unbiased iff $E(\hat{\theta}) = \theta$ ($\text{bias}(\hat{\theta}) = E(\hat{\theta}) - \theta$)

$\hookrightarrow \text{var}(\hat{\theta}) = E[(\hat{\theta} - E(\hat{\theta}))^2]$, $\text{MSE}(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]$. $\text{MSE} = \text{var} + \text{bias}^2$

Defⁿ: degree of freedom $\rightarrow$ number of pieces of information we have to estimate population value.

centering data: $x_{ij}^* = \frac{x_{ij} - \bar{x}_j}{s_j}$ $\rightarrow$ N/B: centre column since
 $s_j \leftarrow$ in sample std deviation. $X_{n \times p} = \begin{pmatrix} 1 & 2 & \dots & p \\ \vdots & & & \end{pmatrix}$ attributes / (features)

For data matrices X, Y ; sample mean $= \bar{X} = \frac{1}{n} 1^T X$,

sample variance $= \text{var}(X) = \frac{1}{n} X^T X$, sample covariance $= \text{cov}(X, Y) = \frac{1}{n} X^T Y$.

SIMPLE LINEAR MODEL $\rightarrow Y_i = a + b X_i + \varepsilon_i$ (ε_i 's iid $\sim N(0, \sigma^2)$)

$\hookrightarrow \varepsilon_i$'s never observed but believed to exist. Assume data $(x_i, y_i)_{i=1}^n$ observed

$\rightarrow$ set of residuals: $e_i(a, b) = y_i - a - b x_i$ ($i=1, \dots, n$).

$\text{RSS} = \frac{1}{n} \sum_{i=1}^n e_i^2(a, b)$. least squares estimator $(\tilde{a}, \tilde{b})$ of (a, b) found by minimising RSS over $(a, b) \in \mathbb{R}^2$.

$\hookrightarrow$ differentiate wrt (a, b) and set derivative $= 0$. $\nearrow$

check residuals after fitting model $\hat{e}_i = \hat{y}_i - y_i$ (where $\hat{y}_i = \tilde{a} + \tilde{b} x_i$)

(Cook's distance $\rightarrow$ how much fit changes after omitting a point)

• If lack of fit $\rightarrow$ transform data; remove outliers; different model.

If errors ε_i 's distributed w/ density $f \rightarrow L(a, b | D) = f(e | (a, b))$

$\hookrightarrow \text{MLE}(\hat{a}, \hat{b})$ found by maximising likelihood $L(a, b | D)$. $(D = \text{data})$ $= \prod_{i=1}^n f(e_i(a, b) | (a, b))$

N/B: when $\varepsilon_i \sim N(0, \sigma^2) \rightarrow$ MLE equivalent to least squares estimator.

If data skewed (use QQ plot for residuals) $\rightarrow$ student's t-dist maybe more appropriate.

Simple linear model invalidated when errors CORRELATED i.e. $\text{cov}(\varepsilon_i, \varepsilon_j) \neq 0$.

MULTIVARIATE LINEAR MODEL $\rightarrow Y_i = X_{i1} \beta_1 + X_{i2} \beta_2 + \dots + X_{ip} \beta_p + \varepsilon_i$

$\therefore$ with $Y = (y_1, \dots, y_n)^T$, $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^T$
 $\beta = (\beta_1, \dots, \beta_p)^T \Rightarrow \underline{Y = X\beta + \varepsilon}$

$$RSS(\beta) = e^T e \quad \text{where } e = Y - X\beta.$$

least squares estimator is $\hat{\beta} = \underset{\beta}{\operatorname{argmin}} RSS(\beta)$.

$$RSS(\beta) = (Y - X\beta)^T (Y - X\beta) = Y^T Y - 2\beta^T X^T Y + \beta^T X^T X \beta.$$

$$\frac{\partial RSS(\beta)}{\partial \beta} = -2X^T Y + 2X^T X \beta = 0 \quad \text{when } \hat{\beta} = (X^T X)^{-1} X^T Y. \quad \left(\begin{array}{l} \text{n/b: } (X^T X)^{-1} \\ \text{exists if} \\ X \text{ full rank} \end{array} \right)$$

$$\text{n/b: } \frac{\partial^2 RSS(\beta)}{\partial \beta} = 2X^T X \rightarrow \text{positive definite} \Rightarrow \text{we have } \hat{\beta} \text{ minimum.}$$

hat matrix $P = X(X^T X)^{-1} X^T$ so fitted values are $\hat{Y} = X\hat{\beta} = PY$.

$$E(Y) = X\beta, \quad \text{var}(Y) = E(\varepsilon \varepsilon^T) = \sigma^2 I_n, \quad E(\hat{\beta}) = \beta, \quad \text{var}(\hat{\beta}) = \sigma^2 (X^T X)^{-1}.$$

If $\varepsilon_i \sim N(0, \sigma^2)$ then since $\hat{\beta}$ linear comb. of normals so $\hat{\beta}$ also normal.

$$\text{var}(\hat{\beta}_j) = \sigma^2 v_j \quad \text{where } v_j \text{ is } j\text{-th diagonal element of } (X^T X)^{-1}. \quad \rightarrow \text{estimator for } \sigma^2.$$

$$\Rightarrow \text{standardised coefficient } z_j = \frac{\hat{\beta}_j}{\hat{\sigma} \sqrt{v_j}} \quad \text{where } \hat{\sigma}^2 = \frac{1}{n-p} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

(p-value is the probability of getting a more extreme value than z_j .)

$\hookrightarrow$ if $\beta_j = 0$ then $z_j \sim t_{n-p}$ so can compute a p-value on z_j to test whether $\beta_j = 0$.

comparing models M_0 (p_0+1 variables) and M_1 (p_1+1 variables) with F statistic:

$$F = \frac{(RSS_0 - RSS_1) / (p_1 - p_0)}{RSS_1 / (n - p_1 - 1)} \rightarrow \text{if } M_0 \text{ true then } F \sim F_{p_1 - p_0, n - p_1 - 1}$$

GAUSS-MARKOV THM $\rightarrow$ LSE $\hat{\beta}$ is the B.L.U.E. (i.e. it has smallest variance).

BUT LSE $\hat{\beta}$ not necessarily best estimator in terms of MSE $\rightarrow$ may exist estimators which are little biased, but better variance $\Rightarrow$ so smaller MSE.

so try modifying coefficients A.K.A. shrinkage.

Pf of Gauss-Markov: if $\tilde{\beta} = CY$ another unbiased estimator define difference

$$D := C - (X^T X)^{-1} X^T \quad \text{so } \tilde{\beta} = \hat{\beta} + DY. \quad \text{Then } E(\tilde{\beta}) = (I_p + DX)\beta \quad \text{so } DX = 0.$$

$$\text{var}(\tilde{\beta}) = \text{var}(\hat{\beta}) + \sigma^2 DD^T \geq \text{var}(\hat{\beta}). \quad \square$$

(Similarly if $\theta = \alpha^T \beta$ (any linear comb of β), $\hat{\theta} = \alpha^T \hat{\beta}$, $\tilde{\theta} = \alpha^T \tilde{\beta}$ both unbiased and:

$$\text{var}(\alpha^T \tilde{\beta}) = \text{var}(\alpha^T \hat{\beta}) + \sigma^2 \alpha^T DD^T \alpha \geq \text{var}(\alpha^T \hat{\beta})$$

RIDGE REGRESSION $\rightarrow \hat{\beta}^{\text{ridge}}(\lambda) = \underset{\beta}{\operatorname{argmin}} \left\{ \underbrace{\sum_{i=1}^n (Y_i - \beta_0 - \sum_{j=1}^p X_{ij} \beta_j)^2}_{\text{OLS}} + \lambda \sum_{j=1}^p \beta_j^2 \right\}$
 (generalised least squares w/ $\lambda > 0$ chosen).

N/B: λ controls balance between accuracy of fit and size of coefficients.

$\hookrightarrow \hat{\beta}^{\text{ridge}}(0) = \hat{\beta}$, $\hat{\beta}^{\text{ridge}}(\lambda) \rightarrow \bar{Y}$ for large λ . (i.e. $\hat{\beta}_0 \rightarrow \bar{Y}$, $\hat{\beta}_j \rightarrow 0$ for $j \neq 0$)

Alternate formulation: $\hat{\beta}^{\text{ridge}} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n (Y_i - \beta_0 - \sum_{j=1}^p X_{ij} \beta_j)^2$,
 subject to $\sum_{j=1}^p \beta_j^2 \leq t$.

$\text{RSS}(\beta) = (Y - X\beta)^T (Y - X\beta) + \lambda \beta^T \beta$ and so $\hat{\beta}^{\text{ridge}} = (X^T X + \lambda I_p)^{-1} X^T Y$.

Condition number $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$ $\rightarrow$ measure how "hard" it is to invert a matrix
 $\searrow$ eigenvalue.

so here adding λI_p to $X^T X$ makes matrix "more" invertible.

SVD for SHRINKAGE consider SVD for $X_{\text{exp}} = U D V^T$ (U, V orthogonal).

for least squares: $X \hat{\beta} = X (X^T X)^{-1} X^T Y = U \cdot U^T Y$

for ridge: $X \hat{\beta}^{\text{ridge}} = U D (D^2 + \lambda I_p)^{-1} D U^T Y = \sum_{j=1}^p \underline{u}_j \frac{d_j^2}{d_j^2 + \lambda} \underline{u}_j^T Y$ ($\underline{u}_j$ are columns of U)

so ridge regression shrinks according to $\frac{d_j^2}{d_j^2 + \lambda}$. If d_j^2 large, $\frac{d_j^2}{d_j^2 + \lambda} \approx 1$;
 if d_j^2 small, ratio is < 1 .

$\Rightarrow$ ridge shrinks coordinates corresponding to small d_j^2 (i.e. directions where singular value small)

PRINCIPAL COMPONENTS: eigenvectors of $X^T X$ are ~~principal components~~ ^{we project data onto}, i.e.

if $X = U D V^T$ then PCs ~~equivalent~~ ^{project onto} to columns of V , v_j .

we call $z_j = X v_j \rightarrow$ projection of data matrix onto v_j . N/B: the z_j are called **PRINCIPAL COMPONENTS**.

$\frac{z_j^T z_j}{n} = \frac{1}{n} v_j^T X^T X v_j = \frac{1}{n} v_j^T V D^2 V^T v_j = \frac{d_j^2}{n}$ } so variance of X in direction v_j is d_j^2/n .

$\Rightarrow$ ridge regression shrinks coordinates in directions w/ smaller variance.

note $\text{var}(\hat{\beta}^{\text{ridge}}) < \text{var}(\hat{\beta})$ due to the constraint.

LASSO REGRESSION $\rightarrow \hat{\beta}^{\text{lasso}}(\lambda) = \underset{\beta}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^n (Y_i - \beta_0 - \sum_{j=1}^p X_{ij} \beta_j)^2 + \lambda \sum_{j=1}^p |\beta_j| \right\}$

(again, generalised least square w/ $\lambda > 0$ chosen).

again, $\hat{\beta}^{\text{lasso}}(0) = \hat{\beta}$ and $\hat{\beta}^{\text{lasso}}(\lambda) \rightarrow \bar{Y}$ for λ large (and $\hat{\beta}_j^{\text{lasso}} \rightarrow 0$ for $j \neq 0$)

Alternate formulation: $\hat{\beta}^{\text{lasso}} = \arg \min_{\beta} \sum_{i=1}^n (Y_i - \beta_0 - \sum_{j=1}^p X_{ij} \beta_j)^2$
subject to $\sum_{j=1}^p |\beta_j| \leq t$.

No closed form solution exists for $\hat{\beta}^{\text{lasso}}(\lambda)$.

LASSO vs. RIDGE $\rightarrow$ penalties $\sum \beta_j^2$ vs. $\sum |\beta_j|$ so e.g. if β_j small β_j^2 tiny.
So Lasso has stronger shrinkage power (shrink more) and some coefficients even shrink to 0 w/ Lasso (i.e. variable removed).

closed form solⁿs exist when X orthogonal:

$$\hat{\beta}^{\text{LS}} = (X^T X)^{-1} X^T Y = X^T Y ; \quad \hat{\beta}^{\text{ridge}} = (I + \lambda I)^{-1} X^T Y = \frac{\hat{\beta}^{\text{LS}}}{1 + \lambda}$$

for Lasso want: $\min_{\beta} \frac{1}{2} (Y - X\beta)^T (Y - X\beta) + \lambda \sum_{j=1}^p |\beta_j| = \min_{\beta} \sum_{j=1}^p \underbrace{\left(-\hat{\beta}_j^{\text{LS}} \beta_j + \frac{1}{2} \beta_j^2 + \lambda |\beta_j| \right)}_{\text{call } M_j}$

i.e. want $\frac{\partial M_j}{\partial \beta_j} = 0 \Rightarrow \hat{\beta}^{\text{lasso}} = \text{sign}(\hat{\beta}^{\text{LS}}) \cdot (|\hat{\beta}^{\text{LS}}| - \lambda)^+$

PRINCIPAL COMPONENTS REGRESSION: let $z_m = X v_m$, $m=1, \dots, p$ (v_m columns of V)

(recall PCs present the data according to a new set of variables $z_1, \dots, z_p$ where the variance of data according to z_1 is greatest and then z_2 etc. (since eigenvalues $d_1 > d_2 > \dots$))

i.e. most information (=variance) in the data set put into z_1 ;

next largest variance put into z_2 (in a direction orthogonal to v_1) etc. etc.

N/B: the variances are just the eigenvalues.

• Pick M s.t. the first M PCs are meaningful (i.e. first M eigenvalues large)

PC regression is: $\hat{y}^{\text{PCR}} = \bar{Y} 1_n + X \sum_{m=1}^M \hat{\theta}_m v_m$ ($= \bar{Y} 1_n + \sum_{m=1}^M \hat{\theta}_m z_m$)

where regression coefficient $\rightarrow \hat{\theta}_m = \frac{\langle z_m, y \rangle}{\langle z_m, z_m \rangle}$.

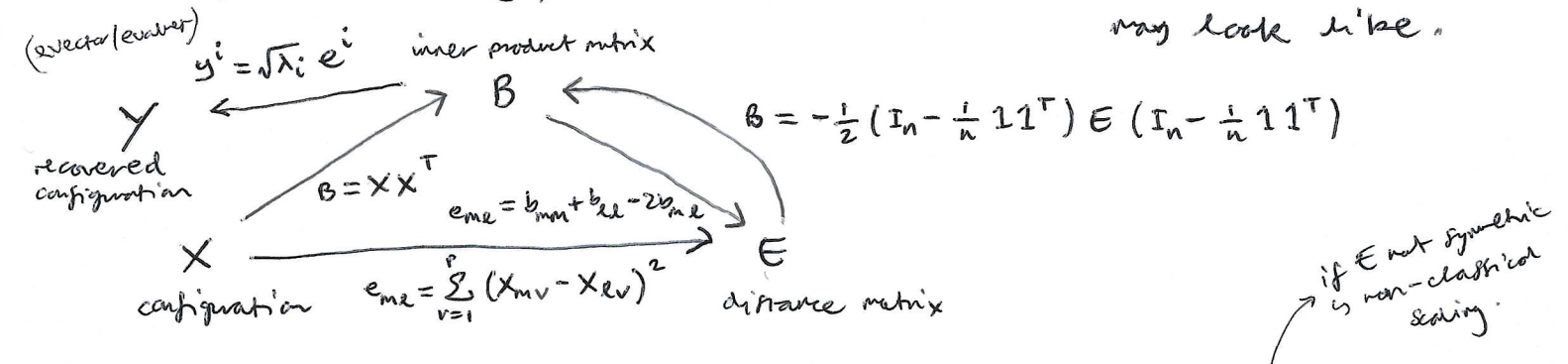
So $\hat{y} = \bar{y} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} + \sum_{m=1}^M \frac{\langle y, z_m \rangle}{\langle z_m, z_m \rangle} z_m$

N/B: setting $\hat{\beta}^{\text{PCR}} = \sum_{m=1}^M \hat{\theta}_m v_m \Rightarrow \hat{y}^{\text{PCR}} = \bar{Y} 1_n + X \hat{\beta}^{\text{PCR}}$.

N/B: choosing $M=p$ restores least squares regression.

DISTANCE-BASED METHODS - CLASSICAL SCALING

if we do not have access to data matrix $X_{n \times p}$. We have information on distance (or dissimilarity) between individuals + wish to recover what X may look like.



For classical scaling $\rightarrow$ Euclidean distances (symmetric), E .

From X : $e_{ml} = \sum_{v=1}^p (X_{mv} - X_{lv})^2 = X_m^T X_m + X_l^T X_l - 2X_m^T X_l$
 $\uparrow$ m -th row of X .

using $B = XX^T \Rightarrow e_{ml} = b_{mm} + b_{ll} - 2b_{ml}$.

N/B: in going from X to B , lose orientation information (so X and a rotated version Y have same inner product matrix)
 $\hookrightarrow$ so can operate equivalently on centered data matrix X .

Further, going from B to E lose position information (so $X_{(m)}$ and $X_{(m)} - \mu$ have same E matrix)

RECOVERING B from E : $\rightarrow$ need to choose an origin so put $\bar{X}_{px1} = \frac{1}{n} 1_n^T X$ at the origin.

$\Rightarrow$ condition: $\bar{X} = 0$ i.e. $1_n^T X = 0$. ($\Leftrightarrow 1_n^T \underbrace{XX^T}_B = 0$ so 1_n vector of B)

using $e_{ml} = b_{mm} + b_{ll} - 2b_{ml} \rightarrow b_{ml} = -\frac{1}{2} (e_{ml} - \frac{e_{m \cdot}}{n} - \frac{e_{\cdot l}}{n} + \frac{e_{\cdot \cdot}}{n^2})$
 (row avg) (col avg) (grand avg)

$\Rightarrow B = -\frac{1}{2} (I_n - \frac{1}{n} 11^T) E (I_n - \frac{1}{n} 11^T)$

RECOVERING X from B : $B = XX^T$ (+ve semi-definite) so use eigendecomposition:

$B = \sum_{i=1}^k \lambda_i e^{(i)} e^{(i)T}$ for eigenvector of B $e^{(i)}$ and eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \lambda_k > 0$ and $\lambda_{k+1} = \dots = \lambda_n = 0$.
 $\uparrow$ Spectral decomp since B symmetric

then define vectors $f^{(i)} = \sqrt{\lambda_i} e^{(i)}$ ($i=1, \dots, k$)

$\Rightarrow Y_{n \times k} = (f^{(1)} f^{(2)} \dots f^{(k)})$ is recovered " X " and have $YY^T = B$.

N/B: if eigenvalue of B negative $\rightarrow$ suggests non-Euclidean distance.
 For k , only take largest few eigenvalues $\rightarrow$ this also indicates underlying dimension of data.

DISSIMILARITIES → not always Euclidean configurations e.g. we may have attributes instead.
 e.g. flower color $\in \{\text{red, blue, yellow}\}$, petal number $\in \{0, 1, 2, \dots\}$ etc.

HAMMING DISTANCE → number of disagreements of attributes.

e.g. $x: 010010011$
 $y: 101011010$

count number of times each object has an attribute.

		Attributes of x	
Attributes of y	yes	a	b
	no	c	d

• Hamming distance = $b + c$

• Jaccard distance = $\frac{b+c}{a+b+c}$

• Simple matching coefficient = $\frac{a+d}{a+b+c+d}$

Require a dissimilarity d to satisfy defⁿ of a metric $d(x, y)$ → to be a good dissimilarity measure.

1) $d(x, y) \geq 0$ and $d(x, y) = 0$ if $x = y$. (not if + only if)

2) $d(x, y) = d(y, x)$.

3) $d(x, y) + d(y, z) \geq d(x, z)$.

NON-METRIC / ORDINAL SCALING → given dissimilarities $\{s_{me}\}$ try to

find good configuration so that Euclidean distances $\{d_{me}\}$ close to $\{s_{me}\}$.

- we can compute "stress funcⁿ" that measures degree of agreement of $\{s_{me}\}$ w/ $\{d_{me}\}$.

- we then minimise stress over all possible configurations.

We create a set of fitted distances $\hat{d}_{me}$ from the d_{me} by computing a MONOTONE LINEAR REGRESSION fit, where d_{me} are in same 'order' as ordered dissimilarities s_{me} (e.g. monotone increasing).

Given observations $\{d_{me}\}_{e=1}^2 \rightarrow$ task is to minimise $\sum_{i=1}^2 (d_{me} - \hat{d}_{me})^2$.

→ want to find the best $\{\hat{d}_{me}\}$.

STRESS FUNCTION: Given configuration and set of dissimilarities $\{s_{me}\}$, we can form distances $\{d_{me}\}$ and the least squares monotone regression fit, $\{\hat{d}_{me}\}$, to the d using s 's.

Let $S^* = \sum_{m \in L} (d_{me} - \hat{d}_{me})^2$; $T^* = \sum_{m \in L} d_{me}^2$.

→ RSS from the least squares monotone regression.

STRESS FUNCTION $\rightarrow S(X) = \sqrt{S^*/T^*}$ (S depends on X since S^* and T^* do.)

$$\text{where } S^* = \sum_{m \in \mathcal{L}} (d_{m\ell} - \hat{d}_{m\ell})^2, \quad T^* = \sum_{m \in \mathcal{L}} d_{m\ell}^2.$$

so we 'create' a configuration X i.e. coordinates for n points in K dimensions.

but must choose $K \rightarrow \underline{K=1}$ doesn't work, but $K=2, 3$ typical.

since S always gets stuck in local optima here.

Differentiation of stress $\rightarrow \frac{\partial S}{\partial x_{ik}} = \frac{1}{2S} \left\{ \frac{T^* \frac{\partial S}{\partial x_{ik}} - S^* \frac{\partial T^*}{\partial x_{ik}}}{(T^*)^2} \right\}$

$\downarrow$
 $= \frac{S}{2} \left[\frac{1}{S^*} \frac{\partial S^*}{\partial x_{ik}} - \frac{1}{T^*} \frac{\partial T^*}{\partial x_{ik}} \right]$

$\therefore$ we need to compute $\frac{\partial T^*}{\partial x_{ik}}, \frac{\partial S^*}{\partial x_{ik}}$ using defⁿ of S^*, T^* , and further

$\frac{\partial \hat{d}_{m\ell}}{\partial x_{ik}} = 0$ since $\hat{d}$ piece-wise constant (by design).

$\Rightarrow S$ exactly once differentiable only.

K-MEANS CLUSTERING, given configuration $X_{n \times p} \rightarrow$ goal is to divide n observations into K clusters.

NB: a K -means clustering solⁿ is one where the K cluster means are defined as being the mean of all points in the cluster.

A point is in cluster $j \in \{1, \dots, K\}$ iff it is closer to the j -th cluster mean than any other cluster mean.

ALGORITHM $\rightarrow$ start with K mean vectors: $\{m_1^{(1)}, \dots, m_K^{(1)}\}$ (p -dimensional)

then at iteration $(s) \rightarrow$ assign each observation x_i to the closest mean vector (by Euclidean distance):

$$C_j^{(s)} := \{x_i : \|x_i - m_j^{(s)}\|^2 \leq \|x_i - m_r^{(s)}\|^2 \quad \forall r=1, \dots, K\}$$

These C_j are K clusters for $j \in \{1, \dots, K\}$.

then update means from observations in the new clusters:

$$m_j^{(s+1)} = |C_j^{(s)}|^{-1} \sum_{x_i \in C_j^{(s)}} x_i$$

then iterate until convergence (i.e. when clusters remain the same).

SELF-ORGANISING MAPS $\rightarrow$ target is same as K-means. SOM starts w/ a rectangular grid of K prototypes (akin to cluster means or representing) the cluster values
 is prototypes parameterised by ^{integer} coordinate pair $m_j \in Q_1 \times Q_2$ w/ $Q_1 = \{1, \dots, q_1\}$
 and so $K = q_1 q_2$.

ALGORITHM $\rightarrow$ Iterate through data X_i and for each data point X_i ,
 find closest ~~map~~ prototype, call it m_j .
 Find the neighbours of m_j , say m_k , and pull $m_j + m_k$'s towards X_i via:
 updating $m_k \mapsto m_k + \alpha(X_i - m_k)$ (for learning rate α).

SUMMARY $\rightarrow$ scaling turns distances / dissimilarities into 'recovered' configurations.
 2 types of scaling $\rightarrow$ classical + ordinal.
 K-means + SOM work on configurations, but use distance information to form groups.

PROCRUSTES ANALYSIS $\rightarrow$ assess performance of scaling methods! And compare
 2 configurations $(X, Y)_{n \times K}$ produced by scaling. $\rightarrow$ we can move Y to be as close
 $\rightarrow$ in order to compare scaling methods to X and then see how close the
 resemblance is.
 CLOSENESS: $Q(X, Y) = \sum_{k=1}^K \sum_{i=1}^n (X_{ik} - Y_{ik})^2$. Goal to minimise Q under
 translation, then under rotation / reflection, then under scale change.

SPLINE MODEL (in linear regression $f(X) = E(Y|X) = a + bX$)
 Linear basis expansion $\rightarrow f(X) = \sum_{m=1}^M \beta_m h_m(X)$ (where $h_m: \mathbb{R}^p \rightarrow \mathbb{R}$)
 $\{h_m\}$ is a fixed set of functions, i.e. transformations of X . (so model still linear in $\{\beta_m\}$)
 [e.g. $h_m(X) = X_{mj}^2$ or $h_m(X) = X_j X_k$ etc. ($m=1, \dots, p$)]
 BUT control complexity! e.g. limit to form $f(X) = \sum_{j=1}^p \sum_{m=1}^N \beta_{jm} h_{jm}(X_j)$
 for fixed N so number of basis functions limited.

PIECEWISE CONSTANT POLYNOMIAL for knots / regions $\{\mathcal{E}_k\}_{k=1, \dots, K}$ with each
 knot being interval $[\xi_{i-1}, \xi_i)$.
 Then we choose $h_i(X) = \mathbb{I}(\xi_{i-1} \leq X < \xi_i)$. e.g. if $f(X) = \sum_{m=1}^K \beta_m h_m(X)$
 then $\hat{\beta} = \bar{Y}_m$ (mean of data in region m)

PIECEWISE LINEAR → use the same basis functions as piecewise constant but also add $\tilde{h}_i = \mathbb{I}(\varepsilon_{i-1} \leq X < \varepsilon_i) \cdot X \quad (\forall i=1, \dots, K)$.

CONTINUOUS PIECEWISE LINEAR → for continuity require constraint that

$f(\varepsilon_i^-) = f(\varepsilon_i^+)$ so obtain basis functions: $h_1 = 1, h_2 = X,$

(CONTINUOUS)

$$h_{2+i} = (X - \varepsilon_i)^+ \quad (\forall i=1, \dots, K)$$

PIECEWISE CUBIC → $h_1 = 1, h_2 = X, h_3 = X^2, h_4 = X^3,$

$$h_{4+i} = (X - \varepsilon_i)^3_+$$

e.g. consider 3 regions so 2 knots $\varepsilon_1, \varepsilon_2$. Each cubic $a + bx + cx^2 + dx^3$ has 4 parameters and 3 regions so $4 \times 3 = 12$ basis functions. But at each knot constraints on matching values to require continuity at 0th, 1st, 2nd derivative $\Rightarrow 2 \times 3 = 6$ d.o.f. taken away $\Rightarrow 12 - 6 = 6$ basis functions.

NATURAL CUBIC SPLINE → 2 end regions are linear (so releases 4 d.o.f.s)

$$N_1(X) = 1, N_2(X) = X, N_{2+K}(X) = d_K(X) - d_{K-1}(X)$$

$$\text{where } d_K(X) = \frac{(X - \varepsilon_K)^3_+ - (X - \varepsilon_{K-1})^3_+}{\varepsilon_K - \varepsilon_{K-1}} \quad \text{for } K=1, \dots, K-2 \quad \left. \vphantom{\frac{(X - \varepsilon_K)^3_+ - (X - \varepsilon_{K-1})^3_+}{\varepsilon_K - \varepsilon_{K-1}}} \right\} K \text{ knots}$$

SMOOTHING SPLINES (no need to select knots) → given data $(x_i, y_i)_{i=1}^n$

goal is to find $f \in \mathcal{F}$ (set of all functions w/ 2 cont. derivatives) that

minimises following (penalised) RSS: $RSS(f, \lambda) = \sum_{i=1}^n (y_i - f(x_i))^2 + \lambda \int f''(t)^2 dt$

where λ is a SMOOTHING PARAMETER so $\int f''(t)^2 dt$ penalises curvature.

2 special cases: $\lambda = 0$, $f \in \mathcal{F}$ that interpolates data.

$\lambda = \infty$, recover simple least squares line fit (no second derivative / curvature allowed).

N/B: if $\lambda = 0$ may not fully interpolate e.g. if there is more than one y_i for an x_i then interpolates their mean.

It can be shown, sol^n to minimise above RSS is a natural cubic spline.

Hence we may write, $f(x) = \sum_{j=1}^n \underbrace{N_j(x)}_{\text{basis function}} \theta_j$

$$\Rightarrow RSS(\theta, \lambda) = (y - N\theta)^T (y - N\theta) + \lambda \theta^T \Omega_n \theta \quad \text{where: } N_{ij} = N_j(x_i)$$

$$\therefore \hat{\theta} = (N^T N + \lambda \Omega_n)^{-1} N^T y \quad (\Omega_n)_{ij} = \int N_i''(t) N_j''(t) dt$$

with $\hat{\theta} = (N^T N + \lambda \Sigma_n)^{-1} N^T y$ then $\hat{f} = N \hat{\theta}$

$$= N (N^T N + \lambda \Sigma_n)^{-1} N^T y$$

so $S_\lambda = N (N^T N + \lambda \Sigma_n)^{-1} N^T$; effective d.o.f.s of S_λ a smoothing spline
 is: $df_\lambda = \text{trace}(S_\lambda)$.

CHOOSING λ $\rightarrow$ 'leave one out cross-validation'. = MISE

Goal: make $\hat{f}_\lambda$ as close to $f(x)$. $\Rightarrow$ aim to minimise $E[\int (\hat{f}_\lambda(x) - f(x))^2 dx]$

estimate MISE by $CV(\hat{f}_\lambda) = \sum_i (y_i - \hat{f}_\lambda^{(-i)}(x_i))^2 = \frac{1}{n} \sum_i \left(\frac{y_i - \hat{f}_\lambda(x_i)}{1 - (S_\lambda)_{ii}} \right)^2$

where $\hat{f}_\lambda^{(-i)}$ is smoothing spline estimator w/ i th point ignored.

• $CV(\hat{f}_\lambda)$ can be computed for each value of λ .

DENSITY ESTIMATION given set of data $x_1, \dots, x_n$ iid w/ density $f(x)$.

observe $x_1, \dots, x_n$ then wish to estimate $f(x)$.

DEFN: KERNEL FUNCTION, $K: \mathbb{R} \rightarrow \mathbb{R}$ s.t. $K(x) \geq 0 \quad \forall x$

$$\int_{\mathbb{R}} K(x) dx = 1$$

and usually we choose symmetric kernel $\rightarrow K(-x) = K(x)$.

Kernel Density Estimator (KDE): $\hat{f}_{n,h,K}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x_i - x}{h}\right)$ $\rightarrow h$ bandwidth

h small $\Rightarrow$ high variance, low bias; h large $\Rightarrow$ low variance, high bias.

e.g. rectangular kernel: $K(x) = \begin{cases} 1 & x \in (-\frac{1}{2}, \frac{1}{2}) \\ 0 & \text{otherwise} \end{cases}$ or gaussian $K(x) = \frac{1}{\sqrt{2\pi}} \exp\{-\frac{1}{2}x^2\}$.

Expectation of KDE $\rightarrow E(\hat{f}(x)) = \frac{1}{nh} \sum_{i=1}^n E\left[K\left(\frac{x_i - x}{h}\right)\right]$

$$= \frac{1}{nh} \sum_{i=1}^n \int K\left(\frac{y-x}{h}\right) f(y) dy$$

we can expand $f(x+\delta) = f(x) + \delta f'(x) + \frac{\delta^2}{2} f''(x) + \dots$ and sub $v = \frac{y-x}{h}$.

$\Rightarrow \int K\left(\frac{y-x}{h}\right) f(y) dy = \int K(v) f(x+hv) h dv$

$$= h f(x) \int K(v) dv + h^2 f'(x) \int v K(v) dv + \frac{1}{2} h^3 f''(x) \int v^2 K(v) dv$$

$\Rightarrow \text{bias}(\hat{f}(x)) \approx \frac{1}{2} h^2 \cdot \text{constant} \cdot f''(x) \rightarrow$ so depends on second derivative of f .

KERNEL REGRESSION → define $K_h(x) = \frac{1}{h} K\left(\frac{x}{h}\right)$. Now given X, Y

we want $E(Y|X) = \int y f(y|x) dy = \int \underbrace{y f(x, y)}_{f(x)} dy$.

Making use of KDE → $\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x_i - x}{h}\right) = \frac{1}{n} \sum_{i=1}^n K_h(x - x_i)$ ↖ use symmetric.

introduce joint 2D KDE → $\hat{f}(x, y) = \frac{1}{n} \sum_{i=1}^n K_h(x - x_i) K_h(y - y_i)$

so estimate $\hat{E}(Y|X) = \frac{\int y \hat{f}(x, y) dy}{\hat{f}(x)} = \frac{\sum_{i=1}^n K_h(x - x_i) \int y K_h(y - y_i) dy}{\sum_{i=1}^n K_h(x - x_i)}$

for the integral $\int y K_h(y - y_i) dy = \int (v + y_i) K(v) dv = y_i$ (since kernel symmetric + integrates to 1)

⇒ $\hat{E}(Y|X) = \frac{\sum_{i=1}^n y_i K_h(x - x_i)}{\sum_{i=1}^n K_h(x - x_i)} \rightarrow$ Nadaraya-Watson estimator for kernel regression.

LOCAL POLYNOMIAL REGRESSION → this generalises Nadaraya-Watson.

define local polynomial estimator $m_{x_0}(x)$ centred on x_0 $= \sum_{j=0}^p \underbrace{\beta_j(x_0)}_{\text{regression coefficients}} (x - x_0)^j$

Then apply a weighted (by kernel) RSS minimiser → $RSS(x_0) = \sum_{i=1}^n (y_i - m_{x_0}(x_i))^2 K_h(x_i - x_0)$

↳ can be written: $RSS(x_0) = (Y - X \beta(x_0))^T W_{x_0} (Y - X \beta(x_0))$

(where $W_{x_0} = \text{diag}\{K_h(x_1 - x_0), \dots, K_h(x_n - x_0)\}$; $X = \begin{pmatrix} | & x_1 - x_0 & \dots & (x_1 - x_0)^p \\ & \vdots & & \vdots \\ & x_n - x_0 & \dots & (x_n - x_0)^p \end{pmatrix}$).

↳ minimiser: $\hat{\beta}(x_0) = (X^T W_{x_0} X)^{-1} X^T W_{x_0} Y$.

ORTHOGONAL SERIES EXPANSIONS → $\{p_v(x)\}$ is orthogonal series basis for funⁿ space.

$f(x) = \sum_v f_v p_v(x)$, $f_v = \langle f, p_v \rangle = \int f(x) p_v(x) dx$

Treating $f(x)$ as a density, $f_v = \int p_v(x) f(x) dx = E[p_v(x)] = \frac{1}{n} \sum_{i=1}^n p_v(x_i)$ ↖ $\hat{f}_v$
we can use $\{\hat{f}_v\}$ as we to estimate $f(x)$.

For $f(x, y)$ use: $f_{v, \mu} = \iint f(x, y) p_v(x) p_\mu(y) dx dy = E[p_v(x) p_\mu(y)]$ ↖ $\hat{f}_{v, \mu}$
 $= \frac{1}{n} \sum_{i=1}^n p_v(x_i) p_\mu(y_i)$

BOTH ABOVE UNBIASED for v , but often $v \in \{-\infty, \dots, \infty\}$ so TRUNCATE!!!

so in practice use (TRUNCATED) linear orthogonal series estimator: $\hat{f}(x) = \sum_{v=-m}^m \hat{f}_v p_v(x)$
 N/B: generally run-resolving for truncated.

FOURIER BASIS EXAMPLE: consider Fourier transform coefficients for KDE:

$$R_n(\omega) = \int_{\mathbb{R}} \underbrace{\hat{f}_n(x)}_{\text{KDE}} \exp\{-2\pi i \omega x\} = \frac{1}{nh} \int_{\mathbb{R}} \sum_{j=1}^n K\left(\frac{x_j - x}{h}\right) \exp\{-2\pi i \omega x\} dx$$

it can be shown $R_n(\omega) = \hat{X}(\omega) \cdot \tilde{K}_n(\omega)$ (subbing in $y = x_j - x$)

where: $\hat{X}(\omega) = \frac{1}{n} \sum_{j=1}^n e^{-2\pi i \omega x_j}$, $\tilde{K}_n(\omega) = \int_{\mathbb{R}} K_n(y) e^{2\pi i \omega y} dy$ } so K decoupled from x_j .

$$\Rightarrow \hat{f}_n^{\text{KDE}}(x) = \int_{\mathbb{R}} \underbrace{\hat{X}(\omega)}_{R_n(\omega)} \underbrace{\tilde{K}_n(\omega)}_{\text{inverse FT}} e^{2\pi i \omega x} d\omega \approx \sum_{q=-M}^M \hat{X}(\omega_q) \tilde{K}_n(\omega_q) e^{2\pi i \omega_q x}$$

WAVELETS: consider Multiresolution Analysis MRA, $V_j \subset L_2(\mathbb{R})$

where V_j is functⁿ of resolution $\leq j$ and $\{V_j\}_{j \in \mathbb{Z}}$ resolution space

$$\bullet \dots \subset V_{-1} \subset V_0 \subset V_1 \subset \dots, \bullet \bigcup_{j \in \mathbb{Z}} V_j = L_2(\mathbb{R}), \bullet \bigcap_{j \in \mathbb{Z}} V_j = \{0\}$$

(rest detail for smaller j)

• $f(x) \in V_j \Rightarrow f(2x) \in V_{j+1}$ (resolution increase)

• $f(x) \in V_0 \Rightarrow f(x-k) \in V_0$ (translation in V_0)

N/B: there exists $\phi(x) \in V_0$ s.t. $\{\phi(x-k)\}_{k \in \mathbb{Z}}$ is ORTHONORMAL BASIS for V_0 .

↳ such a ϕ is father wavelet.

HAAR FATHER WAVELET $\rightarrow \phi(x) = \begin{cases} 1 & x \in (0, 1) \\ 0 & \text{otherwise} \end{cases}$ so $\{\phi(x-k)\}_{k \in \mathbb{Z}}$ forms basis for a V_0 .

in general, $\phi_{j,k} = 2^{j/2} \phi(2^j x - k)$, then $\{\phi_{j,k}\}_{k \in \mathbb{Z}}$ basis for V_j .

projection of f onto $V_j \Rightarrow f_j = \sum_{k \in \mathbb{Z}} c_{j,k} \phi_{j,k}(x)$, where $c_{j,k} = \int f(x) \phi_{j,k}(x) dx$.

e.g. for $f(x) = \begin{cases} a & x \in (0, 1/2) \\ b & x \in (1/2, 1) \end{cases}$ then $f(x) = \frac{a}{\sqrt{2}} \phi_{1,0}(x) + \frac{b}{\sqrt{2}} \phi_{1,1}(x)$.

HAAR MOTHER WAVELET $\rightarrow \psi(x) = \begin{cases} 1 & x \in (0, 1/2) \\ -1 & x \in (1/2, 1) \\ 0 & \text{otherwise} \end{cases}$ N/B: here $j=1$ so this is $= f_1(x)$, (4)

$\therefore f_1(x) = c_{0,0} \phi(x) - d_{0,0} \psi(x)$ where $d_{0,0} = \frac{1}{\sqrt{2}} (c_{0,1} - c_{0,0})$

RELATION OF COEFFICIENTS $\rightarrow c_{0,0} = \frac{1}{\sqrt{2}} (c_{1,1} + c_{1,0})$

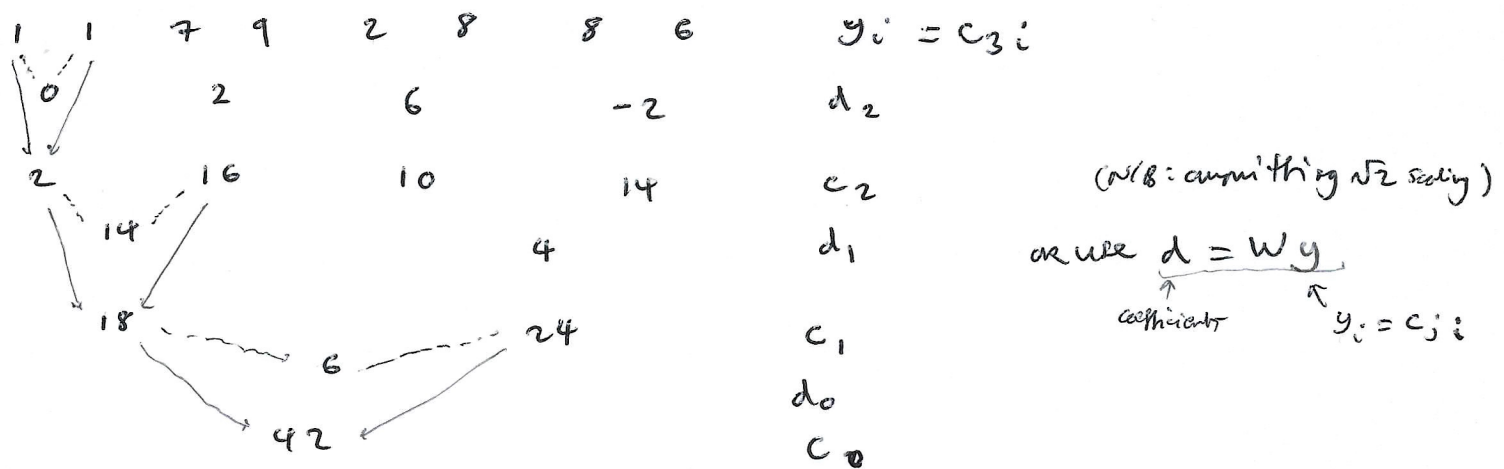
$$d_{0,0} = \frac{1}{\sqrt{2}} (c_{1,1} - c_{1,0})$$

generalized to: $c_{j-1,k} = \frac{1}{\sqrt{2}} (c_{j,2k+1} + c_{j,2k})$
 $d_{j-1,k} = \frac{1}{\sqrt{2}} (c_{j,2k+1} - c_{j,2k})$ } allows reconstruction of f solely from coefficients of highest resolution.

(since we can write $f(x) = \sum_{k \in \mathbb{Z}} c_{j_0,k} \phi_{j_0,k}(x) + \sum_{j=j_0}^{\infty} \sum_{k \in \mathbb{Z}} d_{j,k} \phi_{j,k}(x)$)

starting from primary resolution j_0

Pyramid algorithm allows one to calculate coefficients:



vanishing Moments: A wavelet has m vanishing moments if $\int x^l \phi(x) dx = 0$ for $l = 0, \dots, m-1$.

WAVELET SHRINKAGE, $\rightarrow$ assume model $y_i = f_i + \varepsilon_i$ ($i=1, \dots, n$). ($\varepsilon_i \sim N(0, \sigma^2)$)

let $e = W\varepsilon \rightarrow$ wavelet transform of noise, and $w = Wy$, $d = Wf$.

$\Rightarrow w = d + e$. N/B: d sparse since wavelet transform of sampled f is, but by Parseval's then $\|d\| = \|f\|$ so f gets squeezed into fewer locations, but still has same energy.

so signal to noise ratio improves!

to get rid of small coefficients apply thresholding to w :

$$T_{\text{hard}}(w, \lambda) = w \mathbb{I}(|w| > \lambda), \quad T_{\text{soft}}(w, \lambda) = \text{sgn}(w) (|w| - \lambda) \mathbb{I}(|w| > \lambda)$$

PROJECTION PURSUIT, recall data matrix $X_{n \times p} = UDV^T$,

recall in PCA project, $z_j = XV$ where eigenvector v_j w/ eigenvalue d_j^2 .

PCA projects into components w/ largest variance.

But large variance not always interesting!

if $\dots$ project in 2 dimensions

Let $f(x)$, $g(x)$ be 2 densities. The KL divergence of g from f

(measures difference between 2 distributions) is: $D_{KL}(g||f) = \int_{\mathbb{R}} g(x) \log\left(\frac{g(x)}{f(x)}\right) dx \geq 0$

ENTROPY of density g is: $H(g) = -\int_{\mathbb{R}} g(x) \log(g(x)) dx$ (measures randomness)

N/B: Gaussian maximises entropy i.e. if f PDF $N(0, \sigma^2)$ and g PDF w/ variance σ^2 then $H(f) \geq H(g)$.

Use $D_{KL}(g||f) \geq 0$ to prove this with $D_{KL} = -H(g) - \int_{\mathbb{R}} g(x) \log(f(x)) dx$.

SPHERING → transform a centred matrix to make its variance identity, i.e. individual variables has variance 1 and are uncorrelated.

$S = \frac{1}{n} X^T X$ (sample variance). Then $R = S^{-1/2}$, $W = XR$ is sphered data.

(so W satisfies $\frac{1}{n} W^T W = I_p$). → now variance of projection in any direction is the same.

PROJECTION PURSUIT PROCESS: Start w/ centred, sphered data matrix W .

Pick initial projection direction a and form $u_a = Wa$ ($= (u_{a1}, \dots, u_{an})$)

Solve: $\min_{a: a^T a = 1} H(\hat{f}(u))$ where $\hat{f}(u) = KDE$ formed from $u_1, \dots, u_n$.

INDEPENDENT COMPONENT ANALYSIS: Factor model:

$$\begin{cases} X_{i1} = a_{11} S_{i1} + a_{12} S_{i2} + \dots + a_{1p} S_{ip} \\ \vdots \\ X_{ip} = a_{p1} S_{i1} + a_{p2} S_{i2} + \dots + a_{pp} S_{ip} \end{cases} \left\{ \begin{array}{l} \text{if } S_i\text{'s are statistically} \\ \text{independent (not just uncorrelated)} \\ \text{we have ICA.} \end{array} \right.$$

We sphere X (AKA whitening) and since S, X have identity covariance matrix, we look for ORTHOGONAL A s.t. $X = SA^T$.

here columns of $S_{p \times p}$ are components e.g. set of microphones X_i picking up mixture of independent sources = S_x .

MUTUAL INFORMATION → $I(Y) = \sum_{j=1}^p H(Y_j) - H(Y)$ with $H(Y) = H(g)$ if $Y \sim g$ (pdf of Y).

↑
entropy of component ↑
entropy of entire vector

in the case of ICA, $Y = S = A^T X$ if we have $X = AS$ then $I(Y) = \sum_{j=1}^p H(Y_j) - H(X) = (-\log|\det A|)$ → this is minimised over all orthogonal A . (easiest to make by all or non-Gaussian or sparse)

PROJECTION PURSUIT REGRESSION $\rightarrow X \in \mathbb{R}^{p \times 1}$ p -dimensional explanatory variable vector.
 Y is target / response.

↳ model: $f(X) = \sum_{m=1}^M g_m(w_m^T X) \rightarrow$ additive model derived direction $v_m = w_m^T X$

note w_m are projection direction and $g_m(w_m^T X)$ varies only in direction w_m .

want to find: w_m s.t. the model fits well.

Fit model by minimising error $E = \sum_{i=1}^n \left\{ y_i - \sum_{m=1}^M g_m(w_m^T x_i) \right\}^2$
 over $\{g_m\}, \{w_m\}$.

If w_m are already found, minimisation of E is a smoothing problem (e.g. use smoothing spline).

e.g. if $M=1$, given w form $v_i = w^T x_i$ so this is 1D smoothing problem of y on v_i .

↳ the we've found a suitable g (via smoothing spline) $\rightarrow$ use this g to update w (by requiring error of w is minimised).

use: $g(w^T x_i) \approx g(w_{old}^T x_i) + g'(w_{old}^T x_i) (w - w_{old})^T x_i$

then find g again w/ this new w and subsequently update w . Iterate until convergence.

NEURAL NETWORKS (single layer NN) $x_1, \dots, x_p$ explanatory variables.

Y_k are outputs (w/ $K=1$ for regression).



z_m are created from linear combinations of the inputs $x_1, \dots, x_p$, and then Y_k is modelled as a fcnⁿ of linear combinations of z_m .

MODEL: $z_m = \sigma(\alpha_{0m} + \alpha_m^T X)$, $m=1, \dots, M \rightarrow \sigma = \frac{1}{1+e^{-x}}$ typically

$T_k = \beta_{0k} + \beta_k^T z$, $k=1, \dots, K$

$\hat{Y}_k = f_k(X) = g_k(T_k)$ (for regression, $g_k(T) = T$).

NB: similar to PPR but this has free g_m , whereas here fixed activation fcnⁿ σ used.
 (since we can write $g_m(w_m^T X) = \beta_m \sigma(\alpha_{0m} + \alpha_m^T X)$)

FITTING NN $\rightarrow$ minimise $R(\theta) = \sum_{i=1}^n R_i(\theta) = \sum_{i=1}^n \sum_{k=1}^K (y_{ik} - f_k(x_i))^2$.

use BACKPROPAGATION to minimise $R(\theta)$.

let $z_{mi} = \sigma(\alpha_{0m} + \alpha_m^T x_i)$ and $z_i = (z_{i1}, \dots, z_{iM})$.

BACKPROPAGATION $\rightarrow$ we can find: (chain rule)

$$\frac{\partial R_i(\theta)}{\partial \beta_{km}} = -2(y_{ik} - f_k(x_i)) g'_k(\beta_k^T z_i + \beta_{0k}) z_{mi}$$

$$\frac{\partial R_i(\theta)}{\partial \alpha_{ml}} = -2 \sum_{k=1}^K (y_{ik} - f_k(x_i)) g'_k(\beta_k^T z_i + \beta_{0k}) \beta_{km} \sigma'(\alpha_m^T x_i + \alpha_{0m}) x_{il}$$

Form a backpropagation / gradient-descent algorithm:

$$\beta_{km}^{(r+1)} = \beta_{km}^{(r)} - \gamma_r \sum_{i=1}^n \frac{\partial R_i(\theta)}{\partial \beta_{km}^{(r)}}, \quad \alpha_{ml}^{(r+1)} = \alpha_{ml}^{(r)} - \gamma_r \sum_{i=1}^n \frac{\partial R_i(\theta)}{\partial \alpha_{ml}^{(r)}}$$

$\hookrightarrow \gamma_r$ is called learning rate.

TREE-BASED METHODS e.g. for 3 explanatory variables X_1, X_2, X_3 we have 3D feature space. Assuming a response Y_i for individual i w/ features X_{i1}, X_{i2}, X_{i3}

$\hookrightarrow$ then tree partitions feature space into regions and assigns constant value to each region.

Split parameter space into rectangles $R_1, \dots, R_M$ + predict based on rectangle
 $\hookrightarrow$ regression: $f(x) = \sum_{m=1}^M c_m \mathbb{I}(x \in R_m)$. (constants c_m)

want to minimise $SSQ = \sum_{i=1}^n (Y_i - f(X_i))^2$ so $\hat{c}_m = \frac{1}{n_i} \sum_{i: X_i \in R_m} Y_i$

SEARCHING FOR BEST SPLIT $\rightarrow$ greedy algorithm $= \text{avg}(Y_i | X_i \in R_m)$.

define $R_1(j, s) = \{x | X_j \leq s\}$, $R_2(j, s) = \{x | X_j > s\}$.

seek j, s that minimises: $\min_{c_1} \sum_{X_i \in R_1(j, s)} (Y_i - c_1)^2 + \min_{c_2} \sum_{X_i \in R_2(j, s)} (Y_i - c_2)^2$

$\hookrightarrow$ N/B: c_1, c_2 are taken as average value of regions.

Fix j , then the sum only changes when s cross a data X_i . So only need to search $n+1$ values.

Do for every j and find minimum.

when do we stop? Usually stop when leaves contain S or fewer (X_i, Y_i) .

COST-COMPLEXITY PRUNING $\rightarrow$ given a grown large tree T_0 , search among all sub-trees $T \subset T_0$ for minimiser of cost-complexity.

COST-COMPLEXITY PRUNING: $Q_m(T) = \frac{1}{n_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)$ $\rightarrow$ "average" in R_m

cost-complexity $\rightarrow C_\alpha(T) = \sum_{m=1}^{|T|} n_m Q_m(T) + \alpha |T|$ where α controls balance of size of tree (complexity) vs. fit of data.

overfitting $\rightarrow |T|$ too large

underfitting $\rightarrow$ error $\sum_{m=1}^{|T|} n_m Q_m(T)$ too large.

IDEA is to find for each α , subtree $T_\alpha \subseteq T$ that minimises $C_\alpha(T)$.

CLASSIFICATION TREE $\rightarrow$ outcome is discrete category $1, \dots, K$ (not number)

we: $\hat{p}_{mk} = \frac{1}{n_m} \sum_{x_i \in R_m} \mathbb{I}(y_i = k) \rightarrow$ proportion of class k observations in node m .

we can classify observations in node m to class $k(m) = \max_k \hat{p}_{mk}$, i.e. the majority class in node m .

BOOTSTRAP: suppose we have iid sample $\{x_i\}_{i=1}^n$ from distribution F .

EMPIRICAL DISTRIBUTION $\rightarrow \hat{F}_n(x) = \frac{1}{n} \sum_{i=1}^n \mathbb{I}(x_i \leq x) = \frac{\# \text{ of elements in sample} \leq x}{n}$.

let $x_1^{(1)}, \dots, x_n^{(1)}$ be a random sample from $\{x_i\}_{i=1}^n$ with replacement.

$\{x_i^{(b)}\}_{i=1}^n$ is the b th bootstrap.

BAGGING $\rightarrow$ (bootstrap aggregating) suppose we have regression data and fit regression model to it: $Z = \{(x_1, y_1), \dots, (x_n, y_n)\}$ w/ $\hat{f}(x)$.

We can produce b bootstrap samples $Z^{(b)}$ (for $b=1, \dots, B$) each giving estimate $\hat{f}^{(b)}(x)$.

BAGGING ESTIMATE averages these: $\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{(b)}(x)$.

TRUE BAGGING ESTIMATOR: $\hat{f}_{\text{ag}}(x) = \mathbb{E}_{\hat{P}}(\hat{f}^*(x))$ where $\hat{f}^*$ is obtained from a set $Z^* = \{(x_i^*, y_i^*)\}_{i=1}^n$ where each $(x_i^*, y_i^*) \sim \hat{P}$.

where $\hat{P}$ is a uniform distribution that puts mass $\frac{1}{n}$ onto each (x_i, y_i)

Then $\hat{f}_{\text{bag}}$ is an estimate of TRUE BAGGING ESTIMATE as $B \rightarrow \infty$.

BAGGING works because: $\mathbb{E}[(y - \hat{f}^*(x))^2] \geq \mathbb{E}[(y - \hat{f}_{\text{ag}}(x))^2]$.
so bagging decreases MSE.

BOOTSTRAP TREES tend to be CORRELATED $\rightarrow$ limits effectiveness of bagging.

$\hookrightarrow$ let $w_1, \dots, w_B$ be iid w/ variance σ^2 .

$$\Rightarrow \text{var}(\bar{w}) = \frac{\sigma^2}{B} \rightarrow 0 \text{ as } B \rightarrow \infty.$$

BUT if $\text{cov}(w_b, w_d) = \sigma^2 p$ ($b \neq d$) then

$$\Rightarrow \text{var}(\bar{w}) = B^{-2} \left\{ \sum_{b=1}^B \text{cov}(w_b, w_b) + \sum_{b=1}^B \sum_{d=1, d \neq b}^B \text{cov}(w_b, w_d) \right\} = p\sigma^2 + \frac{1-p}{B} \sigma^2.$$

does NOT tend to 0 as $B \rightarrow \infty$.

$\therefore$ to reduce correlation between bootstrapped trees, select m of p variables at random and pick the best split from the m selected variables.

Random Forest = combination of bagging + random variable selection as above.

BOOSTING $\rightarrow$ technique to combine the outputs of many "weak" classifiers to make a strong classifier.

Consider classifier w/ outputs $Y \in \{-1, 1\}$. Then define $\overline{\text{err}} = \frac{1}{n} \sum_{i=1}^n \mathbb{I}\{y_i \neq g(x_i)\}$ for classifier g .

we build a sequence of ^{stronger} classifiers $g_m(x)$, $m=1, \dots, M$, w/ $g_1 = g(x)$.

$\hookrightarrow$ And obtain g_{m+1} from g_m by focusing more weight on those observations not classified correctly by g_m .

ALGORITHM $\rightarrow$ initially all weights set to $\frac{1}{n} = w_i$.

For $m=1, \dots, M$, Fit classifier $g_m(x)$ to training data using weights $\{w_i\}$.

$$\text{Compute: } \text{err}_m = \frac{\sum_{i=1}^n w_i \mathbb{I}\{y_i \neq g_m(x_i)\}}{\sum_{i=1}^n w_i}.$$

$$\text{Compute: } \alpha_m = \log \left\{ \frac{1 - \text{err}_m}{\text{err}_m} \right\}.$$

update

Then set $w_i \mapsto w_i \exp\{\alpha_m \mathbb{I}\{y_i \neq g_m(x_i)\}\}$. Do this for all $m=1, \dots, M$.

Then output 'strong' classifier $g^*(x) = \text{sgn} \left\{ \sum_{m=1}^M \alpha_m g_m(x) \right\}$.